SUN SEEBEYOND

eWAY™ ADAPTER FOR CICS USER'S GUIDE

Release 5.1.2

Part Number: 819-7401-10

Version 20061012104717

Contents

Chapter 1

Introduction

This chapter includes a brief description of IBM's Customer Information Control System™ (CICS™), and an overview of the Sun SeeBeyond eWay™ Adapter for CICS.

What's in This Chapter

1.1 CICS Transaction Server

IBM's Customer Information Control System (CICS), is IBM's most widely used proprietary, transaction monitor. CICS provides connectivity and online transaction management for mission-critical applications. It supports real-time distributed processing environments and online transaction processing (OLTP). According to IBM, CICS handles more than thirty billion transactions, processing more than one trillion dollars, per day.

CICS is the premier OLTP (On-Line Transaction Processing) products from IBM. It is used to access many file systems and databases including third party products. For IBM product, it interfaces with DB2, VSAM and IMS/DB. For non-IBM products, it interfaces with IDMS, ADABAS, DATACOM, to name a few. Most applications in CICS are written in COBOL, although it supports other languages such as PL/1.

OLTP systems provide accurate, up-to-date information within seconds, from terminals that give direct access to data held as either files or databases. CICS provides a company with numerous transaction processing and resource management functions, allowing the user to concentrate on developing application programs that meet that organization's specific business needs. CICS controls OLTP application programs in a distributed transaction processing (DTP) environment. CICS handles interactions between the terminal user and the application programs. Programs gain access to the CICS facilities with straightforward, high-level commands.

CICS provides:

- Communication functionality to terminals and systems that are required by application programs.
- Control of concurrently running programs that serve online users.
- Facilities for accessing databases and files.
- The ability to communicate with other CICS family members using Transmission Control Protocol/Internet Protocol (TCP/IP).
- Interactive facilities to configure specific systems.
- Recovery processing and data protection, should a problem occur.

1.2 The Sun SeeBeyond eWay™ Adapter for CICS

The Sun SeeBeyond eWay™ Adapter for CICS is an interface that enables remote bidirectional calls to CICS transactional programs. The CICS eWay includes a build tool, the Cobol Copybook Converter, that creates an Object Type Definition (OTD) from a Cobol Copybook file and generates eGate Object Type Definitions for use within the eGate environment. The Copybook file structures are passed into the CICS environment as the data buffer (Commarea).

A fixed Object Type Definition, the **CICS_eWay.CICSClient OTD**, designed to expose various essential portions of the CICS Java API, provides available methods and properties, as well as access to all message attributes.

The eWay can use either the IBM **CICS Transaction Gateway** version 5.1, 6.0, 6.0.1, or 6.1, or the **Sun SeeBeyond CICS Listener** as the underlying connection transport for accessing CICS z/OS transactions.

IBM CICS Transaction Gateway (CTG)

CTG provides an API (the External Call Interface or ECI) to call CICS transactions on the mainframe. The ECI allows a non-CICS application program to call a CICS program in a CICS server. Sun's CICS eWay uses this ECI method to connect to CICS. The CICS eWay connects to CICS with CTG running on a local-host (**Figure 1 on page 9**), on a second computer (**Figure 2 on page 9**), or on the mainframe (**Figure 3 on page 9**).

Figure 1 eGate and CTG running on the same host

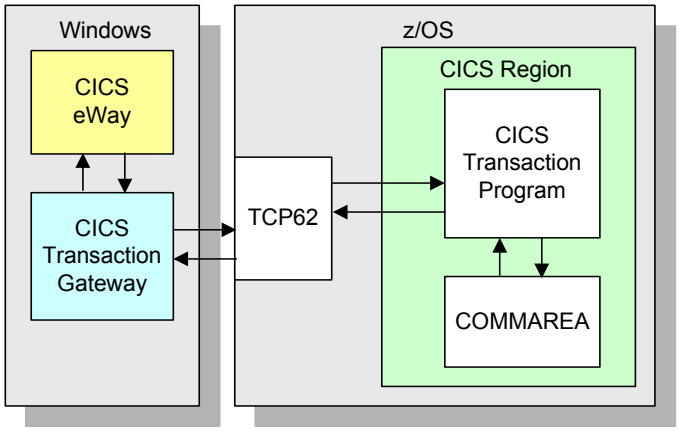


Figure 2 Remote connection with CTG on a UNIX or Windows host

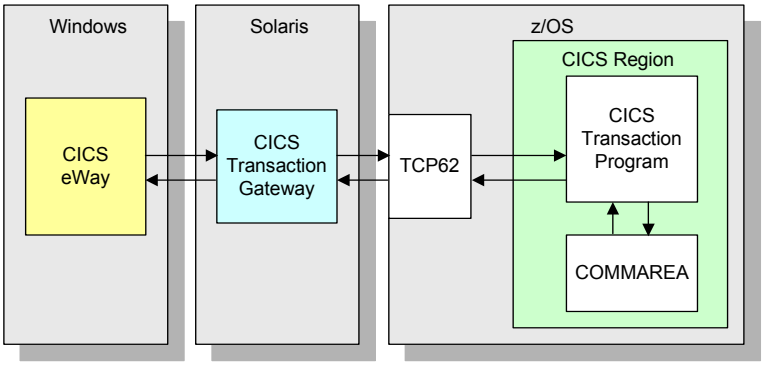
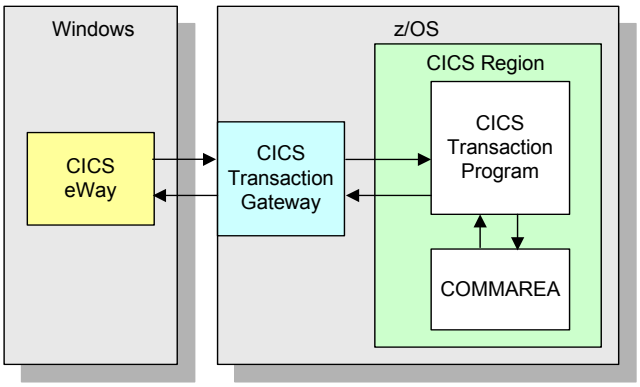


Figure 3 eGate connects with CTG running on the mainframe

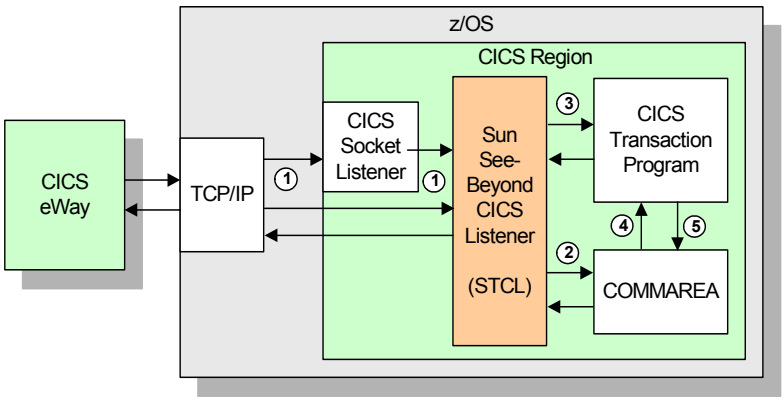


Note: *When using the CICS Transaction Gateway transport, data sent to CICS must be padded with spaces, if necessary, to match the full size of the commarea.*

Sun SeeBeyond CICS Listener

The CICS eWay connects to the IBM CICS Listener running on z/OS via the TCP/IP Sockets. The Listener accepts the incoming request and spawns a new process handing the socket connection off to the newly created process via TCP/IP givesocket()/ takesocket() function calls. The spawned process invokes the user written CICS application program through an EXEC CICS LINK.

Figure 4 Using the Sun SeeBeyond CICS Listener for Synchronous Transactions



The CICS eWay (Java version) communicates with the **Sun SeeBeyond CICS Listener** for Synchronous Transactions (See Figure 4) as follows:

1. An incoming Connect request is handled by the IBM CICS Socket Listener, which starts the Sun SeeBeyond CICS Listener Transaction and hands off the incoming connection via the IBM TCP/IP Give Socket and Take Socket interface.
2. The Sun SeeBeyond CICS Listener allocates a CICS COMMAREA and copies information from the CICS eWay COMMAREA to the actual CICS COMMAREA.
3. The Sun SeeBeyond CICS Listener issues an EXEC CICS LINK to requested CICS Transaction Program passing it the newly allocated COMMAREA.
4. The requested CICS Transaction obtains data from the COMMAREA, performs typical business rule processing and then returns its results in the COMMAREA and returns control back to the Sun SeeBeyond CICS Listener.
5. The Sun SeeBeyond CICS Listener copies information from the CICS COMMAREA back to the CICS eWay COMMAREA.
6. The Sun SeeBeyond CICS Listener goes into a listen mode and waits for the next incoming Transaction Program request.

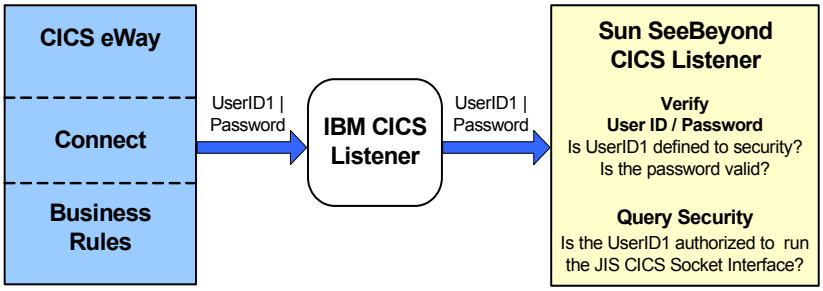
   The process continues until the Sun SeeBeyond CICS Listener Timeout is exceeded or a disconnect request is received from the CICS eWay.

1.3 z/OS CICS Security Considerations

Security Considerations for Sun SeeBeyond CICS Listener

The CICS eWay, using the Sun SeeBeyond CICS Listener as the underlying connection transport, utilizes three modes of security with z/OS: Connection Logic, Request Link to Program, and Request Start Transaction. The userID and password are defined in the eWay properties file. The connection manager uses the userID and password in the properties file to start the Sun SeeBeyond CICS Listener on z/OS. During Business Rules processing, requests that flow into the Sun SeeBeyond CICS Listener can use the userID and password from the properties file, or can be overwritten in the Collaborations.

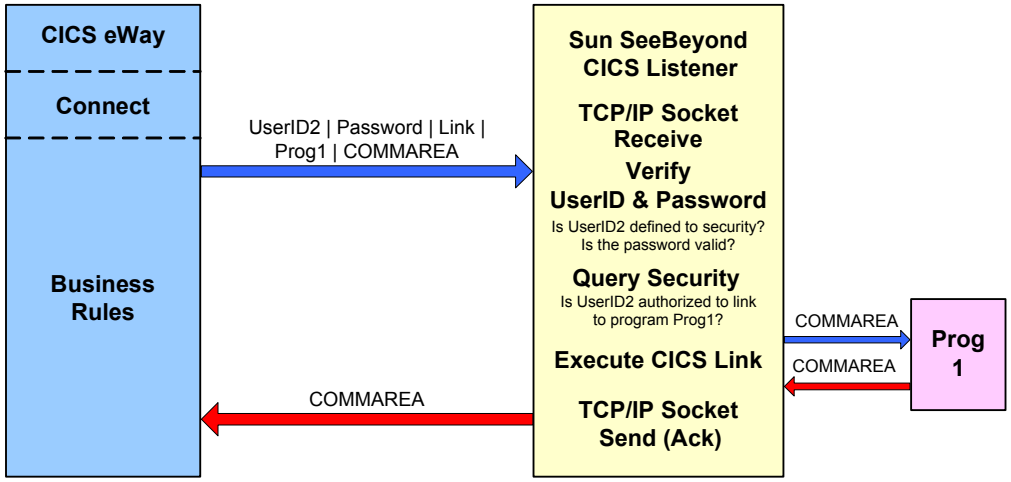
Figure 5 Connection Logic



Connection Logic

For the Connection Logic mode (Figure 5), the userID and password, passed from the CICS eWay through the IBM CICS listener and into the Sun SeeBeyond CICS Listener, must be defined for the z/OS security system (RACF, for example). The userID must be authorized by the z/OS security system to run CICS transaction "xxxx" inside of CICS. The default value for "xxxx" is STCL, and can be changed in the properties of the Connection Manager in the CICS eWay.

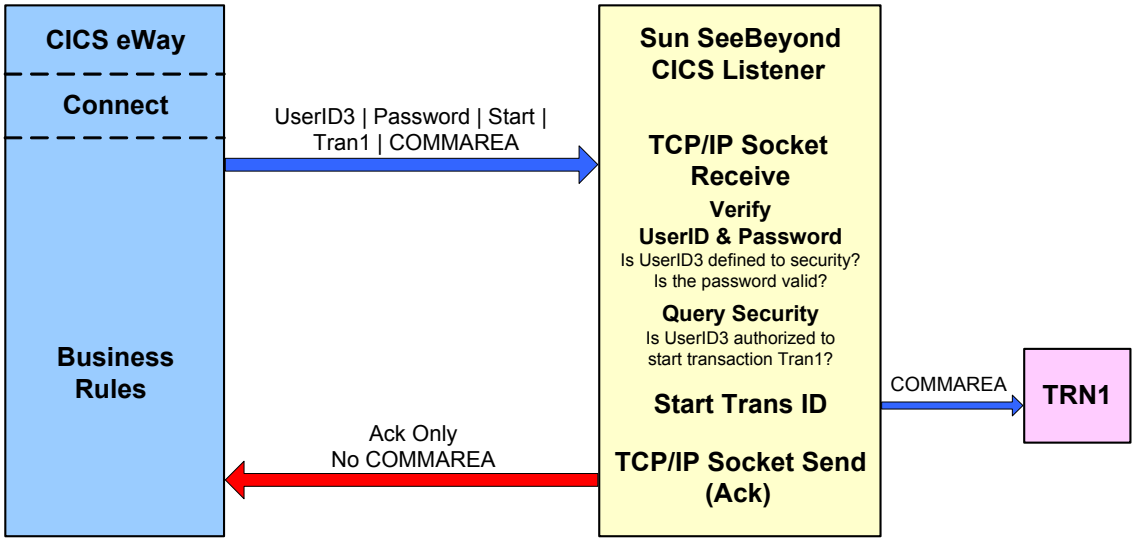
Figure 6 Business Rules Request to Program



Request Link to Program

For the Request Link to Program mode (Figure 6 on page 12), the userID and password passed from the CICS eWay to the Sun SeeBeyond CICS Listener must be defined for the z/OS security system (RACF, for example). The userID must be authorized by the z/OS security system to run CICS program "prog1" inside of CICS. The default value for "prog1" is set in the properties of the CICS eWay, and can be overridden in the Collaboration for each request sent into the Sun SeeBeyond CICS Listener.

Figure 7 Business Rules Request Start Transaction



Request Start Transaction

For the Request Start Transaction mode (Figure 7), the userID and password passed from the CICS eWay to the Sun SeeBeyond CICS Listener must be defined for the z/OS security system (RACF, for example). The userID must be authorized by the z/OS security system to start CICS transaction "TRN1" inside of CICS. The default value for "TRN1" is set in the properties file of the CICS eWay, and can be overridden in the Collaboration for each request sent into the Sun SeeBeyond CICS Listener.

Security Considerations for IBM CICS Transaction Gateway

For information on CICS Transaction Gateway security validation refer to the following:

- IBM documentation *CICS Transaction Gateway z/OS Administration* or the *CICS Transaction Gateway Administration* Guide for your specific operating system, available at:

  http://www-306.ibm.com/software/htp/cics/library/cicstgv5.html

- Readme.txt for CTG 5.1 provided on the CTG 5.1 installation CD_ROM
- APAR OW55570 (for RACF)

1.4 What's New in This Release

The Sun SeeBeyond eWay Adapter for CICS includes the following changes and new features:

New for Version 5.1.2

- This is a maintenance release. No new features.

New for Version 5.1.1

- Added support for CICS Transaction Gateway version 6.1.

New for Version 5.1.0

- Version Control: An enhanced version control system allows you to effectively manage changes to the eWay components.
- Manual Connection Management: Establishing a connection can now be performed automatically (configured as a property) or manually (using OTD methods from the Java Collaboration).
- Multiple Drag-and-Drop Component Mapping from the Deployment Editor: The Deployment Editor now allows you to select multiple components from the Editor's component pane, and drop them into your Environment component.
- Support for Runtime LDAP Configuration: eWay configuration properties now support LDAP key values.
- MDB Pool Size Support: Provides greater flow control (throttling) by specifying the maximum and minimum MDB pool size.
- Connection Retry Support: Allows you to specify the number of attempts to reconnect, and the interval between retry attempts, in the event of a connection failure.
- Connectivity Map Generator: Generates and links your Project's Connectivity Map components using a Collaboration or Business Process.
- Connection Pooling: Connections can be reused without the overhead of creating new connection.

- Added support for CICS Transaction Gateway version 6.0.1.
- Added support for CICS version 3.1.

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*.

1.5 What's in This Document

This book includes the following chapters:

- **Chapter 1 "Introduction"** provides an overview of the CICS eWay Intelligent Adapter, including a brief description of CICS Transaction Gateway, Sun SeeBeyond CICS Listener, and CICS eWay security for z/OS.
- **Chapter 2 "Installing the CICS eWay"** provides directions for installing the CICS eWay, installing and configuring the Sun SeeBeyond CICS Listener, and accessing the accompanying documentation and sample Projects.
- **Chapter 3 "Configuring the CICS eWay"** describes the process of configuring the CICS eWay to run in your environment.
- **Chapter 4 "Using the CICS eWay with eInsight"** describes how to use the CICS eWay with the Sun Java Composite Application Platform Suite's eInsight Business Process Manager and the Web Services interface.
- **Chapter 5 "Implementing a CICS eWay Project"** describes the features and functionality of the CICS eWay using the eGate Integrator and the Collaboration Editor (Java).
- **Chapter 6 "Java Methods"** describes the CICS eWay's Java classes and provides directions for accessing the CICS eWay Javadoc.

CICS eWay Javadoc

A CICS eWay Javadoc is also provided, that documents the Java methods available with the CICS eWay. The Javadoc is uploaded with the eWay's documentation file (CICSeWayDocs.sar) and downloaded from the Documentation tab of the Sun Java 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.

1.5.1 Scope of the Document

This user's guide provides a description of the CICS eWay Intelligent Adapter. It includes directions for installing the eWay, configuring the eWay properties, and implementing the eWay's sample Projects. This document is also intended as a reference guide, listing available properties, functions, and considerations. For a reference of available CICS eWay Java methods, see the associated Javadoc.

1.5.2 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.

1.5.3 Text Conventions

The following conventions are observed throughout this document.

Table 1 Text Conventions

Text Convention	Used For	Examples
Bold	Names of buttons, files, icons, parameters, variables, methods, menus, and objects	▪ Click **OK**. ▪ On the **File** menu, click **Exit**. ▪ Select the **eGate.sar** file.
`Monospaced`	Command line arguments, code samples; variables are shown in ***bold italic***	`java -jar` ***`filename`***`.jar`
Blue bold	Hypertext links within document	See **Text Conventions** on page 15
Blue underlined	Hypertext links for Web addresses (URLs) or email addresses	http://www.sun.com

1.6 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

1.7 Documentation Feedback

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

CAPS_docsfeedback@sun.com

Chapter 2

Installing the CICS eWay

This chapter contains installation information for the CICS eWay. In addition, this chapter also includes information for installing and configuring the **Sun SeeBeyond CICS Listener and IBM CICS Transaction Gateway**. One of these must be installed and configured as the underlying connection transport to access z/OS CICS transactions.

What's in This Chapter

2.1 CICS eWay System Requirements

The CICS eWay Readme contains the latest information on:

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

2.1.1 eWay Requirements for CICS Transaction Gateway

To support the CICS OTD, the CICS Transaction Gateway JAR files, **ctgclient.jar** and **ctgserver.jar**, must be copied to the following location:

```
<JavaCAPS51>\edesigner\lib
```

where *<JavaCAPS51>* is the directory where the Sun Java Composite Application Platform Suite is installed.

This is required when CICS Transaction Gateway 5.1 is used as the underlying connection transport.

Note: *You must restart the Enterprise Designer after copying the JAR files to your system.*

2.1.2 External System Requirements

Software Requirements

- IBM CICS Transaction Server version 1.3, 2.2, 2.3, or 3.1
- An underlying connection transport using either of the following:
  - **I**BM CICS Transaction Gateway 5.1, 6.0, 6.0.1, and 6.1. CTG version 6.0.1 is supported on Windows and UNIX platforms only. CTG 6.1 is supported on z/OS platforms only. See **"Requirements for the CICS Server to use CICS Transaction Gateway" on page 17**.
  - Sun SeeBeyond CICS Listener (see **"Requirements for the CICS Server to use the Sun SeeBeyond CICS Listener" on page 17**)

Requirements for the CICS Server to use CICS Transaction Gateway

For information about configuring CICS Transaction Gateway, version 5.1, 6.0, 6.0.1, or 6.1 to communicate with a CICS server over a TCP/IP usingTCP62, see the "Setting Up Client/Server Communications" section of the *CICS Transaction Gateway Administration* manual for your specific platform available at:

http://www-306.ibm.com/software/htp/cics/ctg/library

Requirements for the CICS Server to use the Sun SeeBeyond CICS Listener

To enable the eWay to communicate correctly with CICS using the Sun SeeBeyond CICS Listener, requires the following:

- OS/390 V2R10 or above (see **"eWay Requirements for CICS Transaction Gateway" on page 17**)
- Resource Access Control Facility (RACF) or an equivalent security product
- IBM CICS Transaction Server version 1.3, 2.2, 2.3, or 3.1
- IBM MVS TCP/IP socket runtime libraries, installed and configured for each CICS region in which the Sun SeeBeyond CICS Listener will be run. For more

information, refer to IBM's TCP/IP V3R2 for MVS: CICS TCP/IP Socket Interface Guide.

- COBOL for z/OS and Language Environments

Sun SeeBeyond CICS Listener Requirements for Invoking DB2 Applications

There are two different techniques that can be used for invoking DB2 application programs through the Sun SeeBeyond CICS Listener:

- RCT entry for the STCL Sun SeeBeyond CICS Listener Transaction
- PPT entry to redirect DB2 application to another AOR (Application Owning Region)

For detailed information refer to **"CICS Listener Considerations for Invoking DB2 Applications" on page 43**.

2.2 Installing the CICS eWay

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

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

2.2.1 Installing the eWay on a JavaCAPS Supported System

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

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

   To upload the CICS eWay User's Guide, Help file, Javadoc, Readme, and sample Projects, select the following:
   - **CICSeWayDocs**

   If your Project uses the Cobol Copybook Converter, select the following:

- **CobolCopyBook**
- **CobolCopyBookDocs**

3 Once you have selected all of your products, click **Next** in the top-right or bottom-right corner of the **Select Sun Java Composite Application Platform Suite Products to Install** box.

4 From the **Selecting Files to Install** box, locate and select your first product's SAR file. Once you have selected the SAR file, click **Next**. Your next selected product appears. Follow this procedure for each of your selected products. The **Installation Status** window appears and installation begins after the last SAR file has been selected.

5 Once your product's installation is finished, continue installing the Sun Java Composite Application Platform Suite as instructed in the *Sun Java Composite Application Platform Suite Installation Guide.*

Adding the eWay to an Existing Suite Installation

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

1 Complete steps 1 through 4 above.

2 Once your product's installation is finished, open the Enterprise Designer and select **Update Center** from the Tools menu. The **Update Center Wizard** appears.

3 For Step 1 of the wizard, simply click **Next**.

4 For Step 2 of the wizard, click the **Add All** button to move all installable files to the **Include in Install** field, then click **Next**.

5 For Step 3 of the wizard, wait for the modules to download, then click **Next**.

6 The wizard's Step 4 window displays the installed modules. Review the installed modules and click **Finish.**

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

After Installation

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

2.2.2 Installing eWay Enterprise Manager plug-ins

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

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

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

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

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

CICS eWay Alert Codes

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

To View the eWay Alert Codes

1. Add the eWay Enterprise Manager plug-in for this eWay.
2. From the Enterprise Manager's **Explorer** toolbar, click the **Configuration** icon.
3. Click the **Web Applications Manager** tab and go to the **Manage Alert Codes** tab. Your installed alert codes are displayed under the **Results** section. If your eWay alert codes are not available displayed under **Results**, do the following
   - A. From the **Install New Alert Codes** section, browse to and select the eWay alert properties file for the application plug-in that you added. The alert properties files are located in the **alertcodes** folder of your Sun Java Composite Application Platform Suite installation directory.
   - B. Click **Deploy**. The available alert codes for your application are displayed under **Results**. A listing of available this eWay's alert codes is displayed in Table 2.

Table 2 CICS eWay Alert Codes

Alert Code	Description	User Action
CICSCLIENTCLIENTEWAY-CONNECT-FAILED000002	Failed to connect to CICS.	▪ CICS is down. Verify that the system you are connecting to is running. ▪ Properties are incorrect. Verify that your parameters are correct (host name, port, and so forth). ▪ CTG configuration is incorrect. Verify CTG installation and configuration is correct. Refer to the log for more information.

An alert code is a warning that an error has occurred. It is not a diagnostic. The user actions noted above are just some possible corrective measures you may take. Refer to the log files for more information. For information on managing and monitoring alert codes and logs, see the *Sun SeeBeyond eGate Integrator System Administration Guide.*

2.3 Sun SeeBeyond CICS Listener Installation for z/OS

The following section provides directions for installing the Sun SeeBeyond CICS Listener to a z/OS operating system from the installation CD-ROM or from a 3480 Tape. Directions are also included for installing CICS CEDA definitions, adding the CICS eWay load modules to the CICS DFHRPL concatenation, and using the Sun SeeBeyond CICS Listener monitoring screen to verify that all components are properly installed and working correctly.

2.3.1 Downloading the Sun SeeBeyond CICS Listener Zip File

The Sun SeeBeyond CICS Listener ZIP file, **CICS_Listener.zip**, can be downloaded during the Sun Java Composite Application Platform Suite Installation. To download **CICS_Listener.zip** to your computer do the following:

1. Locate the Products installation CD-ROM containing the **CICSeWay.sar** file.
2. Open the CD-ROM to view and locate the **CICSeWay.sar** file.
3. Right-click the **CICSeWay.sar** file, and from the shortcut menu, select the "Open With" option and select a zip program.
4. From the opened CICSeWay.sar file, click on and drag the **CICS_Listener.zip** file to your Desktop. Close the zip program. The CICS_Listener.zip file is now available from your computer's Desktop file.

2.3.2 Installing the Sun SeeBeyond CICS Listener from CD for z/OS

The following explains how to restore the Sun SeeBeyond CICS Listener files to your MVS system in a usable state. The files are packaged on MVS, and can be for transferred into a data set by using the TSO transmit (XMIT) command. This converts a PDS into FB 80 files, which can then be sent by FTP. Files are downloaded to a computer and then compressed with ZIP utility.

1. Download the **CICS_Listener.zip file** to your PC (see **"Downloading the Sun SeeBeyond CICS Listener Zip File" on page 21**), and unzip the files.
2. Create two MVS datasets to receive the files, as follows:

```
//DD1     DD DSN=USER.XMIT.CICSLOAD,
//           DISP=(NEW,CATLG,DELETE),
//           RECFM=FB,LRECL=80,BLKSIZE=3120,DSORG=PS,
//           SPACE=(3120,(48,5)),
//           UNIT=diskunit

//DD2     DD DSN=USER.XMIT.JCLLIB,
//           DISP=(NEW,CATLG,DELETE),
//           RECFM=FB,LRECL=80,BLKSIZE=3120,DSORG=PS,
//           SPACE=(3120,(30,5)),
//           UNIT=diskunit
```

3. Upload (FTP, IND$FILE) the unzipped files to MVS using a binary file transfer method (no CRLF or ASCII translation).
4. Restore the files to PDS by using the **Receive** command on MVS:
   A. Issue command: TSO RECEIVE INDATASET(uploaded.dataset)
   B. When prompted by the message:

```
INMR906A Enter restore parameters or 'DELETE' or 'END' +
```

   enter the following:

```
DA(name.of.your.library) UNIT(unit) VOLUME(volume)
```

Note: *The UNIT() and VOLUME() operands are optional but shown in case your installation requires them.*

We suggest using the following names for your received datasets:

```
STC.XMIT.CICS.CICSLOAD
STC.XMIT.CICS.JCLLIB
```

These names match our documentation. Change them as your facility requires.

2.3.3 Installing the Sun SeeBeyond CICS Listener from 3480 Tape

The Sun SeeBeyond CICS Listener installation for z/OS is provided on an installation tape containing the following datasets (Table 3):

Table 3 Tape Contents

File	Dataset Name	Contents
1	TAPE.STC.RESTORE.JCL	JCL sample to load this tape to disk (RESTORE JCL).
2	TAPE.STC.CICS.JCLLIB	Installation jobs and control cards for the CICS eWay.
3	TAPE.STC.CICS.CICSLOAD	Load library that containing the load modules for the CICS eWay.

Copying the Tape Contents to Disk

1 Create, customize, and submit the following job to copy the RESTORE JCL to disk:

```
1  //JOBNAME JOB (SYSTEM),'SYSPROG',NOTIFY=&SYSUID,REGION=6M,
2  //     CLASS=A,MSGLEVEL=(1,1),MSGCLASS=X
3  //*
4  //* COPY SAMPLE JCL TO DISK
5  //*
6  //IEBGENER EXEC PGM=IEBGENER
7  //SYSPRINT DD   SYSOUT=*
8  //SYSUT1   DD   DSN=TAPE.STC.RESTORE.JCL,
9  //              UNIT=tapeunit,
10 //              VOL=SER=STC390,
11 //              LABEL=(1,SL,,,EXPDT=98000),
12 //              DISP=OLD
13 //SYSUT2   DD   DSN=customer.dataset,
14 //              SPACE=(TRK,(10,10),RLSE),
15 //              UNIT=diskunit,
16 //              VOL=SER=diskvol,
17 //              DCB=(LRECL=80,BLKSIZE=3120,RECFM=FB),
18 //              DISP=(NEW,CATLG,DELETE)
19 //SYSIN    DD   DUMMY
```

2 Edit the fore mentioned code for your system as follows:

A Replace the Jobcard in the first two lines with one that meets your installation standards.

B Replace **tapeunit** in line 9 with the unit name of your tape drive: for example, UNIT=TAPE.

C Some Tape Management systems require an **EXPDT** value (see line 11) to indicate a foreign tape.

D Specify a **customer.dataset** name (see line 13) that meets your installation standards.

E Specify a unitname for the disk (replacing **diskunit** in line 15), to meet your installation standards: for example, UNIT=3390.

F Specify the volume serial number of the disk (replacing **diskvol** in line 16), where the dataset is to be created.

3 Once the job has completed successfully, customize and submit the sample JCL in CUSTOMER.DATASET job to copy the entire contents of the installation tape to disk. Customize displayed in the first 15 lines below

```
1  //JOBNAMEx   JOB(SYSTEM),'SYSPROG',NOTIFY=&SYSUID,REGION=6M,
2  //            MSGLEVEL=(1,1),MSGCLASS=X,CLASS=A
3  //*
4  //TAPECOPY PROC PREFIX=custpref, <== CUSTOMER HIGH LEVEL
QUALIFIER
5  //         TAPEUNIT=tapeunit,    <== UNIT NAME FOR TAPE DRIVE
6  //         DISKUNIT=3390,        <== UNIT NAME FOR DISK DRIVE
7  //          VOLSER=diskvol,       <== DISK VOLUME FOR INSTALL
DATASETS
8  //          LOADBLK=TRK,          <== BLOCKING FACTOR FOR CICS
LOAD
9  //          LOADPRI=45,           <== PRIMARY ALLOCATION FOR
CICS LOAD
10 //          LOADSEC=15,           <== SECONDARY ALLOCATION FOR
CICS LOAD
11 //          LOADDIR=10,           <== DIRECTORY BLOCKS FOR CICS
LOAD
12 //          JCLLBLK=TRK,          <== BLOCKING FACTOR FOR JCL
LIBRARY
13 //          JCLLPRI=1,            <== PRIMARY ALLOCATION FOR JCL
LIBRARY
14 //          JCLLSEC=1,            <== SECONDARY ALLOCATION FOR
JCL LIB
15 //          JCLLDIR=10            <== DIRECTORY BLOCKS FOR JCL
LIBRARY:
```

A Replace the Jobcard in the first two lines with one that meets your installation standards.

B Change the **custpref** value (see line 4) in the TAPECOPY procedure to a high-level-qualifier for your installation datasets.

C Replace **tapeunit** in line 4 with the unit name of your tape drive: for example, TAPEUNIT=TAPE.

D Change the **3390** for the DISKUNIT in line 6, to meet your installation standards: for example, DISKUNIT=SYSDA.

E Specify the volume serial number of the disk (replacing **diskvol** in line 7), where the dataset is to be created.

2.3.4 Creating the STCCLCFG File

Customize the **Jobcard** and **Dataset** names, and submit the **STCCLCFG** job found in **JCLLIB**.

2.3.5 Installing the CICS CEDA Definitions

Customize the file definition in the JCLLIB member CEDALCFG to conform to your file naming conventions, and to match the file name created in the previous step. Customize and submit job STCLCEDA to create CICS CEDA definitions for the eWay.

Using the CICS RDO interface, install the new definitions in group STCLSN into CICS, and add the group name STCLSN, to your start-up list.

2.3.6 Adding the CICS eWay Load Modules to CICS DFHRPL Concatenation

Add the following data set to the DFHRPL concatenation under CICS:

```
//   DD DSN=&PREFIX..STC.CICS.CICSLOAD,DISP=SHR
```

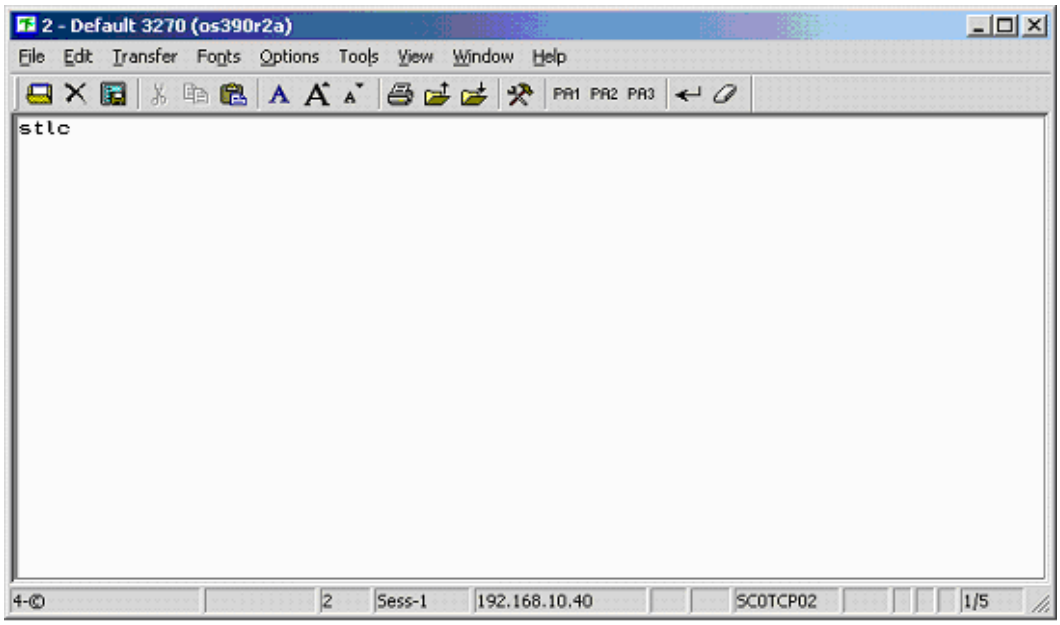
Note: *Remember to re-cycle your CICS region to pick up this new library.*

2.4 Sun SeeBeyond CICS Listener Configuration Maintenance for z/OS

After the STCCLCFG Configuration File has been created and the CEDA file definition has been installed, use the Sun SeeBeyond CICS Configuration File Maintenance screen to view and update the contents of the STCCLCFG configuration file.

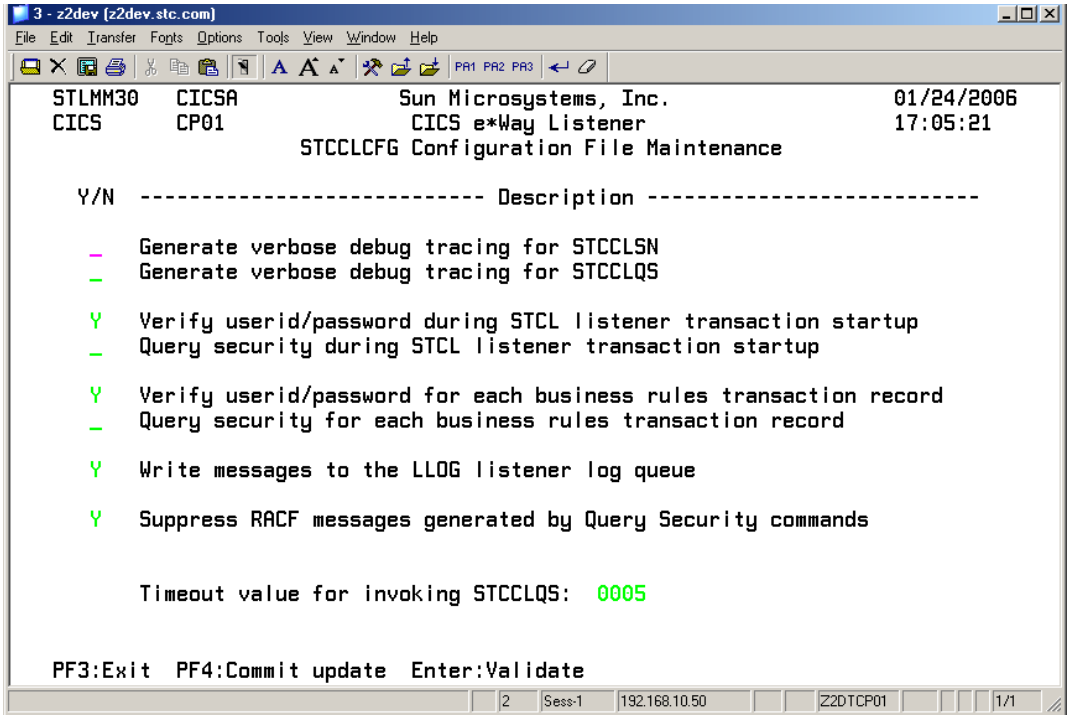
1 Logon to the CICS region that the Sun SeeBeyond CICS Listener is running in. Clear the screen, type in **stlc** as shown in Figure 8, and press **Enter**.

Figure 8



2 The **Configuration File Maintenance** screen appears as displayed in **Figure 9 on page 26**.

Figure 9



3 The fields of the Listener Configuration Maintenance screen are described as follows:

- **Generate verbose debug tracing for STCCLSN**:This flag controls the logging of debug tracing messages issued by the STCCLSN Sun SeeBeyond CICS Listener Program. These tracing messages are written to the LLOG transient data queue.

  If the "**Write messages to the LLOG listener log queue**" field on this screen is turned off, then debug tracing will NOT be logged.

  - **Y**: yes, log debug tracing messages.
  - **N** or blank: no, do not log debug tracing messages.

- **Generate verbose debug tracing for STCCLQS**:This flag controls the logging of debug tracing messages issued by the STCCLQS Sun SeeBeyond CICS Listener Query Security Program. These tracing messages are written to the LLOG transient data queue.

  If the "**Write messages to the LLOG listener log queue**" field on this screen is turned off, then debug tracing will NOT be logged.

  - **Y**: yes, log debug tracing messages.
  - **N** or blank: no, do not log debug tracing messages.

- **Verify userid/password during STCL listener transaction startup**:This flag controls userid/password verification at the STCL Sun SeeBeyond CICS

Listener Transaction startup. The userid and password are passed from the CICS eWay to the IBM Listener program in the user area of Connection Request Record. The IBM Listener passes this user area to the Sun SeeBeyond CICS Listener when it spawns it as a child listener task.

If this flag is turned on, the Sun SeeBeyond CICS Listener Program startup logic issues a Verify Password command to check whether the userid is defined for the RACF or any equivalent security system, and if the password is valid and current.

- **Y**: yes, verify userid/password.
- **N** or blank: no, do not verify userid/password.

- **Query Security during STCL listener transaction startup**: This flag controls Query Security processing at the STCL Sun SeeBeyond CICS Listener Transaction startup. The userid and password are passed from the CICS eWay to the IBM Listener program in the user area of Connection Request Record. The IBM Listener passes this user area to the Sun SeeBeyond CICS Listener when it spawns it as a child listener task.

  If this flag is turned on, the Sun SeeBeyond CICS Listener Program startup logic invokes the STLQ Sun SeeBeyond Listener Query Security transaction to check if the userid is authorized to run the STCL Sun SeeBeyond Listener transaction.

  - **Y**: yes, perform Query Security processing.
  - **N** or blank: no, do not Query Security processing

- **Verify userid/password for each business rules transaction record**: This flag controls userid/password verification for each Business Rules transaction sent in from the CICS eWay. The userid and password are passed in from the CICS eWay in the Application Control Record (ACR) that contains the Business Rules transaction data.

  If this flag is turned on, the Sun SeeBeyond CICS Listener program Business Rules logic issues a Verify Password command to check whether the userid is defined for the RACF or any equivalent security system, and if the password is valid and current.

  - **Y**: yes, verify userid/password.
  - **N** or blank: no, do not verify userid/password.

- **Query Security for each business rules transaction record:** This flag controls Query Security processing for each Business Rules transaction sent in from the CICS eWay. The userid and password are passed in from the CICS eWay in the Application Control Record (ACR) that contains the Business Rules transaction data.

  If this flag is turned on, the Sun SeeBeyond CICS Listener Program Business Rules logic invokes the STLQ Sun SeeBeyond Listener Query Security transaction to check whether the userid is authorized to run the requested customer application program or transaction that is specified in the ACR.

- **Y**: yes, perform Query Security processing.
- **N** or blank: no, do not Query Security processing.

- **Write messages to the LLOG listener log queue:** This flag controls the logging of messages to the LLOG listener log queue.

  If this flag is turned off, NO messages (normal information as well as debug tracing) will be written by either the STCCLSN (Sun SeeBeyond CICS Listener) or STCCLQS (Sun SeeBeyond Listener Query Security) programs.

  - **Y**: yes, write log messages to the LLOG listener log queue.
  - **N** or blank: no, do not write log messages to the LLOG listener log queue.

- **Suppress RACF messages generated by Query Security commands:** This flag controls RACF informational message logging for Query Security exceptions.

  If this flag is turned off, then every Query Security command that results in a negative result for Control, Alter, Update, or Read will cause respective RACF information messages to be written to the system message log. This could create unnecessary, high volume logging on the system message log files.

  It is recommended that this flag be turned off for debugging or low volume testing purposes only.

  In a production environment, it is recommended that this flag be turned on to suppress the RACF informational messages generated by Query Security exception conditions.

  - **Y**: yes, suppress RACF messages generated by Query Security processing.
  - **N** or blank: no, do not suppress (in other words, allow) RACF messages to be generated by Query Security processing.

- **Timeout value for invoking STCCLQS:** This value is used to control the amount of time the STCCLSN Sun SeeBeyond CICS Listener program waits for the STLQ Listener Query Security Transaction to return a response. Valid values are 0000 to 9999. Since the STLQ transaction, under normal CICS processing conditions, executes within one second, it is recommended that the value be set above 1 and below 10. However, if the CICS region that the Sun SeeBeyond CICS Listener is running in experiences occasional performance bottle necks, you may choose to set this value higher to avoid unnecessary STLQ timeout errors during peek load conditions.

  This value does not affect the time STLQ takes to respond. It simply sets a timeout threshold above which the STCCLSN stops waiting for a response from STLQ and issues an error message back to the CICS eWay indicating that an STLQ timeout has occurred. For example, if this timeout value is set to 5, and STLQ executes in 1 second, then STCCLSN will “wake up” after 1 second, not the full 5 seconds.

4 The pfkeys for this screen are as follows:

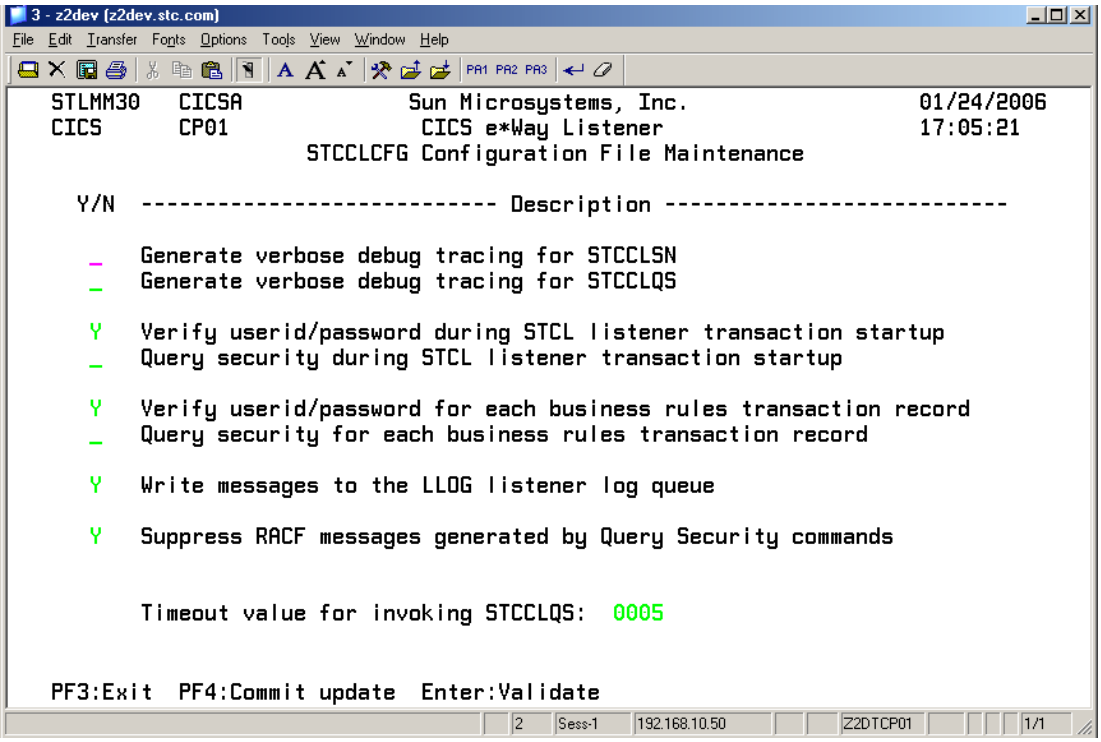
- **PF3**: exit.

- **PF4**: commit the updates on the screen to the STCCLCFG file and update all existing Listener Program Areas (LPA's) in CICS storage.
- **Enter**: validates the onscreen data for errors.

The user may make any necessary changes, then use the **Enter** key to validate any new values that have been entered. IF no error conditions occur for the data on the screen, then the user must press the PF4 key to commit the updates to the STCCLCFG Listener Config VSAM File. The updated screen appears as displayed in Figure 10. A message appears at the bottom of the screen that indicating the file and whether any LPA's were updated.

During commit processing, the program will also automatically propagate the new data to all existing Sun SeeBeyond Listener Program Areas (LPA's) in the same CICS region. There is one LPA for each instance of the Sun SeeBeyond CICS Listener in the region. The STCCLSN (Sun SeeBeyond CICS Listener) program uses its copy of the LPA to store monitoring statistics and to retrieve data used to control the execution of the Sun SeeBeyond CICS Listener. Live updates to the LPAs provide the capability for real time control of the configuration flags and timers for all active Sun SeeBeyond CICS Listeners executing at that time.

Figure 10



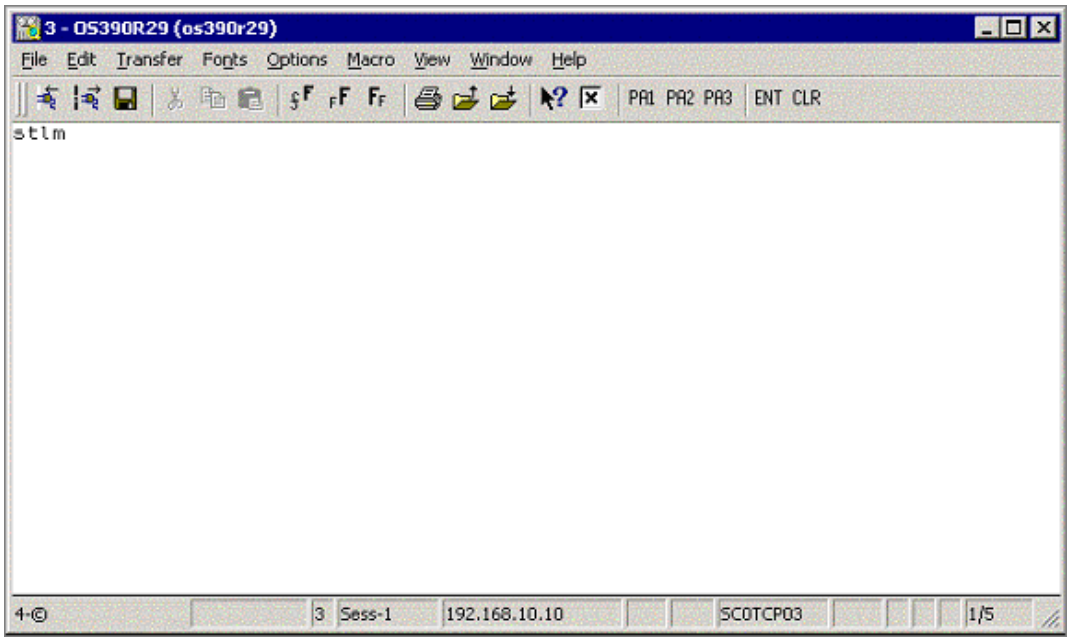
5 Continue making any necessary changes, committing them by pressing **PF4**. To exit the screen press **PF3**.

2.4.1 The Sun SeeBeyond CICS Listener Monitor Screen for z/OS

After the components are installed, use the Sun SeeBeyond CICS Listener monitoring screen to verify that all components are properly installed and working correctly.

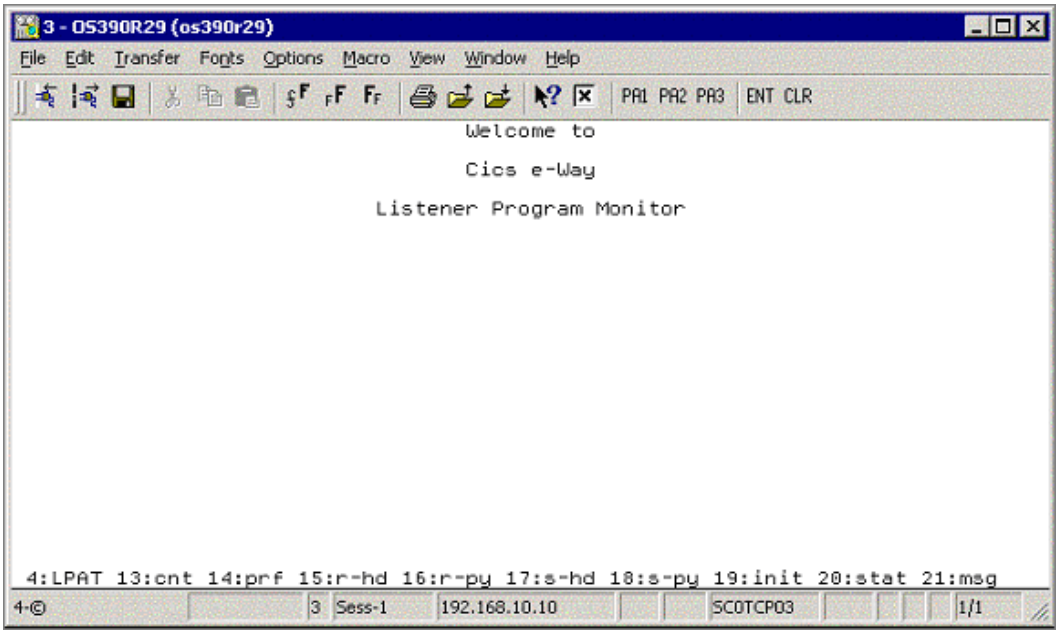
1 Logon to the CICS region that the Sun SeeBeyond CICS Listener is running in. Clear the screen, and type in **stlm** as displayed in Figure 11. Press **Enter**.

Figure 11



2 The menu screen appears as displayed in Figure 12.

Figure 12

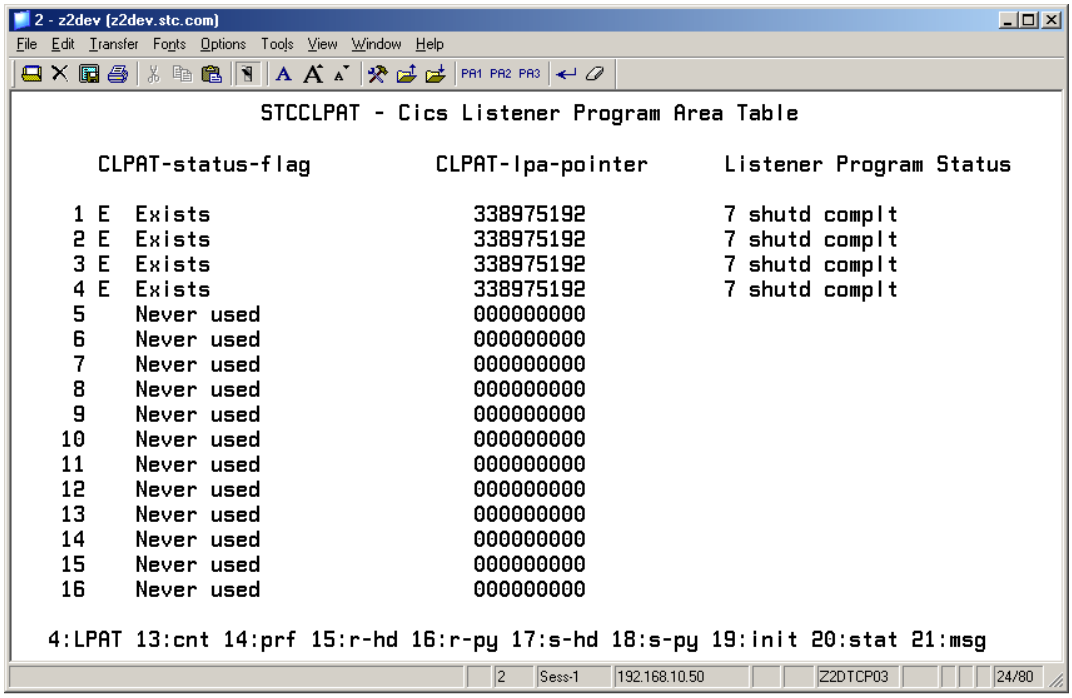


3 The **pfkeys** for this screen, and all of the monitor screens shown in this document, are:

- **PF4**: displays the LPAT (Listener Program Area Table).
- **PF7**: Scrolls up through the list of LPAs.
- **PF8**: Scrolls down through the list of LPAs
- **PF13**: displays the LPA counts and polling rates.
- **PF14**: displays the LPA performance statistics.
- **PF15**: displays the LPA last request header received from the CICS eWay.
- **PF16**: displays the LPA last request payload received from the CICS eWay.
- **PF17**: displays the LPA last response header sent to the CICS eWay.
- **PF18**: displays the LPA last response payload sent to the CICS eWay.
- **PF19**: displays the LPA initial record sent from the IBM Listener (EZACIC02).
- **PF20**: displays the LPA status.
- **PF21**: displays the LPA last error message sent to the CICS eWay.

Press **PF4**. The **Listener Program Area Table** screen appears as displayed in Figure 13.

Figure 13

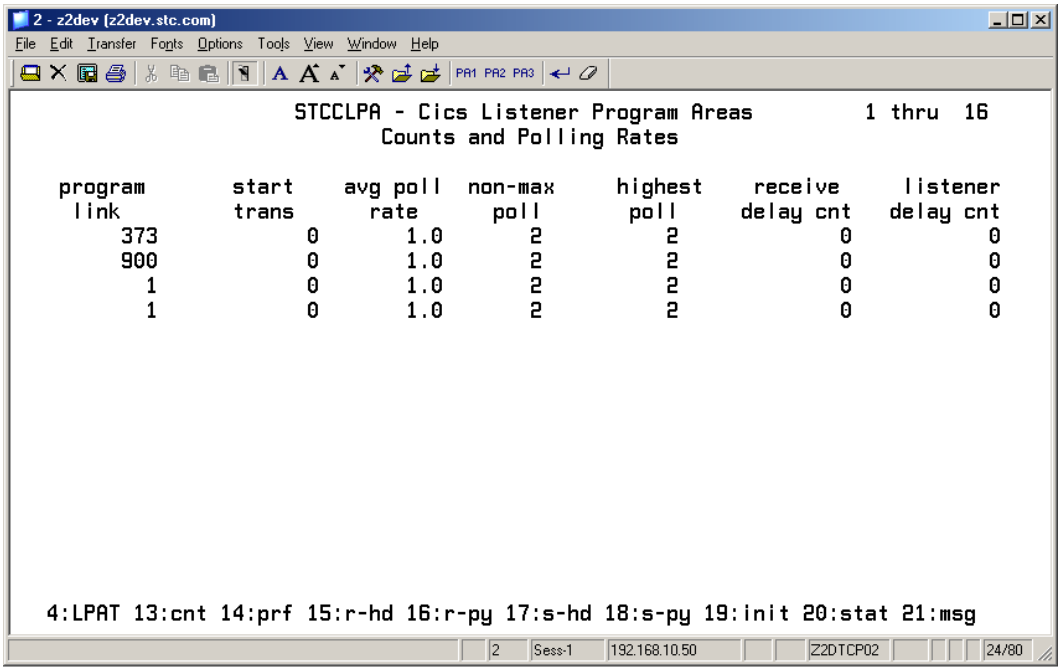


4 This screen displays the Listener Program Area Table (LPAT). Each line contains information about a Listener Program Area (LPA). There is one LPA associated with each instance of the Sun SeeBeyond CICS Listener Program that is running or has run in this CICS region. The fields on the screen are:

- **CLPAT-status-flag**: shows the current status of the LPAT entry as follows:
  - **E**: Exists.
  - **N**: Never used.
  - **C**: Corrupted.
- **CLPAT-lpa-pointer**: shows the address of the most recent LPA in this entry.
- **Listener Program Status**: shows the last status of the Listener Program using this LPA.

Press **PF13.** The **LPA Counts and Polling Rates** screen appears as displayed in Figure 14.

Figure 14



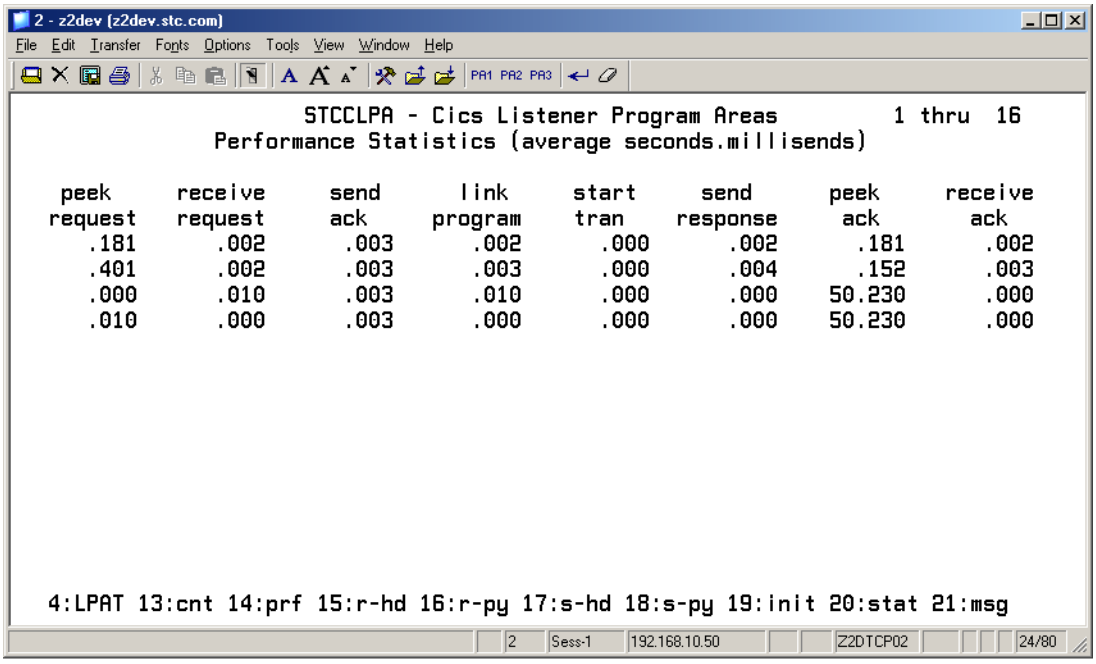
5 This screen displays counts and polling rates information for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond CICS Listener Program).

The fields on the screen are:

- **program link**: the number of program links that have been requested.
- **start trans**: the number of transaction starts that have been requested.
- **avg poll rate**: the average number of receives per polling cycle to satisfy a full message receive from the CICS eWay.
- **non-max cnt**: the 'high water mark' of receives within a polling cycle without reaching the polling rate (polling rate is sent in from the CICS eWay when it initializes connection with the CICS Listener program).
- **highest poll**: the 'high water mark' of receives within a polling cycle that can include reaching the maximum allowed polling rate.
- **receive delay cnt**: the number of one second delays that occurred due to the Comm Timeout limit being reached (Comm. Timeout is sent in from the CICS eWay when it initializes connection with the CICS Listener program).
- **listener delay cnt**: number of one second delays that occur due to the Listener Timeout limit being reached (Listener Timeout is sent in from the CICS eWay when it initializes connection with the CICS Listener program).

Press **PF14**. The **LPA Performance Statistics** screen appears as displayed in Figure 15.

Figure 15



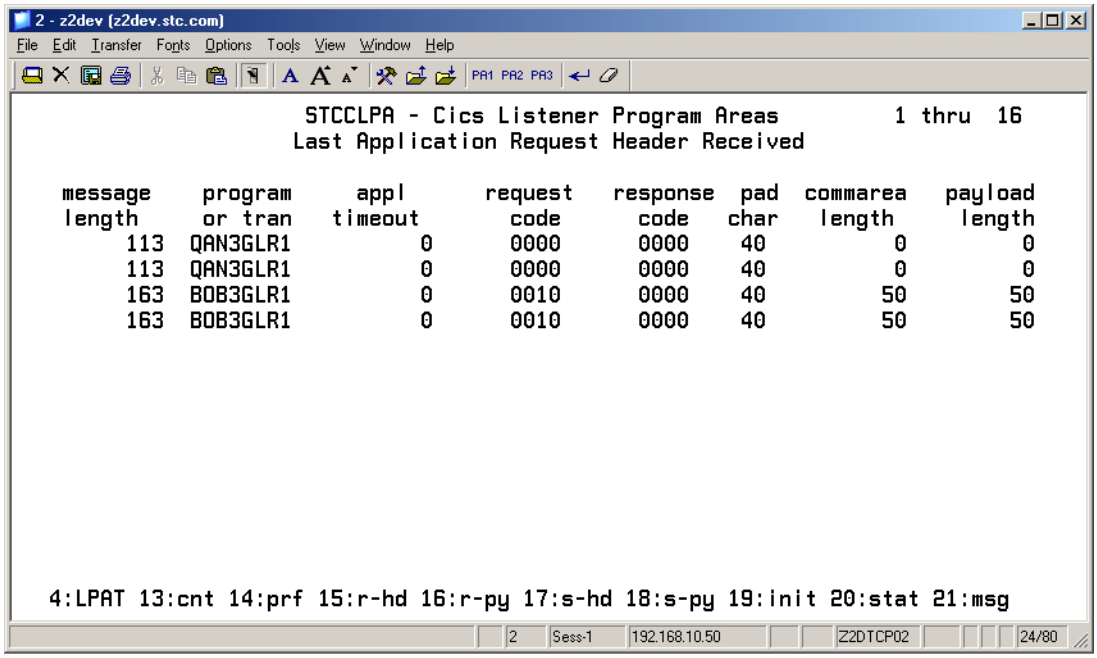
6 This screen (Figure 15) displays the performance statistics for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond Listener Program).

All times displayed are in sss.mmm format (sss = seconds, mmm=milliseconds). The fields on the screen are:

- **peek request**: the average time spent peeking for the next incoming application request message.
- **receive request**: the average time spent receiving the next incoming application request message.
- **send ack**: the average time spent sending an ack for an incoming application request message.
- **link program**: the average time spent link to requested application programs.
- **start tran**: the average time spent starting requested application transactions.
- **send response**: the average time spent sending the application response (including the updated COMMAREA for program links) back to the eWay.
- **peek ack**: the average time spent peeking for the incoming ack to the preceding send response.
- **receive ack**: the average time spent receiving the incoming ack to the preceding send response.

Press **PF15.** The **Last Application Request Header Received** screen appears as displayed in Figure 16.

Figure 16

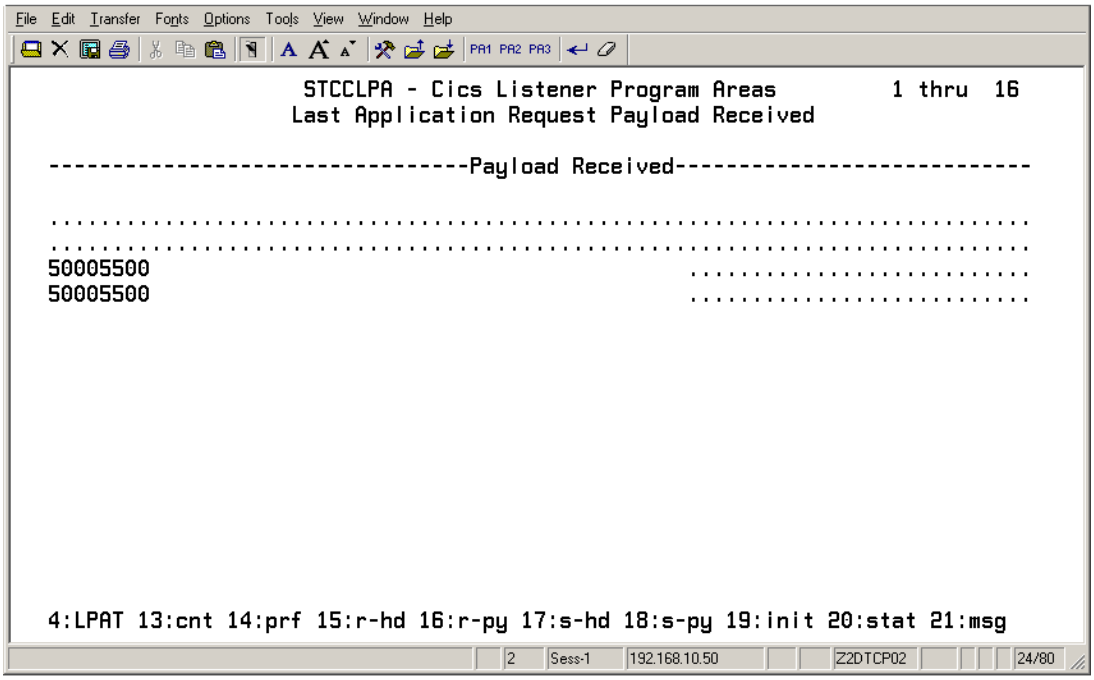


7 This screen (Figure 16) displays the last application request header received from the CICS eWay for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond Listener Program). The fields on the screen are:

- **message length**: the entire length of the incoming message including the header and payload.
- **program or tran**: the requested application program or transaction.
- **appl timeout**: (reserved for future development).
- **request code**: which action is being requested.
  - 0010 = link to application program.
  - 0020 = start application transaction.
  - 0111 = ack for a link response.
  - 0121 = ack for a start response.
  - 9000 = shutdown the Listener program.
- **response code**: always set to zero from the CICS eWay.
- **pad char**: padding character for the COMMAREA if the payload length is less than the COMMAREA length.
- **commarea length**: the length of the data to pass to the application program (link) or the application transaction (start).
- **payload length**: the length of the payload portion of the incoming message.

Press **PF16.** The **Last Application Request Payload Received** screen appears as displayed in Figure 17.

Figure 17

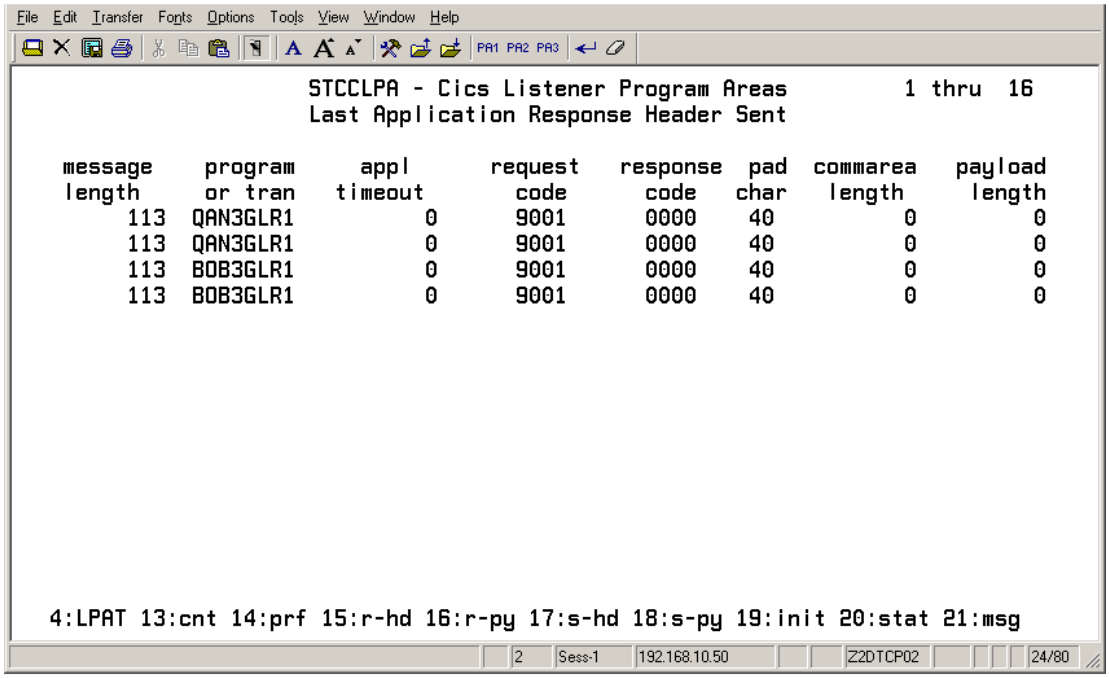


8 This screen (Figure 17) displays the last application request payload received from the CICS eWay for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond Listener Program). There is one field on the screen:

- **Payload Received**: the incoming payload from the CICS eWay.

Press **PF17**. The **Last Application Response Header** screen appears as displayed in **Figure 18 on page 37**.

Figure 18



9 The **Last Application Response Header** screen (Figure 18) displays the last response header sent to the CICS eWay for each LPA. Each line shows one LPA (each instance of the Sun SeeBeyond Listener Program). The fields on the screen are:

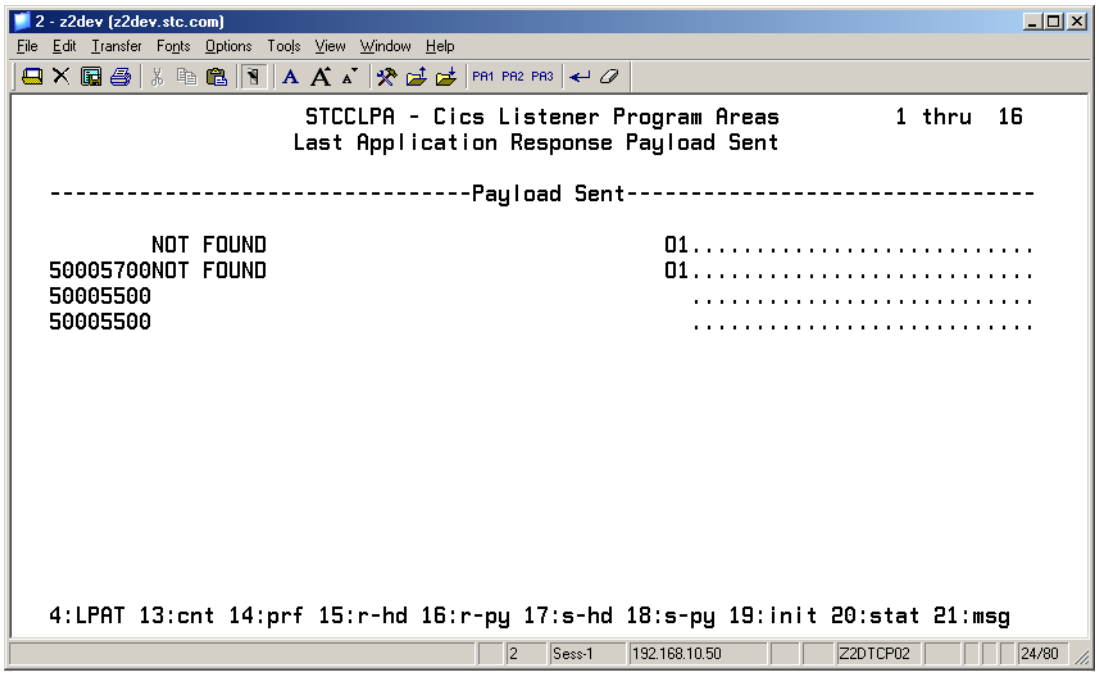
- **message length**: entire length of the outgoing message (header and payload).
- **program or tran**: the application or transaction that was executed or started.
- **appl timeout**: (reserved for future development).
- **request code**: what action is being requested.
  - 0001 = ack the init request.
  - 0011 = ack for a link program request.
  - 0021 = ack for a start trans response.
  - 0110 = response from a linked application program (including COMMAREA).
  - 0120 = response from starting an application transaction (no COMMAREA included).
  - 9001 = ack for a shutdown request.
- **response code**: response code indicating what occurred while processing the previous incoming request from the eWay.
  - 0000 = ok.
  - 0011 = bad startcode during initialization.
  - 0012 = bad password on init record.
  - 0013 = bad comm timeout on init record.

- 0014 = bad appl timeout on init record.
- 0015 = bad Listener timeout on init record.
- 0016 = error reading the init record from transient data queue.
- 0017 = error retrieving the init record from the start data.
- 0018 = error while setting blocking mode.
- 0051 = bad payload length.
- 0052 = error while linking to a program.
- 0053 = error while starting a transaction.
- 0054 = bad request code.
- 0055 = bad response code.
- 0056 = partial message received.
- 0057 = bad program name.
- 0058 = bad message length.
- 0059 = bad commarea length.

- **pad char**: padding char that is sent in from the CICS eWay.
- **commarea length**: COMMAREA length that was sent in from the CICS eWay.
- **payload length**: payload length that was sent in from the CICS eWay (the Listener sets this to zero if this is a response to a start transaction request).

Press **PF18**. The **Last Application Response Payload Sent** screen appears as displayed in Figure 19.

Figure 19

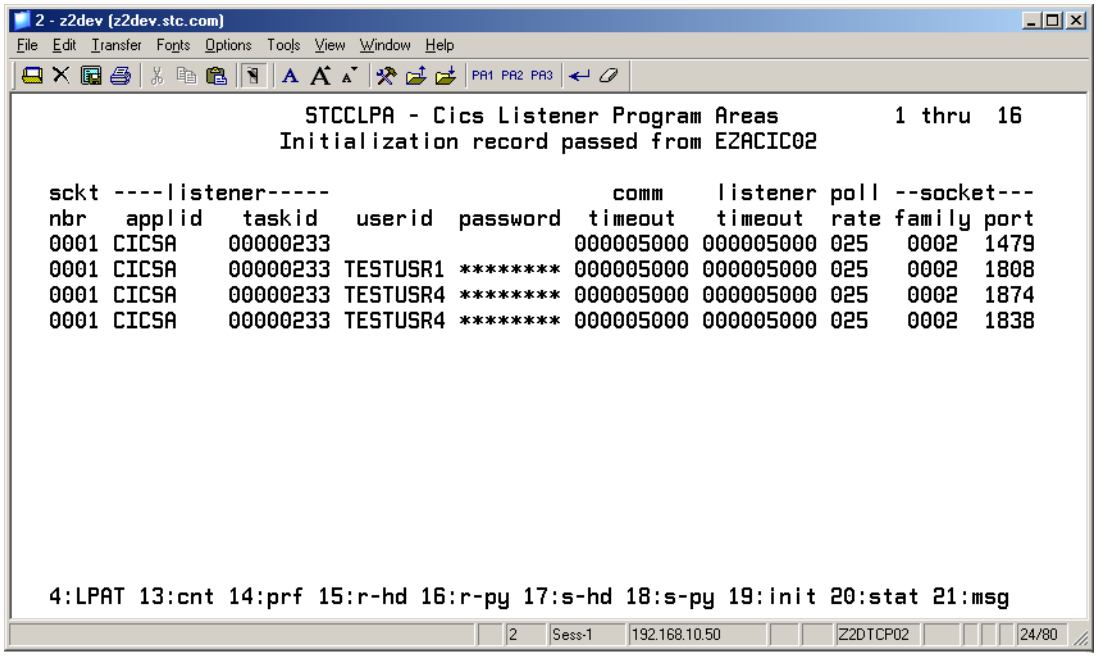


10 This screen (Figure 19) displays the last application response payload sent to the CICS eWay for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond Listener Program). There is one field on the screen:

- Payload sent: the outgoing payload being sent to the CICS eWay.

Press **PF19**. The **Initialization Record Passed from EZACIC02** screen appears as displayed in Figure 20.

Figure 20

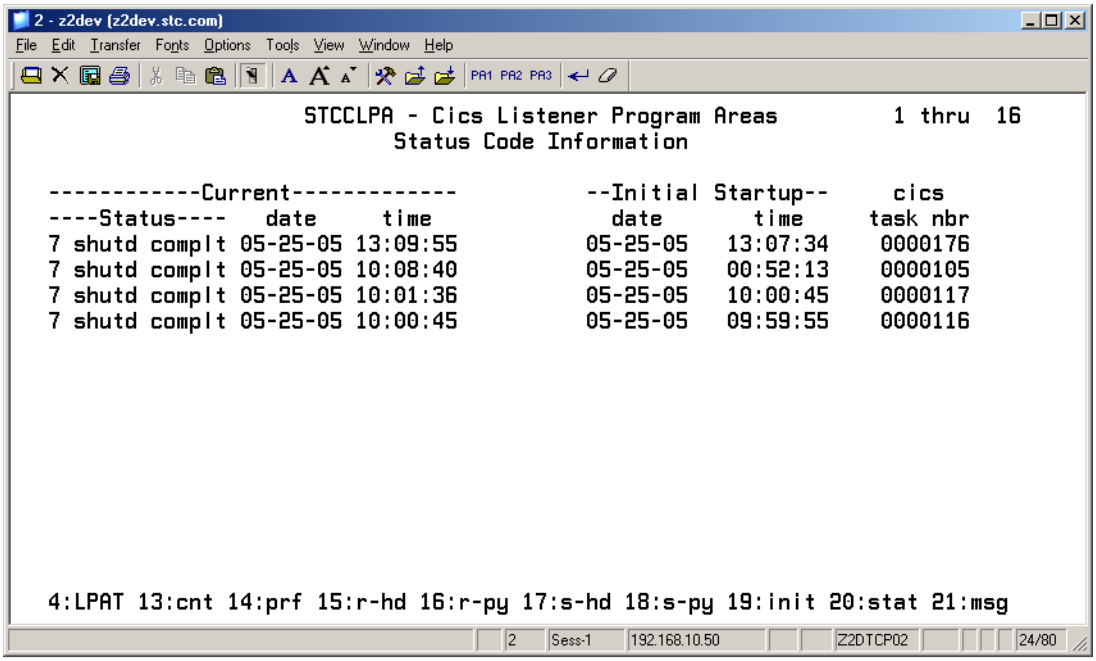


11 This screen (Figure 20) displays the init record that is passed to the Sun SeeBeyond CICS Listener program from the IBM Listener program (ezacic02) for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond CICS Listener program). The fields on the screen are:

- **sckt nbr**: the socket number that is passed to the Sun SeeBeyond CICS Listener.
- **Listener applid**: the applid of the CICS region that the IBM Listener is executing in that started this instance of the Sun SeeBeyond CICS Listener.
- **Listener tasked**: the CICS taskid for the IBM Listener program that started this instance of the Sun SeeBeyond CICS Listener.
- **userid**: userid passed in from the CICS eWay.
- **password**: user password passed in from the CICS eWay.
- **comm. Timeout**: this timeout value is the threshold limit for waiting for all the bytes of an incoming message.
- **Listener timeout**: this timeout value is the threshold limit for waiting for a new incoming message.
- **poll rate**: how many times the Sun SeeBeyond CICS Listener will perform a receive loop to satisfy a complete message receive, after which it will wait one second before trying again.
- **socket family**: the TCP/IP family to which this socket belongs.
- **socket port**: the port this instance of the Sun SeeBeyond CICS Listener is using.

Press **PF20**. The **Status Code Information** screen appears as displayed in Figure 21.

Figure 21



12 This screen (Figure 21) displays the status for each LPA. Each line shows one LPA (one instance of the Sun SeeBeyond CICS Listener program). The fields are:

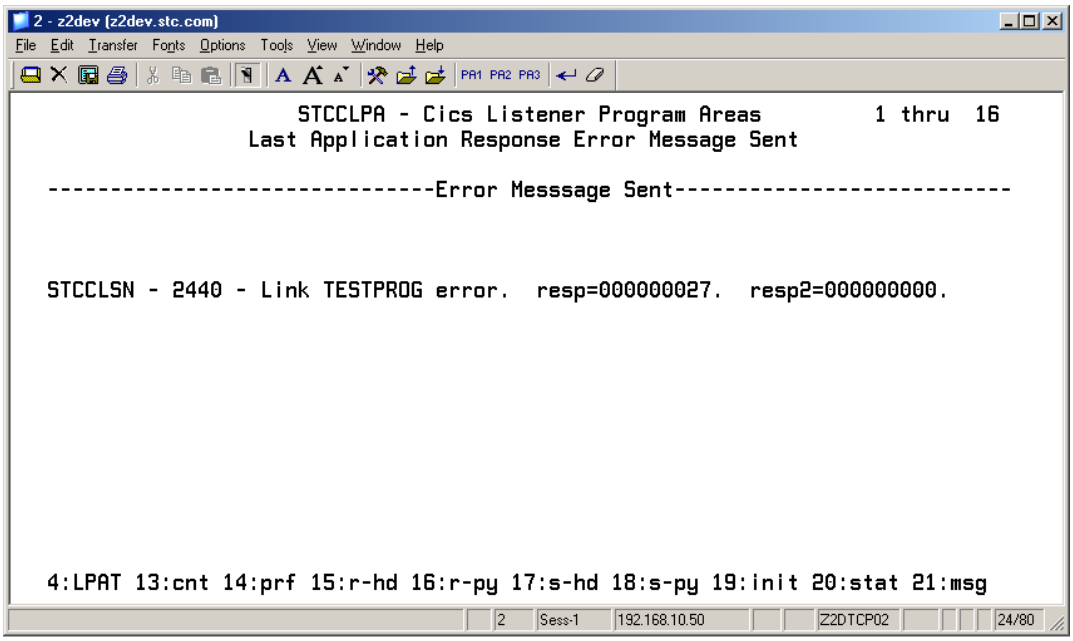
- **Current status**: the current status of an active Sun SeeBeyond CICS Listener program, or the last known status of a previously executing Sun SeeBeyond CICS Listener program.
  - C = LPA is initialized.
  - E = about to get ezacic02 data.
  - G = about to take socket from IBM Listener program.
  - I = about to set mode to blocking.
  - K = about to send ack for init request.
  - M = peeking for length of next incoming request.
  - O = peeking for entire incoming request.
  - Q = receiving full incoming request message.
  - S = sending ack for application request.
  - U = linking to requested application program.
  - W = starting a request application transaction.
  - Y = sending a application response.
  - 0 = peeking for application response ack.
  - 2 = receiving an application response ack.
  - 4 = a shutdown request has been received.
  - 6 = sending an ack for the shutdown request.

- 7 = shutdown is complete, this LPA is now available for reuse.
- 8 = Sun SeeBeyond CICS Listener program appended.

- **Current Status date**: date for this status.
- **Current Status time**: time for this status.
- **Initial Startup date**: date this instance of the Sun SeeBeyond CICS Listener was started.
- **Initial Startup time**: time this instance of the Sun SeeBeyond CICS Listener was started.
- **cics task nbr**: the CICS *eibtaskn* for this instance of the Sun SeeBeyond CICS Listener.

Press **PF21**. The **Error Message Sent** screen appears as displayed in Figure 22.

Figure 22



13 This screen (Figure 22) shows the last error message sent to the CICS eWay for each LPA. Each line shows one LPA (one for each instance of the Sun SeeBeyond CICS Listener program). There is one field on the screen:

- **Error Message Sent**: the last error message sent to the CICS eWay for this instance of the Sun SeeBeyond CICS Listener program.

2.5 CICS Listener Considerations for Invoking DB2 Applications

One of the following two techniques are required when using the CICS eWay to run CICS applications that invoke a DB2 table:

- RCT entry for the STCL Sun SeeBeyond CICS Listener Transaction
- PPT entry to redirect DB2 application to another AOR

RCT entry for the STCL Sun SeeBeyond CICS Listener Transaction

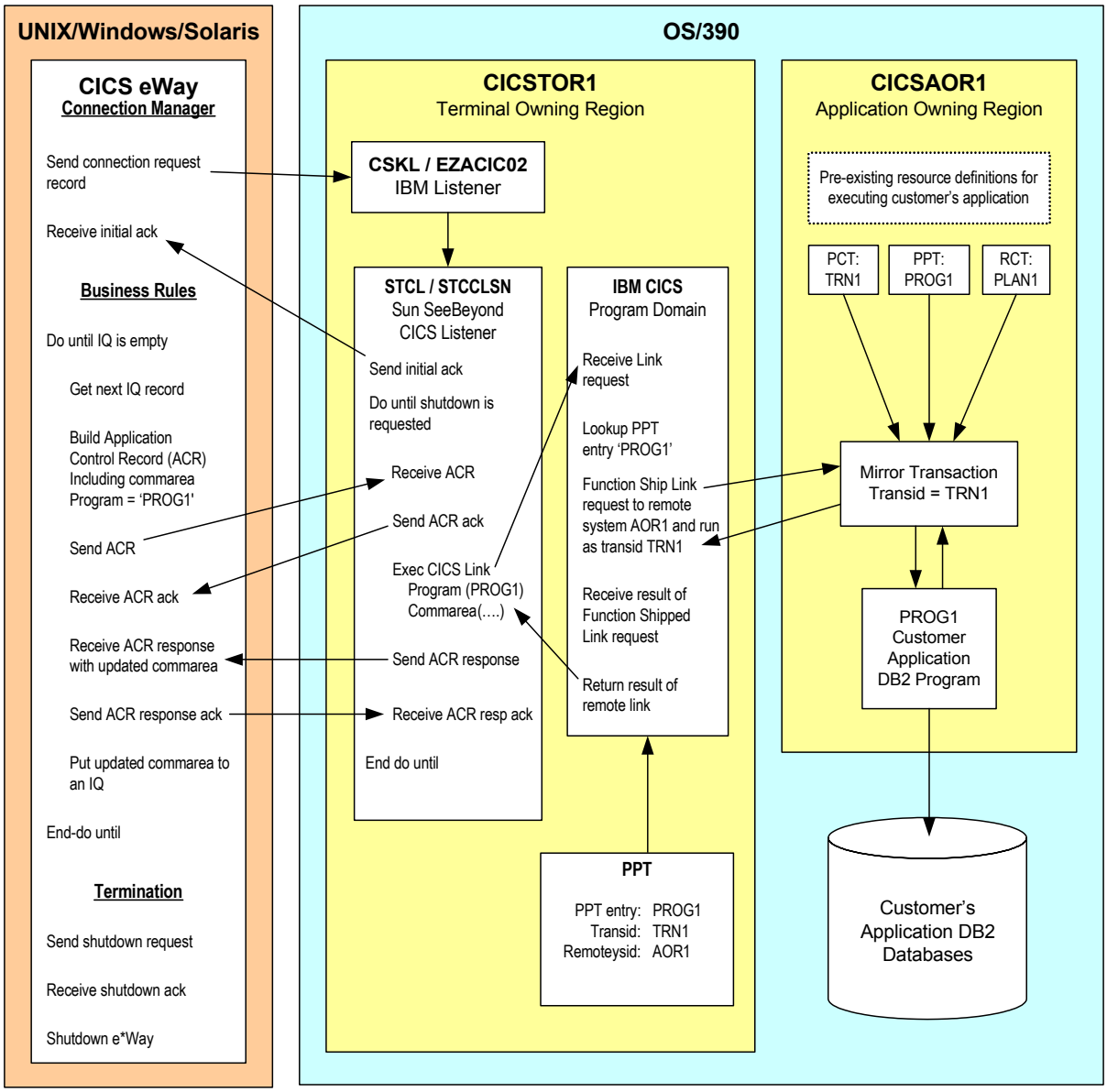
To run DB2 application programs under the STCL Sun SeeBeyond CICS Listener transid, an RCT entry for the STCL transaction must be created using an RCT macro definition similar to the one shown below or by using an equivalent RDO entry.

```
DSNCRCT TYPE=ENTRY,TXID=STCL,THRDM=0,THRDA=0,THRDS=0,
TWAIT=POOL,PLAN=HD45LE06,AUTH=(SIGNID,*,*)
```

PPT entry to redirect DB2 application to another AOR

To redirect DB2 application programs to another AOR, the PPT entry for the requested DB2 application program in the TOR (where the STCL Sun SeeBeyond CICS Listener transaction is running) must be defined to run the DB2 application program in a different region (usually an AOR) and under a specific PCT in that AOR. Refer to **Figure 23 on page 44** for a detailed flow diagram showing the use of the PPT and PCT.

Figure 23 Sun SeeBeyond CICS Listener invoking DB2 programs



2.6 CICS Transaction Gateway Installation

For installation and configuration information about IBM's CICS Transaction Gateway 5.1, 6.0, 6.0.1, or 6.1, consult the latest CTG documentation at IBM's CICS Library web site:

http://www-306.ibm.com/software/htp/cics/ctg/library

This documentation should also be consulted for information on configuring CTG for use with RACF and the latest APARs.

IBM CICS Transaction Gateway properties are set using the CTG Configuration Tool. The Configuration Tool is located under the CICS Transaction Gateway program menu.

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 5.1.2. 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-protected each component. These protected components appear in the Explorer tree with a red padlock in the bottom-left corner of each icon. Before you can deploy the imported Project, the Project's components must first be *"checked out"* of Version Control from both the Project Explorer and the Environment Explorer. To *"check out"* all of the Project's components, do the following:

- A From the Project Explorer, right-click the Project and select **Version Control > Check Out** from the shortcut menu. The Version Control - Check Out dialog box appears.
- B Select **Recurse Project** to specify all components, and click **OK**.
- C Select the Environment Explorer tab, and from the Environment Explorer, right-click the Project's Environment and select **Version Control > Check Out** from the shortcut menu.
- D Select **Recurse Environment** to specify all components, and click **OK**.

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 Sun Java™ Composite Application Platform Suite.*

Chapter 3

Configuring the CICS eWay

This chapter describes how to create and configure the CICS eWay properties.

What's in This Chapter

3.1 Creating and Configuring the CICS eWay

All eWays contain a set of parameters with properties that are unique to that eWay type. The CICS eWay properties 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 CICS External System window.
- **Collaboration or Business Process**: CICS eWay properties may also be set from your Collaboration or Business process, 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.

3.1.1 Selecting CICS as the External Application

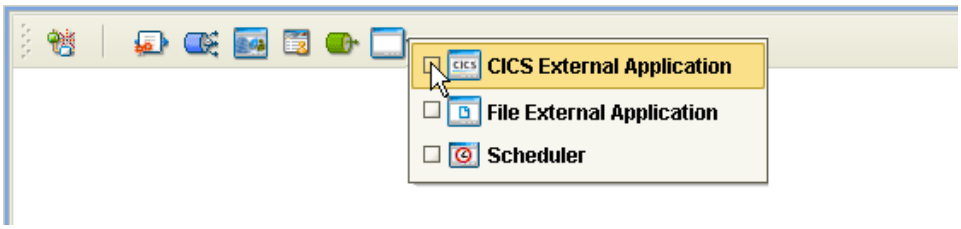
To create a CICS eWay you must first create a CICS External Application in your Connectivity Map. CICS eWays are located between a CICS External Application and a Service. Services are containers for Collaborations, Business Processes, eTL processes, and so forth.

To create the CICS External Application

1 From the Connectivity Map toolbar, click the **External Applications** icon.

2 Select the **CICS External Application** from the menu (see Figure 24). The selected CICS External Application icon now appears on the Connectivity Map toolbar.

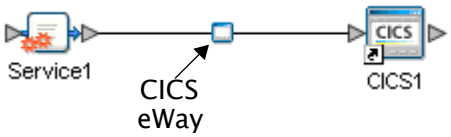
Figure 24 External Applications Selection Menu



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

From the Connectivity Map, you can associate (bind) the External Application to the Service to establish an eWay (see Figure 25).

Figure 25 eWay Location



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

3.1.2 Configuring the CICS eWay Properties

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

Configuring the CICS 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** appears with a template containing the CICS eWay Connectivity Map properties. Make any necessary changes to the property values and click **OK** to save the settings.

Configuring the CICS eWay (Environment Explorer) Properties

1 From the **Environment Explorer** tree, right-click the CICS External System. Select **Properties** from the shortcut menu. The **Properties Editor** opens with the CICS eWay Environment properties.

2 Make any necessary changes to the Environment property values, and click **OK** to save the settings.

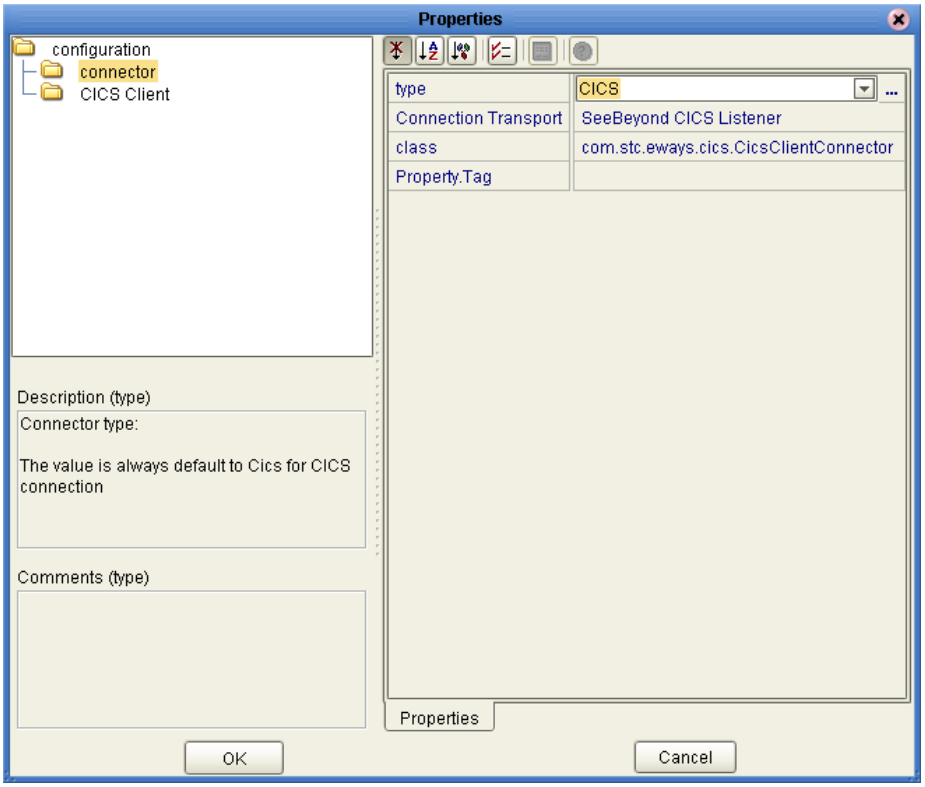
3.1.3 Using the Properties Editor

Modifications to the eWay configuration properties are made from the CICS eWay **Properties Editor**.

Modifying the Default eWay Configuration Properties

1. From the Connectivity Map or the Environment Explorer, open the Properties Editor to the CICS eWay default properties.
2. From the upper-right pane of the Properties Editor, select a subdirectory of the configuration directory. The parameters contained in that subdirectory are now displayed in the Properties pane of the Properties Editor. For example, if you click on the **connector** subdirectory, the editable **connector** parameters are displayed in the right pane (see Figure 26).

Figure 26 Properties Editor -- CICS Properties



3. Click on any property field to make it editable. For example, click on the **class** property to edit the class value. If a property value is true/false or multiple choice, the field displays a submenu of property options.
4. Click on the ellipsis (. . .) in the properties field to open a separate configuration dialog box. This is helpful for large values that cannot be fully displayed in the parameter's property field. Enter the property value in the dialog box and click **OK**. The value is now displayed in the property field.

5 A description of each property is displayed in the **Description** pane when that property is selected. This provides a brief explanation of the required settings or options.

6 The **Comments** pane provides an area to record notes and information regarding the currently selected property. These comments are saved when you close the editor.

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

3.2 CICS eWay Connectivity Map Properties

The CICS eWay configuration parameters, accessed from the Connectivity Map, are organized into the following sections:

- **Connector** on page 50
- **CICS Client** on page 52
- **Connection Mode** on page 55

3.2.1 Connector

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

Table 4 Connectivity Map - Connector Properties

Name	Description	Required Value
Type	Specifies the connector type.	**Enter CICS**. The value always defaults to **CICS** for CICS connections.
Connection Transport	Specifies the underlying connection transport used by the CICS eWay to send requests to and get responses from a CICS region.	**Select one of the following underlying connection transports:** ♦ **Sun SeeBeyond CICS Listener** ♦ **CICS Transaction Gateway (specifies the IBM CICS Transaction Gateway)** **Sun SeeBeyond CICS Listener is the configured default.**

Table 4 Connectivity Map - Connector Properties (Continued)

Name	Description	Required Value
Class	Specifies the class name of the CICS Client connector object.	The (class) package name for the CICS Client connector object. The default is **com.stc.eways.cics.CicsClient Connector**.
Property.Tag	Specifies the data source identity. This parameter is required by the current EBobConnectorFactory.	The data source package name.

3.2.2 CICS Client

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

Table 5 Connectivity Map - CICS Client Properties

Name	Description	Required Value
ECI call type	Specifies whether the ECI call type is Synchronous. Synchronous calls wait for the transaction to complete, then return the contents of the COMMAREA. Only Synchronous calls are supported.	**Synchronous** is the configured default.
CICS Program	Specifies the CICS program to be run on the server. Maximum length is eight characters.	A CICS program name, eight characters or less.
CICS TransId	CTG specific. Specifies the ID of a CICS transaction. Maximum length is four characters. Attributes are dependent upon the value set for **"Use TransId as ECI_TPN" on page 53**, as follows: ♦ If **EciTPN** is set to **FALSE**, the value of the transid is stored in EIBTRNID for the duration of the LINK to the program specified in the Program parameter. The called program runs under the mirror transaction **CPMI**, but is linked to under the Transid transaction name. This name is available to the called program for querying the transaction ID. Some servers use the transaction ID to determine security and performance attributes for the called program. ♦ If **EciTPN** is set to **TRUE**, the Transid will be interpreted as the **ECI_TPN** transid, a transaction that will be used in the server to process the ECI request. This transaction must be defined in the server as a CICS mirror transaction. If the ECI request is extended, this parameter has a meaning only for the first request.	A CICS transaction ID of four characters or less.

Table 5 Connectivity Map - CICS Client Properties (Continued)

Use TransId as ECI_TPN	**CTG** specific. Specifies whether the TransId is interpreted as ECI_TPN or if the called program runs under the default mirror transaction CPMI. Options are: • **TRUE**: indicates that the TransId is interpreted as ECI_TPN. • **FALSE**: indicates that the called program runs under the default mirror transaction CPMI, and is linked to under the TransId (if present).	Select **TRUE** or **FALSE**. **FALSE** is the configured default.
COMMAREA Length	Specifies the length (in bytes) of the COMMAREA passed to the ECI.	A number indicating the byte length of the COMMAREA. ***Note:*** When using the CICS Transaction Gateway transport, data sent to CICS must be padded with spaces, if necessary, to match the full size of the commarea.
ECI extend mode	Specifies whether a logical unit of work is terminated at the end of a call.	**Yes** or **No**. **Yes** indicates that the work unit is terminated at the end of a call. **T**he configured default is **No**.

Table 5 Connectivity Map - CICS Client Properties (Continued)

ECI LUW Token	CTG specific. Specifies an integer used to identify the logical unit of work (LUW) to which a call belongs. This must be set to **0** (zero) at the start of an LUW, even if the LUW is to be extended. The ECI updates the value upon the first (or only) call of the LUW. If the LUW is to be extended, this value is used as input to any subsequent calls associated with the same LUW. If the return code is not ECI_NO_ERROR, and a call is ending or continuing an existing LUW, then this field is used to report the state of the LUW as follows: ▪ A code of 0 (zero) indicates that the LUW has ended and its updates have been backed out. ▪ Any non-zero code indicates the current input value. It also indicates that the LUW is continuing, and updates are still pending. See the **Logical units of work in ECI** table in the **CICS Transaction Gateway: Programming Guide** for more information.	An integer used to identify the ECI logical unit of work.
Encoding	Specifies the canonical name for the encoding set.	The canonical name for any encoding set supported by Sun's Java Runtime Environment 1.1.8 (contained in rt.jar and i18n.jar). Examples are **ASCII** and **Cp500** (EBCDIC). When running the CICS eWay on a **z/OS** platform, set the Encoding value to **Cp500**.

3.2.3 Connection Mode

The **Connection Mode** section of the CICS Connectivity Map properties contains the top-level parameters displayed in Table 6.

Table 6 Connectivity Map Properties - Connection Mode Section

Name	Description	Required Value
CICS Connection Mode	**Specifies whether a physical connection is established when an external connection is instantiated. The options are:** • **Automatic: Establishes a physical connection when an external connection is instantiated.** • **Manual: Does not automatically establish a physical connection when an external connection is instantiated.** If a physical connection is not automatically established, a physical connection must be established from the Collaboration (for example, by calling the **connect()** method).	**Automatic** or **Manual**. The configured default is **Automatic**.

3.3 CICS eWay Environment Configuration Properties

The CICS eWay configuration parameters, accessed from the Environment Explorer tree, are organized into the following sections:

3.3.1 Sun SeeBeyond CICS Listener

The **Sun SeeBeyond CICS Listener** section of the CICS Environment properties contains the top-level parameters displayed in Table 7.

Table 7 Environment Properties - Sun SeeBeyond CICS Listener Section

Name	Description	Required Value
Host	Specifies the name of the mainframe host you are connecting to.	**Enter CICS**. The value always defaults to **CICS** for CICS connections.
Port	Specifies the TCP/IP port where the Sun SeeBeyond CICS Listener (and the CICS Listener) is listening. This is the port to which the CICS eWay will connect.	**The TCP/IP port to which Sun SeeBeyond CICS Listener is listening.**
Sun SeeBeyond CICS Listener TransId	Specifies the Sun SeeBeyond CICS Listener TransId on the mainframe host. This is the CICS Transaction that the Sun SeeBeyond CICS Listener is installed under.	**The valid TransId of the Sun SeeBeyond CICS Listener. The default is STCL.**
Start Type	Specifies the startup type. This can be either **IC** for CICS interval control, or **TD** for CICS transient data. This is the CICS Startup type for the program being executed.	**Select IC or TD. The default value is IC.**
Start Delay	Specifies the hours, minutes and seconds (interval of time) to delay starting the transaction program (TP) on the CICS server for the **IC** Start Type. This field is optional, but must specify all 6 digits if used.	A 6 digit integer. All 6 digits must be given if this is specified (for example, 123456). The default value is **000000**.
Listener Timeout	Specifies the amount of time (in milliseconds) for the Sun SeeBeyond CICS Listener to wait for the next incoming transaction program request from the CICS eWay.	A number indicating the Listener timeout in milliseconds (for example, 120000 milliseconds equals 2 minutes).

Table 7 Environment Properties - Sun SeeBeyond CICS Listener Section (Continued)

Name	Description	Required Value
TP Timeout	Specifies the amount of time the CICS eWay will wait for the Sun SeeBeyond CICS Listener to return results for a current transaction program request.	A number indicating the TP Timeout in milliseconds (for example, 120000 milliseconds equals 2 minutes).
Polling Rate	Specifies the polling rate. This is the number of times the Sun SeeBeyond CICS Listener queries the current TCP connection for incoming traffic before issuing an EXEC CICS DELAY for one second.	An integer indicating the Sun SeeBeyond CICS Listener polling rate.
Transport Timeout	Specifies the timeout used by both the local and host side for send or receive.	A number indicating the Transport Timeout in milliseconds (for example, 5000 milliseconds equals 5 seconds).
COMMAREA Padding Character	Specifies the EBCDIC code for the character used by the SBYND listener to pad the COMMAREA at the CICS server when the actual length of the payload in the COMMAREA is shorter than the length given by CommAreaLength. The default value is hexadecimal 40 - EBCDIC space.	A character value coded in Hexadecimal. For example: **40** for Blanks, **00** for Low Values, **FF** for High Values, and so forth. The default value is **40**.
SendBufSize	Specifies the Send Buffer Size (in bytes) for the underlying socket.	A number indicating the Buffer Size in bytes (for example, 2048 bytes equals 2 KB).
ReceiveBufSize	Specifies the Receive Buffer Size (in bytes) for the underlying socket (provided as a hint).	A number indicating the Receive Buffer Size in bytes (for example, 10240 bytes equals 10 kilobytes).
NoDelay	Specifies whether the system can delay connections or requests. Generally, **True** (no delay) is required for high-volume or critical transactions. In cases of low-volume and noncritical transactions, you can use **False**. (Specifies whether to disable Nagle's Algorithm)	Enter **TRUE** or **FALSE**. **TRUE** is the default.

Table 7 Environment Properties - Sun SeeBeyond CICS Listener Section (Continued)

Name	Description	Required Value
KeepAlive	Specifies whether to enable the socket's **SO_KEEPALIVE** option when it creates a socket connection to the CICS listener. SO_KEEPALIVE maintains active connections by enabling periodic transmission of messages (if this is supported by the protocol). If the connected socket fails to respond to these messages, the connection is broken and the processes writing to that socket are notified with an ENETRESET errno. This option takes an **int** value in the optval argument. This is a BOOL option. The socket's SO_KEEPALIVE option is used to enable pinging of the connection to the peer during connection to keep the connection "alive". This is used to prevent connections from going idle and timing out. SO_KEEPALIVE periodically sends a message to the connection socket of the peer to ensure that the connection is still "alive" (active). One of three responses is expected: 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 The peer fails to return a response. The socket is closed. The purpose of this option is to detect whether the peer host has crashed.	Enter **TRUE** to enable SO_KEEPALIVE, or **FALSE** to disable the option. **TRUE** is the configured default.

3.3.2 CICS Gateway

The **CICS Gateway** section of the CICS Environment properties contains the top-level parameters displayed in Table 8.

Table 8 Environment Properties - CICS Gateway Section

Name	Description	Required Value
URL	Specifies the remote or local Gateway to which you are connecting. ***Important:*** This parameter requires specific JAR files when using "**local:**" as the value. For directions on installing the required files, see **"Adding JAR Files to the LogicalHost" on page 65**. ***Note:*** The default value "**local:**", does not work with CTG running on **z/OS**. For CTG running on a z/OS system, the URL property value must be set to **localhost** or to the server name.	The remote or local Gateway node name or IP address. The configured default is **local:**.
Port	Specifies the TCP/IP port where CTG is running.	An number indicating the TCP/IP port.
Server	Specifies a server to use from the servers listed in the CTG configuration.	The name of a server as specified in the CTG server list. If this value is left blank, the first server specified in the list is used by default.
SSL KeyRing Class	Specifies the classname of the SSL KeyRing class.	The full classname of the SSL KeyRing class.
SSL KeyRing Password	Specifies the password for the encrypted KeyRing class.	The password for the SSL KeyRing class.

3.3.3 CICS Client

The **CICS Gateway** section of the CICS Environment properties contains the top-level parameters displayed in Table 9.

Table 9 Environment Properties - CICS Client Section

Name	Description	Required Value
CICS UserId	Specifies the ID of the CICS user. Maximum length is eight characters.	A CICS user ID that uses eight characters or less.
CICS Password	Specifies the password for the CICS user. Maximum length is eight characters.	A password of eight characters or less.

3.3.4 Tracing

The **Tracing** section of the CICS Environment properties contains the top-level parameters displayed in Table 10.

Table 10 Environment Properties - Tracing Section

Name	Description	Required Value
Level	**CTG** specific. Specifies the level of trace information recorded available. The options are: • **0**: **None**. No CICS Java client application tracing. • **1**: **Standard**. Only the first 128 bytes of any data block (for example the COMMAREA, or network flows) are displayed by default. This trace level is equivalent to the Gateway trace set by the ctgstart -trace option. (This can also be set using the system property **gateway.T.trace=on**). • **2**: **Full Debug**. Traces out the whole of any data blocks by default. The trace contains more information about CICS Transaction Gateway than the standard trace level. This trace level is equivalent to the Gateway debug trace set by the ctgstart -x option. (This can also set using the system property **gateway.T=on**). • **3**: **Exception Stacks**. Traces most Java exceptions, including exception which are expected during normal operation of the CICS Transaction Gateway. No other tracing is written. This trace level is equivalent to the Gateway stack trace set by the ctgstart -stack option. (This can also set using the system property **gateway.T.stack=on**).	An integer from **0** to **3** that indicates the specified trace information level. The configured default is **0**.
Filename	**CTG** specific. Specifies a file location for writing the trace output. This is an alternative to the default output on **stderr**. Long filenames must be surrounded by quotation marks; for example: "trace output file.log". ***Note:*** The filename can also be set using the system property **gateway.T.setTFile=xxx**, where **xxx** is a filename.	The output file name.

Table 10 Environment Properties - Tracing Section (Continued)

Name	Description	Required Value
Truncation Size	**CTG** specific. Specifies the maximum size of any data blocks written in the trace. ***Note:*** The truncation size can also be set using the system property **gateway.T.setTruncationSize=xxx**, where **xxx** is a number.	A number indicating the maximum data block size. A value of **0** indicates that no data blocks will be written in the trace. No value (leaving the property blank) indicates that no truncation size is specified.
Dump Offset	**CTG** specific. Specifies the offset from which the display of any data blocks will start. ***Note:*** The dump offset can also be set using the system property **gateway.T.setDumpOffset=xxx**, where **xxx** is a number.	CTG specific. Specifies the offset from which the display of any data blocks will start.
Timing	Specifies whether or not to display time-stamps in the trace.	**On** or **Off**. **On** indicates that the time-stamp is displayed in the trace. The default setting is **On**. ***Note:*** Timing can also set using the system property **gateway.T.timing=on**.

3.3.5 Connection Retry Settings

The **Connection Retry Settings** section of the CICS Environment properties contains the top-level parameters displayed in Table 11.

Table 11 Environment Properties - Connection Retry Settings Section

Name	Description	Required Value
Maximum Retries	Specifies the maximum number of connection retries.	A number indicating the number of times the eWay will try to establish a connection. The configured default is **5**.
Retry Interval [ms]	Specifies the number of milliseconds to wait between connection retries.	A number indicating the time (in milliseconds) that the eWay waits between connection attempts. The configured default is **5000** (or 5 seconds).

3.3.6 Connection Pool Settings

The **Connection Pool Settings** section of the CICS Environment properties contains the top-level parameters displayed in Table 12.

Table 12 Environment Properties - Connection Pool Settings Section

Name	Description	Required Value
Steady Pool Size	Specifies the initial and minimum number of connections to be maintained.	A number indicating the initial and minimum number of connections to be maintained. The configured default is **2**.
Max Pool Size	Specifies the maximum size of the connection to EIS.	A number indicating the maximum size of the connection to EIS. The configured default is **10**.

3.4 Adding JAR Files to the LogicalHost

When using **local:** as the URL value in the Environment properties, both the **ctgclient.jar** and the **ctgserver.jar** files must be installed to the following location prior to running a CICS eWay Project:

```
<JavaCAPS51>\logicalhost\is\lib
```

where <*JavaCAPS51*> is the directory where Sun Java Composite Application Platform Suite is installed.

Note: *If the Logical Host is running, it must be recycled to pickup the new JAR file.*

Chapter 4

Using the CICS eWay with eInsight

This chapter describes how to use the CICS eWay with the Sun SeeBeyond eInsight Business Process Manager and the Web Services interface.

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

What's in This Chapter

4.1 The eInsight Engine and Components

eGate components can be deployed as Activities in eInsight Business Processes. Once a component is associated with an Activity, eInsight invokes it using a Web Services interface. eGate components that can interface with eInsight in this way include the following:

- Object Type Definitions (OTDs)
- eWays
- Collaborations

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

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

4.2 The CICS eWay With eInsight

An eInsight Business Process Activity can be associated with the CICS eWay during the system design phase. To make this association, select the desired operators under the eWay in the Enterprise Explorer and drag it onto the eInsight Business Process Designer canvas.

The CICS eWay has the following operators available:

- getServerList
- execute

The operation is automatically changed to an Activity with an icon identifying the component that is the basis for the Activity. At run time, eInsight invokes each step in the order defined in the Business Process. Using eInsight's Web Services interface, the Activity in turn invokes the CICS eWay.

Dynamic Configuration

The CICS eWay web service message structure exposes properties such as **server**, **transId**, **password**, **userId**, and so forth, that correspond to eWay configuration parameters. Populating these properties dynamically at runtime from the Business Rules Designer, allows the user to override the corresponding configuration parameters.

4.3 The CICS eWay eInsight Sample Project

This following pages provide directions for creating a simple Project that demonstrates how eInsight Business Processes are used with the CICS eWay. The same Project can be downloaded from the Installation CD-ROM in a near-complete state.

For more information on creating ICAN Projects see the *eInsight™ Business Process Manager User's Guide* and the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Expanded BPEL Properties

The 5.0.3 version of the CICS eWay displays additional CICS BPEL properties in the message structure, visible from the Business Rules Designer. The packaged sample that is included on the installation CD-ROM is designed to work with eGate, versions 5.0.1 and above, and will not display these additional properties when loaded. To see these additional parameters, create the sample manually, as directed in this chapter.

4.4 Importing a Sample Project

Sample eWay Projects are included as part of the installation package. To import a sample eWay Project to the Enterprise Designer do the following:

1. The sample files are uploaded with the eWay's documentation .sar file and downloaded from the Sun Java Composite Application Platform Suite Installer's Documentation tab. The **CICS_eWay_Sample.zip** file contains the various sample Project zip files. Extract the samples to a local file.
2. Save all unsaved work before importing a Project.
3. From the Enterprise Designer's Project Explorer pane, right-click the Repository and select **Import** from the shortcut menu. The **Import Manager** appears.
4. Browse to the directory that contains the sample Project zip file. Select the sample file (for this sample, **CICS_BP_Sample.zip**) and click **Import**. After the sample Project is successfully imported, click **Close**.
5. Before an imported sample Project can be run you must do the following:
   - Create an **Environment** (see **"Creating an Environment" on page 76**)
   - Configure the eWays for your specific system (see **"Configuring the eWays" on page 77**)
   - Create a **Deployment Profile** (see **"Creating the Deployment Profile" on page 79**)
   - Create and start a domain (see **"Creating and Starting the Domain" on page 80**)
   - Build and deploy the Project (see **"Building and Deploying the Project" on page 80**)

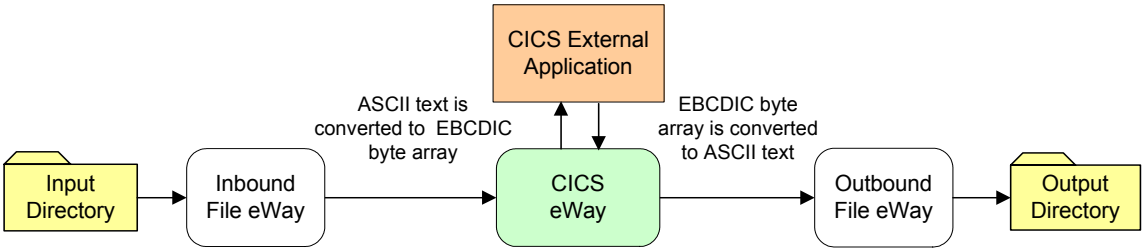
   The following pages provide step by step directions for manually creating the prjCICS_BP_Sample Project.

4.5 The CICS_BP_Sample Project Overview

The CICS eWay Project, **prjCICS_BP_Sample**, demonstrates the following:

- The Inbound File eWay subscribes to an external directory and receives input data (ASCII text).
- The ASCII text data is converted to EBCDIC (byte array) using cp037 encoding, and published by the CICS eWay, to the CICS external system.
- Input EBCDIC data from the CICS external system, is received by the CICS eWay and converted back to String ASCII data using cp037 encoding. This data is published to the outbound File eWay.
- The outbound File eWay publishes the data to an external directory (see **Figure 27 on page 69**).

Figure 27 prjCICS_BP_Sample Project



4.6 Creating the prjCICS_BP_Sample Project

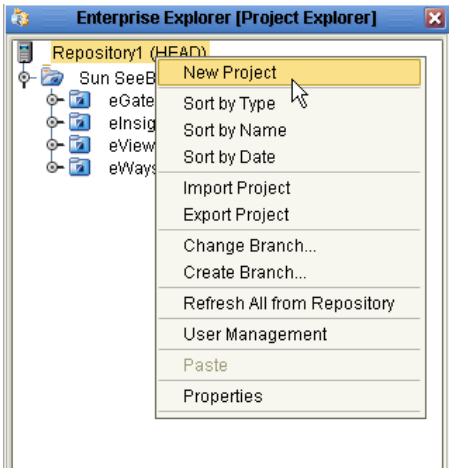
This section provides step by step directions for creating the prjCICS_BP_Sample Project.

4.6.1 Creating a Project

The first step is to create a new Project in the Enterprise Designer.

1. Start the Enterprise Designer.
2. From the Project Explorer tree, right-click the Repository and select **New Project** (see Figure 28). A new Project (**Project1**) appears on the Project Explorer tree.

Figure 28 Enterprise Explorer - New Project



3. Click twice on **Project1** and rename the Project (for this sample, **prjCICS_BP_Sample**).

4.6.2 Creating a Business Process

To create the CICS eInsight Business Process, do the following:

Creating the Business Process Flow

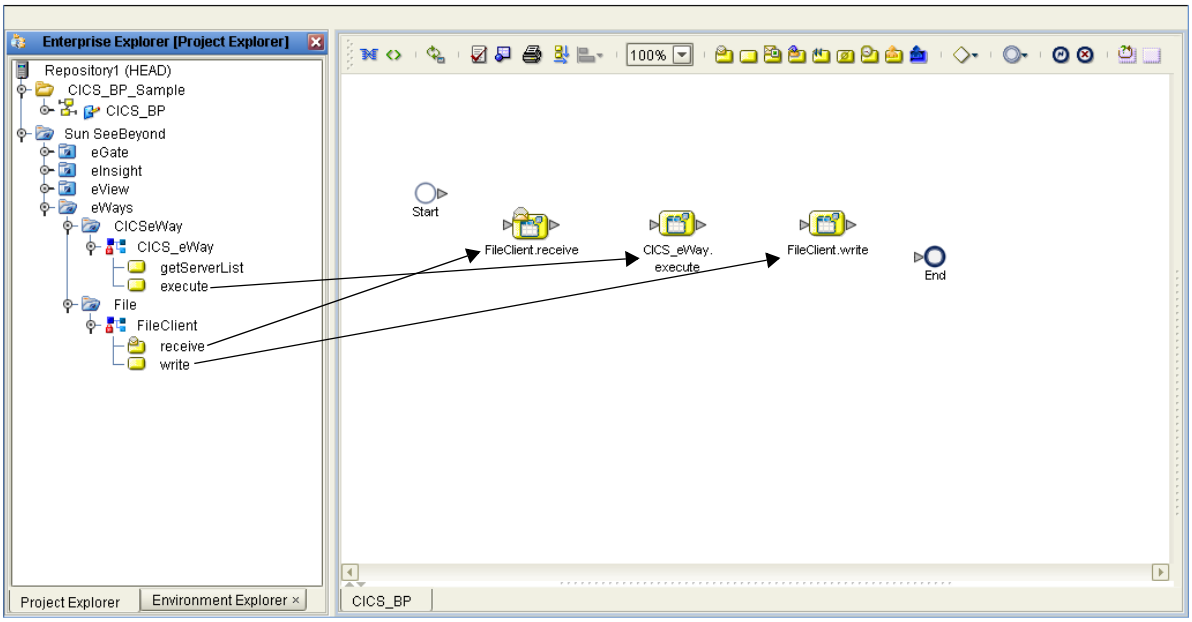
1 Right-click your new Project in the Enterprise Designer's Project Explorer, and select **New** > **Business Process** from the shortcut menu. The eInsight Business Process Designer appears and **BusinessProcess1** is added to the Project Explorer tree. Rename **BusinessProcess1** to **CICS_BP**.

2 From the Project Explorer tree, expand the Sun SeeBeyond > eWays > CICSeWay > CICS_eWay, and File > FileClient nodes.

3 Populate the eInsight Business Process Designer's canvas with the following activities from the Project Explorer tree, as displayed in Figure 29:

- receive, under Sun SeeBeyond > eWays > File > FileClient
- execute, under Sun SeeBeyond > eWays > CICSeWay > CICS_eWay
- write, under Sun SeeBeyond > eWays > File > FileClient

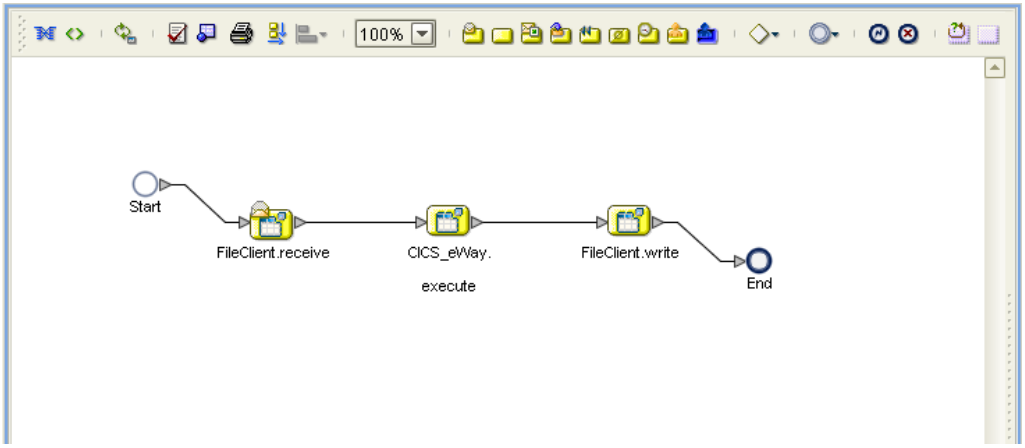
Figure 29 eInsight Business Process Designer - Populate the Canvas



4 Link the modeling elements by clicking on the element connector and dragging the cursor to the next element connector, making the following links as displayed in **Figure 30 on page 71**.

- Start -> FileClient.receive
- FileClient.receive -> CICS_eWay.execute
- CICS_eWay.execute -> FileClient.write
- FileClient.write -> End

Figure 30 eInsight Business Process Designer - Populate the Canvas



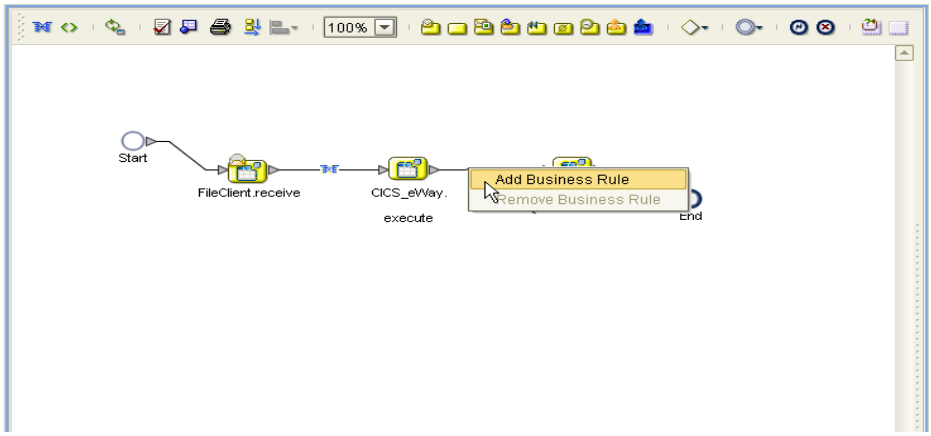
Configuring the Modeling Elements

Business Rules, created between the Business Process Activities, allow you to configure the relationships between the input and output Attributes of the Activities using the Business Process Designer's Business Rule Designer.

Adding Business Rules

1. Right-click the link between the **FileClient.receive** and **CICS_eWay.execute** Activities and select **Add Business Rule** from the shortcut menu.
2. Repeat step 1 for the **CICS_eWay.execute** to **FileClient.write** link (see Figure 31).

Figure 31 eInsight Business Process Designer - Adding Business Rules



Using the Business Rule Designer

1. From the eInsight Business Process Designer toolbar, click the **Display Business Rules Designer** button. The Business Rule Designer appears at the bottom of the eInsight Business Process Designer.
2. Click on the Business Rule icon in the link between **FileClient.receive** and **CICS_eWay.execute** to display the Business Rule's Input and Output Attributes in the Business Rule Designer. These Attributes can now be modified.
3. Create the **FileClient.receive -> CICS_eWay.execute** Business Rule:

A From the Business Rule Designer's **String** menu, select Settings. The Method Palette appears. From the **String** tab, select **bytes to text** and **text to bytes**, and click **Close**. The **bytes to text** and **text to bytes** options are added to the String menu.

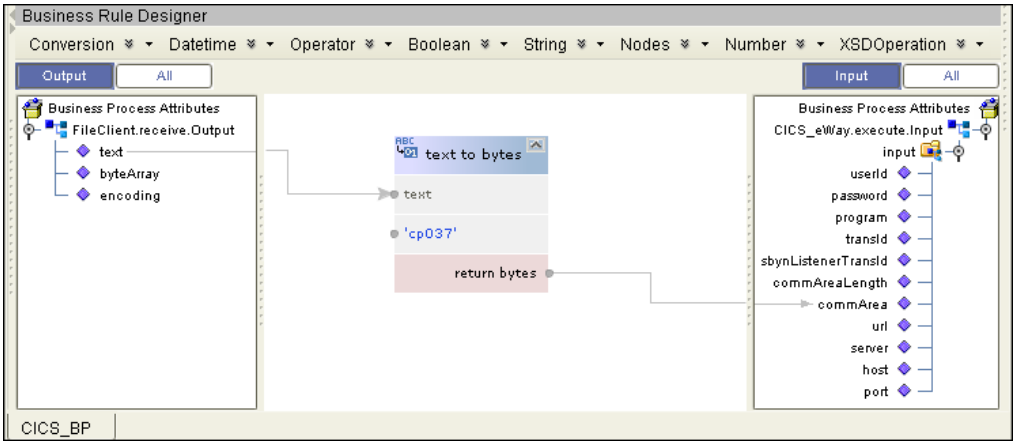
B From the Business Rule Designer's **String** menu, select **text to bytes**. The **text to bytes** method box appears on the Business Rule Designer canvas.

C From the **text to bytes** method box, double-click the **encoding** node and enter **cp037** as the literal value.

D Map **text**, under **FileClient.receive.Output** to the **text** input node of the **text to bytes** method box. To do this, click on **text** under **FileClient.receive.Output** in Output pane of the Business Rule Designer, and drag your cursor to the **text** input node of the **text to bytes** method box.

E Map the **return bytes** output node of the **text to bytes** method box, to **commArea** under **CICS_eWay.execute.Input** > **Input** in the Input pane of the Business Rule Designer (see Figure 32).

Figure 32 eInsight Business Rule Designer



4 Create the **CICS_eWay.execute -> FileClient.write** Business Rule:

A Click on the Business Rule icon in the link between the **CICS_eWay.execute** and **FileClient.write** activities. This Business Rule Designer now displays the Input and Output attributes.

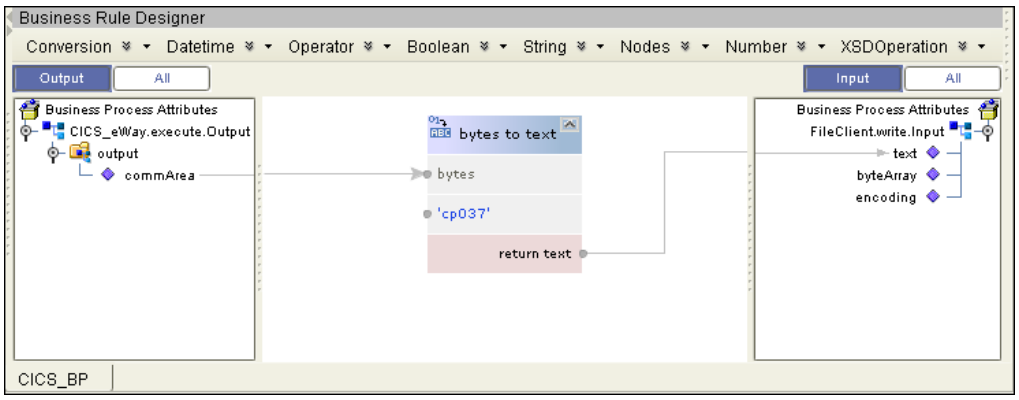
B From the Business Rule Designer's **String** menu, select **bytes to text**. The **bytes to text** method box appears.

C From the **bytes to text** method box, double-click the **encoding** node and enter **cp037** as the literal value.

D Map **commArea**, under **CICS_eWay.execute.Output** > **output** in the Output pane, to the **bytes** input node of the **bytes to text** method box.

E Map the **return text** output node of the **bytes to text** method box, to **text** under **FileClient.write.Input** in the Input pane (see **Figure 33 on page 73**).

Figure 33 eInsight Business Rule Designer



5 From the Business Process Designer toolbar, click **Synchronize Graphical Model and Business Process** icon to synchronize the graphical interface to the Business Process code.

6 Save your changes to the Repository.

4.6.3 Creating a Connectivity Map

The Connectivity Map provides a canvas for assembling and configuring a Project's components.

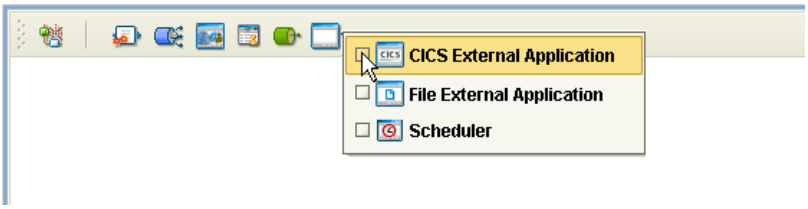
1 From the Project Explorer tree, right-click the new **prjCICS_BP_Sample** Project and select **New > Connectivity Map** from the shortcut menu.

2 The New Connectivity Map appears and a node for the Connectivity Map is added under the Project on the Project Explorer tree labeled **CMap1**. Rename the Connectivity Map **cmCICS_BPEL**.

Selecting the External Applications

In the Connectivity Map, the eWays are associated with External Systems. For example, to establish a connection to CICS, you must first select CICS as an External System to use in your Connectivity Map (see Figure 34).

Figure 34 Connectivity Map - External Applications



1 Click the **External Application** icon on the Connectivity Map toolbar,

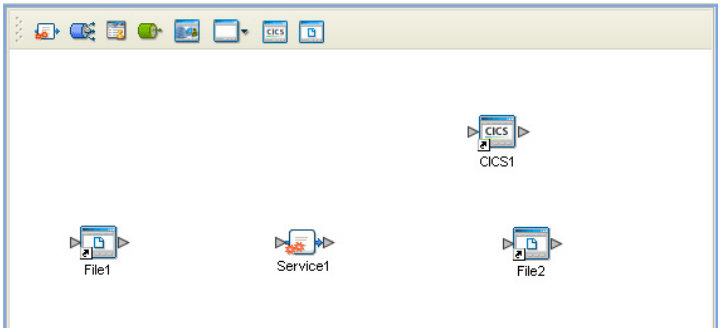
2 Select the external systems necessary to create your Project (for this sample, **CICS** and **File**). Icons representing the selected external systems are added to the Connectivity Map toolbar.

Populating the Connectivity Map

The icons in the toolbar represent the available components used to populate the Connectivity Map canvas. Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1. For this sample, drag the following components onto the Connectivity Map canvas as displayed in Figure 35:
   - **File External System** (2)
   - **Service** (A service is a container for Collaborations, Business Processes, eTL processes, and so forth)
   - **CICS External System**

Figure 35 Connectivity Map with Components



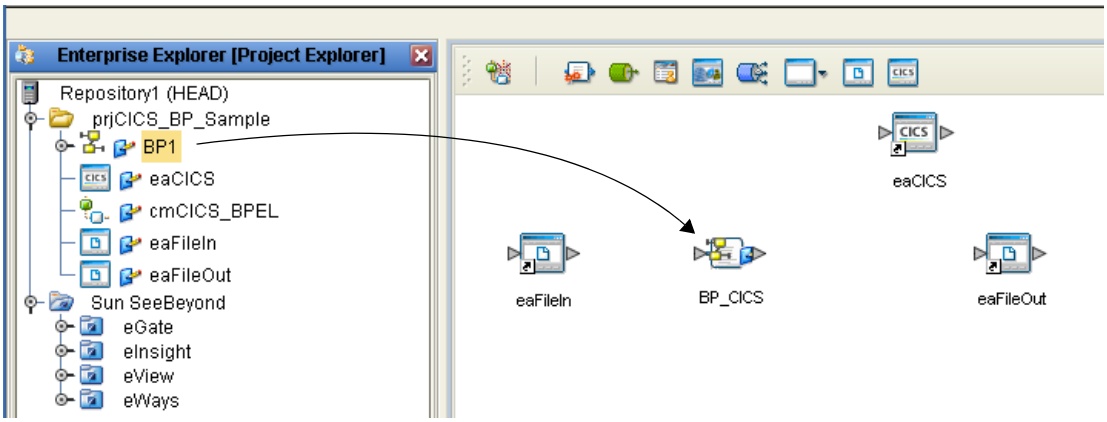
2. Rename the **File1** External Application to **eaFileIn** by right-clicking the object, selecting **Rename** from the shortcut menu, and typing in the new name. In the same way, rename the other Connectivity Map components as follows:
   - **File2** to **eaFileOut**
   - **CICS1** to **eaCICS**
   - **Service1** to **BP_CICS**.
3. Save your current changes to the Repository.

4.6.4 Binding the eWay Components

Once the Connectivity Map has been populated, components are associated and Bindings are created in the Connectivity Map.

1. From the Project Explorer, double-click **cmCICS_BPEL**. The Enterprise Designer canvas now displays the Connectivity Map.
2. Drag and drop the **CICS_BP** Business Process from the Project Explorer to the Service (**BP_CICS**). If the Business Process was successfully associated, the Service's icon changes to a Business Process icon (see **Figure 36 on page 75**).

Figure 36 Connectivity Map - Binding the eWay Components



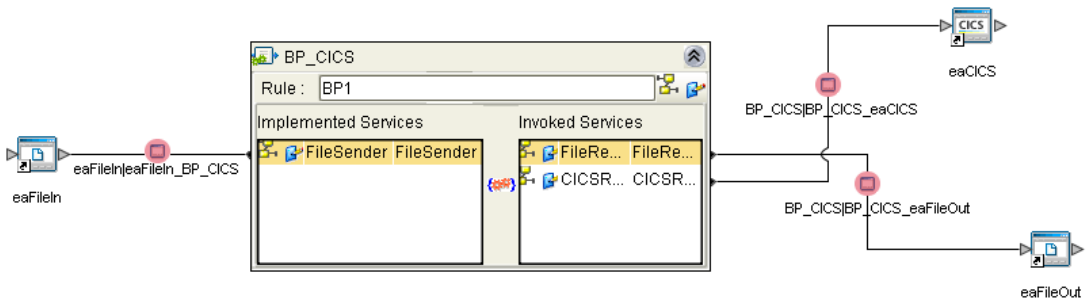
3 Double-click the **BP_CICS Service**. The **BP_CICS** Binding dialog box appears using the **BP1** Rule.

4 From the BP_CICS Binding dialog box, map **FileSender** (under Implemented Services) to the **eaFileIn** (File) External Application. To do this, click on **FileSender** in the BP_CICS Binding dialog box, and drag the cursor to the **eaFileIn** External Application in the Connectivity Map. A link, **eaFileIn|eaFileIn_BP_CICS**, is now visable between the eaFileIn External Application and the BP_CICS.

5 From the BP_CICS Binding dialog box, map **CICS_Receiver** (under Invoked Services) to the **eaCICS** External Application.

6 From the BP_CICS Binding dialog box, map **FileReceiver** to the **eaFileOut** External Application (see Figure 37).

Figure 37 Connectivity Map - Associating (Binding) the Project's Components



7 Minimize the BP_CICS Binding dialog box by clicking the chevrons in the upper-right corner.

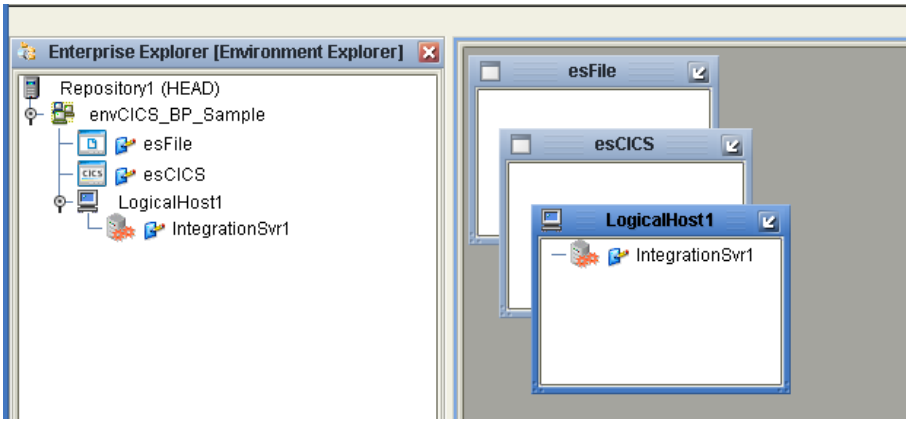
8 Save your current changes to the Repository.

4.6.5 Creating an Environment

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

1. From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.
2. Right-click the Repository and select **New Environment**. A new Environment is added to the Environment Explorer tree.
3. Rename the new Environment to **envCICS_BP_Sample**.
4. Right-click **envCICS_BP_Sample** and select **New > CICS External System**. Name the External System **esCICS** Click **OK**. **esCICS** is added to the Environment Editor.
5. Right-click **envCICS_BP_Sample** and select **New > File External System**. Name the External System **esFile**. Click **OK**. **esFile** is added to the Environment Editor.
6. Right-click **envCICS_BP_Sample** and select **New > Logical Host**. The **LogicalHost1** box is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.
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** (see Figure 38).

Figure 38 Environment Editor - envCICS_BP_Sample

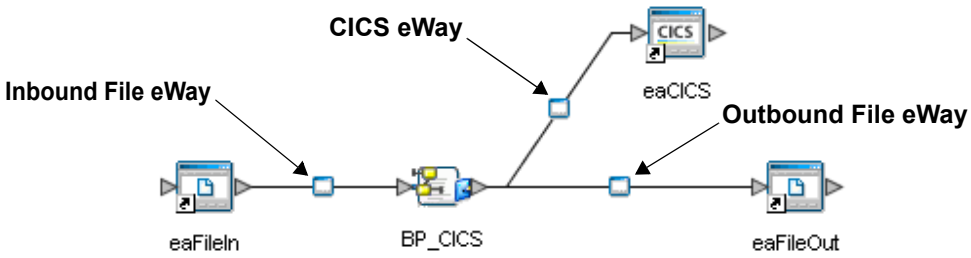


8. Save your current changes to the Repository.

4.6.6 Configuring the eWays

The prjCICS_BP_Sample Project uses three eWays, each represented in the Connectivity Map as a node between an External Application and a Service (see Figure 39). eWays facilitate communication and movement of data between the external applications and the eGate system.

Figure 39 eWays



The eWay properties are set from both the Connectivity Map and the Environment Explorer. For more information on configuring eWay properties see **"Configuring the CICS eWay" on page 47**.

Configure the File eWays Connectivity Map Properties

1 From the Project's Connectivity Map, double-click the **eaFileIn eWay**. The **Properties Editor** opens to the **eaFileIn** eWay properties. Modify the properties for your system, include the settings in Table 13, and click **OK**.

Table 13 eaFileIn eWay properties

eaFileIn (inbound) eWay Connection Parameters	
Input file name	Input*.txt

2 In the same way, modify the **eaFileOut** eWay properties for your system, include the settings in Table 14, and click **OK**.

Table 14 eaFileOut eWay properties

eaFileOut eWay Connection Parameters	
Output file name	output%.dat

Configure the File eWay Environment Explorer Properties

3 From the **Environment Explorer** tree, right-click the File External System (**esFile** in this sample), and select **Properties**. The Properties Editor opens to the File eWay Environment configuration.

4 Modify the File eWay Environment configuration properties for your system, include the settings in **Table 15 on page 78**, and click **OK**.

Table 15 File eWay Environment Explorer Properties

File eWay Environment Explorer Properties	
Inbound File eWay - Set as directed, otherwise use the default settings	
Parameter Settings > Directory	*C:/temp or the input directory of your choice*
Outbound File eWay - Set as directed, otherwise use the default settings.	
Parameter Settings > Directory	*C:/temp or the output directory of your choice*

Configure the CICS eWay Connectivity Map Properties

1 From the **Connectivity Map**, double-click the **CICS eWay**. The **Properties Editor** opens to the CICS eWay Connectivity Map configuration properties.

2 Modify the CICS eWay Connectivity Map properties for your system, include the settings in Table 16, and click **OK**.

Table 16 CICS eWay Connectivity Map Properties

CICS eWay Connectivity Map Properties	
Connector - Set as directed, otherwise use the default settings	
Connection Transport	*Sun SeeBeyond CICS Listener or CICSTransaction Gateway*
CICS Client - Set as directed, otherwise use the default settings.	
CICS Program	*The appropriate CICS program*

Configure the CICS eWay Environment Explorer Properties

3 From the **Environment Explorer** tree, double-click the CICS External System (**esCICS** in this sample) to opens to the CICS eWay Environment properties.

4 Modify the CICS eWay Environment configuration properties for your system, including the settings in Table 17, and click **OK**.

Table 17 CICS eWay Environment Explorer Properties

CICS eWay Environment Explorer Properties	
Sun SeeBeyond CICS Listener (if you use Sun SeeBeyond CICS Listener as the connection transport) - Set as directed, otherwise use the default settings	
Host	*The name of the mainframe host*
Port	*The TCP/IP port where Sun SeeBeyond CICS Listener is listening*
CICS Gateway (if you use CTG as the connection transport) - Set as directed, otherwise use the default settings	
URL	*The remote or local Gateway node name or IP address*
Port	*The TCP/IP port where CTG is running*
CICS Client - Set as directed, otherwise use the default settings.	
CICS UserId	*The login ID of the CICS user.*
CICS Password	*The login password of the CICS user.*

Note: *CICS eWay properties may also be set from the Collaboration Editor (Java) or eInsight Business Rules Designer, in which case they will override the corresponding properties in the eWay's configuration file. Any properties that are not overridden retain their saved settings.*

4.6.7 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.

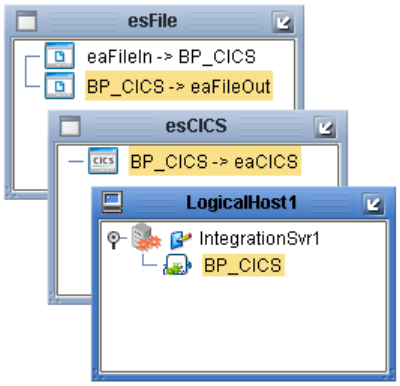
For more information on deploying a Project see the *Sun SeeBeyond Java™ Composite Application Platform Suite Deployment Guide*.

4.6.8 Creating the Deployment Profile

A Deployment Profile is used to assign services and message destinations to the integration server and message server. Deployment profiles are created using the Deployment Editor.

1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjCICS_BP_Sample**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpCICS_BP_Sample**). Select **envCICS_BP_Sample** as the Environment and click **OK**.
3. From the Deployment Editor toolbar, click the **Automap** icon. The Project's components are automatically mapped to their system windows (see Figure 40).

Figure 40 Deployment Profile



4.6.9 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java™ Composite Application Platform Suite installation.
2. Double-click the **domainmgr.bat** file. The **Domain Manager** appears.
3. If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
4. If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.
5. Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

4.6.10 Building and Deploying the Project

The Build process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed. You can now test your sample.

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.*

4.6.11 Running the Sample

To run your deployed sample Project do the following

1. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.
2. From your output directory, verify the output data.

Chapter 5

Implementing a CICS eWay Project

This chapter provides an introduction to the CICS eWay components and information on how these components are created and implemented in an eGate Project. It is assumed that the reader understands the basics of creating a Project using the Enterprise Designer. For more information on creating an eGate Project see the *Sun SeeBeyond eGate™ Tutorial* and the *Sun SeeBeyond eGate™ Integrator User's Guide*.

What's in This Chapter

5.1 CICS eWay Java Collaboration Implementation

This chapter presents sample CICS eWay Projects that use Collaboration Definitions (Java) to implement the Business Logic. These Projects are created using the same procedures as the sample end-to-end Project provided in the *Sun SeeBeyond eGate Tutorial*.

5.1.1 CICS eWay Components

CICS eWay components that are unique to this eWay include the following:

CICS eWay Properties File

The Properties file for the CICS eWay contains the parameters that are used to connect with a specific external system. These parameters are set using the **Properties Editor**. For more information about the CICS eWay Configuration File and the **Properties Editor** see **"Creating and Configuring the CICS eWay" on page 47**.

CICS_eWay.CICSClient OTD

The CICS_eWay.CICSClient OTD is provided with the eWay and contains methods and attributes used to create the Business Rules that invoke the CICS program.

Cobol Copybook Wizard

The Cobol Copybook Wizard allows the eGate Integrator to convert Cobol Copybook files into eGate Object Type Definition (OTD) files.

5.2 CICS eWay Sample Projects

Two sample Projects are defined in the following sections.

- **prjCICS_JCD_Sample**: demonstrates the CICS eWay receiving data from a file, passing the data to a CICS External System, and writing the output from the CICS System to another file.
- **prjCICS_COBOL_JCD_Sample**: demonstrates a simple CICS Project that uses the Cobol Copybook Converter.

5.3 Importing a Sample Project

Sample eWay Projects are included as part of the installation package. To import a sample eWay Project to the Enterprise Designer do the following:

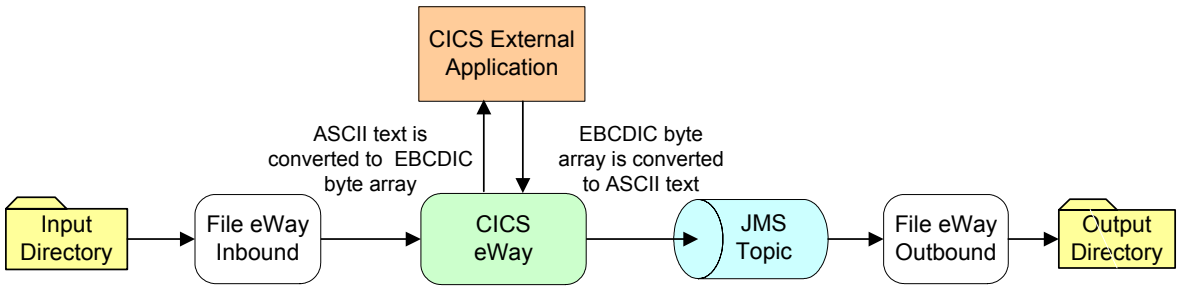
1. The sample files are uploaded with the eWay's documentation SAR file and downloaded from the Sun Java Composite Application Platform Suite Installer's **Documentation** tab. The **CICS_eWay_Sample.zip** file contains the various sample Project ZIP files. Extract the samples to a local file.
2. Save all unsaved work before importing a Project.
3. From the Enterprise Designer's Project Explorer pane, right-click the Repository and select **Import** from the shortcut menu. The **Import Manager** appears.
4. Browse to the directory that contains the sample Project zip file. Select the sample file (for this sample, **CICSjms_JCE_Sample.zip**) and click **Import**. After the sample Project is successfully imported, click **Close**.
5. Before an imported sample Project can be run you must do the following:
   - Create an **Environment** (see **"Creating an Environment" on page 96**)
   - Configure the eWays for your specific system (see **"Configuring the eWays" on page 97**)
   - Create a **Deployment Profile** (see **"Creating the Deployment Profile" on page 100**)
   - Create and start a domain (see **"Creating and Starting the Domain" on page 115**)
   - Build and deploy the Project (see **"Building and Deploying the Project" on page 116**)

5.4 The CICS JCD Sample Project Overview

The CICS eWay Project, **prjCICS_JCD_Sample**, demonstrates the following:

- Input data (ASCII text) is read from an external directory by the Inbound File eWay.
- The ASCII text data is converted to EBCDIC (byte array) using cp037 encoding, and published by the CICS eWay to the CICS external system.
- The CICS eWay subscribes to the CICS external system, receives EBCDIC input data, and converts it to String ASCII data using cp037 encoding. The data is then published to the JMS Topic.
- The outbound File eWay subscribes to the JMS Topic and publishes the data to an external directory.

Figure 41 prjCICS_JCD_Sample Project



5.5 Creating the prjCICS_JCD_Sample Project

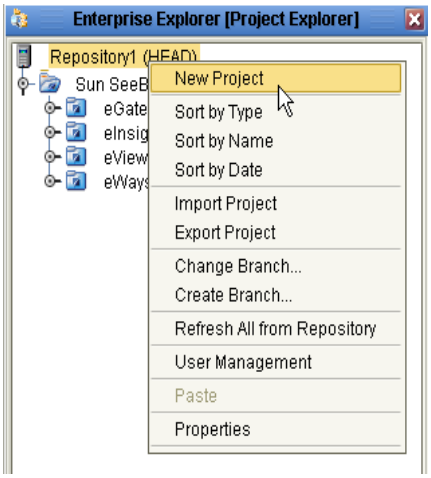
The following pages provide step by step directions for manually creating the prjCICS_JCD_Sample Project's components.

5.5.1 Creating a Project

To create a new Project in the Enterprise Designer, do the following:

1. Start the Enterprise Designer.
2. From the Enterprise Explorer's Project Explorer tab, right-click your Repository and select **New Project from the shortcut menu** (see **Figure 42 on page 85**). A new Project (Project1) appears on the Project Explorer tree.

Figure 42 Enterprise Explorer - New Project



3 Rename the Project (for this sample, **prjCICS_JCD_Sample**).

5.5.2 Creating a Connectivity Map

The Connectivity Map provides a canvas for assembling and configuring a Project's components.

1 From the Project Explorer tree, right-click the new **prjCICS_JCD_Sample** Project and select **New > Connectivity Map** from the shortcut menu.

2 The New Connectivity Map appears and a node for the Connectivity Map is added under the Project on the Project Explorer tree labeled **CMap1**. Rename the Connectivity Map to **cmCICS_JCD_Sample**.

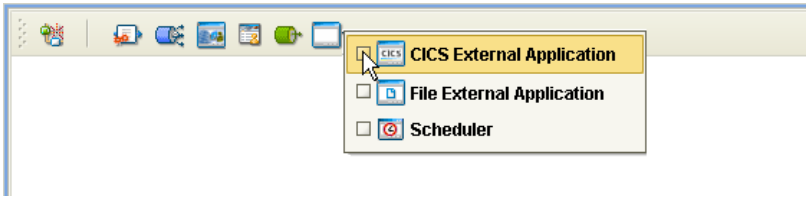
The icons in the toolbar represent the available components used to populate the Connectivity Map canvas.

Selecting the External Applications

The icons in the toolbar represent the available components used to populate the Connectivity Map canvas.

In a Connectivity Map, the eWays are associated with External Systems. For example, to establish a connection to CICS, you must first select CICS as an External System to use in your Connectivity Map (see Figure 43).

Figure 43 Connectivity Map - External Applications



1 Click the **External Application** icon on the Connectivity Map toolbar,

2 Select the external systems needed for your Project (for this sample, the **CICS** and **File** External Applications). Icons representing the selected external systems are added to the Connectivity Map toolbar.

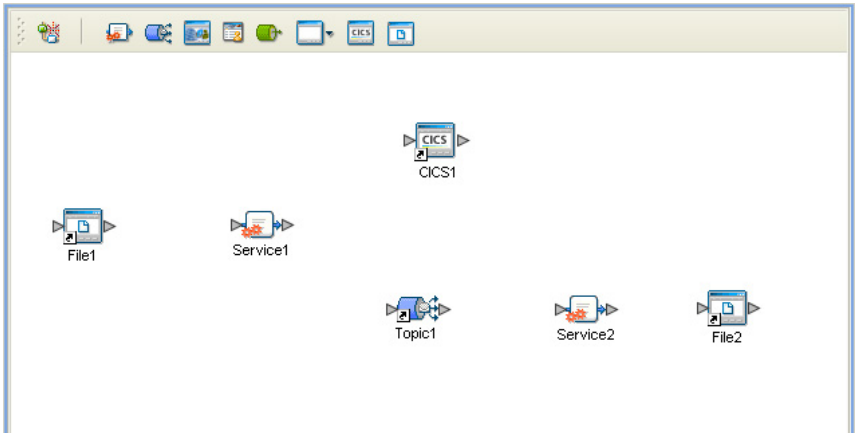
Populating the Connectivity Map

Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1 For this sample, drag the following components onto the Connectivity Map canvas as displayed in Figure 44:

- **File External System** (2)
- **Service** (2) A service is a container for Collaborations, Business Processes, eTL processes, and so forth.
- **Topic**
- **CICS External System**

Figure 44 Connectivity Map with Components



2 Rename the objects by right-clicking the object, selecting **Rename** from the shortcut menu, and typing in the new name. Change the names to the following:

- File1 to eaFileIn
- **Service1** to **jcolCICS1**
- **CICS1** to **eaCICS**
- **Topic1** to **Topic**
- **Service2** to **jcolCICS2**
- **File2** to **eaFileOut**

3 From the Enterprise Designer toolbar, click the **Save All** icon to save your current changes to the Repository.

5.5.3 Creating the Collaboration Definitions

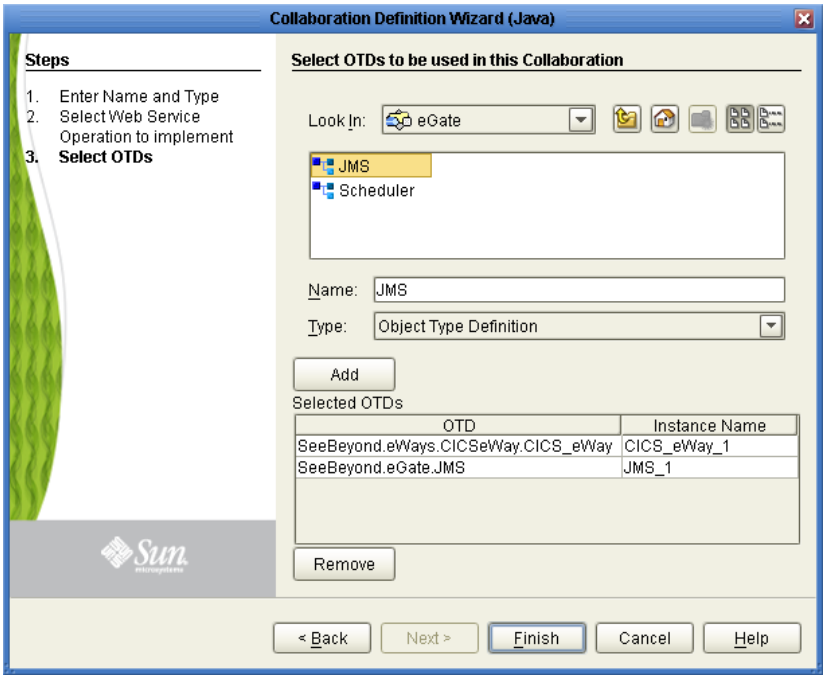
The next step is to create two Collaborations using the **Collaboration Definition Wizard (Java)**. Once you create the Collaboration Definitions, you can write the Business Rules of the Collaborations using the Collaboration Editor.

Creating the jcdCICS1 Collaboration

The **jcdCICS1** Collaboration defines transactions from the inbound file application to the CICS eWay and the CICS application, and back to the JMS Topic.

1. From the Project Explorer, right-click the sample Project and select **New** > **Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.
2. Enter a Collaboration Definition name (for this sample **jcdCICS1**) and click **Next**.
3. For Step 2 or the wizard, from the Web Services Interfaces selection window, double-click **Sun SeeBeyond** > **eWays** > **File** > **FileClient** > **receive**. The File Name field now displays **receive.** Click **Next**.
4. For Step 3 of the wizard, from the Select OTDs selection window, double-click **Sun SeeBeyond** > **eWays** > **CICSeWay** > **CICS_eWay**. The **CICS_eWay** OTD is added to the Selected OTDs field.
5. Click the **Up One Level** button to return to the Repository. Double-click **Sun SeeBeyond** > **eGate** > **JMS**. The **Selected OTDs** field now lists the **JMS OTD** (see Figure 45).

Figure 45 Collaboration Definition Wizard (Java) - Select Web Service Interface



6. Click **Finish**. The Collaboration Editor with the new **jcdCICS1** Collaboration appears in the right pane of the Enterprise Designer.

Creating the jcdCICS2 Collaboration (Java)

The **jcdCICS2** Collaboration defines transactions made from the JMS Topic to the Outbound File eWay.

1. From the Project Explorer, right-click the sample Project and select **New** > **Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.
2. Enter a Collaboration Definition name (**jcdCICS2**) and click **Next**.
3. For Step 2 of the wizard, from the Web Services Interfaces selection window, double-click **Sun SeeBeyond** > **eGate** > **JMS** > **receive**. The File Name field now displays **receive.** Click **Next**.
4. For Step 3, **Select OTDs**, from the Select OTDs selection window, double-click **Sun SeeBeyond** > **eWays** > **File** > **FileClient**. The Selected OTDs field now lists the **FileClient** OTD.
5. Click **Finish**. The **jcdCICS2** Collaboration appears.

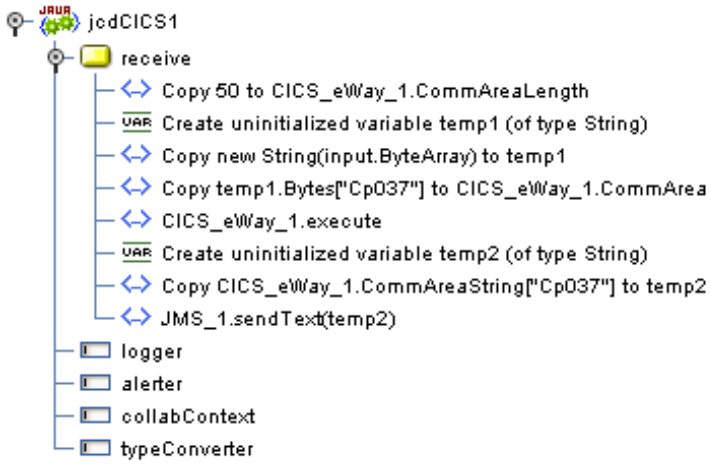
5.5.4 Using the Collaboration Editor (Java)

The next step in the sample is to create the Business Rules of the Collaboration using the Collaboration Editor.

Creating the jcdCICS1 Business Rules

The **jcdCICS1** Collaboration contains the Business Rule displayed in Figure 46.

Figure 46 jcdCICS1 Business Rules



To create the **jcdCICS1** Collaboration Business Rules do the following:

1. From the Project Explorer tree, double-click **jcdCICS1** to open the Collaboration Editor (Java) to the jcdCICS1 Collaboration.
2. To create comments for the Business Rules, click the comment icon on the Business Rules toolbar. The **Enter a Comment** dialog box appears. Enter the comment and

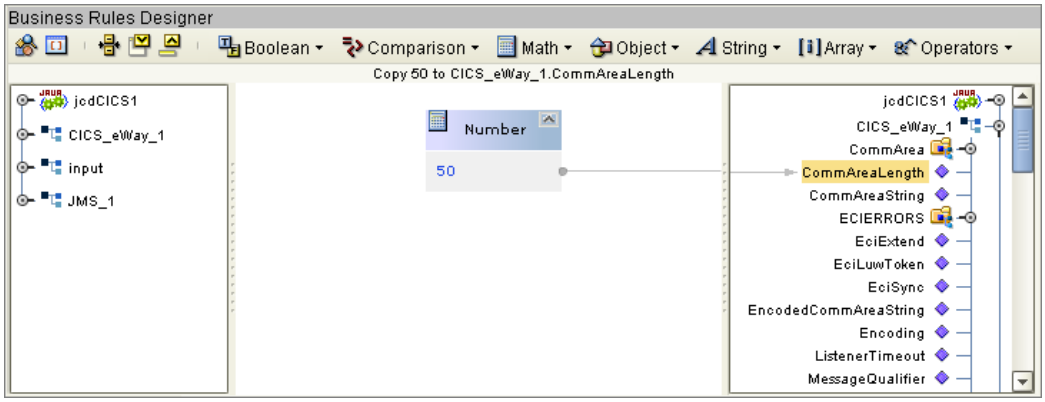
click **OK**. The comment is placed on the Business Rules tree under the last selected item. Once the comment is created, it can be moved by clicking the comment and dragging it up or down the Business Rules tree to a new location.

3 Create the **Copy 50 to CICS_eWay_1.CommAreaLength** Business Rule:

A From the Business Rules Designer Math menu, select **Literal Number**. The **Number** method box appears. Enter **50** as the value.

B Map the **50** output node of the **Number** method box to **CommAreaLength** under **CICS_eWay_1** in the right pane of the Collaboration Editor. To do this, click on the **50** output node in the **Number** method box and drag the cursor to **CommAreaLength** under **CICS_eWay_1** in the right pane of the Business Rules Designer (see Figure 47).

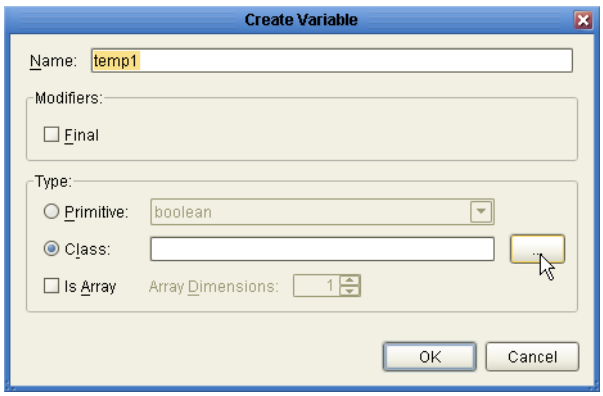
Figure 47 jcdCICS1 - Copy 50 to CICS_eWay_1.CommAreaLength Rule



4 Create the **Create uninitialized variable temp1 (of type String)** rule (variable):

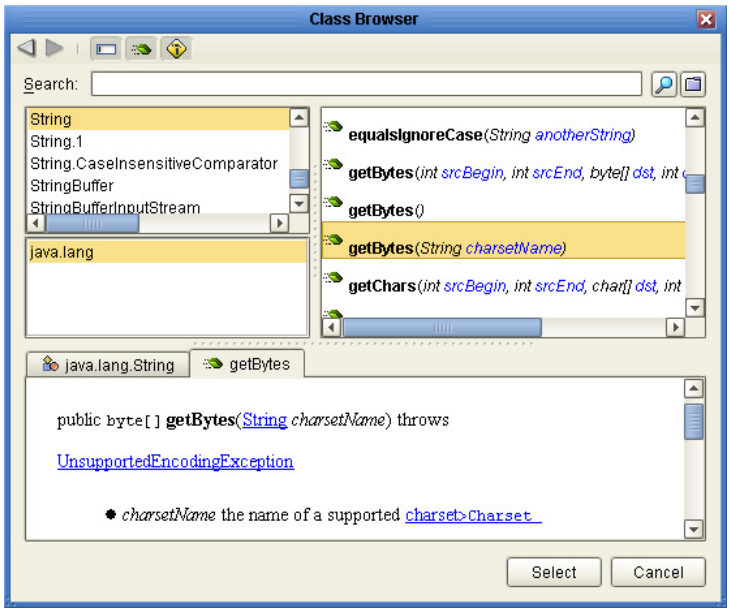
A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears (see Figure 48).

Figure 48 Create Variable dialog box



B From the Create Variable dialog box, enter **temp1** as the Name, and for Type, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears (see **Figure 49 on page 90**).

Figure 49 Class Browser



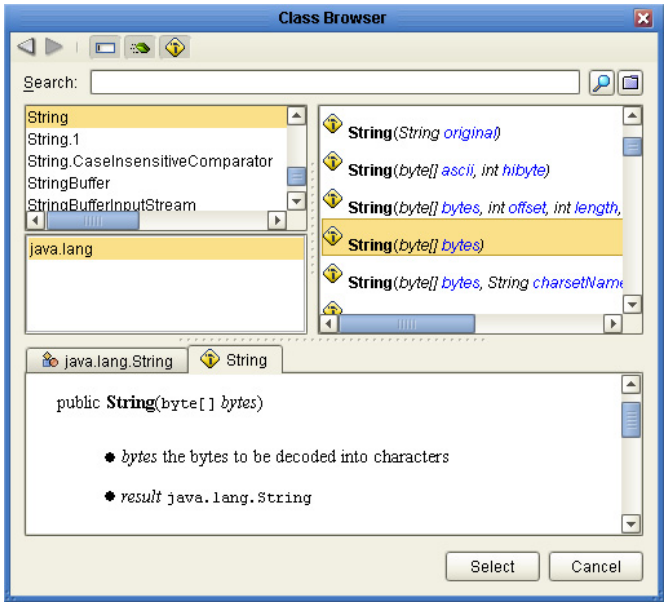
C From the **Class Browser** dialog box, select **String** in the All Classes field, and **String(byte[] bytes)** in the String field. Click **Select**. From the **Create Variable** dialog box, click **OK**.

5 Create the **Copy new String(input.ByteArray) to temp1 rule:**

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the Business Rules Designer toolbar, click the **Class Browser** icon. The Class Browser appears. Select **String** in the All Class field, select the **String(byte[] bytes)** constructor in the **String** field, and click **Select** (see Figure 50). The **String** constructor box appears.

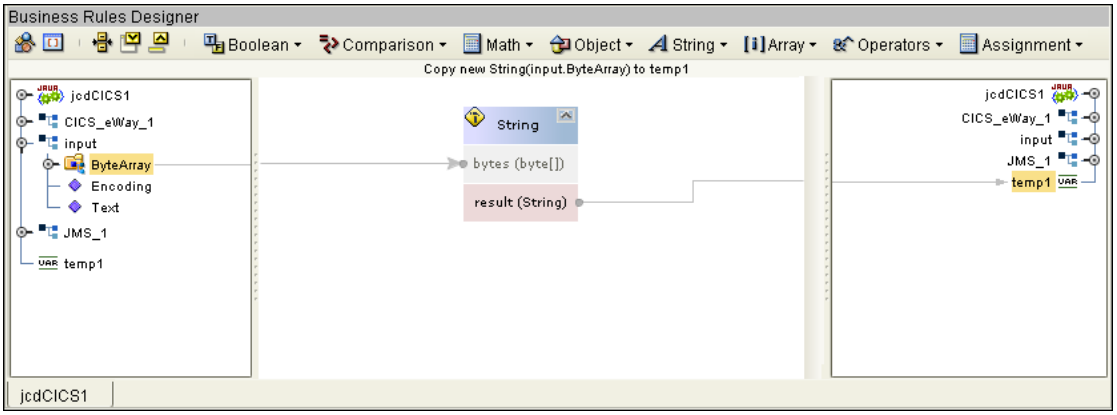
Figure 50 Class Browser



C Map **ByteArray**, under **input** in the left pane of the Business Rules Designer, to the **bytes (byte[])** input node of the **String** constructor box.

D Map the **result (String)** output node of the **String** constructor box to **temp1** in the right pane of the Business Rules Designer (see Figure 51).

Figure 51 Copy new String(input.ByteArray) to temp1 Rule



6 Create the **Copy temp1.Bytes["Cp037"] to CICS_eWay_1.CommArea** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

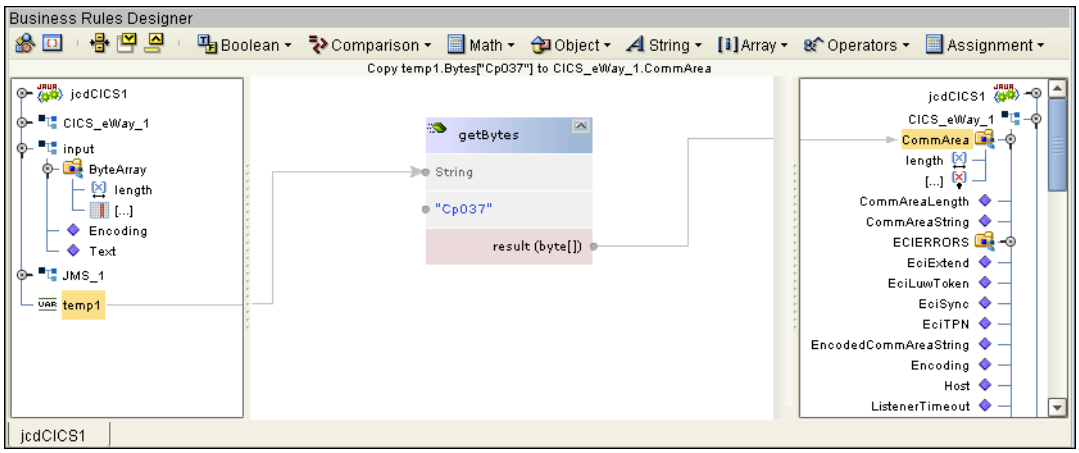
B From the left pane of the Business Rules Designer, right click the **temp1** variable and click **Select method to call** from the shortcut menu.

C From the method selection box, double-click **getBytes(String charsetName)**. The **getBytes** method box appears.

D From the **getBytes** method box, change the value of the **charsetName (String)** field to **Cp037**. To do this, double-click the **charsetName (String)** field and enter **Cp037**.

E Map the **result (byte[])** output node of the **getBytes** method box to **CommArea** under **CICS_eWay_1** in the right pane of the Business Rules Designer (see).

Figure 52 Copy temp1.Bytes["Cp037"] to CICS_eWay_1.CommArea rule



7 Create the **CICS_eWay_1.execute** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right-click **CICS_eWay_1** and select **Select method to call** from the shortcut menu.

C From the method selection box, double-click **execute**(). The **execute** method box appears.

8 Create the (variable) **Create uninitialized variable temp2 (of type String)** rule:

A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears.

B From the Create Variable dialog box, enter **temp1** as the Name, and for Type, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears.

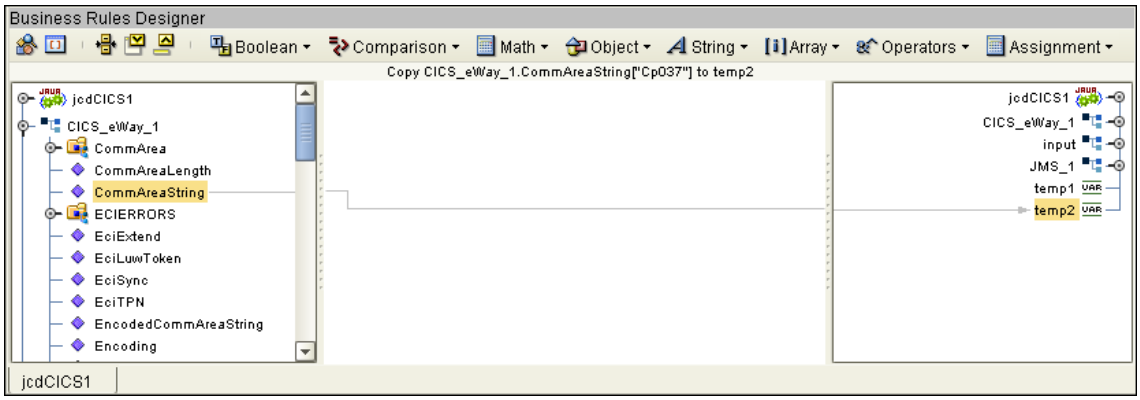
C From the **Class Browser** dialog box, select **String** in the All Classes field, and **getBytes(String charsetName)** in the String field. Click **Select**. From the **Create Variable** dialog box, click **OK**.

9 Create the **Copy CICS_eWay_1.CommAreaString["Cp037"] to temp2** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B Map CommAreaString, under CICS_eWay_1 in the left pane of the Business Rules Designer, to temp2 in the right pane (see Figure 53).

Figure 53 Copy CICS_eWay_1.CommAreaString["Cp037"] to temp2 rule



10 Create the **JMS_1.sendText(temp2)** rule:

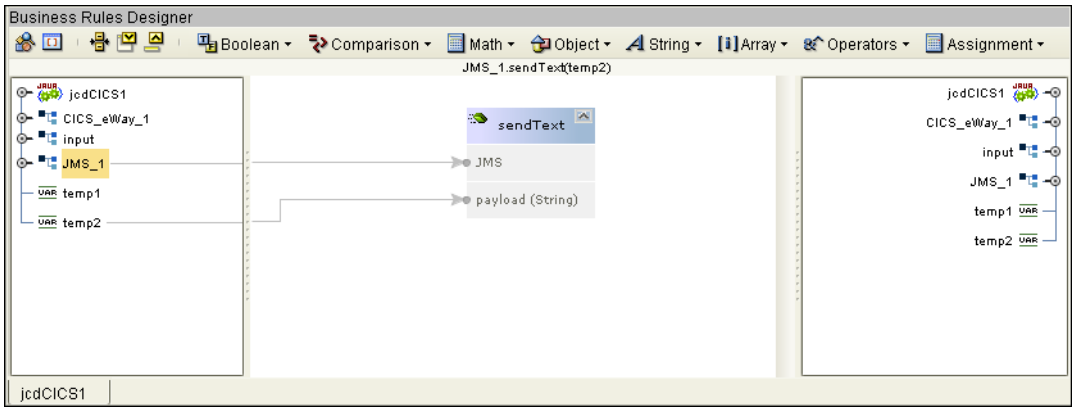
A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right click **JMS_1** and click **Select method to call** from the shortcut menu.

C From the method selection box, double-click **sendText(String payload)**. The **sendText** method box appears.

D Map **jmstext** in the left pane of the Business Rules Designer to the **payload (String)** output node of the **sendText** method box (see **Figure 54 on page 93**).

Figure 54 JMS_1.sendText(temp2) Rule



11 From the editor's toolbar, click **Validate** to check the Collaboration for errors.

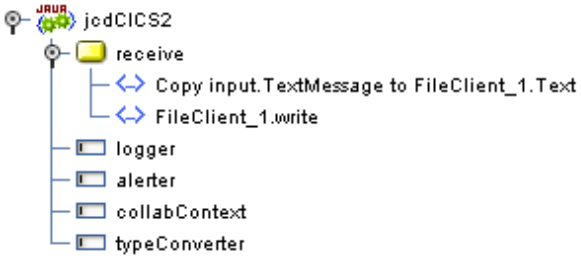
12 Save your current changes to the Repository.

Note: *See the Sun SeeBeyond eGate™ Integrator User's Guide for more information on editing Collaborations.*

Creating the jcdCICS2 Collaboration Business Rules

Be careful to open all nodes specified in the directions to connect to the correct item. The jcdCICS2 Collaboration contains the Business Rule displayed in Figure 55

Figure 55 jcdCICS2 Business Rules



To create the **jcdCICS2** Collaboration Business Rules do the following:

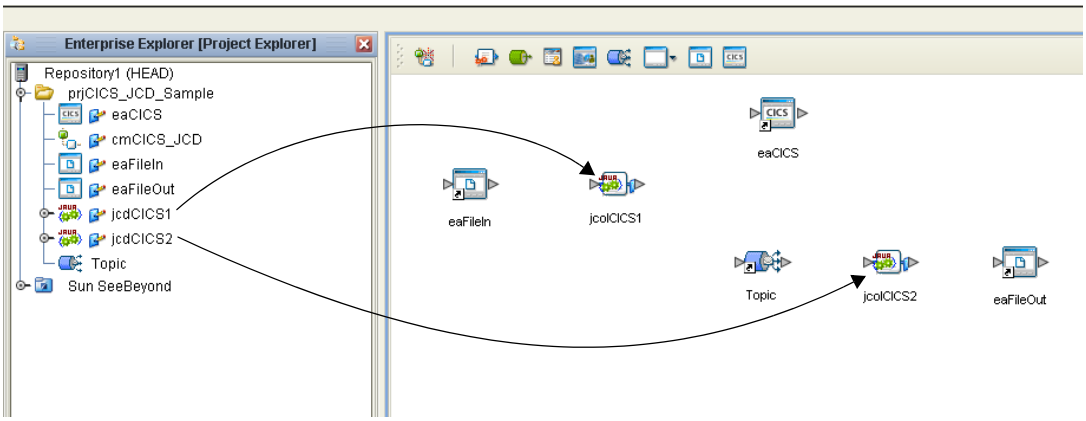
1. From the Project Explorer tree, double-click **jcdCICS2** to open the Collaboration Editor to the **jcdCICS2** Collaboration.
2. To create comments for the Business Rules, from the Business Rules toolbar, click the **comment** icon, enter your comment and click **OK**. The comment is placed on the Business Rules tree under the last selected item. Once the comment is created, it can be moved by clicking the comment and dragging it up or down the Business Rules tree to a new location.
3. Create the **Copy input.TextMessage to FileClient_1.Text** rule:
   A. Select the **New rule** in the Business Rules pane.
   B. Map **TextMessage,** under **input** in the left pane of the Business Rules Designer, to **Text,** under **FileClient_1** in the right pane.
4. Create the **FileClient_1.write** rule:
   A. From the Business Rules toolbar click the **rule** icon to add a new rule.
   B. From the left pane of the Business Rules Designer, right-click **FileClient_1** and click **Select method to call** from the shortcut menu. Select the **write method**. The **write** method box appears.
5. From the editor's toolbar, click **Validate** to check the Collaboration for errors.
6. Save your current changes to the Repository.

5.5.5 Binding the eWay Components

After the Collaborations have been written, the components are associated and Bindings are created in the Connectivity Map.

1. From the Project Explorer tree, double-click **cmCICS_JCD_Sample** to display the Connectivity Map in the Enterprise Designers canvas.
2. Drag and drop the **jcdCICS1** Collaboration from the Project Explorer to the first Service (**jcolCICS1**). If the Collaboration was successfully associated, the Service's "gears" icon changes from red to green (see Figure 56).

Figure 56 Connectivity Map - Binding eWay Components



3 Drag and drop **jcdCICS2** from the Project Explorer to the second Service (**jcolCICS2**).

4 From the Connectivity Map canvas, double-click **jcolCICS1**. The **jcolCICS1** Binding box appears with the **jcdCICS1** Rule.

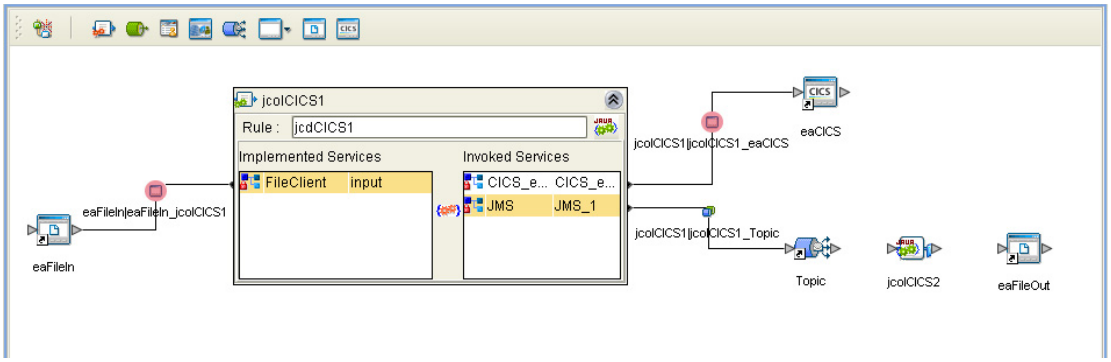
5 From the **jcolCICS1** Binding box, map your connections by clicking on the appropriate port of the Binding box and dragging it to the intended destination (the input or output port of a queue, topic or External Application) as follows:

A Map **FileClient Input** (under Implemented Services) to the inbound **eaFileIn** External Application.

B Map the **CICS_eWay** node (under Invoked Services) to the **eaCICS** External Application.

C Map **JMS_1** (under Invoked Services) to **Topic (see Figure 57 on page 95)**.

Figure 57 Connectivity Map - Associating (Binding) the Project's Components



6 Minimize the **jcolCICS1** Binding box by clicking the chevrons in the upper-right corner.

7 Double-click **jcolCICS2**. The **jcolCICS2 Binding box** appears with the **jcdCICS2** Rule.

8 From the **jcolCICS2** Binding box, create the following connections:

A Map **JMS_1** (under Implemented Services) to **Topic**.

B Map **FileClient_1** (under Invoked Services) to the **eaFileOut** Application.

9 Minimize the Collaboration Binding box, and save your current changes to the Repository

5.5.6 Creating an Environment

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

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. A new Environment is added to the Environment Explorer tree.

3 Rename the new Environment to **envCICS_JCD_Sample**.

4 Right-click **envCICS_JCD_Sample** and select **New CICS External System**. Name the External System **esCICS** Click **OK**. **esCICS** is added to the Environment Editor.

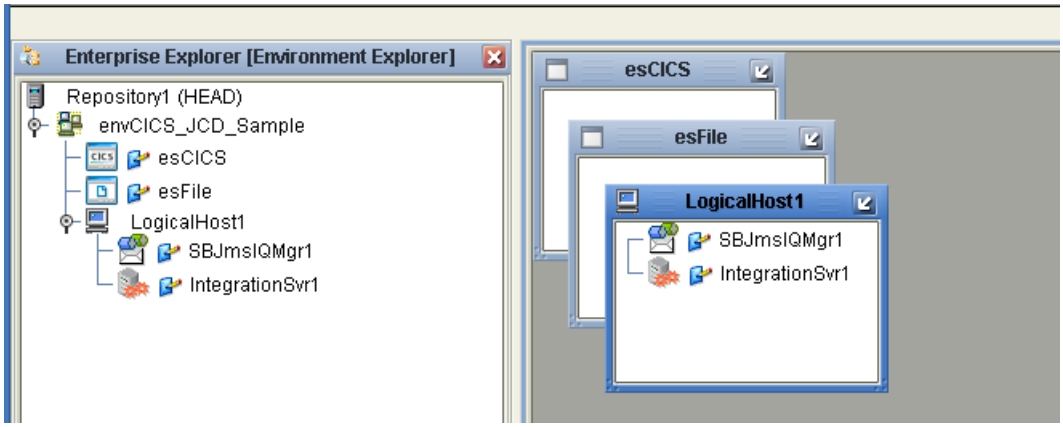
5 Right-click **envCICS_JCD_Sample** and select **New File External System**. Name the External System **esFile**. Click **OK**. **esFile** is added to the Environment Editor.

6 Right-click **envCICS_JCD_Sample** and select **New Logical Host**. The **LogicalHost1** box is added to the Environment and **LogicalHost1** is added to the Environment Editor tree.

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** (see Figure 58).

Figure 58 Environment Editor - envCICS_JCD_Sample

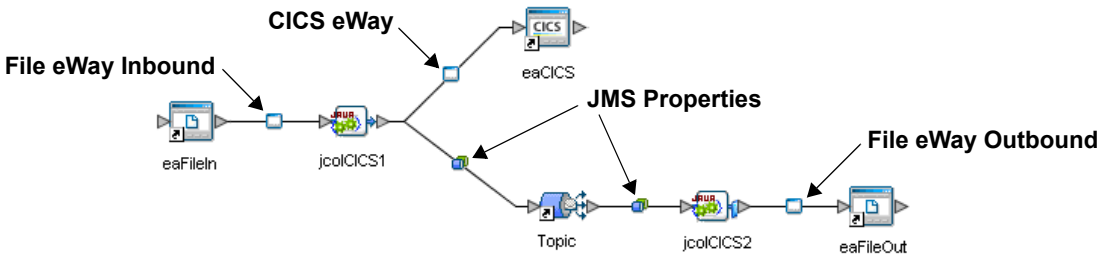


9 Save your current changes to the Repository.

5.5.7 Configuring the eWays

The prjCICS_JCD_Sample Project uses three eWays, each represented in the Connectivity Map as a node between an External Application and a Service (see Figure 59). eWays facilitate communication and movement of data between the External Application and the eGate system.

Figure 59 eWays and JMS Properties



The eWay properties are set from both the Connectivity Map and the Environment Explorer. For more information on configuring eWay properties see **"Configuring the CICS eWay" on page 47**:

Configure the eWays

The prjCICS_JCD_Sample Project uses three eWays, each represented in the Connectivity Map as a node between an External Application and a Collaboration.

The eWay properties are set from both the Project Explorer's Connectivity Map and the Environment Explorer. To configure the eWays do the following:

Configure the File eWays Connectivity Map Properties

1 From the Project's Connectivity Map, double-click the **eaFileIn eWay**. The **Properties Editor** opens to the **eaFileIn** eWay properties. Modify the properties for your system, including the settings in Table 18, and click **OK**.

Table 18 eaFileIn eWay Connectivity Map Properties

eaFileIn (inbound) eWay Connection Parameters	
Parameter Settings- Set as directed, otherwise use the default settings	
Input file name	Input*.txt

2 In the same way, modify the eaFileOut eWay properties for your system, including the settings in Table 19, and click **OK**.

Table 19 eaFileOut eWay Connectivity Map Properties

eaFileOut eWay Connection Parameters	
Parameter Settings- Set as directed, otherwise use the default settings	
Output file name	Output*.txt

Configure the File eWay Environment Explorer Properties

3 From the **Environment Explorer** tree, right-click the File External System (**esFile** in this sample), and select **Properties**. The Properties Editor opens to the File eWay Environment configuration.

4 Modify the File eWay Environment configuration properties for your system, including the settings in Table 20, and click **OK**.

Table 20 File eWay Environment Explorer Properties

File eWay Environment Explorer Properties	
Inbound File eWay - Set as directed, otherwise use the default settings	
Parameter Settings > Directory	*C:/temp or the input directory of your choice*
Outbound File eWay - Set as directed, otherwise use the default settings.	
Parameter Settings > Directory	*C:/temp or the output directory of your choice*

Configuring the CICS eWay

Configure the CICS eWay properties from both the Project Explorer and Environment Explorer. For more information on the CICS eWay properties, see **"CICS eWay Connectivity Map Properties" on page 50**.

Configure the CICS eWay Connectivity Map Properties

1 From the **Connectivity Map**, double-click the **CICS eWay**. The **Properties Editor** opens to the CICS eWay Connectivity Map configuration properties.

2 Modify the CICS eWay Connectivity Map properties for your system, including the settings in **Table 21 on page 98**, and click **OK**.

Table 21 CICS eWay Connectivity Map Properties

CICS eWay Connectivity Map Properties	
Connector - Set as directed, otherwise use the default settings	
Connection Transport	*Sun SeeBeyond CICS Listener or CICS Transaction Gateway*
CICS Client - Set as directed, otherwise use the default settings.	
CICS Program	*The appropriate CICS program*

Configure the CICS eWay Environment Explorer Properties

3 From the **Environment Explorer** tree, right-click the CICS External System (**esCICS** in this sample), and select **Properties**. The Properties Editor opens to the CICS eWay Environment configuration.

4 Modify the CICS eWay Environment configuration properties for your system, including the settings in Table 22, and click **OK**.

Table 22 CICS eWay Environment Explorer Properties

CICS eWay Environment Explorer Properties	
Sun SeeBeyond CICS Listener (if you use Sun SeeBeyond CICS Listener as the connection transport) - Set as directed, otherwise use the default settings	
Host	*The name of the mainframe host*
Port	*The TCP/IP port where Sun SeeBeyond CICS Listener is listening*
CICS Gateway (if you use CTG as the connection transport) - Set as directed, otherwise use the default settings	
URL	*The remote or local Gateway node name or IP address*
Port	*The TCP/IP port where CTG is running*
CICS Client Set as directed, otherwise use the default settings.	
CICS UserId	*The login ID of the CICS user.*
CICS Password	*The login password of the CICS user.*

5.5.8 Configuring the JMS Clients

When a Service is linked with a Queue (or Topic), the Enterprise Designer adds a JMS properties handle between the two (see **Figure 59 on page 97**). The JMS properties facilitate the transfer and, if necessary, translation of data within the eGate system. JMS configuration properties must be configured in both the Connectivity Map and the Environment Explorer.

For more information on JMS configuration parameters see the *Sun SeeBeyond eGate™ Integrator User's Guide*.

Modifying the JMS Connectivity Map Configuration Parameters

1. From the cmCICS_JCD_Sample Connectivity Map, double-click the JMS properties handle located between the **jcolCICS1** and **Topic.**
2. The **Properties Editor** appears, containing the default environment-configuration properties. Click **OK** to keep the default settings.
3. Double-click the JMS properties handle located between **Topic** and the **jcolCICS2** Collaboration**.**
4. The **Parameters** dialog box appears, containing the default configuration parameters. Click **OK** to keep the default settings.

Modifying the JMS Environment Explorer Configuration Parameters

5. From the **Environment Explorer** tree, right-click the JMS IQ Manager (**SBJmsIQMgr1**) found under the Logical Host, and select **Properties**. The **Properties Editor** opens to the JMS Environment-configuration properties.
6. Modify the JMS Environment-configuration for your specific system and click **OK**.

5.5.9 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment Profiles are created using the Deployment Editor.

1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjCICS_JCD_Sample**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpCICS_JCD_Sample**). Make sure that the selected Environment is **envCICS_JCD_Sample**. Click **OK**.
3. Click the **Automap** icon as displayed in Figure 60. The Project's components are automatically mapped to their system window as seen in **Figure 61 on page 100**.

Figure 60 Deployment Profile - Automap

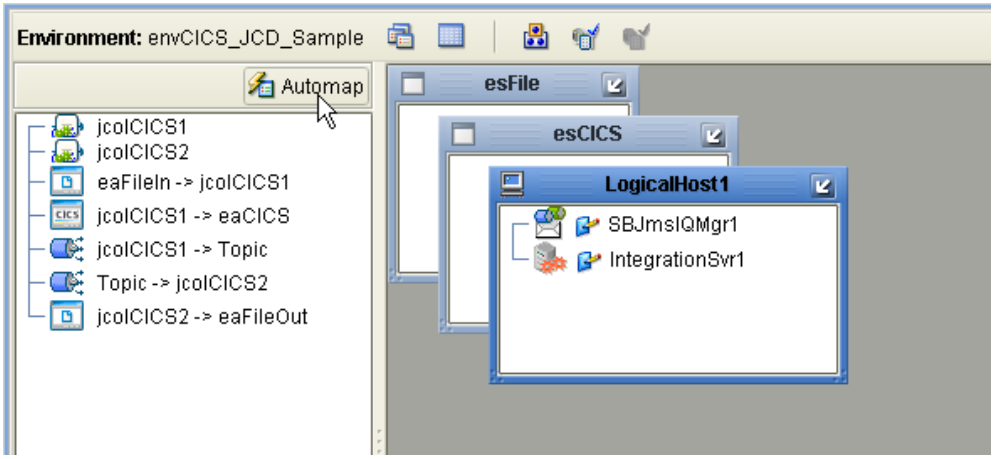
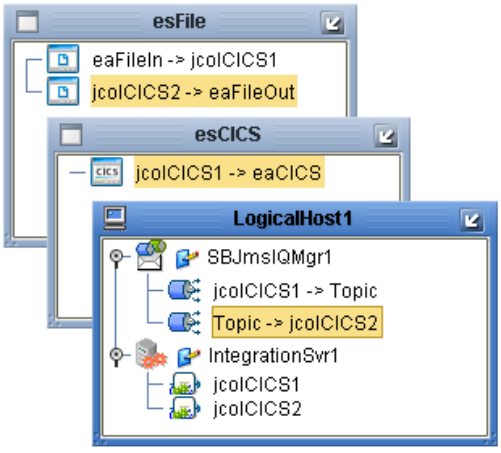


Figure 61 Deployment Profile



4. Save your changes to the Repository.

5.5.10 Creating and Starting the Domain

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

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.
2. Double-click the **domainmgr.bat** file. The **Domain Manager** appears.
3. If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
4. If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.
5. Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

For more information about creating and managing domains see the *Sun SeeBeyond eGate Integrator System Administration Guide*.

5.5.11 Building and Deploying the Project

The Build process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed.

5.5.12 Running the Sample

To run your deployed sample Project do the following

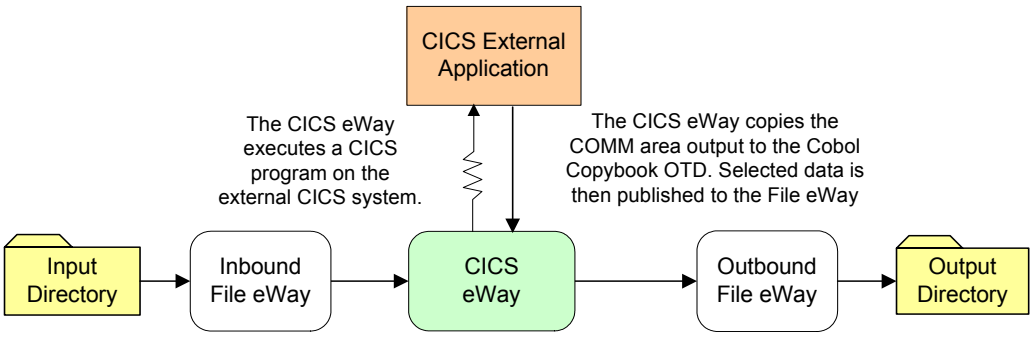
1. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.
2. From your output directory, verify the output data.

5.6 The CICS COBOL Sample Project

The CICS eWay Project, **prjCICS_COBOL_JCD_Sample**, demonstrates the following:

- The Inbound File eWay subscribes to an external directory, receives the data, and publishes the data to the CICS eWay.
- The CICS eWay, triggered by the data from the File eWay, does the following:
  - Executes a CICS program on the external CICS system (mainframe)
  - Reads the program's output (the contents of its COMM area)
  - Copies the output into a Cobol Copybook OTD that is generated from the same data area description
  - The OTD accessors selectively extract the value of select fields
- This data is published to the outbound File eWay, which publishes the data to an external directory.

Figure 62 prjCICS_COBOL_JCD_Sample Project



The **prjCICS_COBOL_JCD_Sample** is available to import from the Installation CD-ROM package (see **"Importing a Sample Project" on page 83** for more information). The following pages review how the prjCICS_COBOL_JCD_Sample Project is created.

5.6.1 Creating a Project

The first step is to create a new Project in eGate Enterprise Designer.

1. Start the Enterprise Designer.
2. From the Enterprise Explorer's Project Explorer tab, right-click the Repository and select **New Project**. A new Project (Project1) appears on the Project Explorer tree.
3. Rename the Project (for this sample, **prjCICS_COBOL_JCD_Sample**).

5.6.2 Creating a Connectivity Map

The Connectivity Map provides a canvas for assembling and configuring your Project's components.

1. From the Project Explorer tree, right-click the new Project (prjCICS_COBOL_JCD_Sample) and select **New > Connectivity Map** from the shortcut menu.
2. The New Connectivity Map appears and a node for the Connectivity Map is added under the Project on the Project Explorer tree labeled **CMap1**. Rename the **CMap1** Connectivity Map to **cmCICS_COBOL_JCD**.

Selecting the External Applications

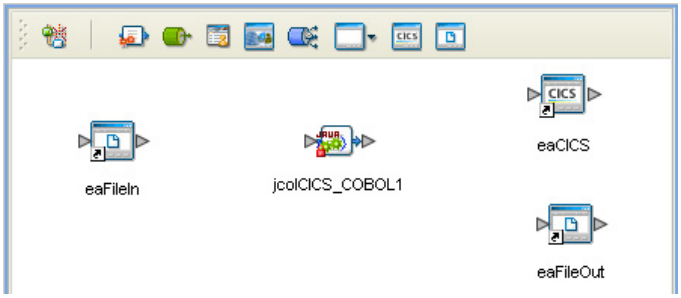
1. Click the **External Application** icon on the Connectivity Map toolbar.
2. Select the External Systems needed for your Project (for this sample, **CICS** and **File**). Icons representing these External Systems are added to the toolbar.

Populating the Connectivity Map

Add the Project components to the Connectivity Map by dragging the icons from the toolbar to the canvas.

1. For the **prjCICS_COBOL_JCD_Sample** Project, drag the following components onto the Connectivity Map canvas as displayed in Figure 63:
   - **File** External System (2)
   - Service
   - **CICS** External System

Figure 63 Connectivity Map with Components



2. Rename the Connectivity Map components as follows:
   - File1 to eaFileIn
   - **Service1** to **jcolCICS_COBOL1**
   - **CICS1** to **eaCICS**
   - **File2** to **eaFileOut**
3. Save your current changes to the Repository.

5.6.3 Creating an Object Type Definition

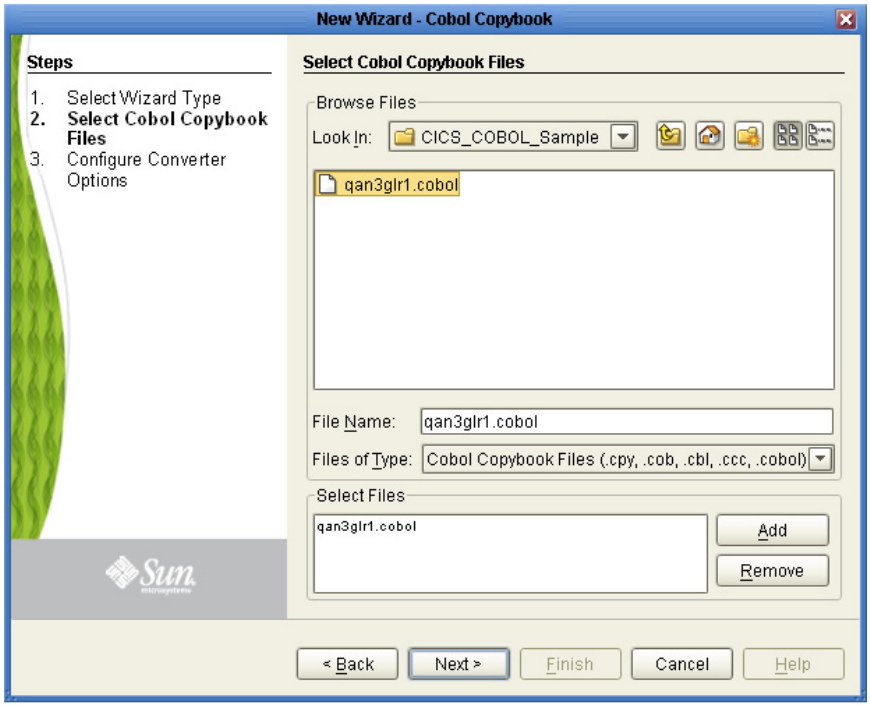
Object Type Definitions (OTDs) contain the data structures and rules that define objects. OTDs are used in the Enterprise Designer to create Collaboration Definitions for building data transformations. OTDs are created using the **Object Type Definition Wizard**.

Using the Cobol Copybook Wizard

The **Cobol Copybook Wizard** converts a .cobol file to an OTD. A sample Cobol data file, **qan3glr1.cobol** is available from the downloaded CICS eWay Samples folder. To create the Cobol Copybook OTD using the sample .cobol file do the following:

1. Copy the sample Cobol file, **qan3glr1.cobol**, from the installation CD-ROM to a temporary file.
2. From the Project Explorer tree, right-click the **prjCICS_COBOL_JCD_Sample** Project, and select **New > Object Type Definition** from the shortcut menu. The OTD Wizard Selection page appears. Select the **Cobol Copybook Wizard** and click **Next**.
3. From the **Select Cobol Copybook Files** page of the Wizard, navigate to the directory that contains the sample Cobol file. The sample file is now visible in the Available Files field (see Figure 64).

Figure 64 Cobol Copybook Wizard - Select Cobol Files



4. Select the sample Cobol file and click the **Add** button. The file is added to the **Selected Files** field. Click **Next**.

5 From the **Configure Converter Options** page, leave both converter options checked and click **Finish**.

6 The OTD Editor appears displaying the new OTD, and the OTD is added to the Project Explorer tree as **qan3glr1**.

5.6.4 Creating a Collaboration Definition (Java)

The next step is to create the **jcdCICS_COBOL** Collaboration using the Collaboration Definition Wizard (Java). Once the Collaboration Definition has been created, the Business Rules of the Collaboration can be written using the Collaboration Editor (Java).

Creating the jcdCICS_COBOL Collaboration

1 From the Project Explorer, right-click the **prjCICS_COBOL_JCD_Sample** Project and select **New** > **Collaboration Definition (Java)** from the shortcut menu. The **Collaboration Definition Wizard (Java)** appears.

2 Enter a Collaboration Definition name (for this sample **jcdCICS_COBOL**) and click **Next**.

3 For Step 2 of the Wizard, from the Web Services Interfaces selection window, double-click **Sun SeeBeyond** > **eWays** > **File** > **FileClient** > **receive**. The File Name field now displays **receive.** Click **Next**.

4 For Step 3 of the Wizard, from the Select OTDs selection window, double-click **Sun SeeBeyond** > **eWays** > **CICSeWay** > **CICS_eWay**. The **CICS_eWay** OTD is added to the Selected OTDs field.

5 Click the **Up One Level** button to return to the Repository directory. Double-click **Sun SeeBeyond** > **eWays** > **File** > **FileClient**. The **FileClient** OTD is added to the Selected OTDs field.

6 Click the **Up One Level** button to return to the Repository directory. Double-click **prjCICS_COBOL_JCD_Sample** > **qan3glr1**. The **qan3glr1** OTD is added to the Selected OTDs field.

7 Click **Finish**. The Collaboration Editor appears containing the to the new Collaboration. You can now write the Collaboration's

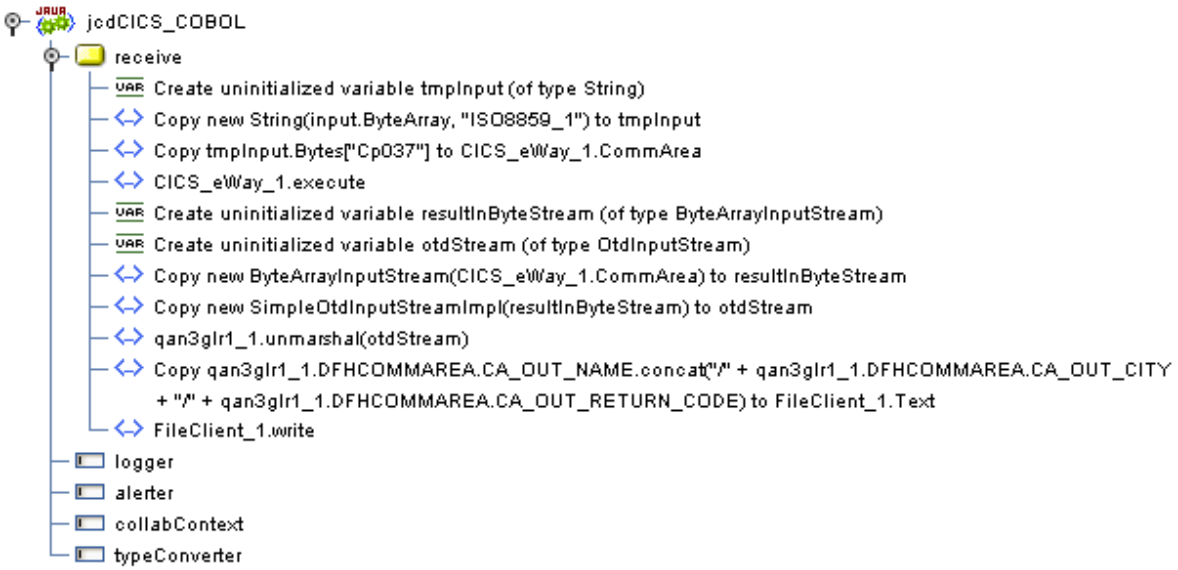
5.6.5 The jcdCICS_COBOL Business Rules

The the **prjCICS_COBOL_JCD_Sample** Project uses one Collaboration created in the previous section, **jcdCICS_COBOL**.

jcdCICS_COBOL Business Rules

The jcdCICS_COBOL Collaboration contains the Business Rule displayed in **on page 106**. These rules are created using the Collaboration Editor (Java).

Figure 65 jcdCICS_COBOL Collaboration Business Rules



Note: *Wrapped Business Rule code is for display purposes only.*

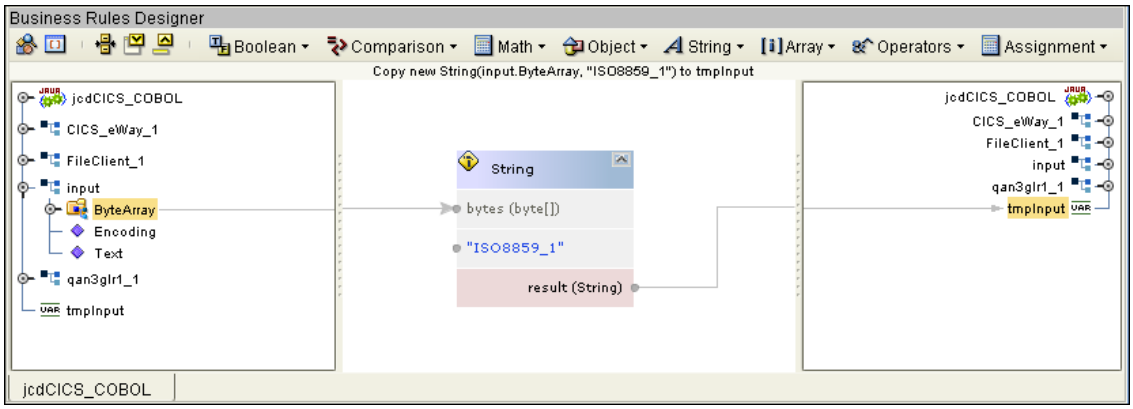
To create the jcdCICS_COBOL Business Rules do the following:

1. From the Project Explorer tree, double-click **jcdCICS_COBOL** to open the Collaboration Editor to the Collaboration.
2. Create the **Create uninitialized variable tmpInput (of type String)** rule:
   - A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears.
   - B From the Create Variable dialog box, enter **tmpInput** as the Name, and for Type, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears.
   - C From the **Class Browser** dialog box, select **String** in the All Classes field, and **String(byte[] bytes)** in the String field. Click **Select**. From the **Create Variable** dialog box, click **OK**.
3. Create the **Copy new String(input.ByteArray, "ISO8859_1") to tmpInput** rule:
   - A From the Business Rules Designer toolbar, click the **Class Browser** icon. The **Class Browser** dialog box appears.
   - B From the Class Browser dialog box, select **String** under **All Classes**, select **String*(bytes[] bytes, String charsetName)*** under **String**, and click **Select**. The **String** method box appears.
   - C From the **String** method box, double-click **charsetName (String)** and enter **ISO8859_1** as the value.
   - D Map **ByteArray** under **input** in the left pane of the Business Rules Designer, to the **bytes (bytes[])** input node of the **String** method box. Do this by clicking on **ByteArray** in the left pane of the Business Rule Designer and dragging your

cursor to the **bytes (bytes[])** input node of the **String** method box. A visible link is created between the two nodes in the Business Rules Designer.

E Map the **result (String)** output node of the **String** method box to **tmpInput** in the right pane of the Business Rules Designer (see Figure 66).

Figure 66 Copy new String(input.ByteArray, "ISO8859_1") to tmpInput rule



4 Create the **Copy tmpInput.Bytes["Cp037"] to CICS_eWay_1.CommArea** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right click **tmpInput** and select **Select method to call** from the shortcut menu.

C From the method selection box, select **getBytes*(String charsetName)***. The **getBytes** method box appears on the Business Rules Designer canvas.

D From the **getBytes** method box, double-click **charsetName (String)** and enter **Cp037** as the value.

E Map the **result (byte[])** output node of the **getBytes** method box to **CommArea** in the right pane of the Business Rules Designer.

5 Create the **CICS_eWay_1.execute** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right click **CICS_eWay_1** and select **Select method to call** from the shortcut menu.

C From the method selection box, select **execute()**. The **execute** method box appears on the Business Rules Designer canvas.

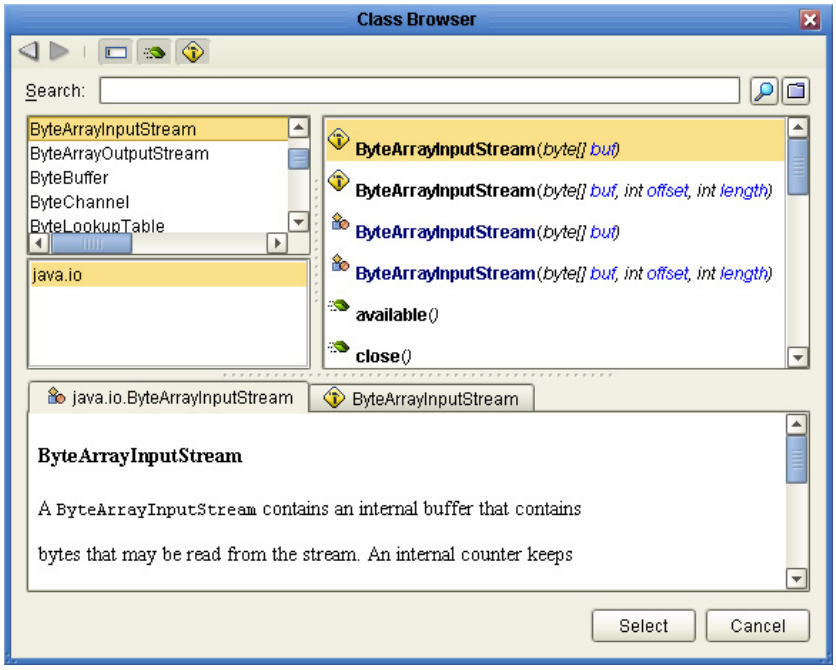
6 Create the (variable) **Create uninitialized variable resultInByteStream (of type ByteArrayInputStream)** rule:

A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears. Enter **resultInByteStream** as the Variable Name, and for Type, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears.

B From the **Class Browser** dialog box, select **ByteArrayInputSteam** in the **All Classes** field, select **ByteArrayInputSteam(byte[] buf)** in the

ByteArrayInputSteam field (as displayed in Figure 67), and click **Select**. From the **Create Variable** dialog box, click **OK**.

Figure 67 Class Browser



7 Create the (variable) **Create uninitialized variable otdStream (of type OtdInputStream)** rule:

A From the Business Rules toolbar, click the **Local Variable** icon. The **Create Variable** dialog box appears. Enter **otdStream** as the Variable Name, and for Type, select **Class** and click the ellipsis (...) button. The **Class Browser** dialog box appears.

B From the **Class Browser** dialog box, select **OtdInputStream** in the **All Classes** field, select **OtdInputStream()** in the **OtdInputStream** field, and click **Select**. From the **Create Variable** dialog box, click **OK**.

8 Create the **Copy new ByteArrayInputStream(CICS_eWay_1.CommArea) to resultInByteStream** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

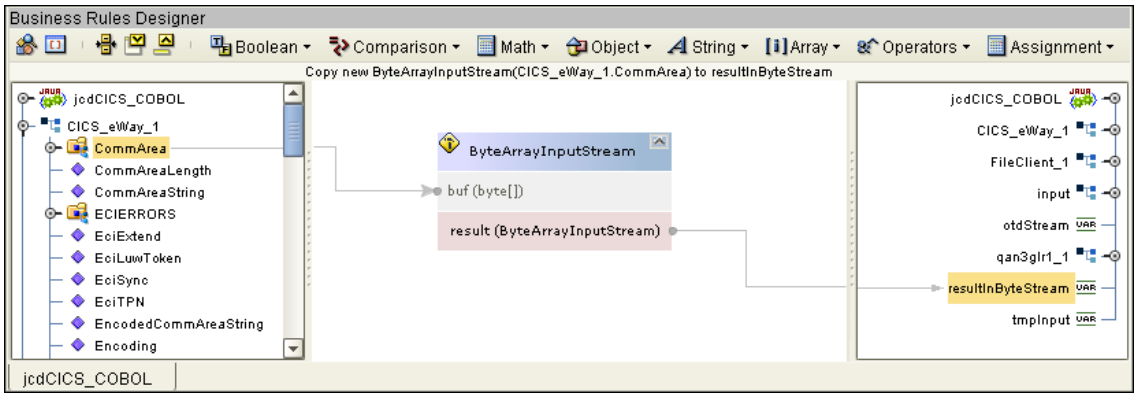
B From the Business Rules Designer toolbar, click the **Class Browser** icon. The Class Browser dialog box appears.

C From the **Class Browser** dialog box, select **ByteArrayInputSteam** in the **All Classes** field, select **ByteArrayInputSteam(byte[] buf)** in the **ByteArrayInputSteam** field (as displayed in Figure 67), and click **Select**. The **ByteArrayInputStream** method box appears.

D Map **CommArea**, under **CICS_eWay_1** in the left pane of the Business Rules Designer, to the **buf (byte[])** input node of the **ByteArrayInputStream** method box.

E Map the **result (ByteArrayInputStream)** output node of the String method box to **resultInByteStream** in the right pane of the Business Rules Designer (See Figure 68).

Figure 68 Copy new ByteArrayInputStream to resultInByteStream rule



9 Create the **Copy new SimpleOtdInputStreamImpl(resultInByteStream) to otdStream** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

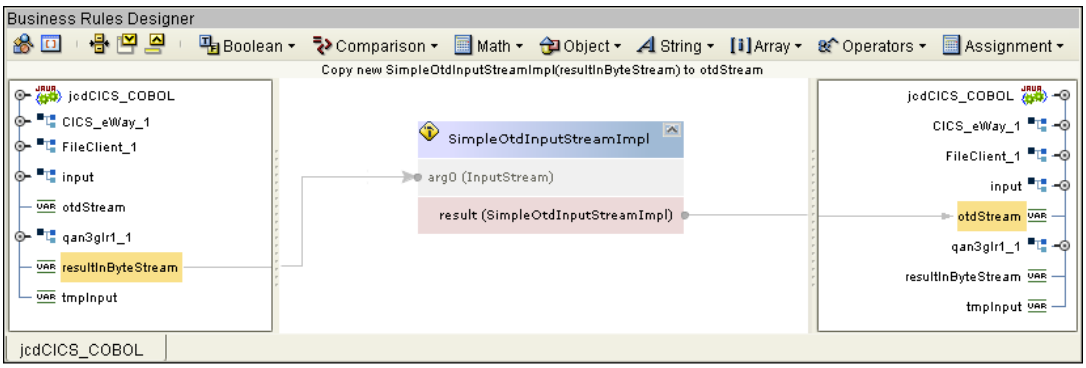
B From the Business Rules Designer toolbar, click the **Class Browser** icon. The Class Browser dialog box appears.

C From the **Class Browser** dialog box, select **SimpleOtdInputSteamImpl** in the **All Classes** field, select **SimpleOtdInputSteamImpl(java.io.InputStream arg0)** in the **SimpleOtdInputSteamImpl** field, and click **Select**. The **SimpleOtdInputSteamImpl** method box appears.

D Map **resultInByteStream** in the left pane of the Business Rules Designer, to the **arg0 (InputStream)** input node of the **SimpleOtdInputSteamImpl** method box.

E Map the **result (SimpleOtdInputSteamImpl)** output node of the String method box to **otdStream** in the right pane of the Business Rules Designer (see Figure 69).

Figure 69 Copy new SimpleOtdInputStreamImpl to otdStream rule



10 Create the **qan3glr1_1.unmarshal(otdStream)** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right click **qan3glr1_1** and select **Select method to call** from the shortcut menu.

C From the method selection box, select **unmarshal(com.stc.otd.runtime.OtdInputStream in)**. The **unmarshal** method box appears on the Business Rules Designer canvas.

D Map **qan3glr1_1** in the left pane of the Business Rules Designer, to the **qan3glr1_Tree** input node of the **unmarshal** method box.

11 Create the **Copy qan3glr1_1.DFHCOMMAREA.CA_OUT_NAME.concat("/" + qan3glr1_1.DFHCOMMAREA.CA_OUT_CITY + "/" + qan3glr1_1.DFHCOMMAREA.CA_OUT_RETURN_CODE) to FileClient_1.Text** rule:

A From the Business Rules Designer's String menu, select Concat. The Concat method box appears on the Business Rules Designer canvas.

B Map **CA-OUT-NAME,** under qan3glr1_1 > DFHCOMMAREA in the left pane of the Business Rules Designer, to the String input node of the Concat box.

C Double-click the subsequent String field (the second input node of the Concat method box) and change the value from **str(String)** to **/**.

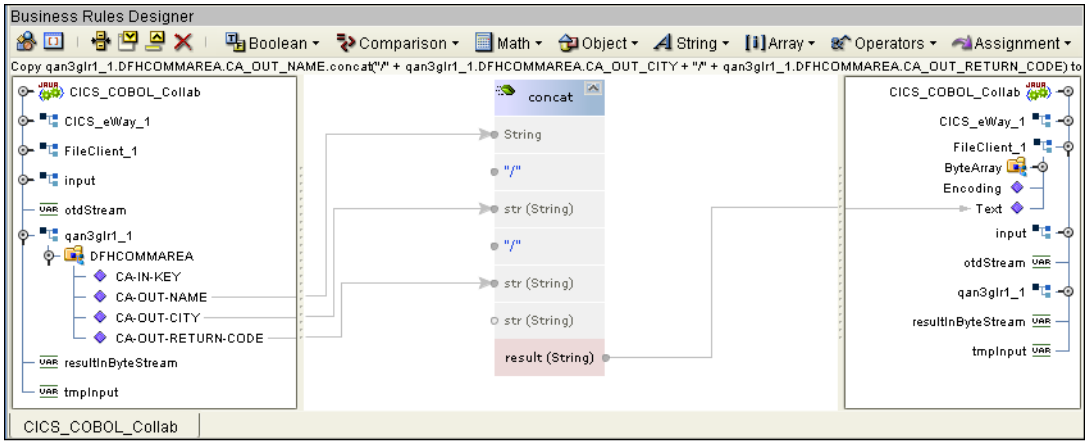
D Map **CA-OUT-CITY,** under qan3glr1_1 > DFHCOMMAREA in the left pane of the Business Rules Designer, to the third String input node of the Concat method box.

E As in step C, double-click the subsequent String field (the forth input node of the Concat method box) and change the value from **str(String)** to **/**.

F Map **CA-OUT-RETURN-CODE,** under qan3glr1_1 > DFHCOMMAREA in the left pane of the Business Rules Designer, to the fifth String input node of the Concat method box.

G Map **result(String)** output node of the Concat method box to Text under FileClient_1 in the right pane of the Business Rules Designer (see Figure 70).

Figure 70 Copy qan3glr1_1.DFHCOMMAREA.CA_OUT_NAME.concat to FileClient_1.Text



12 Create the **FileClient_1.write** rule:

A From the Business Rules toolbar, click the **rule** icon to add a new rule.

B From the left pane of the Business Rules Designer, right click **FileClient_1** and click **Select method to call** from the shortcut menu.

C From the method selection box, double-click **write()**. The **write** method box appears.

13 From the editor's toolbar, click **Validate** to check the Collaboration for errors.

14 Save your current changes to the Repository.

Note: *See the Sun SeeBeyond eGate™ Integrator User's Guide for information on editing Collaborations.*

5.6.6 Creating Collaboration Bindings

After the Collaborations have been written, the components are associated and the Collaboration Bindings are created in the Connectivity Map.

1 From the Project Explorer, double-click **cmCICS_COBOL_JCD** to display the Connectivity Map.

2 Drag and drop the **jcdCICS_COBOL** Collaboration from the Project Explorer onto **jcolCICS_COBOL1** in the **cmCICS_COBOL_JCD** Connectivity Map. If the Collaboration is successfully associated, the Collaboration's "gears" icon changes from red to green.

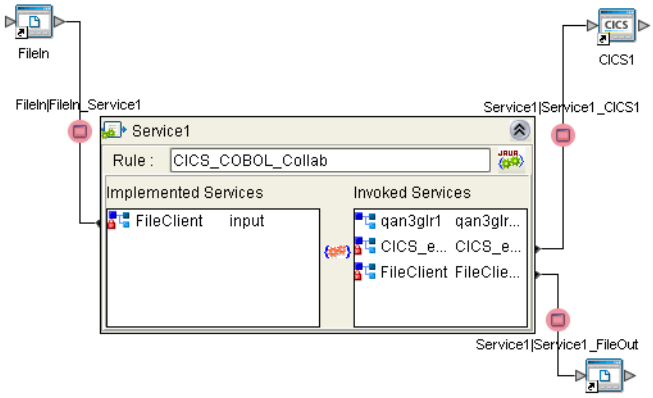
3 Double-click **jcolCICS_COBOL1**. The **jcolCICS_COBOL1** Binding dialog box appears.

4 From the **jcolCICS_COBOL1** Binding dialog box, drag **FileClient input** (under Source) to the inbound **esFileIn** External Application.

5 From the **jcolCICS_COBOL1** Binding dialog box, drag the **CICS_eWay CICS_eWay_1** OTD (under Destination) to the **esCICS** External Application.

6 From the **jcolCICS_COBOL1** Binding dialog box, drag the **FileClient FileClient_1** OTD (under Destination) to the outbound **esFileOut** External Application (see Figure 71).

Figure 71 Connectivity Map - Binding the Project's Components



7 Minimize the **jcolCICS_COBOL1** Binding dialog box and save your changes.

5.6.7 Creating an Environment

Environments include the external systems, Logical Hosts, integration servers and message servers used by a Project and contain the configuration information for these components.

1 From the Enterprise Designer's Enterprise Explorer, click the **Environment Explorer** tab.

2 Right-click the Repository and select **New Environment**. A new environment is added to the Environment Explorer tree.

3 Rename the new environment to **envCICS_COBOL_JCD**.

4 Right-click **envCICS_COBOL_JCD** and select **New > CICS External System**. Name the External System **esCICS** and click **OK**. **esCICS** is added to the Environment Editor.

5 Right-click **envCICS_COBOL_JCD** and select **New > File External System**. Name the External System **esFile** and click **OK**. **esFile** is added to the Environment Editor.

6 Right-click **envCICS_COBOL_JCD** and select **New > Logical Host**. The **LogicalHost1** box is added to the Environment.

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 Save your current changes to the repository.

5.6.8 Configuring the eWays

The **prjCICS_COBOL_JCD_Sample** Project uses three eWays, each represented in the Connectivity Map as a node between an External Application and a Collaboration.

The eWay properties are set from both the Project Explorer's Connectivity Map and the Environment Explorer. To configure the eWays do the following:

Configure the File eWays Connectivity Map Properties

1 From the Project's Connectivity Map, double-click the **eaFileIn eWay**. The **Properties Editor** opens to the **eaFileIn** eWay properties. Modify the properties for your system, including the settings in Table 23, and click **OK**.

Table 23 eaFileIn eWay properties

eaFileIn (inbound) eWay Connection Parameters	
Input file name	Input*.txt

2 In the same way, modify the eaFileOut eWay properties for your system, including the settings in **Table 24 on page 113**, and click **OK**.

Table 24 eaFileOut eWay properties

eaFileOut (inbound) eWay Connection Parameters	
Output file name	Output*.txt

Configure the File eWay Environment Explorer Properties

1 From the **Environment Explorer** tree, right-click the File External System (**esFile** in this sample), and select **Properties**. The Properties Editor opens to the File eWay Environment configuration.

2 Modify the File eWay Environment configuration properties for your system, including the settings in Table 25, and click **OK**.

Table 25 File eWay Environment Explorer Properties

File eWay Environment Explorer Properties	
Inbound File eWay - Set as directed, otherwise use the default settings	
Parameter Settings > Directory	*C:/temp or the input directory of your choice*
Outbound File eWay - Set as directed, otherwise use the default settings.	
Parameter Settings > Directory	*C:/temp or the output directory of your choice*

Configuring the CICS eWay

Configure the CICS eWay properties from both the Project Explorer and Environment Explorer. For more information on the CICS eWay properties, see **"Creating and Configuring the CICS eWay" on page 47**.

Configure the CICS eWay Connectivity Map Properties

1 From the **Connectivity Map**, double-click the **CICS eWay**. The **Properties Editor** opens to the CICS eWay Connectivity Map configuration properties.

2 Modify the CICS eWay Connectivity Map properties for your system, including the settings in Table 26, and click **OK**.

Table 26 CICS eWay Connectivity Map Properties

CICS eWay Connectivity Map Properties	
Connector - Set as directed, otherwise use the default settings	
Connection Transport	*Sun SeeBeyond CICS Listener or CICS Transaction Gateway*
Class	com.stc.eways.cics.CicsClientConnector
CICS Client - Set as directed, otherwise use the default settings.	
CICS Program	*QAN3GLR1*

Configure the CICS eWay Environment Explorer Properties

1. From the **Environment Explorer** tree, right-click the CICS External System (**esCICS** in this sample), and select **Properties**. The Properties Editor opens to the CICS eWay Environment configuration.
2. Modify the CICS eWay Environment configuration properties for your system, including the settings in Table 27, and click **OK**.

Table 27 CICS eWay Environment Explorer Properties

CICS eWay Environment Explorer Properties	
Sun SeeBeyond CICS Listener (if you use Sun SeeBeyond CICS Listener as the connection transport) - Set as directed, otherwise use the default settings	
Host	*The name of the mainframe host*
Port	*The TCP/IP port where Sun SeeBeyond CICS Listener is listening*
CICS Gateway (if you use CTG as the connection transport) - Set as directed, otherwise use the default settings	
URL	*The remote or local Gateway node name or IP address*
Port	*The TCP/IP port where CTG is running*
CICS Client Set as directed, otherwise use the default settings.	
CICS UserId	*The login ID of the CICS user.*
CICS Password	*The login password of the CICS user.*

5.6.9 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.

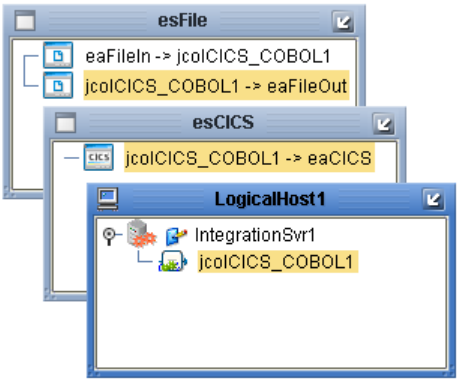
For more information on deploying a Project see the *Sun SeeBeyond Java™ Composite Application Platform Suite Deployment Guide*.

5.6.10 Creating the Deployment Profile

A Deployment Profile is used to assign Collaborations and message destinations to the integration server and message server. Deployment profiles are created using the Deployment Editor.

1. From the Enterprise Explorer's Project Explorer, right-click the Project (**prjCICS_COBOL_JCD_Sample**) and select **New** > **Deployment Profile**.
2. Enter a name for the Deployment Profile (for this sample **dpCICS_COBOL_Sample**). Select **envCICS_COBOL_JCD** as the Environment and click **OK**.
3. From the Deployment Editor toolbar, click the **Automap** icon. The Project's components are automatically mapped to their system windows (see Figure 72).

Figure 72 Deployment Profile



4. Save your current changes to the repository.

5.6.11 Creating and Starting the Domain

To deploy your Project, you must first create a domain. A domain is an instance of a Logical Host. After the domain is created, the Project is built and then deployed.

Create and Start the Domain

1. Navigate to your **<*JavaCAPS51*>\logicalhost** directory (where <*JavaCAPS51*> is the location of your Sun Java Composite Application Platform Suite installation.
2. Double-click the **domainmgr.bat** file. The **Domain Manager** appears.
3. If you have already created a domain, select your domain in the Domain Manager and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.
4. If there are no existing domains, a dialog box indicates that you can create a domain now. Click **Yes**. The **Create Domain** dialog box appears.
5. Make any necessary changes to the **Create Domain** dialog box and click **Create**. The new domain is added to the Domain Manager. Select the domain and click the **Start an Existing Domain** button. Once your domain is started, a green check mark indicates that the domain is running.

For more information about creating and managing domains see the *Sun SeeBeyond eGate™ Integrator System Administration Guide*.

5.6.12 Building and Deploying the Project

The Build process compiles and validates the Project's Java files and creates the Project EAR file.

Build the Project

1. From the Deployment Editor toolbar, click the **Build** icon.
2. If there are any validation errors, a **Validation Errors** pane will appear at the bottom of the Deployment Editor and displays information regarding the errors. Make any necessary corrections and click **Build** again.
3. After the Build has succeeded you are ready to deploy your Project.

Deploy the Project

1. From the Deployment Editor toolbar, click the **Deploy** icon. Click **Yes** when the **Deploy** prompt appears.
2. A message appears when the project is successfully deployed. You can now test your sample.

Note: *Projects can also be deployed from the Enterprise Manager. For more information about using the Enterprise Manager to deploy, monitor, and manage your projects, see the Sun SeeBeyond eGate™ Integrator System Administration Guide.*

5.6.13 Running the Sample

To run your deployed sample Project do the following

1. From your configured input directory, paste (or rename) the sample input file to trigger the eWay.
2. From your output directory, verify the output data.

Chapter 6

Java Methods

A number of Java methods have been exposed to make it easier to set information in the eWay's OTD Editor and to get information from it. These methods are contained in the CicsClient Class.

What's in This Chapter

6.1 The CicsClient and PackedDecimal Classes

The CicsClient and PackedDecimal classes represent an OTD through which a Collaboration can invoke transaction programs on a CICS server. Nodes and methods are exposed so that the Collaboration can conveniently prepare a request for a CICS program, invoke the program, and get result from the program. The OTDs expose a number of nodes that correspond to eWay configuration parameters, to provide the means to dynamically override the preset eWay settings at runtime.

CICS Javadoc

The Javadoc is uploaded with the eWay's documentation file (**CICSeWayDocs.sar**) and downloaded from the **Documentation** tab of the Sun Java 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.

6.1.1 ECIRequest Class

The ECIRequest class contains the details of an External Call Interface (ECI) request to the CICS Transaction Gateway. The **ECIRequest** object contains all the necessary data to invoke a CICS program through CICS Transaction Gateway. To make the ECIRequest class available, you must first import the IBM **ctgclient.jar** and **ctgserver.jar** files into your Project. To do this:

1. From the Project Explorer, right-click your Project and select **New > File** from the shortcut menu. The **Import Files** dialog box appears.
2. Locate and select the **ctgclient.jar** and **ctgserver.jar** files and click **Import**. These files now appear in your Project Explorer tree.

3 To import these jar files into your Collaboration, from the Collaboration Editor's toolbar, click the **Import JAR File** icon. The **Add/Remove Jar Files** dialog box appears.

4 Click **Add**. Locate and select the **ctgclient.jar** and **ctgserver.jar** from your Project file and click **Close**.

5 After you have imported the two JAR files you can create a variable of type **ECIRequest** by using the **Field** or **Local Variable** options on the Business Rules toolbar. To initialize this variable, use the Class Browser option, located on the Business Rules Designer toolbar.

6.2 Packed Decimal Java Helper Methods

The Java helper methods for the toPackedDecimal class. These methods are not exposed in the Collaboration but are available for use. For example, to use the **CopyTo** method to obtain a Hex value in a byte array you would use the following code:

```
MyPacked.toPackedDecimal("327.00") ;
    System.out.println(  " Decimal value is : "  + MyPacked.toString() ) ;
    byte[] work_buf = new byte[7];
    MyPacked.CopyTo(work_buf, 7);
    for ( int  Ii = 0 ; Ii < 7 ; Ii++ )
    {
      int Ib = (int) new Byte(work_buf[Ii]).intValue();
      if (Ib < 16)
      {
        System.out.println("  Byte" + Ii  + " Hexvalue = 0" + Integer.toHexString(Ib));
      }
      else
      {
        System.out.println("  Byte" + Ii  + " Hexvalue = " + Integer.toHexString(Ib));
      }
    }
```

This produces the following output:

```
Decimal value is : 327.00
Byte0 Hexvalue = 00
Byte1 Hexvalue = 00
Byte2 Hexvalue = 00
Byte3 Hexvalue = 00
Byte4 Hexvalue = 32
Byte5 Hexvalue = 70
Byte6 Hexvalue = 0c
```

These methods are described in detail on the following pages:

- **ContainerExists** on page 119
- **CopyBack** on page 119
- **CopyTo** on page 119
- **GiveElem** on page 120
- **SetElem** on page 120

ContainerExists

Description

Checks to see if contents of a packed decimal are available.

Syntax

```
public com.stc.eways.cics.PackedDecimal boolean ContainerExists()
```

Parameters

None.

Return Values

Boolean

Throws

None.

CopyBack

Description

Builds a packed decimal from the payload in a byte array. The byte array must contain a valid packed decimal number.

Syntax

```
public com.stc.eways.cics.PackedDecimal void CopyBack(byte Origin[],
int Size)
```

Parameters

Name	Type	Description
Origin[]	byte	Payload containing a valid packed decimal number.
Size	int	Size of the payload that contains the packed decimal number.

Return Values

None.

Throws

None.

CopyTo

Description

Copies a packed decimal number to a byte array.

Syntax

```
public com.stc.eways.cics.PackedDecimal void CopyTo(byte Dest[], int
Size)
```

Parameters

Name	Type	Description
Dest[]	byte	A byte array that the packed decimal number will be copied to.
Size	int	Size of the payload that contains the packed decimal number.

Return Values

None.

Throws

None.

GiveElem

Description

Returns a specified byte of a packed decimal number.

Syntax

```
public com.stc.eways.cics.PackedDecimal int GiveElem(int Ii)
```

Parameters

Name	Type	Description
Ii	int	Position of the byte to be returned.

Return Values

int - Returns the value of the position.

Throws

None.

SetElem

Description

Sets the value of a specified byte in a packed decimal number.

Syntax

```
public com.stc.eways.cics.PackedDecimal void SetElem(int Ii, byte
Value)
```

Parameters

Name	Type	Description
Ii	int	Position of the byte to be set.
Value	byte	Value to set.

Return Values

None.

Throws

None.

Index

R

S

T

U