SeeBeyond ICAN Suite

e*Way Intelligent Adapter for Portal Infranet

Release 5.0.5 for Schema Run-time Environment (SRE)

SEEBEYOND®

Version 20050406030914.

Contents

Chapter 1

Introduction

This chapter includes a brief description of SeeBeyond™ Technology Corporation's (SeeBeyond™) e*Way Intelligent Adapter for Portal Infranet (Portal e*Way), as well as system requirements for using the e*Way.

1.1 Portal e*Way: Introduction

Portal Software provides customer management and billing software for the internet and emerging, next generation telecommunication services. Infranet software links Internet/IP services, subscribers, and revenues, enabling service providers to provide price and provision services effectively managing customer usage and billing in real time.

An Infranet server often needs to be integrated seamlessly with existing services, legacy applications and custom built applications. The Portal e*Way provides this interface to Portal's Infranet software. The e*Way utilizes the Portal APIs enabling e*Gate and the e*Way to access and update information contained in the Infranet database, as well as subscribe to Infranet events.

The e*Way supports two messaging modes: inbound to e*Gate using publication via Portal's EAI Manager, and outbound from e*Gate following the Request/Reply scenarios via OPCODEs.

The OPCODE builder for e*Gate Event Type Definitions (ETDs) creates the appropriate ETD with elements for each of the specific parameters and methods required to perform PCM API calls.

For the EAI Manager, two possibilities exist, XML using the standard SeeBeyond DTD builder, or an FLIST builder given a valid FLIST creates a valid ETD that can then be parsed.

1.2 Intended Reader

The reader of this guide is presumed to be a developer or System Administrator with responsibility for maintaining the e*Gate system; to have high-level knowledge of Windows operations and administration; to be thoroughly familiar with Portal Infranet and with Windows-style GUI operations.

1.3 Functional Description

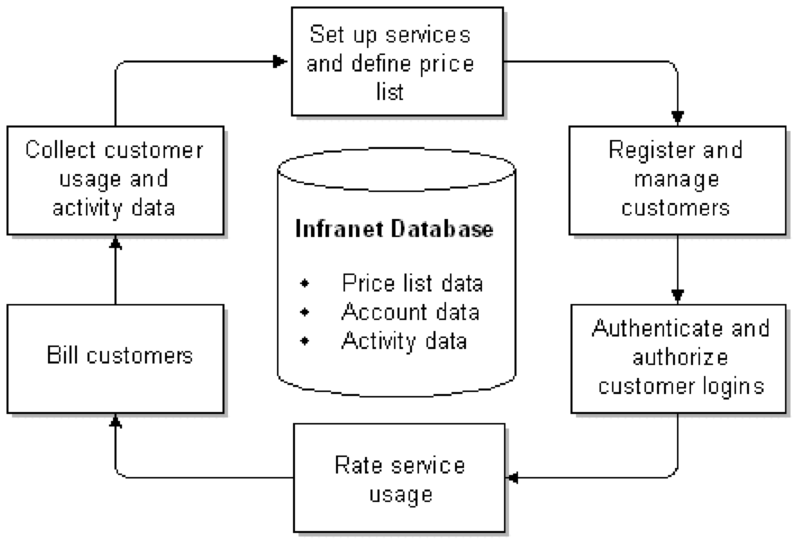
The Portal e*Way is comprised of two components, the first, referred to as the Portal Client e*Way, manages OPCODEs, the second, referred to as the Portal Subscriber e*Way handles Events generated by the EAI Manager. These e*Ways allow the user to exchange data with Infranet through the configuration parameters defined by the user.

1.3.1 Feature Summary

Companies that utilize Infranet provide online services such as Internet access, e-mail and telephone services. Infranet integrates with these services to provide customer registration, service provisioning, authentication and authorization, customer activity tracking, billing and account management, along with data collection and report administration.

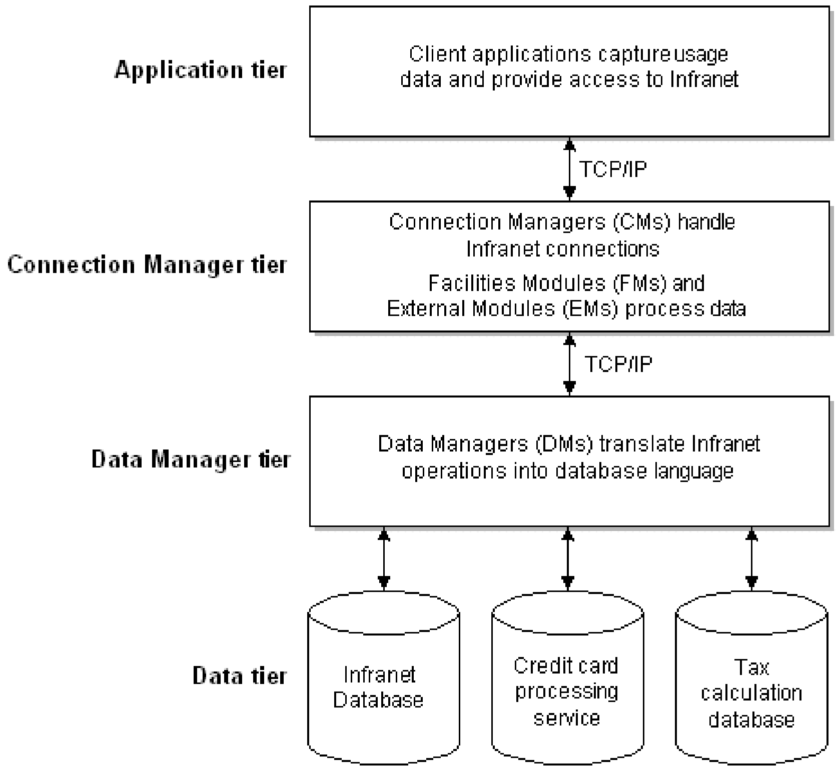
The figure below show an example of the Infranet business cycle:

Figure 1 Infranet Business Cycle


Set up services and define price list
Register and manage customers
Authenticate and authorize customer logins
Rate service usage
Bill customers
Collect customer usage and activity data
Infranet Database
Price list data
Account data
Activity data

The Infranet system is based on a four-tier architecture:

Figure 2 Infranet System Architecture


Application tier
Client applications capture usage data and provide access to Infranet
TCP/IP
Connection Manager tier
Connection Managers (CMs) handle Infranet connections
Facilities Modules (FMs) and External Modules (EMs) process data
TCP/IP
Data Manager tier
Data Managers (DMs) translate Infranet operations into database language
Data tier
Infranet Database
Credit card processing service
Tax calculation database

Application Tier

The Application tier consists of two types of client applications:

- Applications that obtain data based on customer usage.
- Infranet client applications that are operated by a human user.

Connection Manager Tier

The Connection Manager tier consists of Connection Managers, referred to as CMs, Facilities Modules, referred to as FMs, and External Modules, referred to as EMs. CMs provide an interface between Infranet clients and the remainder of the Infranet system. CMs include FMs that process the data obtained via the Infranet client. EMs perform functions similar to that of FMs, but must be started separately as a service or process.

Data Manager Tier

The Data Manager tier consists of Data Managers, referred to as DMs, that translate requests made by the CMs into a language that the database can understand. The Infranet database language is SQL. There are also DMs for credit card processors and other data usage types.

The data tier consists of the Infranet database and other data access systems. The Infranet database stores customer account data along with business data such as price lists, general ledgers, or records of all events that occur within the Infranet system.

1.3.2 Portal Client e*Way Functionality: Sending to Portal

The Portal Client e*Way acts as a client application, requesting the Infranet server to perform tasks by making OPCODE calls. These OPCODE calls provide context management, basic object manipulation and Facilities Modules (FM) object manipulation.

OPCODE ETD Generator

The OPCODE ETD Generator creates the appropriate ETD containing the elements for each of the specific parameters and methods required to perform OPCODE calls.

An OPCODE represents an Infranet method that allows customers to query or change Infranet data. An OPCODE requires an input Portal FLIST, and returns an output FLIST. The OPCODE builder parses an OPCODE specification (HTML) into an ETD representing the OPCODE. The OPCODE input and output FLISTs are represented along with the methods needed for execution.

1.3.3 Portal Subscriber e*Way Functionality - Receiving from Portal

The Portal Subscriber e*Way listens for Events generated by Infranet. The e*Way accepts data/Events in the form of XML or FLISTS residing within the EAI framework.

Infranet to e*Gate

The Infranet EAI package can publish events to external systems such as e*Gate. The EAI Manager performs the following tasks to collect Infranet events and publish them:

- The EAI Manager uses the Infranet event notification mechanism to cache Infranet events in the Payload Generator External Module (EM).
- The Payload Generator EM generates the data required as defined in **payloadconfig.xml** of the cached complete business object to publish the business event by performing one for both of these tasks:
    - Read the fields in the incoming flists for the event
    - Access the Infranet database.
- The Payload Generator EM generates the data in XML (default) or FLIST format as specified in the configuration file, and send the generated data (payload) to the EAI Data Manager (DM) via the Connection Manager (CM).
- The EAI Data Manager publishes the payload as a business event to the external system

The Portal Subscriber e*Way is a connector DLL that is installed on the EAI Manager and a JMS e*Way connection residing on the Participating Host. The DLL sends published Events to e*Gate via JMS. Messages (Events) stored in the JMS server may be accessed in collaborations through the JMS connections.

Custom ETD Builder

The SeeBeyond Custom ETD builder can also be used to generate ETDs from XML.

Infranet FLIST Wizard

The Infranet FLIST wizard creates the appropriate ETD from an FLIST. The FLIST specifications can be found within the Portal installation or written by hand.

Infranet OPCODE Wizard

The Infranet OPCODE wizard creates the appropriate ETD from an OPCODE. The OPCODE specifications can be found within the Portal installation or written by hand.

1.3.4 Logging

In general, an e*Way relays any pertinent information as to the state, protocol position, and any conditions that are helpful to the user to understand what is taking place according to the debug level settings, and either logs the information to a file, or to notifies the Alert Agent.

Standard debug levels are set through the standard GUI.

1.3.5 Errors

Any error condition are written to the log file. The inability to write to the log file or any fatal/unrecoverable errors result in the e*Way shutting down after it sends an alert to the Alert Agent.

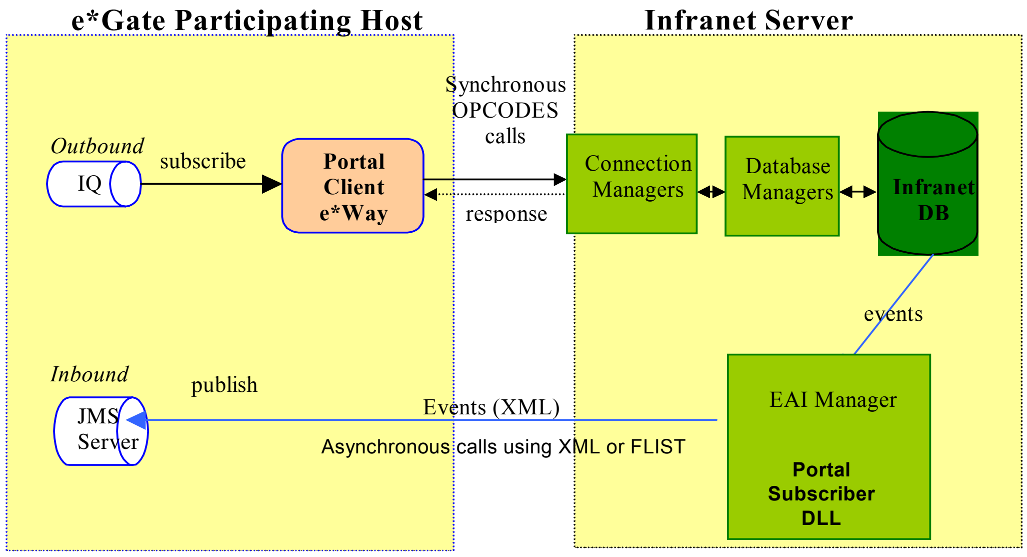
1.3.6 Alerting

Any errors that affect the operation of the e*Way preventing the successful delivery of a message cause an alert to be sent. If the alert can not be sent, the e*Way shuts down.

1.4 Architecture: Component Interrelations

The following architectural diagram illustrates the components involved in calling OPCODEs and receiving Events from Infranet.

Figure 3 Portal Infranet e*Way System Architecture



1.4.1 Components

Each section represents a component of the Portal e*Way

Portal Client e*Way

The Portal Client e*Way provides an implementation of and conforms to outbound functionality as related to e*Gate allowing for synchronous OPCODE calls.

OPCODE ETD Generator

This feature is an ETD generator that creates ETDs from the HTML specifications of Portal Infranet OPCODEs.

Portal Subscriber e*Way

The Portal Subscriber e*Way provides an implementation of and conforms to inbound functionality as related to e*Gate using the JMS Server to receive messages published from the Portal Subscriber DLL residing on the EAI Manager.

FLIST ETD Generator and Standard e*Gate ETD Generator

A generator that creates ETDs used for parsing incoming (to e*Gate) FLISTs from the EAI Manager. Since the EAI Manager can be configured to generate Events in XML format as well as FLIST format, the standard e*Gate DTD Builder can also be used to build an ETD by hand if no FLIST specification is available.

1.4.2 Protocols/APIs

TCP/IP is used as the communication protocol between the Portal Infranet client program and the e*Way. The APIs conform to the Java PCM API as defined by Portal.

1.5 Supported Operating Systems

The Portal e*Way is supported on the following platforms:

- Windows 2000 and Windows Server 2003
- HP-UX 11.0 and 11i (PA-RISC)
- Sun Solaris 8

1.6 System Requirements

To use the Portal e*Way, you need to meet the following system requirements:

- An e*Gate Participating Host.
- The amounts of disk space required on both the Participating and the Registry Host vary. See the **Readme.txt** file in the root directory of the e*Gate installation CD-ROM, for specific version information.

Note: *Additional disk space is required to process and queue the data that this e*Way processes; the amount necessary can vary based on the type and size of the data being processed, and any external applications performing the processing.*

- A TCP/IP network connection, a phone line, and so on.
- The Portal Infranet application components have specific requirements of their own; see that system's documentation for more details.

1.7 External System Requirements

To enable the e*Way to communicate correctly with Portal, you need the following:

- Portal Infranet Application Server version 6.1 or 6.5, SP2 and above
- Portal Infranet 6.1 or 6.5 documentation
- Portal Infranet SDK
- See the External System Configuration section to establish which files are required on the EAI Manager.

Chapter 2

Installation

This chapter explains procedures for installing the Portal Client and Subscriber e*Ways.

2.1 Windows Systems

2.1.1 Pre-installation

- Exit all Windows programs before running the setup program, including any anti-virus applications.
- You must have Administrator privileges to install this e*Way.

2.1.2 Installation Procedure

To install the Portal e*Way on Windows systems

1. Log in as an Administrator to the workstation on which you are installing the e*Way.
2. Insert the e*Way installation CD-ROM into the CD-ROM drive.
3. If the CD-ROM drive's Autorun feature is enabled, the setup application launches automatically; skip ahead to step 4. Otherwise, use the Windows Explorer or the Control Panel's **Add/Remove Applications** feature to launch the file **setup.exe** on the CD-ROM drive.
4. The InstallShield setup application launches. Follow the installation instructions until you come to the **Please choose the product to install** dialog box.
5. Select **e*Gate Integrator**, then click **Next**.
6. Follow the on-screen instructions until you come to the second **Please choose the product to install** dialog box.
7. Clear the check boxes for all selections except **Add-ons**, and then click **Next**.

8. Follow the on-screen instructions until you come to the **Select Components** dialog box.
9. Highlight (but do not check) **e*Ways**, and then click the **Change** button. The **SelectSub-components** dialog box appears.
10. Select the **Portal e*Way**. Click the continue button to return to the **Select Components** dialog box, then click **Next**.
11. Follow the rest of the on-screen instructions to install the Portal e*Way. Be sure to install the e*Way files in the suggested client installation directory. The installation utility detects and suggests the appropriate installation directory. **Unless you are directed to do so by SeeBeyond support personnel, do not change the suggested installation directory setting.**

Note: *Once you have installed and configured this e*Way, you must incorporate it into a schema by defining and associating the appropriate Collaborations, Collaboration Rules, IQs, and Event Types before this e*Way can perform its intended functions. For more information about any of these procedures, please see the online Help.*

*For more information about configuring e*Ways or how to use the e*Way Editor, see the **e*Gate Integrator User's Guide**.*

2.2 UNIX Systems

2.2.1 Pre-installation

You do not require root privileges to install this e*Way. Log in under the user name that you wish to own the e*Way files. Be sure that this user has sufficient privileges to create files in the e*Gate directory tree.

2.2.2 Installation Procedure

To install the Portal e*Way on a UNIX system

1. Log in on the workstation containing the CD-ROM drive, and insert the CD-ROM into the drive.
2. If necessary, mount the CD-ROM drive.
3. At the shell prompt, type

   cd /cdrom
4. Start the installation script by typing

   setup.sh
5. A menu of options will appear. Select the **Install e*Way** option. Then, follow the additional on-screen directions.

Note: *Be sure to install the e*Way files in the suggested* ***client*** *installation directory. The installation utility detects and suggests the appropriate installation directory.* ***Unless you are directed to do so by SeeBeyond support personnel, do not change the suggested "installation directory" setting.***

6. After installation is complete, exit the installation utility and launch the Schema Designer.

Note: *Once you have installed and configured this e*Way, you must incorporate it into a schema by defining and associating the appropriate Collaborations, Collaboration Rules, IQs, and Event Types before this e*Way can perform its intended functions. For more information about any of these procedures, please see the online Help system.*

*For more information about configuring e*Ways or how to use the e*Way Editor, see the* ***e*Gate Integrator User's Guide****.*

2.3 Files/Directories Created by the Installation

The Portal e*Way installation process will install the following files, see the table **"Files Created by the Installation" on page 16**, within the e*Gate directory tree. Files will be installed within the **egate\server** tree on the Participating Host and committed to the **default** schema on the Registry Host.

Table 1 Files Created by the Installation

e*Gate Directory	File(s)
classes\	stcportal.jar stcportalflist.jar stcportalOPCODE.jar stcexception.jar
bin\	stc_ewportaljms.dll
configs\portal\	portal.def
dtd\	statupdate.dtd srvcpasswd.dtd srvclogin.dtd pdtpurch.dtd pdtcanc.dtd payinfoupdt.dtd nameinfoupdt.dtd dealpurch.dtd dealcanc.dtd custcreate.dtd common.dtd billinfoupdt.dtd
etd\	portal.ctl OPCODEwizard.ctl flistwizard.ctl
ThirdParty\antlr-2.7.1\classes\	antlrall.jar
ThirdParty\jdom-b7\classes\	jdom.jar
ThirdParty\xml\Apache\classes\	xerces.jar

2.3.1 Additional Required Files

In addition to the files created by the installation above, there are files that must be copied from Infranet to the several locations.

Upon installing the InfranetSDK, locate and copy **pcm.jar** and **pcmext.jar**, which are located in the following directory as relative to the root directory (i.e., :\Program Files):

```
..\Portal Software\InfranetSDK\jars
```

Manually install them into the following directories:

```
Server\registry\repository\default\ThirdParty\portal\classes
```

```
client\ThirdParty\portal\classes
```

If not present, the OPCODE wizard and FLIST wizard can provide an error message that the Portal software is not available.

0.1 Installing the SeeBeyond EAI Manager Connector DLL

In addition to the below configuration steps required on the machine on which the EAI Manager is running, the Portal Connector Shared Library must be installed. To do this, uncompress the corresponding platform file (for example, win32.taz) from the Installation CD **setup\addons\ewportal** directory to **%infranetrootdir%/sys/dm_eai**. The .**dll** files must be in the same directory, not under bin as stored in the .**taz** file.

0.1.1 Objective

Events that occur in Infranet may be published to e*Gate via Infranet's Application Manager Interface (EAI).

Portal provides EAI Framework that can be installed on a base Infranet system. The EAI Framework provides a link between Infranet and external systems. It is used to link Infranet with e*Gate.

Installing on EAI running on Windows

- Windows systems
- Infranet 6.1 or 6.5
- Infranet EAI 6.1 or 6.5
- Infranet Administrator 6.1 or 6.5

Note: *The steps below use* ***C:\Infranet*** *as the root directory location where Infranet is installed. You must use the applicable root directory for all steps.*

Solution Steps

Follow steps 1 -2 if you don't have EAI Framework Manager installed already. Otherwise skip to step 3.

1. Download the 6.1 or 6.5 EAI Framework software from the Portal web site (**www.pin.com**). See your Strategic Alliance Manager for a **www.pin.com** login if you don't have one.
2. Stop all Infranet services and unzip the **6.1_EAI_Framework.zip** file and invoke the **setup.exe** program. Follow the instructions, selecting the default option for all entries.
3. Make the following changes to the **C:\Infranet\sys\cm\pin.conf** file:

```
- cm cm_loglevel 3
- cm dm_pointer 0.0.9.1 ip hostname 11970  # dm_eai
- cm em_pointer publish ip hostname 11930
- fm_publish enable_publish 1
```

Note: *"hostname" is the hostname of the system where the Infranet Data Manager is running.*

4 Make the following changes to the **C:\Infranet\sys\dm_eai\pin.conf** file:

```
- dm loglevel 3
- dm dm_name hostname
```

Note: *"hostname" is the hostname of the system where the Infranet EAI Data Manager is running.*

5 Add the following changes to the **C:\Infranet\sys\dm_eai\pin.conf** file::

```
#EGate
#- dm egatejmstopic PortalEventTopic
  - dm egateeventouttype FLIST
  - dm egatejmshots <hostname>
  - dm egatejmsport <portnumber>
  - dm egatejmspriority <priority level>
  - dm egatejmstimetolive <expirationtime>
  - dm egatejmspubuuid <JMS publisher uuid>
```

Note: *egateeventouttype can be either FLIST or XML, egatejmshost is the e*Gate JMS server host, egatejmsport is the e*Gate JMS server port.*

6 In the **C:\Infranet\sys\dm_eai\pin.conf** file, comment out the line:

```
- dm plugin_name ./plugin_xml.dll
```

Add the following line:

```
- dm plugin_name ./stc_ewportaljms.dll
```

7 Make the following change to the **C:\Infranet\sys\eai_js\Infranet.properties** file: infranet.log.level=3

8 Extract **stc_ewportaljms.dll** from **CD\setup\addons\ewportal\win32.tar** to **Infranet\sys\dm_eai** directory.

9 Copy **stc_msapi.dll**, **stc_msclient.dll**, and **stc_mscommon.dll** from the **eGate\client\bin** directory to **Infranet\sys\dm_eai** directory.

10 Start all Infranet services, including the Infranet EAI Data Manager and the Infranet EAI Java Server. The EAI Data Manager should start successfully with the custom DLL.

11 From **C:\Infranet\sys\data\config**, do the following:

```
C:\Infranet\sys\data\config>load_pin_notify pin_notify_eai
```

Expected Output:

Using config file (pin_notify_eai)

Object '/config/notify' created successfully

12 Invoke an .xml file editor, such as Internet Explorer and open the file **C:\Infranet\sys\eai_js\payloadconfig.xml**. Select the drop down menu View and select Source.

13 The default publishing format in the payloadconfig.xml is set to "XML" as displayed in the line:

```
<Publisher DB="0.0.9.1" Format="XML">.
```

Note: *Change "XML" to "FLIST" if you want to publish FLISTs, otherwise do not change.*

14 To demonstrate publishing events, invoke Administrator and create an account, selecting the Custom plan. The CustCreate and CustomService entries in the payloadconfig.xml file will be executed and the results published in FLIST form which you can see in the **C:\Infranet\var\eai_js\eai_js.pinlog** file.

Installing on EAI running on Unix

- Solaris 2.6, 7, or 8
- HP-UX 11.0 or 11i
- Infranet 6.1 or 6.5
- Infranet EAI Framework Manager 6.1 or 6.5
- Infranet Administrator 6.1 or 6.5

Note: *The steps below use $PIN_HOME as the root directory location where Infranet is installed. You must use the applicable root directory for all steps.*

Solution Steps

Follow steps 1 -2 if you don't have EAI Framework Manager installed already. Otherwise skip to step 3.

1 Download the 6.1 EAI Framework software from the Portal web site **www.pin.com**. See your Strategic Alliance Manager for a www.pin.com login if you don't have one.

2 Stop all Infranet services and unzip the **6.1_EAI_Framework.zip** file and invoke the **setup.exe** program. Follow the instructions, selecting the default option for all entries.

3 Make the following changes to the **$PIN_HOME/sys/cm/pin.conf** file, where **$PIN_HOME** is the value of the environmental variable PIN_HOME:

```
- cm cm_loglevel 3
- cm dm_pointer 0.0.9.1 ip hostname 11970  # dm_eai
- cm em_pointer publish ip hostname 11930
- fm_publish enable_publish 1
```

Note: *"hostname" is the hostname of the system where the Infranet Data Manager is running.*

4 Make the following changes to the **$PIN_HOME/sys/dm_eai/pin.conf** file:

```
- dm loglevel 3
- dm dm_name hostname
```

Note: *"hostname" is the hostname of the system where the Infranet EAI Data Manager is running.*

5 Add the following changes to the **$PIN_HOME/sys/dm_eai/pin.conf** file::

```
#EGate
#- dm egatejmstopic PortalEventTopic
  - dm egateeventouttype FLIST
  - dm egatejmshots <hostname>
  - dm egatejmsport <portnumber>
  - dm egatejmspriority <priority level>
  - dm egatejmstimetolive <expirationtime>
  - dm egatejmspubuuid <JMS publisher uuid>
```

Note: *egateeventouttype can be either FLIST or XML, egatejmshost is the e*Gate JMS server host, egatejmsport is the e*Gate JMS server port.*

6 In the **PIN_HOME/sys/dm_eai/pin.conf** file, comment out the line:

```
- dm plugin_name ./plugin_xml.dll
```

Add the following line:

```
- dm plugin_name ./stc_ewportaljms.dll
```

7 Make the following change to the **$PIN_HOME/sys/eai_js/Infranet.properties** file: infranet.log.level=3

8 Extract **stc_ewportaljms.dll** from **CD/setup/addons/ewportal/hpux11.taz** to **$PIN_HOME/sys/dm_eai** directory.

9 Start all Infranet services, including the Infranet EAI Data Manager and the Infranet EAI Java Server. The EAI Data Manager should start successfully with the custom DLL.

10 From **$PIN_HOME/sys/data/config**, do the following:

```
$PIN_HOME/sys/data/config>load_pin_notify pin_notify_eai
```

Expected Output:

Using config file (pin_notify_eai)

Object '/config/notify' created successfully

11 Invoke an .xml file editor, such as Internet Explorer and open the file **$PIN_HOME/sys/eai_js/payloadconfig.xml**. Select the drop down menu View and select Source.

12 The default publishing format in the payloadconfig.xml is set to "XML" as displayed in the line:

```
<Publisher DB="0.0.9.1" Format="XML">.
```

Note: *Change "XML" to "FLIST" if you want to publish FLISTs, otherwise do not change.*

13 To demonstrate publishing events, invoke Administrator and create an account, selecting the Custom plan. The CustCreate and CustomService entries in the

payloadconfig.xml file will be executed and the results published in FLIST form which you can see in the **$PIN_HOME/var/eai_js/eai_js.pinlog** file.

Chapter 3

e*Way Connection Configuration

This chapter describes how to configure the Java-enabled Portal e*Way Connection Configuration.

3.1 Configuring e*Way Connections

e*Way Connections are set using the Schema Designer.

To create and configure e*Way Connections:

1. In the Schema Designer's **Component** editor, select the **e*Way Connections** folder.
2. On the palette, click the **Create a New e*Way Connection** button.
3. The **New e*Way Connection Component** dialog box opens, enter a name for the new **e*Way Connection**. Click **OK**.
4. Double-click on the new **e*Way Connection**. For this example, the connection has been defined as **portalConn**.
5. The **e*Way Connection Properties** dialog box opens.
6. From the **e*Way Connection Type** drop-down box, select **Portal**.
7. Enter the **Event Type "get"** interval in the dialog box provided. The configured default is 100 milliseconds.
8. From the **e*Way Connection Configuration File**, click **New** to create a new Configuration File for this e*Way Connection. (To use an existing file, click **Find**.)
9. The **e*Way Connection Edit Settings** window opens. Make any necessary changes to the Portal e*Way Connection parameters.
10. Go to **File**, **Save** to save settings.
11. Go to **File**, **Promote to Run Time**.

The Portal e*Way Connection configuration parameters are organized into the following sections:

- **Connector**
- **Connection**

3.1.1 Connector

This section contains a set of top level parameters:

- type
- class
- Property.Tag

Type

Description

Specifies the type of connector.

Required Values

A string . The value always defaults to **Portal** for Portal connections.

Class

Description

Specifies the class name of the Portal connector object.

Required Values

A valid package name. The default is com.stc.eways.portal.runtime.PortalConnector.

Property.Tag

Description

Specifies the data source identity. This parameter is required by the current **EBobConnectorFactory**.

Required Values

A valid data source package name.

3.1.2 Connection

This section assist in setting the parameters necessary to connect to the Portal Infranet :

- Host
- Port
- Username
- Password
- LoginType

Host

Description

Specifies the host on which the Portal Infranet server is running.

Required Values

A valid host name. *This parameter is mandatory.*

Port

Description

Specifies the TCP/IP port to which to connect to the Portal Infranet.

Required Values

A valid port number. *This parameter is mandatory.*

Username

Description

Specifies the username needed to connecting to the Portal Infranet.

Required Values

A valid username. *This parameter is mandatory*.

Password

Description

Specifies the Password for associated with the above username.

Required Values

A valid password. *This parameter is mandatory.*

LoginType

Description

Specifies the Login Type for connecting to the Portal Infranet.

Required Values

A valid Login Type.

Chapter 4

Multi-Mode e*Way Configuration

This chapter describes how to configure the e*Gate Integrator's Multi-Mode e*Way Intelligent Adapter.

4.1 Multi-Mode e*Way Properties

Set the Multi-Mode e*Way properties using the e*Gate Schema Designer.

To set properties for a new Multi-Mode e*Way

1. Select the Navigator pane's Components tab in the Main window of the Schema Designer.
2. Open the host and Control Broker where you want to create the e*Way.
3. On the Palette, click on the icon to create a new e*Way.
4. Enter the name of the new e*Way, then click **OK**.
5. Select the new component, then click the Properties icon to edit its properties.

   The **e*Way Properties** dialog box opens
6. Click **Find** beneath the **Executable File** field, and select an executable file (**stceway.exe** is located in the **bin** directory).
7. Under the **Configuration File** field, click **New**.

   The e*Way Configuration Editor window opens.
8. When the **Settings** page opens, set the configuration parameters for this e*Way's configuration file, under **JVM Settings** (see **"JVM Settings" on page 26** and **"General Settings" on page 29** for details).
9. After selecting the desired parameters, click **Save** on the **File** menu to save the configuration (**.cfg**) file.
10. Close the **.cfg** file and e*Way Configuration Editor.
11. Set the properties for the e*Way in the **e*Way Properties** dialog box.
12. Click **OK** to close the dialog box and save the properties.

4.2 JVM Settings

To correctly configure the e*Way Intelligent Adapter for Portal Infranet, you must configure the Java Virtual Machine (JVM) settings. This section explains the configuration parameters in the e*Way Configuration Editor window, which control these settings.

JNI DLL Absolute Pathname

Description

Specifies the absolute path name to where the JNI **.dll** (Windows) or shared library (UNIX) file is installed by the Java SDK on the Participating Host.

Required Values

A valid path name.

Additional Information

The JNI **.dll** or shared library file name varies, depending on the current operating system (OS). The following table lists the file names by OS:

OS	Java 2 JNI DLL Name
Windows systems	jvm.dll
Solaris 2.6, 2.7, 2.8	libjvm.so
HP-UX	libjvm.sl
AIX 4.3	libjvm.a

The value assigned can contain a reference to an environment variable, by enclosing the variable name within a pair of “%” symbols, for example:

```
%MY_JNIDLL%
```

Such variables can be used when multiple Participating Hosts are used on different OS/platforms.

Caution: *To ensure that the JNI **.dll** file loads successfully, the Dynamic Load Library search path environment variable must be set appropriately to include all the directories under the Java SDK installation directory, which contain shared library or **.dll** files.*

CLASSPATH Prepend

Description

Specifies the paths to be prepended to the CLASSPATH environment variable for the JVM.

Required Values

An absolute path or an environmental variable. This parameter is optional.

Additional Information

If left unset, no paths are prepended to the CLASSPATH environment variable.

Existing environment variables may be referenced in this parameter by enclosing the variable name in a pair of "%" symbols, for example:

```
%MY_PRECLASSPATH%
```

CLASSPATH Override

Description

Specifies the complete CLASSPATH variable to be used by the JVM. This parameter is optional. If left unset, an appropriate CLASSPATH environment variable (consisting of required e*Gate components concatenated with the system version of CLASSPATH) is set.

Note: *All necessary **.jar** and **.zip** files needed by both e*Gate and the JVM must be included. It is advised that the **CLASSPATH Prepend** parameter be used.*

Required Values

An absolute path or an environmental variable. This parameter is optional.

Additional Information

Existing environment variables may be referenced in this parameter by enclosing the variable name in a pair of "%" symbols, for example:

```
%MY_CLASSPATH%
```

CLASSPATH Append From Environment Variable

Description

Specifies whether the path is appended for the CLASSPATH environmental variable to jar and zip files needed by the JVM.

Required Values

YES or **NO**. The configured default is YES.

Initial Heap Size

Description

Specifies the value for the initial heap size in bytes. If set to 0 (zero), the preferred value for the initial heap size of the JVM is used.

Required Values

An integer between 0 and 2147483647. This parameter is optional.

Maximum Heap Size

Description

Specifies the value of the maximum heap size in bytes. If set to 0 (zero), the preferred value for the maximum heap size of the JVM is used.

Required Values

An integer between 0 and 2147483647. This parameter is optional.

Maximum Stack Size for Native Threads

Description

Specifies the value of the maximum stack size in bytes for native threads. If set to 0 (zero), the default value is used.

Required Values

An integer between 0 and 2147483647. This parameter is optional.

Maximum Stack Size for JVM Threads

Description

Specifies the value of the maximum stack size in bytes for JVM threads. If set to 0 (zero), the preferred value for the maximum heap size of the JVM is used.

Required Values

An integer between 0 and 2147483647. This parameter is optional.

Disable JIT

Description

Specifies whether the Just-In-Time (JIT) compiler is disabled.

Required Values

YES or **NO**.

Remote debugging port number

Description

Specifies whether to allow remote debugging of the JVM.

Required Values

YES or **NO**.

Suspend Option for Debugging

Description

Indicates whether to suspend Option for Debugging on JVM startup.

Required Values

YES or **NO**.

4.3 General Settings

This section contains the parameters for rollback wait and IQ messaging priority.

Note: *For more information on the **General Settings** configuration parameters see the **e*Gate Integrator User's Guide**.*

4.3.1 Rollback Wait Interval

Description

Specifies the time interval to wait before rolling back the transaction.

Required Values

A number within the range of 0 to 99999999, representing the time interval in milliseconds.

4.3.2 Standard IQ FIFO

Description

Specifies whether the highest priority messages from all SeeBeyond Standard IQs are delivered in the first-in-first-out (FIFO) order.

Required Values

Select **Yes** or **No**. **Yes** indicates that the e*Way retrieves messages from all SeeBeyond Standard IQs in the first-in-first-out (FIFO) order. **No** indicates that this feature is disabled; **No** is the default.

Chapter 5

Implementation

This chapter includes basic information pertinent to implementing the Portal e*Way in a production environment. A sample schema is included with the installation CD-ROMs for the Portal Client e*Way. In addition, the following pages demonstrate how the components of the sample schema are created.

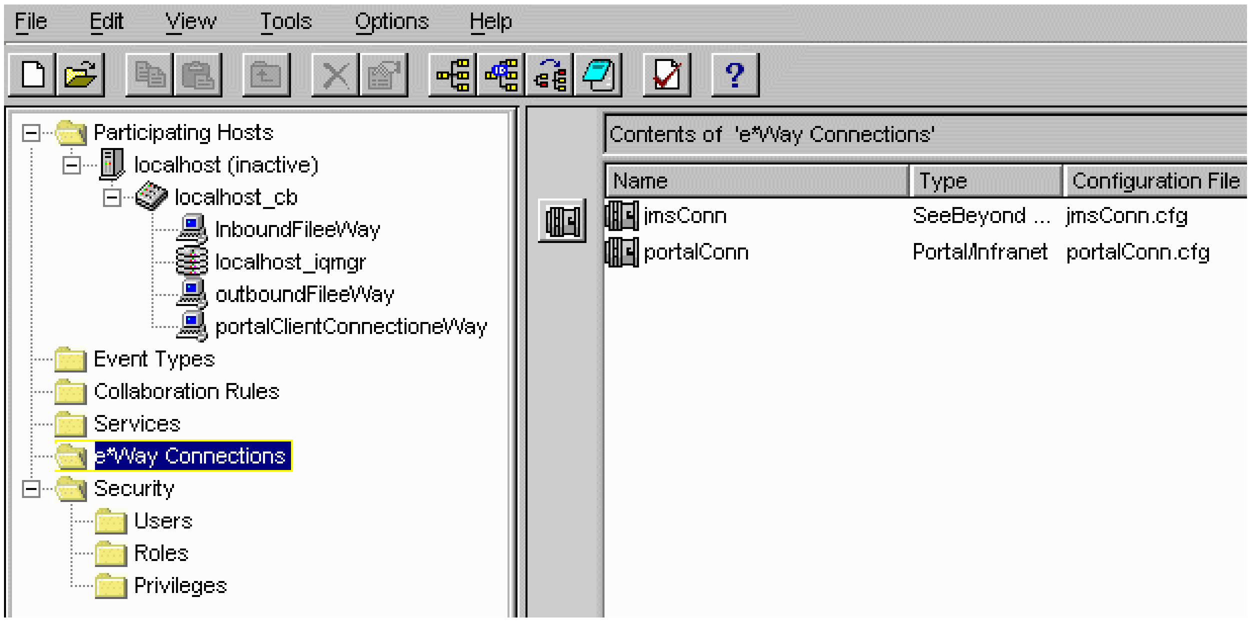
5.1 Portal Client Sample Implementation

The following directions assume that the e*Way has been successfully installed and that all necessary JAR files are accessible.

During installation, the host and Control Broker are automatically created and configured. The default name of each is the name of the host on which you are installing the e*Gate Schema Designer GUI.

Figure 4 shows the components of the Sample Portal Client e*Way schema as viewed from the Components tab of the Schema Designer.

Figure 4 Portal Client e*Way Schema Components



- Participating Hosts
  - localhost (inactive)
    - localhost_cb
      - InboundFileeWay
      - localhost_iqmgr
      - outboundFileeWay
      - portalClientConnectioneWay
- Event Types
- Collaboration Rules
- Services
- e*Way Connections
- Security
  - Users
  - Roles
  - Privileges

Contents of 'e*Way Connections'

Name	Type	Configuration File
jmsConn	SeeBeyond ...	jmsConn.cfg
portalConn	PortalInfranet	portalConn.cfg

The sample schema included on the Installation CD-ROM provides the sample in a near - complete form. To create the sample manually includes the following procedures:

- Make sure that the Control Broker is activated.
- In the e*Gate Schema Designer, define and configure the following as necessary:
  - A SeeBeyond JMS IQ Manager
  - Inbound e*Way using **stcewfile.exe**
  - Outbound e*Way using **stcewfile.exe**
  - The Multi-Mode e*Way component as described in **Chapter 4**
  - Event Type Definitions used to package the data to be exchanged with the external system.
  - Collaboration Rules to process Events.
  - The Portal e*Way Connection as described in **Chapter 3**.
  - A JMS connection
  - Collaborations, to be associated with each e*Way component, to apply the required Collaboration Rules.
  - The destination to which data will be published prior to being sent to the external system.

The following sections describe how to create and configure each of the above components. For directions on importing the sample schema, see **Importing the Sample Schema** on page 58.

5.1.1 Creating a New Schema

The first task in deploying the sample implementation is to create a new schema name. While it is possible to use the default schema for the sample implementation, it is recommended that you create a separate schema for testing purposes. After you install the e*Way, do the following:

1. Start the e*Gate Schema Designer GUI.
2. When the Schema Designer prompts you to log in, select the host that you specified during installation, and enter your password.
3. You will then be prompted to select a schema. Click **New**.
4. Enter a name for the new Schema. In this case, enter **Portal_Test**, or any name as desired.
   The e*Gate Schema Designer opens under your new schema. You are now ready to begin creating the necessary components for this sample schema.

5.1.2 Designating a SeeBeyond JMS IQ Manager

To create and configure a new SeeBeyond JMS IQ Manager

1. In the Schema Designer's Navigator pane, click the **Components** tab (if necessary) and click the Control Broker of the Participating Host where you want to create the new IQ manager.
2. In the component pane, select the IQ Manager and then edit its properties.

   The **IQ Manager Properties** dialog box appears.
3. From the **IQ Manager Type** list, click **SeeBeyond JMS**.
4. Under **Configuration File**, click **New**. When the **Edit Settings** dialog box appears, set the configuration parameters for this configuration file. Parameters are listed and explained in the *SeeBeyond JMS Intelligent Queue User's Guide*.

Note: *You may want to add user notes to flag or explain any nonstandard settings.*

5. After setting the configuration parameters and adding user notes, save the **.cfg** file (keeping the same name as the name of the IQ Manager and accepting the default location), and close the **Edit Settings** dialog box.

5.1.3 Creating Event Type Definitions

The Portal Client e*Way requires using the OPCODE builder and the Portal HTML specifications to create the necessary ETDs.

Creating an Event Type from Portal HTML Specifications

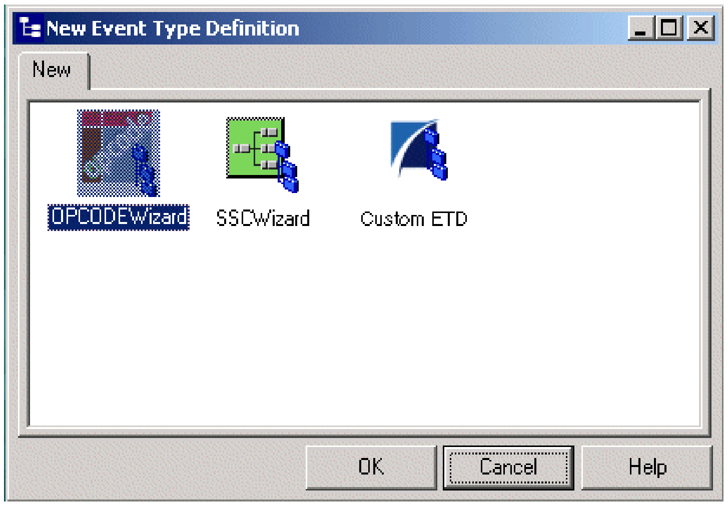
For the purpose of this example, the following procedure shows how to create an Event Type Definition (ETD) from the Portal Infranet 6.1 or 6.5 documented OPCODE HTML specification using "PCM_OP_CUST_POL_GET_PLANS.input.html" as the input file.

To create an ETD using Portal HTML specifications

1. From the e*Gate Schema Designer, open the ETD Editor by double-clicking on the button in the button bar (be sure your editor default is set to the Java programming language).
2. When the ETD Editor Main window opens, select **New** from the **File** menu.

The **New Event Type Definition** dialog box appears (see Figure 5).

Figure 5 New Event Type Definition Dialog Box



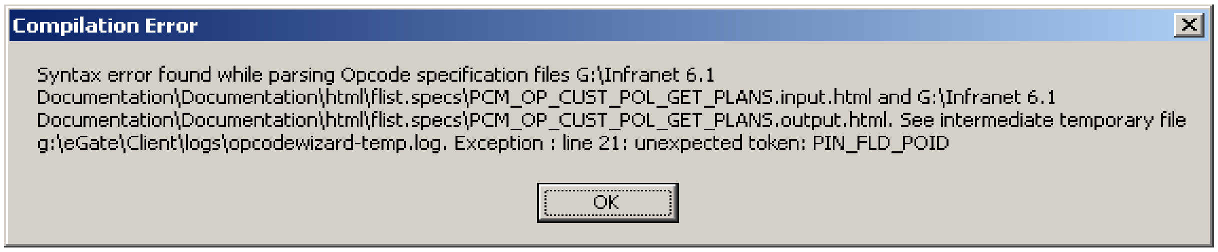
3 Select the **OPCODEWizard** icon.

4 Follow the wizard instructions, enter a package name, and navigate to the location of the OPCODE specifications locating the desired OPCODE.

Note: *It is not necessary to open the Infranet documentation, navigate to <rootdirectory>\Infranet 6.1 or 6.5 Documentation\Documentation\html\flist.specs*

For the sample, the PCM_OP_CUST_POL_GET_PLANS.input.html has been used.

Note: *If there are any errors contained within the HTML specifications an error message will appear providing a log file and exception line information which acts as an intermediary file to review the specification. There is the possibility that the Portal specifications contain errors. These errors must be fixed by hand before continuing. Portal is continuously repairing the specifications for Infranet, you should contact them for the latest version if you find a problem.*

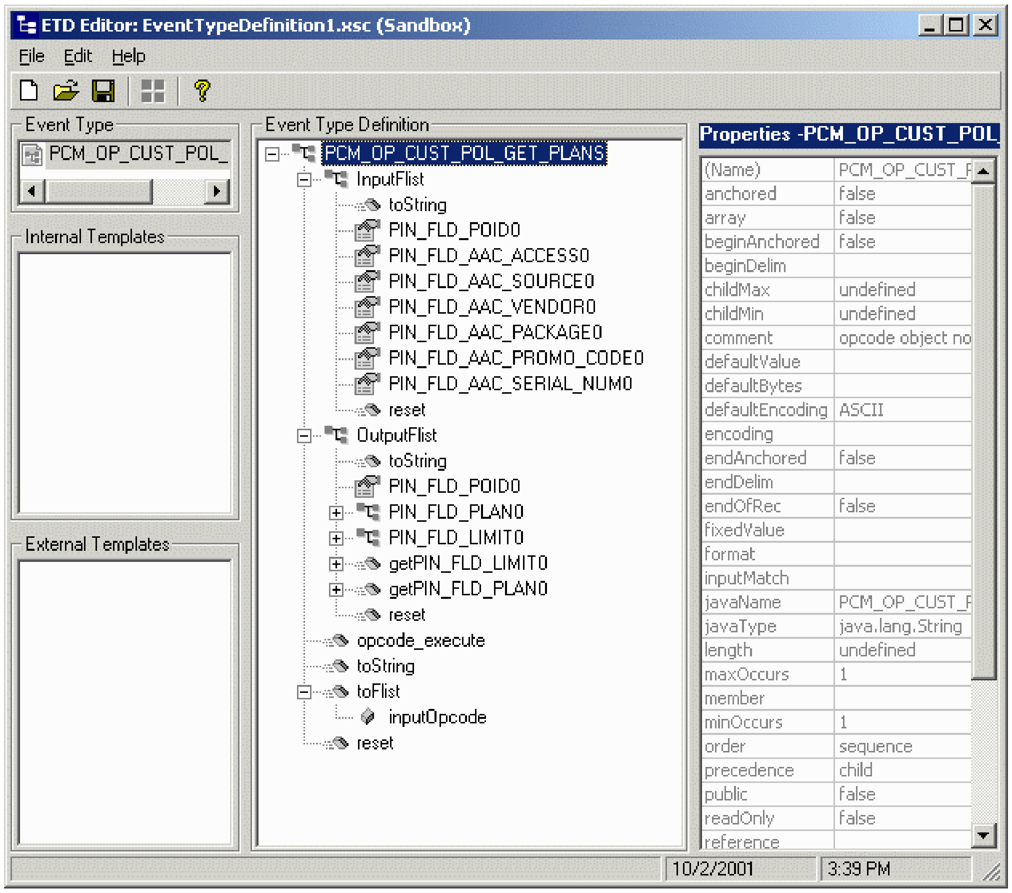


```
Syntax error found while parsing Opcode specification files G:\Infranet 6.1
Documentation\Documentation\html\flist.specs\PCM_OP_CUST_POL_GET_PLANS.input.html and G:\Infranet 6.1
Documentation\Documentation\html\flist.specs\PCM_OP_CUST_POL_GET_PLANS.output.html. See intermediate temporary file
g:\eGate\Client\logs\opcodewizard-temp.log. Exception : line 21: unexpected token: PIN_FLD_POID
```

Once the error has been located, in the egate\client\logs, the user must modify the actual input.html file and save to another location. Once completed, return to the builder to re-attempt the creation. If there are no further errors, continue.

5 If no error occurs, or all errors have been corrected, the ETD Editor opens displaying the newly converted .xsc file.

An ETD is a graphical representation of the layout of data in an Event. Both the input FLIST and output FLIST information has been incorporated into a the resulting .xsc.



Event Type: PCM_OP_CUST_POL_

```
PCM_OP_CUST_POL_GET_PLANS
  InputFlist
    toString
    PIN_FLD_POID0
    PIN_FLD_AAC_ACCESS0
    PIN_FLD_AAC_SOURCE0
    PIN_FLD_AAC_VENDOR0
    PIN_FLD_AAC_PACKAGE0
    PIN_FLD_AAC_PROMO_CODE0
    PIN_FLD_AAC_SERIAL_NUM0
    reset
  OutputFlist
    toString
    PIN_FLD_POID0
    PIN_FLD_PLAN0
    PIN_FLD_LIMIT0
    getPIN_FLD_LIMIT0
    getPIN_FLD_PLAN0
    reset
  opcode_execute
  toString
  toFlist
    inputOpcode
  reset
```

(Name)	PCM_OP_CUST_F
anchored	false
array	false
beginAnchored	false
beginDelim	
childMax	undefined
childMin	undefined
comment	opcode object no
defaultValue	
defaultBytes	
defaultEncoding	ASCII
encoding	
endAnchored	false
endDelim	
endOfRec	false
fixedValue	
format	
inputMatch	
javaName	PCM_OP_CUST_F
javaType	java.lang.String
length	undefined
maxOccurs	1
member	
minOccurs	1
order	sequence
precedence	child
public	false
readOnly	false
reference	

6 Save the file as “**pcmOpCustPolGetPlans.xsc**.”

7 Click **File, Promote File to Run Time**.

Note: *For more information on the creation and modification of Java-enabled ETDs, please see the “Java-based ETD Editor” user guide.*

5.1.4 Creating and Configuring the e*Ways

The first components to be created are the following e*Ways.

- **“Inbound e*Way” on page 35**
- **“Outbound e*Way” on page 36**
- **“Portal Client e*Way” on page 36**

The following sections provide instructions for creating each e*Way.

Inbound e*Way

1 Select the Navigator's **Components** tab.

2 Open the host on which you want to create the e*Ways.

3 Select the **Control Broker** that will manage the new e*Ways.

4 On the palette, click the **Create a New e*Way** button.

5 Enter the name of the new e*Way (in this case "**InboundFileeWay**"**)**, then click **OK**.

6 Right-click the new e*Way and select **Properties** to edit its properties.

7 The e*Way Properties window opens. Click the **Find** button beneath the **Executable File** field, and select **stcewfile.exe** as the executable file.

8 Under the **Configuration File** field, click the **New** button. The Edit Settings window opens. Select the following settings for this configuration file.

:

Table 2 Configuration Parameters for the Inbound e*Way

Parameter	Value
General Settings (unless otherwise stated, leave settings as default)	
AllowIncoming	YES
AllowOutgoing	NO
Outbound Settings	Default
Poller Inbound Settings	
PollDirectory	../DATA/input/portaltest (input file folder)
InputFileExtension	*.dat (input file extension) For the example, the contents of the input file is that of 46 "admintool".
PollMilliseconds	1000
Remove EOL	YES
MultipleRecordsPerFile	YES
MaxBytesPerLine	4096
BytesPerLineIsFixed	NO
File Records Per eGate Event	1
Performance Testing	Default

9 After selecting the desired parameters, save the **configuration** file (as "**portalIn.cfg**").

10 Click **File**, **Promote to Run Time**. This will close the **.**cfg file.

11 In the e*Way - Properties window, use the **Startup**, **Advanced**, and **Security** tabs to modify the default settings for each e*Way you configure.

A Use the **Startup** tab to specify whether the e*Way starts automatically, or restarts after abnormal termination or due to scheduling, and so forth.

B Use the **Advanced** tab to specify or view the activity and error logging levels, as well as the Event threshold information.

C Use **Security** to view or set privilege assignments.

12 Select **OK** to close the e*Way Properties window.

Outbound e*Way

1 Select the Navigator's **Components** tab.

2 Open the host on which you want to create the e*Ways.

3 Select the **Control Broker** that will manage the new e*Ways.

4 On the palette, click the **Create a New e*Way** button.

5 Enter the name of the new e*Way (in this case "**OutboundFileeWay**"), then click **OK**.

6 Select the new e*Way, right-click and select **Properties** to edit its properties.

7 When the **e*Way Properties** window opens, click the **Find** button beneath the **Executable File** field. Select **stcewfile.exe** as the executable file.

8 Under the **Configuration File** field, click the **New** button. The **Edit Settings** window opens. Select the following settings for this configuration file.

:

Table 3 Configuration Parameters for the Outbound e*Way

Parameter	Value
General Settings (unless otherwise stated, leave settings as default)	
AllowIncoming	NO
AllowOutgoing	YES
Outbound Settings	
OutputDirectory	../DATA/OUTPUT/portalTest
OutputFileName	output%d.dat
MultipleRecordsPerFile	NO
MaxRecordsPerFile	10000
AddEOL	YES
Poller Inbound Settings	Default
Performance Testing	Default

9 Save the .cfg file (**outboundfileeWay.cfg**), and click **File**, **Promote to Run Time**, to close the Edit Settings window.

10 Click **OK** to close the e*Way Properties window.

Portal Client e*Way

1 Select the e*Gate Schema Designer's **Components** tab.

2 Open the host on which you want to create the e*Way.

3 Select the **Control Broker** that manages the new e*Way.

4. On the palette, click the **Create e*Way** button.
5. Enter the name of the new e*Way (in this case, **PortalClienteWay**), then click **OK**.
6. Right-click the new e*Way and select **Properties** to edit its properties.
7. When the **e*Way Properties** dialog box opens, click the **Find** button beneath the **Executable File** field, and select **stceway.exe** as the executable file.
8. To edit the JVM Settings, select **New** under **Configuration File**. When the e*Way Configuration Editor opens, edit these settings as shown in Table 4.

:

Table 4 Configuration Parameters for the Portal Client e*Way

Parameter	Value
JVM Settings (unless otherwise stated, leave settings as default)	
JNI DLL absolute pathname	C:\eGate\client\bin\Jre\jvm.dll (or absolute path to proper JNI DLL)
CLASSPATH Prepend	Default
CLASSPATH Append From Environment Variable	YES

Note: *See* **Chapter 4** *for details on the parameters associated with the Multi-Mode e*Way.*

9. Save the **.cfg** file, and click **File**, **Promote to Run Time** to close the e*Way Configuration Editor window.
10. In the **e*Way Properties** dialog box, use the **Startup**, **Advanced**, and **Security** tabs to modify the default settings for each.
    A. Use the **Startup** tab to specify whether the e*Way starts automatically, restarts after abnormal termination or due to scheduling, and so on.
    B. Use the **Advanced** tab to specify or view the activity and error logging levels, as well as the Event threshold information.
    C. Use **Security** to view or set privilege assignments.
11. Click **OK** to close e*Way Properties window.

5.1.5 Create the Portal e*Way Connection

The e*Way Connection configuration file contains the connection information along with the information needed to communicate using Infranet.

To create and configure a New e*Way Connection

1. Select the **e*Way Connection** folder on the **Components** tab of the e*Gate Navigator.
2. On the palette, click the **Create a New e*Way Connection** button.
3. Enter the name of the e*Way Connection (for this sample, "**portalClientConnection**"), then click **OK.**
4. Double-click the new e*Way Connection to edit its properties.

5. The e*Way Connection Properties window opens. Select **PortalInfranet** from the **e*Way Connection Type** drop-down menu.
6. Under e*Way Connection Configuration File, click the **New** button.
7. The e*Way Connection editor opens, select the necessary parameters.

   For more information on the Portal e*Way Connection parameters, see **"e*Way Connection Configuration" on page 22**.
8. Save the .cfg file and click **File**, **Promote to Run Time**.

5.1.6 Create the JMS e*Way Connection

The e*Way Connection configuration file contains the connection information along with the information needed to communicate using Infranet.

To create and configure a New e*Way Connection

1. Select the **e*Way Connection** folder on the **Components** tab of the e*Gate Navigator.
2. On the palette, click the **Create a New e*Way Connection** button.
3. Enter the name of the e*Way Connection (for this sample, "**jms_conn**"), then click **OK.**
4. Double-click the new e*Way Connection to edit its properties.
5. The e*Way Connection Properties window opens. Select **SeeBeyondJMS** from the **e*Way Connection Type** drop-down menu.
6. Under e*Way Connection Configuration File, click the **New** button.
7. The e*Way Connection editor opens, select the necessary parameters.

   For more information on the JMS Connection parameters, see the SeeBeyond JMS Intelligent Queue User's Guide.
8. Save the .cfg file and click **File**, **Promote to Run Time**.

5.1.7 Collaboration Rules

The next step is to create the Collaboration Rules that will extract and process selected information from the source Event Type defined above, according to its associated Collaboration Service. The **Default Editor** can be set to either **Monk** or **Java**.

From the **Schema Designer Task Bar,** select **Options** and click **Default Editor**. The default should be set to **Java**.

The sample schema calls for the creation of three collaboration Rules files.

- **passThrough** (Java)
- **GetPlanList** (Java)
- **OutboundCollab** (Java)

Creating Collaboration Rules files

javaPassThroughCollab (inbound)

1 Select the Navigator's **Components** tab in the e*Gate Schema Designer.

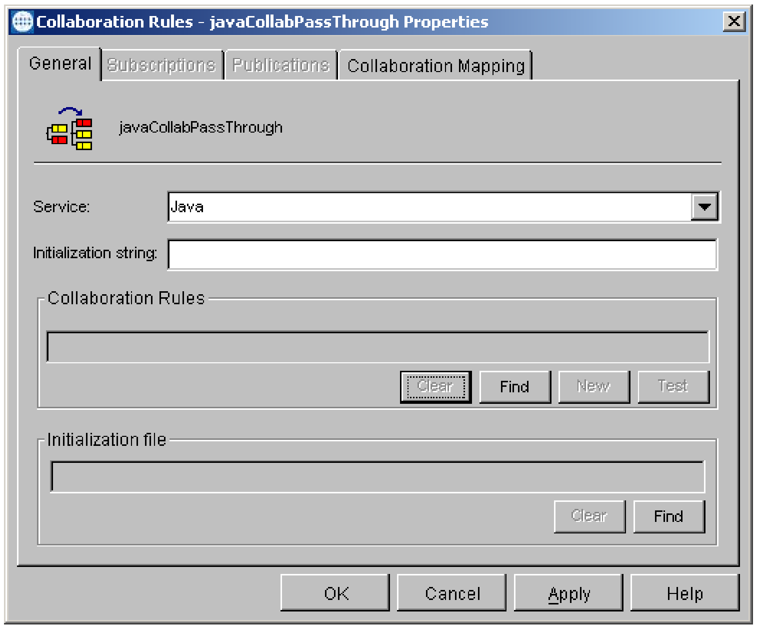
2 In the Navigator, select the **Collaboration Rules** folder.

3 On the palette, click the **Create New Collaboration Rules** button.

4 Enter the name of the new Collaboration Rule Component (for this case "**javaCollabPassThrough**"), then click **OK**.

5 Double-click the new Collaboration Rules Component. The **Collaboration Rules Properties** window opens.

Figure 6 Collaboration Properties



Collaboration Rules - javaCollabPassThrough Properties

javaCollabPassThrough

Service: Java

Initialization string:

6 From the **Service** field drop-down box, select **Java**.

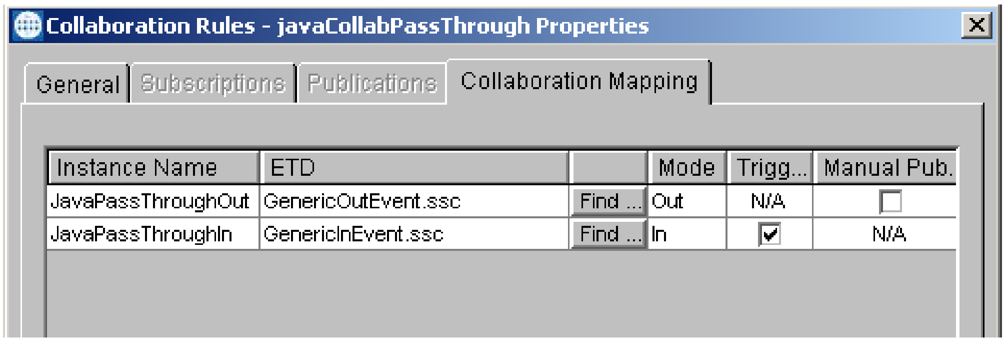
7 Under the **Collaboration Rules** dialog, select **Find**. Navigate to collaboration_rules\STCLibrarySTCJavaPassThrough.class, select and the class appears in the dialog box.

In the **Initialization string** box, enter any required initialization string that the Collaboration Service may require. This field can be left blank.

8 The **Collaboration Mapping** tab is now enabled, and the **Subscriptions** and **Publications** tabs are disabled.

9 Select the **Collaboration Mapping** tab. You will see default Instance and ETD names for this Collaboration Rule.

Figure 7 Collaboration Rules - Collaboration Mapping Properties



Instance Name	ETD		Mode	Trigg...	Manual Pub.
JavaPassThroughOut	GenericOutEvent.ssc	Find ...	Out	N/A	☐
JavaPassThroughIn	GenericInEvent.ssc	Find ...	In	✓	N/A

GetPlanList (Java)

1 Select the Navigator's **Components** tab in the e*Gate Schema Designer.

2 In the **Navigator**, select the **Collaboration Rules** folder.

3 On the palette, click the **Create New Collaboration Rules** button.

4 Enter the name of the new Collaboration Rule, then click **OK** (for this case, use **GetPlanList**).

5 Double-click the new Collaboration Rules Component to edit its properties. The Collaboration Rules Properties window opens.

6 From the **Service** field drop-down box, select Java. The **Collaboration Mapping** tab is now enabled, and the **Subscriptions** and **Publications** tabs are disabled.

7 In the **Initialization string** box, enter any required initialization string that the Collaboration Service may require. This field can be left blank.

8 Select the **Collaboration Mapping** tab.

9 Using the **Add Instance** button, create instances to coincide with the Event Types.

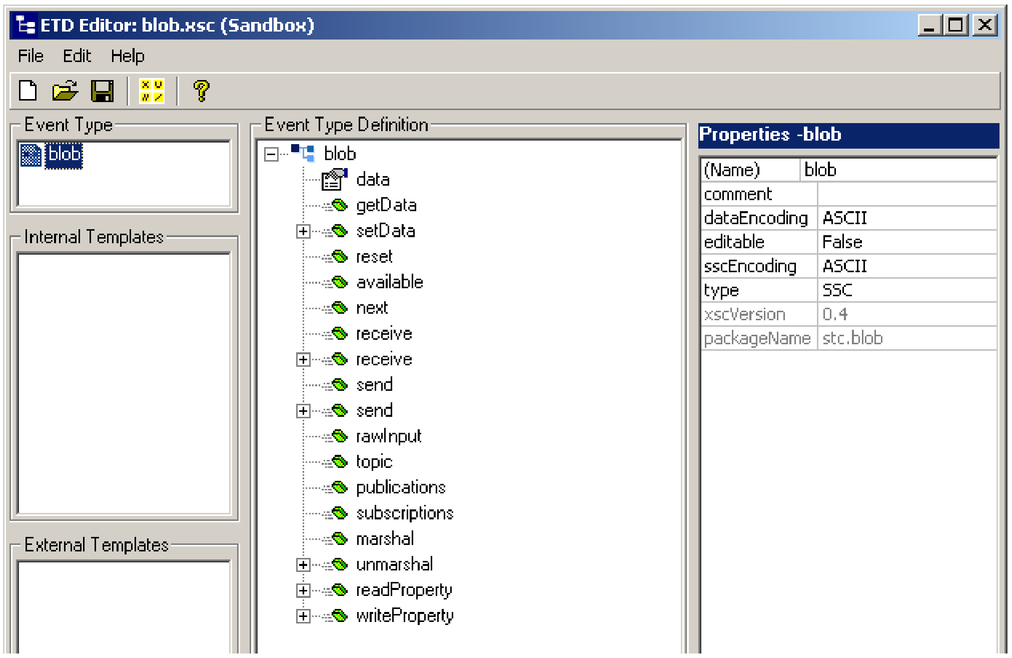
For this sample, do the following:

10 In the Instance Name column, enter **inBlob** for the instance name.

11 Click **Find**, navigate to **blob.xsc**, double-click to select. **blob.xsc** is added to the **ETD** column for this instance.

blob.xsc is a single-element ETD created using the e*Gate Custom ETD wizard

Figure 8 blob.xsc



(Name)	blob
comment	
dataEncoding	ASCII
editable	False
sscEncoding	ASCII
type	SSC
xscVersion	0.4
packageName	stc.blob

12 In the Mode column, select **In** from the drop–down menu available.

13 In the **Trigger** column, check the box to enable trigger mechanism.

14 Repeat steps 9–13 using the following values:

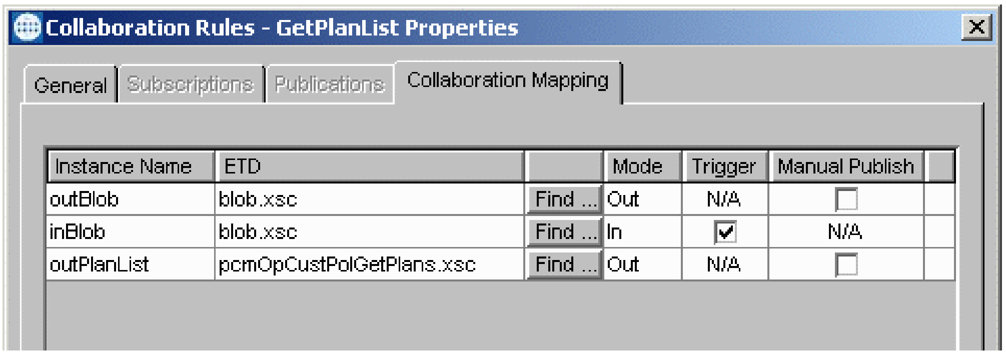
- Instance Name — **outBlob**
- ETD — **blob.xsc**
- Mode — **Out**
- Trigger — do not select

15 Repeat steps 9–13 again using the following values:

- Instance Name — **outPlanList**
- ETD — **pcmOpCustPolGetPlans.xsc**
- Mode — **Out**

16 Trigger — do not select

Figure 9 Collaboration Rules - Collaboration Mapping Properties

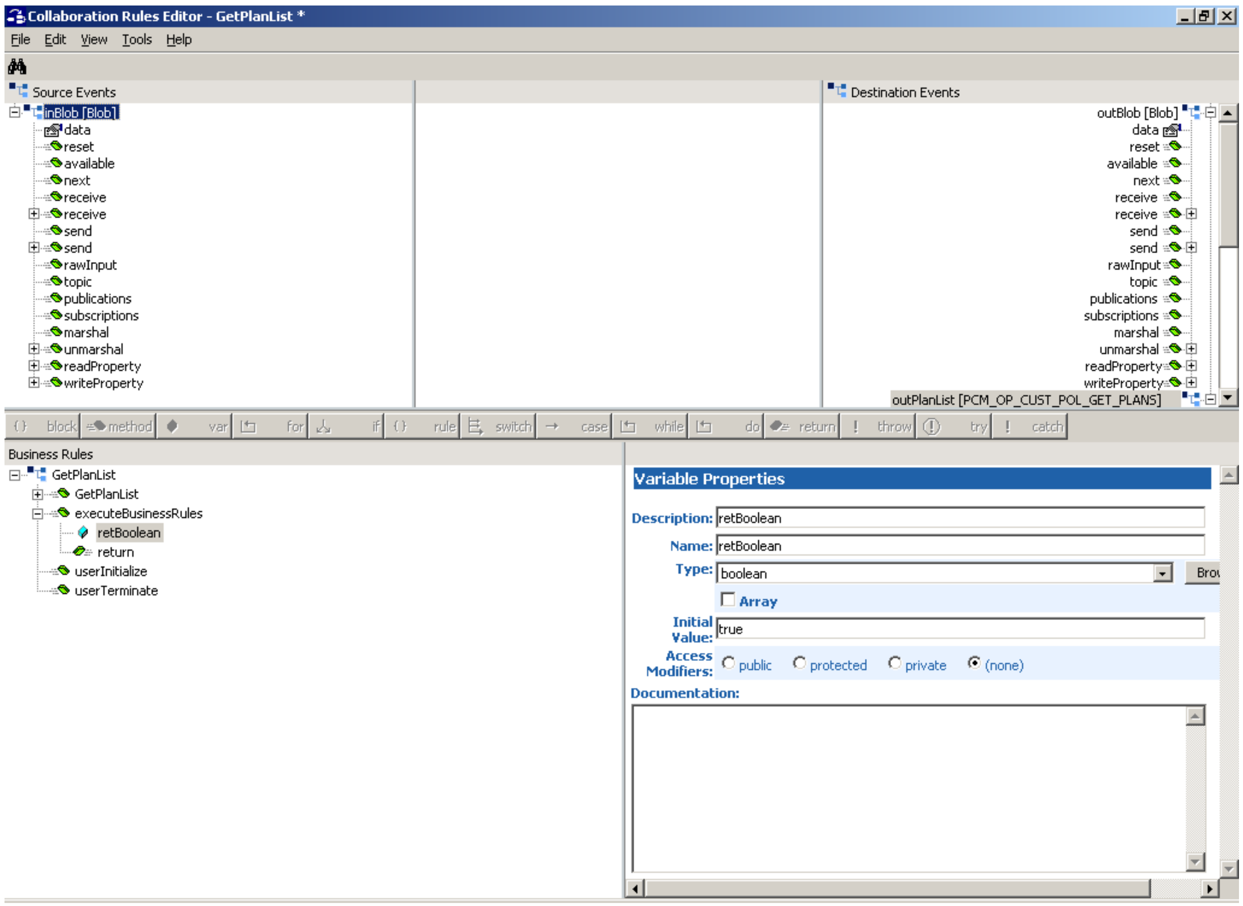


Instance Name	ETD		Mode	Trigger	Manual Publish
outBlob	blob.xsc	Find ...	Out	N/A	☐
inBlob	blob.xsc	Find ...	In	✓	N/A
outPlanList	pcmOpCustPolGetPlans.xsc	Find ...	Out	N/A	☐

Select the **General** tab, under the Collaboration Rule box, select **New**. The **Collaboration Rules Editor** opens.

17 Expand to full size for optimum viewing, expanding the Source and Destination Events as well.

Figure 10 Collaboration Rules — Collaboration Mapping Properties



Description: retBoolean
Name: retBoolean
Type: boolean
☐ Array
Initial Value: true
Access Modifiers: ○ public ○ protected ○ private [✓ (none)]
Documentation:

Creating the Collaboration Rules Class

In order to send or receive data, the following tasks must be completed:

A Populate the InputFlist - In this case the poid was set, ACCESS0 was set with a specific value, thereby populating the InputFlist.

B Call to OPCODE_execute() made, which sent this particular OPCODE to Portal along with the modified InputFlist.

C The OPCODE was executed on the Portal side. The result of which is then used to populate **OutputFlist**.

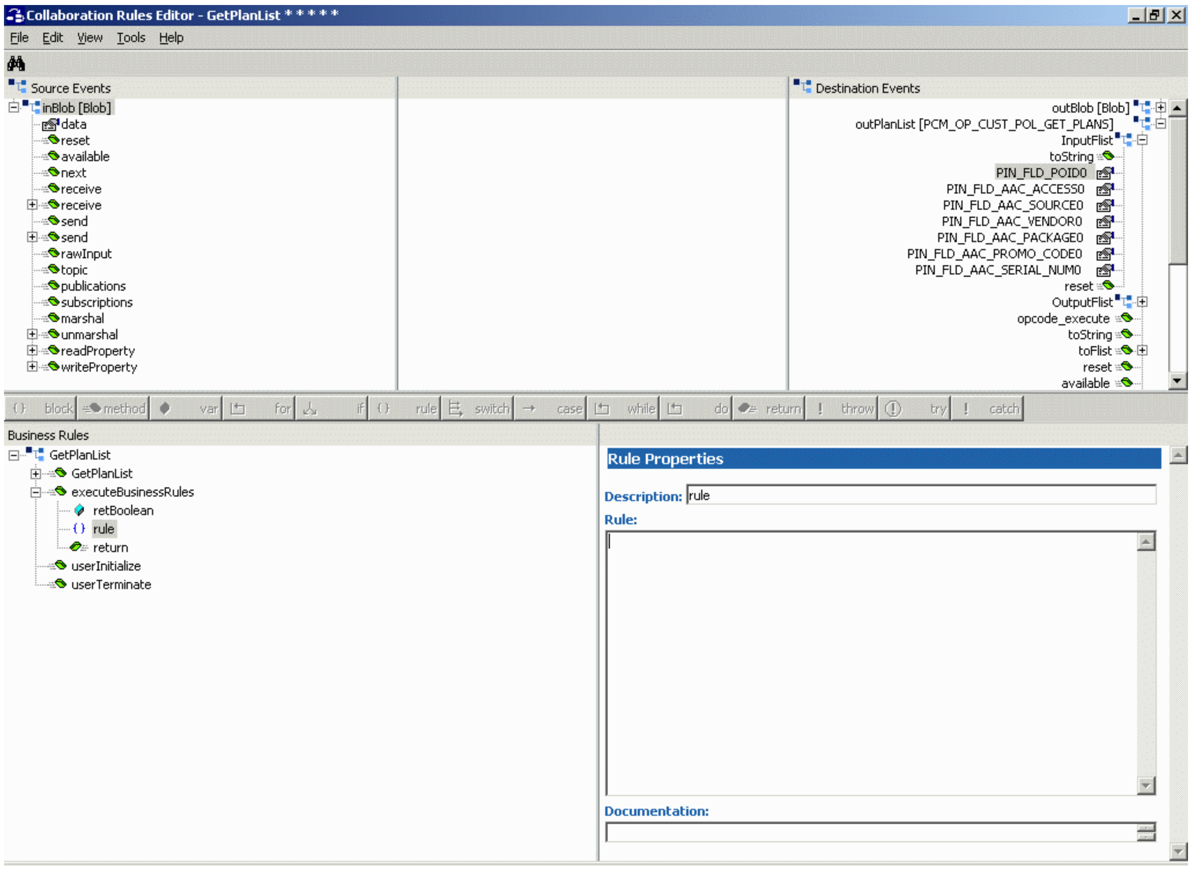
D The toString() method is provided to traverse the Flist and convert it to a string, which can then be copied to our simple structure.

1 Highlight **retBoolean** in the **Business Rules** pane.

All of the user–defined business rules are added as part of this method.

2 Select {} **rule** from the center pane. A rule appears below the **retBoolean** in the Business Rules pane.

Figure 11 Collaboration Rules — Collaboration Mapping Properties

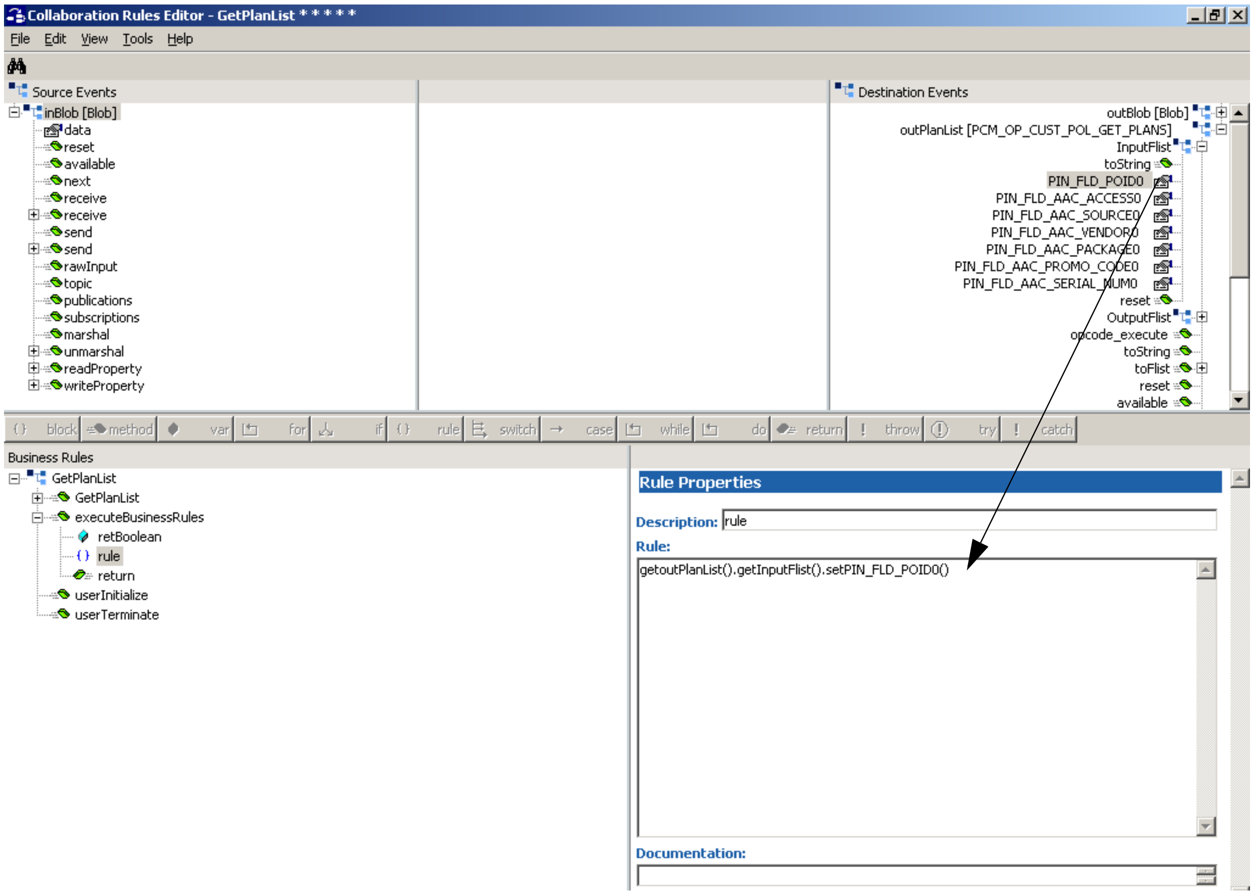


Description: rule

3 Select the newly created **rule** highlighted as above,

4 Drag the PIN_FLD_POID0 property value located under InputFlist into the rules dialog box on the right pane.

Figure 12 Collaboration Rules — Collaboration Mapping Properties



Description: rule

Rule:

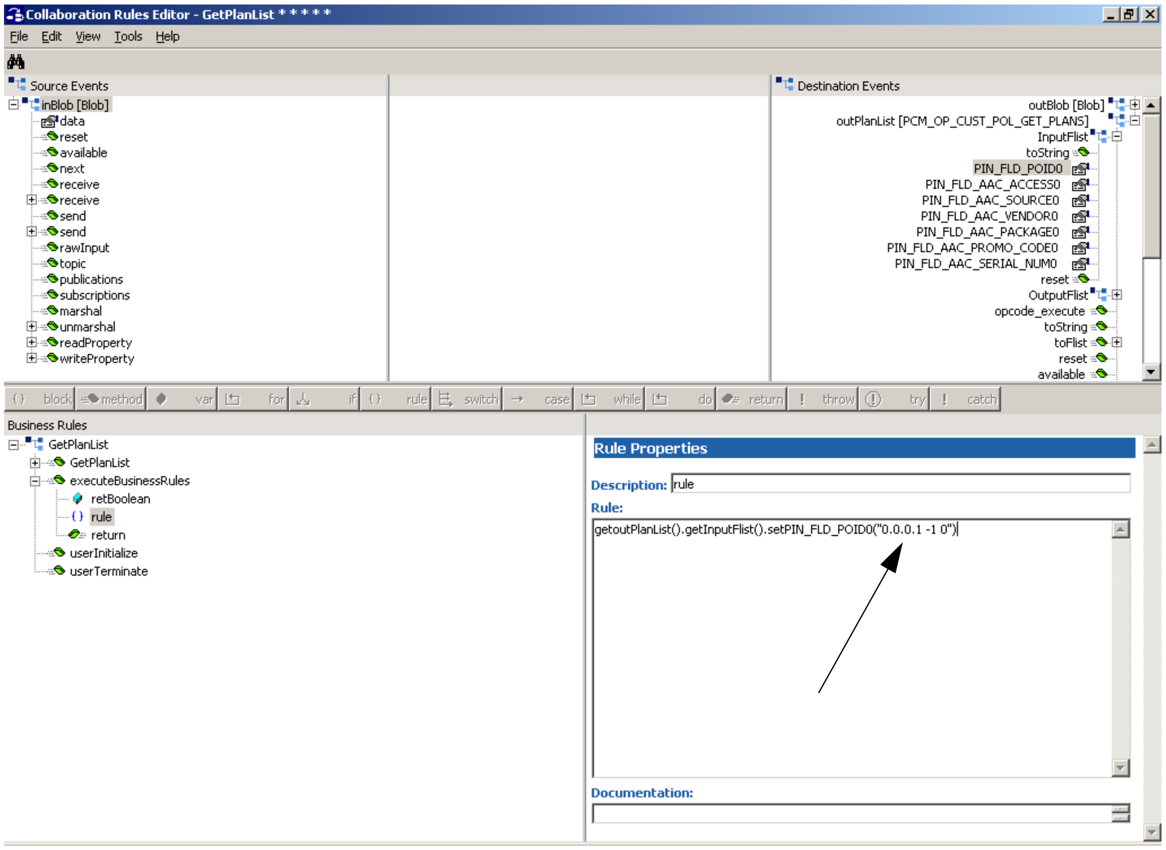
```
getoutPlanList().getInputFlist().setPIN_FLD_POID0()
```

5 Enter the required parameter information into the method parentheses. For the sample, the following is used:

```
"0.0.0.1 -1 0"
```

The “0.0.0.1” represents the Portal internal database location.

Figure 13 Collaboration Rules — Collaboration Mapping Properties



Description: rule

Rule:

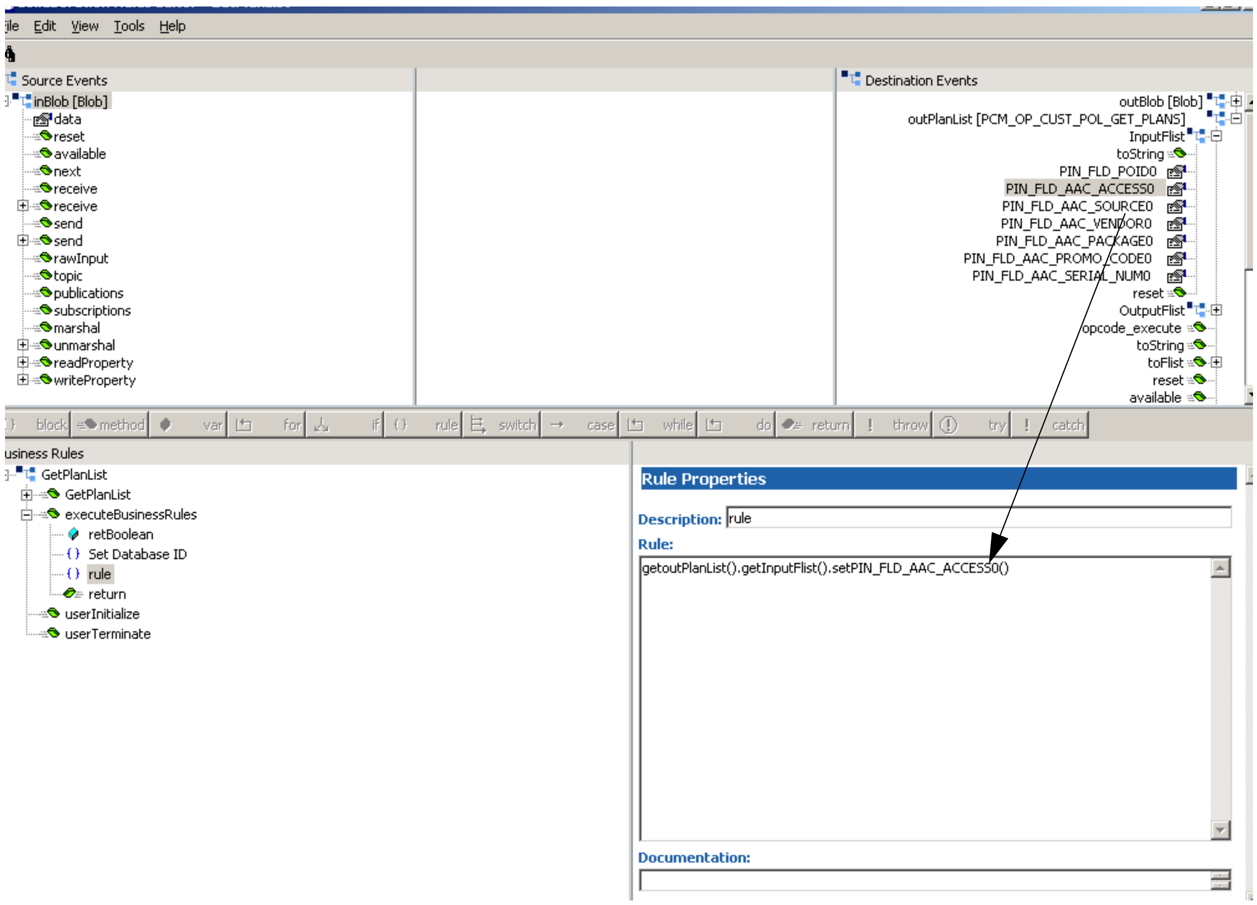
```
getoutPlanList().getInputFlist().setPIN_FLD_POID0("0.0.0.1 -1 0")
```

6 Change the description of the method from **rule** to **Set Database ID**. Reselect the **rule** to affect the updated description name.

7 Select {} **rule** from the center pane. A rule appears below the **Set Database ID** in the Business Rules pane.

8 Drag the PIN_FLD_AAC_ACCESS0 property into the rules dialog box.

Figure 14 Collaboration Rules — Collaboration Mapping Properties



Description: rule

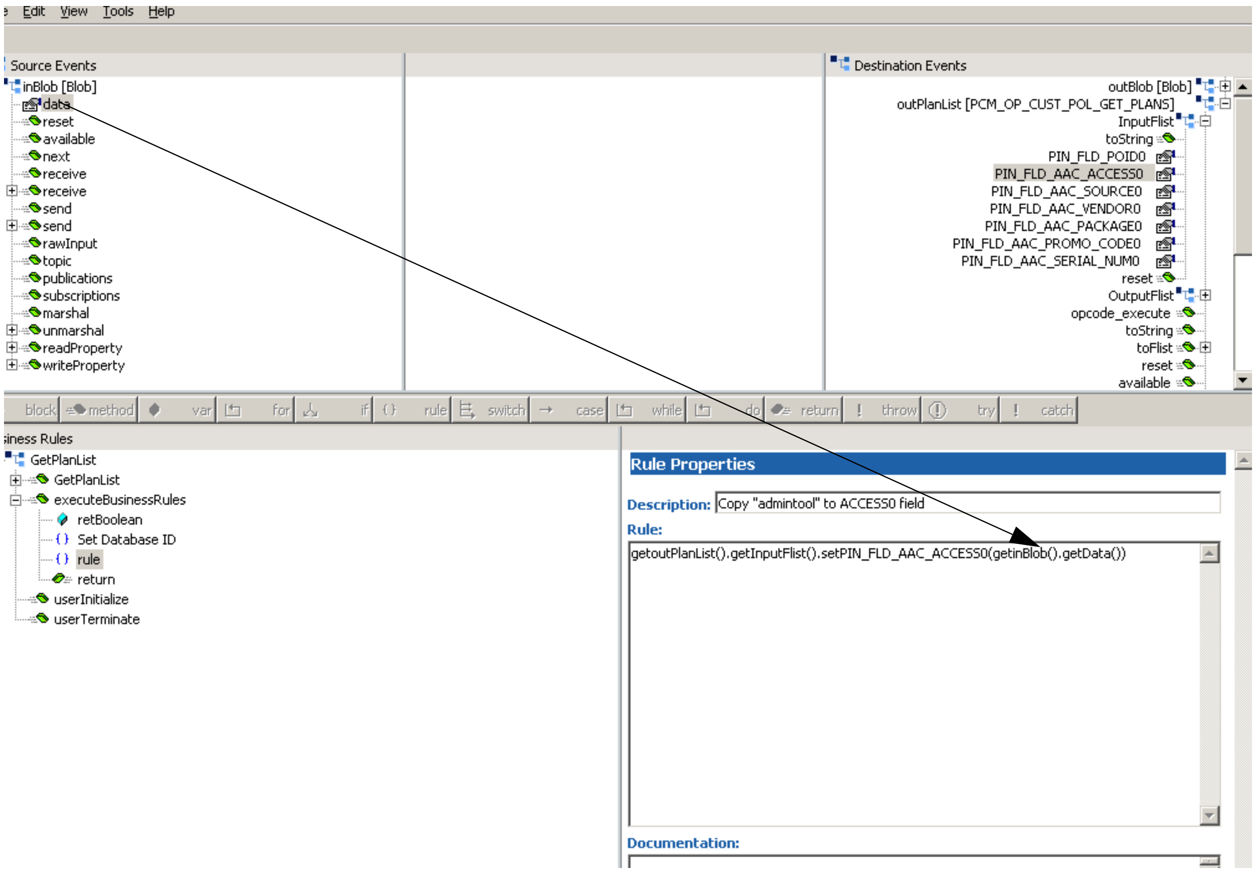
Rule:

```
getoutPlanList().getInputFlist().setPIN_FLD_AAC_ACCESS0()
```

9 Change the description of the method from **rule** to **Copy "admintool" to ACCESS0 field**. Reselect the **rule** to affect the updated description name.

10 Drag the data property located under the inBlob Source Event into the setPIN_FLD_AAC_ACCESS0 parentheses.

Figure 15 Collaboration Rules — Collaboration Mapping Properties



Description: Copy "admintool" to ACCESS0 field

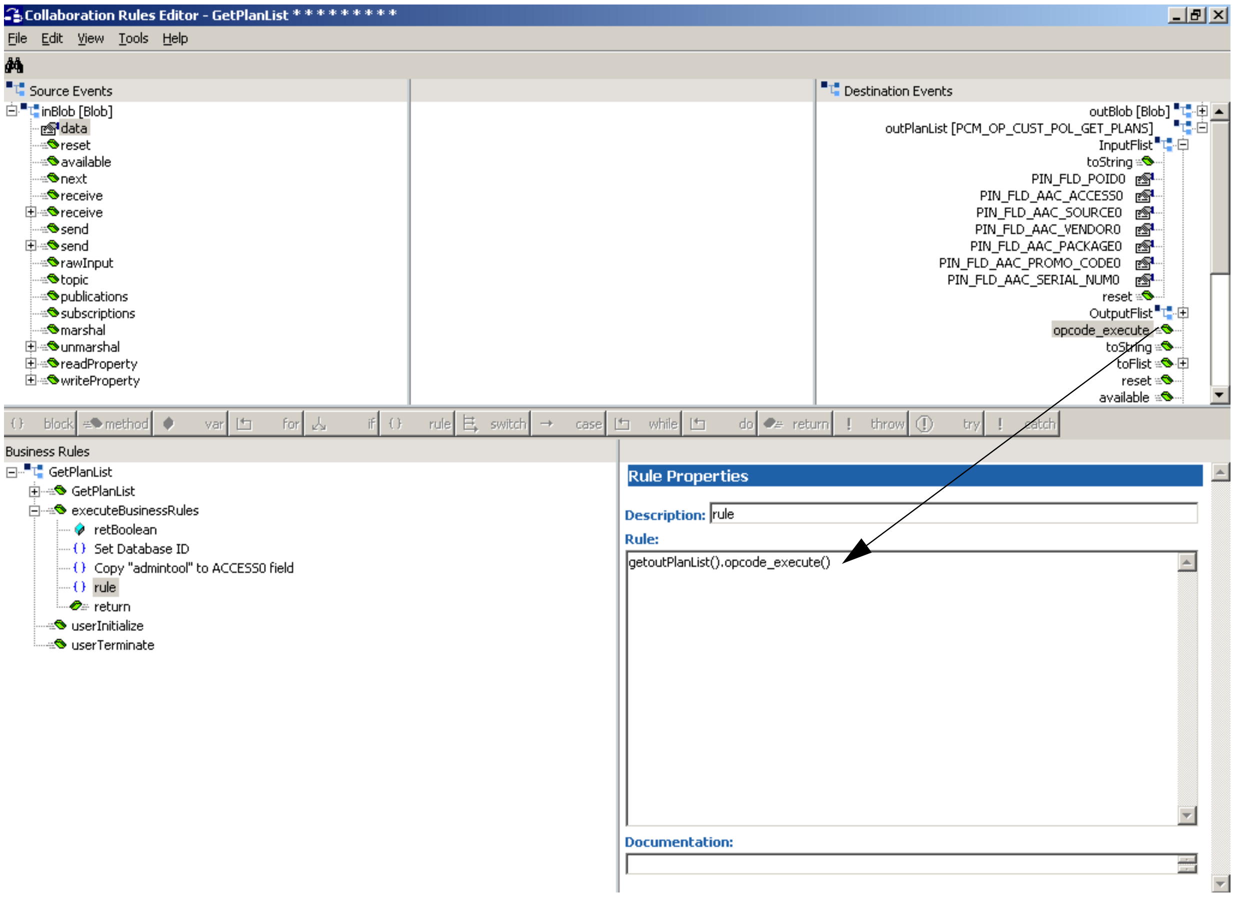
Rule:

```
getoutPlanList().getInputFlist().setPIN_FLD_AAC_ACCESS0(getinBlob().getData())
```

11 Select {} **rule** from the center pane. A rule appears below the **Copy "admintool" to ACCESS0 field** in the Business Rules pane.

12 Drag the OPCODE_execute method into the rules dialog box.

Figure 16 Collaboration Rules — Collaboration Mapping Properties



Description: rule

Rule:

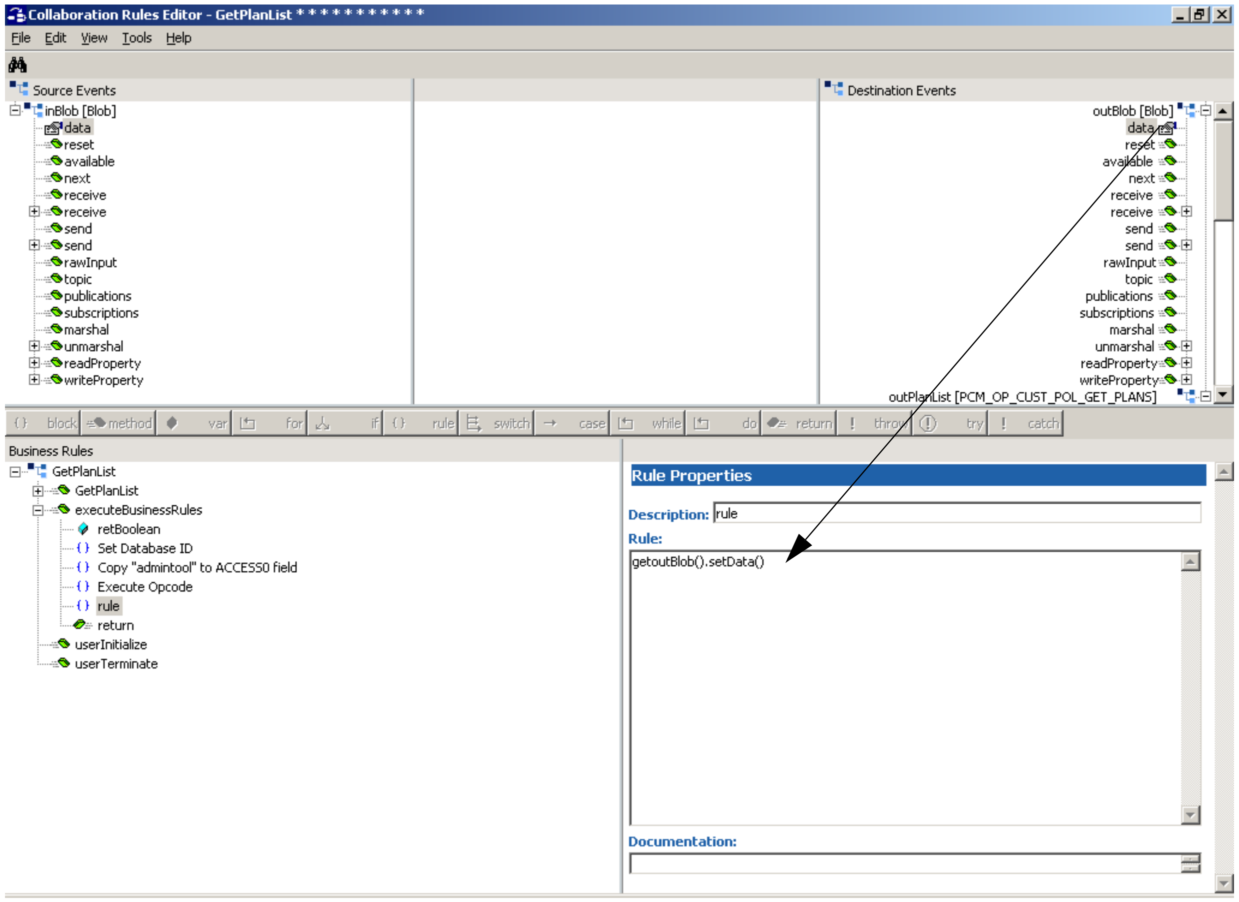
```
getoutPlanList().opcode_execute()
```

13 Change the description of the method from **rule** to **Execute OPCODE**. Reselect the **rule** to affect the updated description name.

14 Select {} **rule** from the center pane. A rule appears below the **Execute OPCODE** in the Business Rules pane.

15 Drag the data property located under the Destination Event into the rules dialog box.

Figure 17 Collaboration Rules — Collaboration Mapping Properties



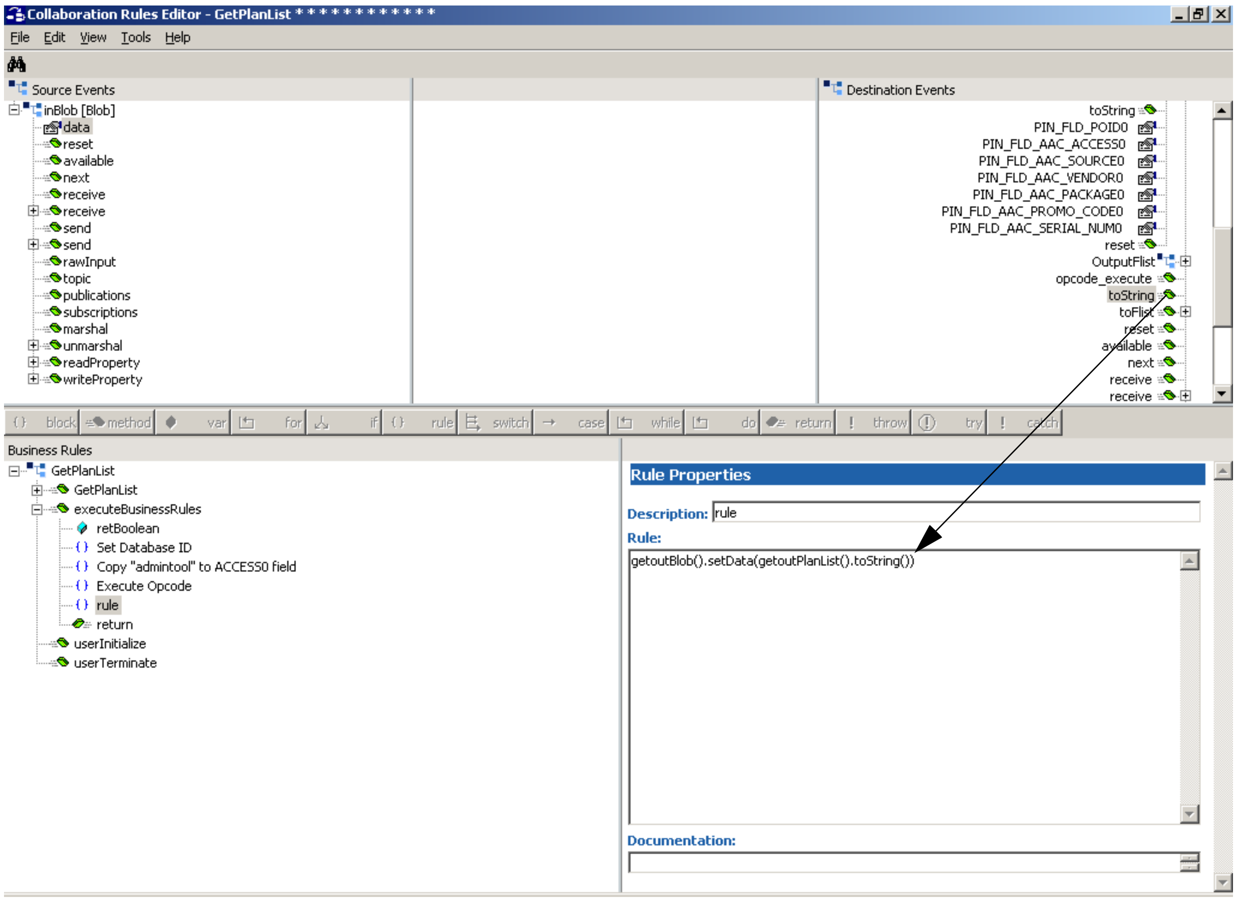
Description: rule

Rule:

```
getoutBlob().setData()
```

16 Drag the **data** property located under the **Destination Event** into the rules dialog box.

Figure 18 Collaboration Rules — Collaboration Mapping Properties



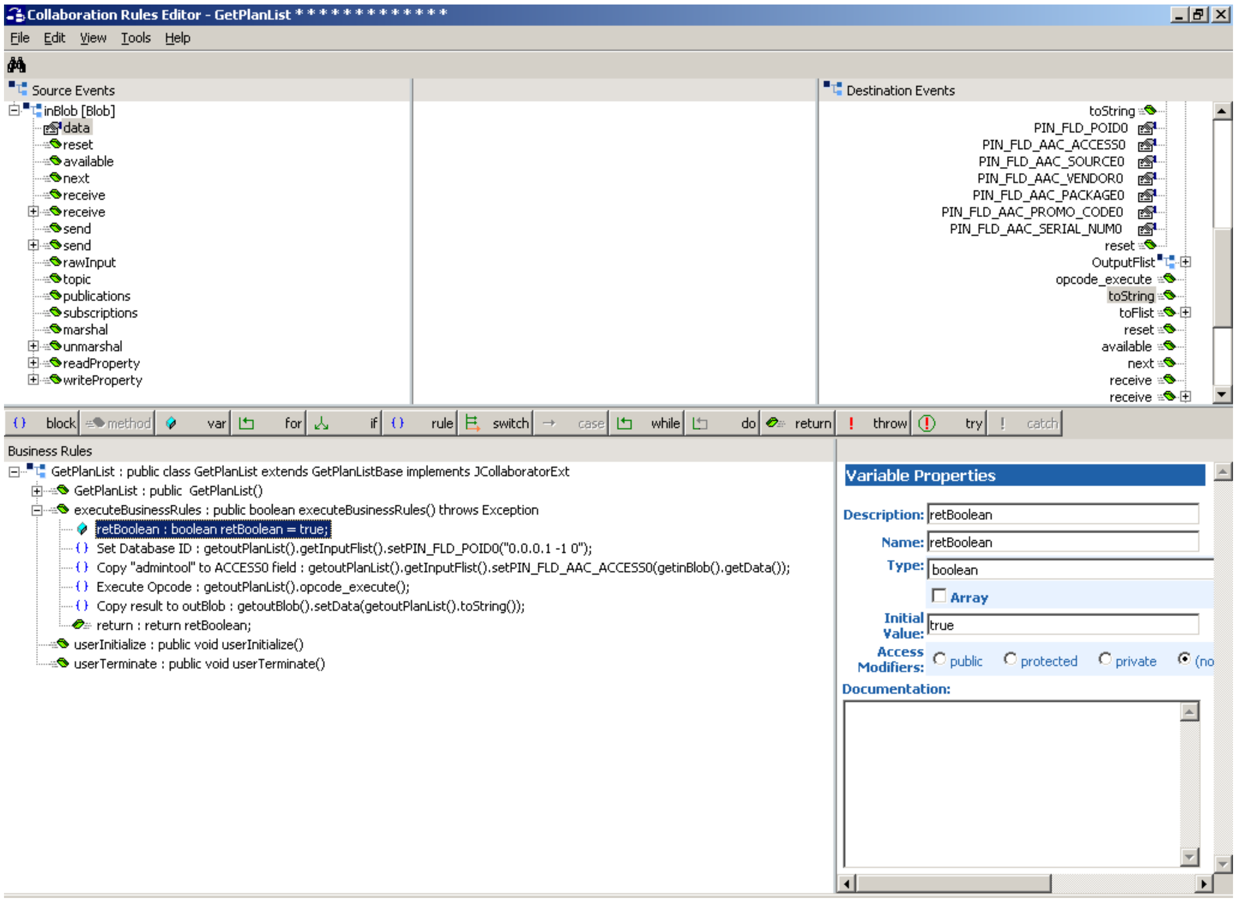
Description: rule

Rule:
```
getoutBlob().setData(getoutPlanList().toString())
```

17 Change the description of the method from **rule** to **Copy result to outBlob**. Reselect the **rule** to affect the updated description name.

18 Select View from the Menu, select Display Code.

Figure 19 Collaboration Rules — Collaboration Mapping Properties



```
GetPlanList : public class GetPlanList extends GetPlanListBase implements JCollaboratorExt
    GetPlanList : public  GetPlanList()
    executeBusinessRules : public boolean executeBusinessRules() throws Exception
        retBoolean : boolean retBoolean = true;
        Set Database ID : getoutPlanList().getInputFlist().setPIN_FLD_POID0("0.0.0.1 -1 0");
        Copy "admintool" to ACCESS0 field : getoutPlanList().getInputFlist().setPIN_FLD_AAC_ACCESS0(getinBlob().getData());
        Execute Opcode : getoutPlanList().opcode_execute();
        Copy result to outBlob : getoutBlob().setData(getoutPlanList().toString());
        return : return retBoolean;
    userInitialize : public void userInitialize()
    userTerminate : public void userTerminate()
```

Variable Properties

Description:	retBoolean
Name:	retBoolean
Type:	boolean
	☐ Array
Initial Value:	true
Access Modifiers:	○ public ○ protected ○ private ● (no
Documentation:	

19 When all the business logic has been defined, the code can be compiled by selecting **Compile** from the **File** menu. The **Save** menu opens, provide a name for the **.xpr** file. For the sample, use **GetPlan.xpr**.

If the code compiles successfully, the message **Compile Completed** appears. If the outcome is unsuccessful, a Java Compiler error message appears.

Once the compilation is complete, save the file, promote to runtime and exit.

20 Under the **Collaboration Rules**, the path for the .class file created appears. (For the sample, the path **"collaboration_rules\GetPlan.class** "appears.)

21 Under **Initialization** file, the path for the **.ctl** file created appears. (For the sample the path **"collaboration_rules\GetPlan.ctl"** appears.)

22 Click **OK** to exit the **Properties** Box.

Augmenting OPCODE Specifications

Many OPCODEs serve as a temple to perform certain general function, for example, PCM_OP_CREATE_OBJ, PCM_OP_SEARCH. Portal does not specify the full input/output list definition in these OPCODE specifications. The users must know the exact object they want to create or perform the search upon, and augment those specifications to enable the OPCODE ETD Builder to correctly build the ETD based on the augmented specifications.

For example, in the original OPCODE specifications for PCM_OP_CREATE_OBJ, nothing is listed about new object structure. Instead you see the following comment:

```
! Fields for object being created are added to input flist here. See
! the object spec for the object type being created for a detailed
! list of Mandatory and Optional fields.
```

To follow Portal's recommendation, if we want to create a nameinfo object, this is what we add the specs: PCM_OP_CREATE_OBJ.input.html:

```
! manually added in SeeBeyond to illustrate create an account obj
field  PIN_FLD_NAME (
    type=        PIN_FLDT_STR,
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
field PIN_FLD_ACCOUNT_TYPE (
    type=    PIN_FLDT_ENUM,
    perms=   O,
);
! manually added in SeeBeyond to illustrate create an account obj
array PIN_FLD_NAMEINFO (
    type=        PIN_FLDT_ARRAY,
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Mr. or Mrs. or Ms.
field * PIN_FLD_SALUTATION (
    type=        PIN_FLDT_STR(25),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Last name
field * PIN_FLD_LAST_NAME (
    type=        PIN_FLDT_STR(60),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! First name
field * PIN_FLD_FIRST_NAME (
    type=        PIN_FLDT_STR(30),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Middle name
field * PIN_FLD_MIDDLE_NAME (
    type=        PIN_FLDT_STR(30),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Job title
field * PIN_FLD_TITLE (
    type=        PIN_FLDT_STR(60),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Company affiliation
field * PIN_FLD_COMPANY (
    type=        PIN_FLDT_STR(60),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Street address
field * PIN_FLD_ADDRESS (
    type=        PIN_FLDT_STR(255),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! City
field * PIN_FLD_CITY (
    type=        PIN_FLDT_STR(30),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! State / Province
field * PIN_FLD_STATE (
    type=        PIN_FLDT_STR(30),
    perms=       O,
);
! Zip / Postal code
field * PIN_FLD_ZIP (
    type=        PIN_FLDT_STR(12),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Country
field * PIN_FLD_COUNTRY (
    type=        PIN_FLDT_STR(60),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Email address for this person
field * PIN_FLD_EMAIL_ADDR (
    type=        PIN_FLDT_STR(1023),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Contact type comment for this person (for undefined element-ids)
field * PIN_FLD_CONTACT_TYPE (
    type=        PIN_FLDT_STR(255),
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Phone array
array * PIN_FLD_PHONES (
    type=        PIN_FLDT_ARRAY,
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Phone Type
field * * PIN_FLD_TYPE (
    type=        PIN_FLDT_ENUM,
    perms=       O,
);
! manually added in SeeBeyond to illustrate create an account obj
! Phone Number
field * * PIN_FLD_PHONE (
    type=        PIN_FLDT_STR(25),
    perms=       M,
);

! end array PIN_FLD_PHONES
!
```

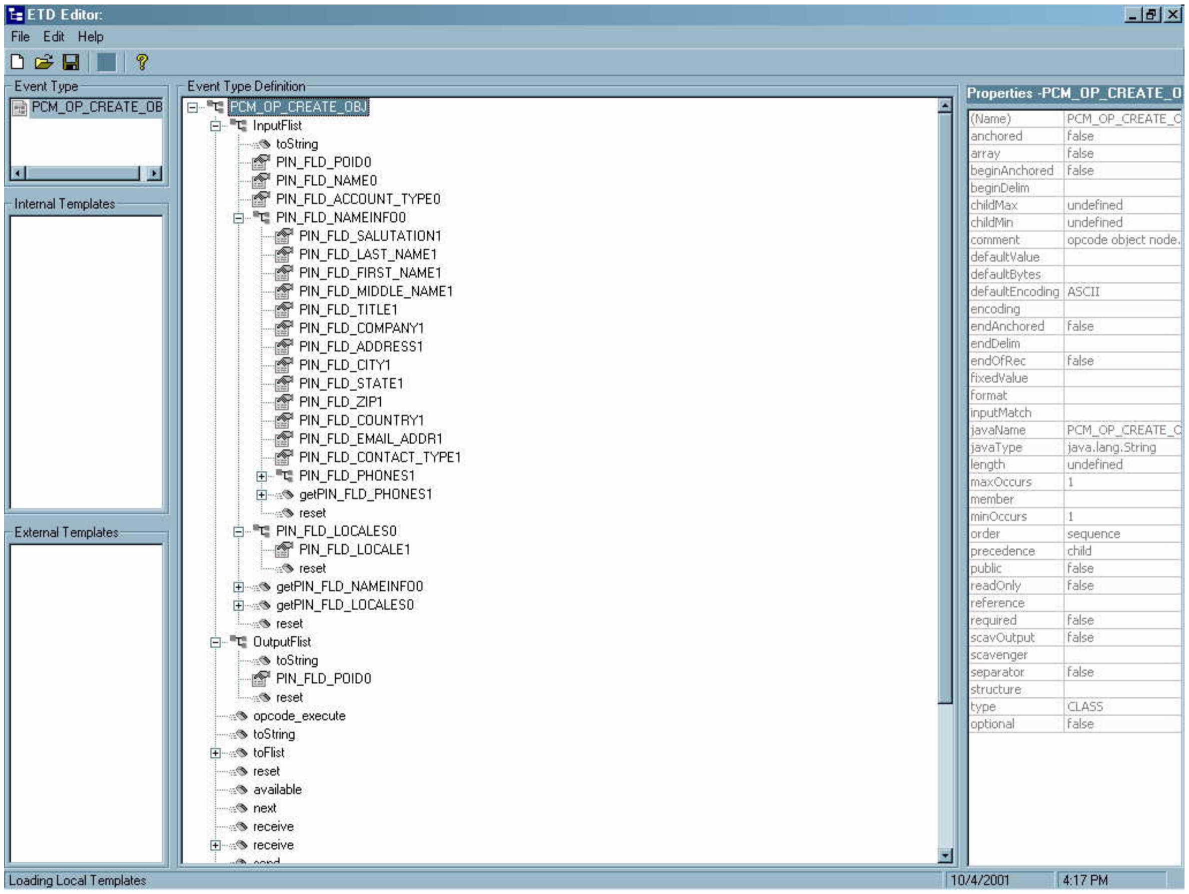
```
! manually added in SeeBeyond to illustrate create an account obj
! Locales array.
!
! There can only be one element in this array.
! The element-id is not significant.
array PIN_FLD_LOCALES (
        type    =               PIN_FLDT_ARRAY,
        perms   =               O,
);
! manually added in SeeBeyond to illustrate create an account obj
! New locale
field * PIN_FLD_LOCALE (
        type    =               PIN_FLDT_STR(255),
        perms   =               M,
);
! manually added in SeeBeyond to illustrate create an account obj
! end array PIN_FLD_NAMEINFO
```

Save the file using the same file name. Run the OPCODE ETD Builder on the newly augmented specifications.

Note: *The OPCODE ETD Builder assumes the prefix of the file name is the OPCODE name being called, therefore, the augmented file must be saved using the same file name. The file can be saved to a different directory to distinguish the original template specification from the final augmented specifications.*

When the OPCODE ETD Builder is invoked, you need to point to new directory to implement the augmented OPCODE. In the same manner, you need to change the PCM_OP_*.output.html, if the existing one does not contain the structure definition and therefore, is only a template. You need to save the PCM_OP_*.output.html to the same directory as the PCM_OP_*.input.html. The OPCODE ETD Builder parses both files, if either one is missing, the OPCODE ETD Builder will error out.

Figure 20 Creating an Augmented OPCODE Specification



```
PCM_OP_CREATE_OB

PCM_OP_CREATE_OBJ
  InputFlist
    toString
    PIN_FLD_POID0
    PIN_FLD_NAME0
    PIN_FLD_ACCOUNT_TYPE0
    PIN_FLD_NAMEINFO0
      PIN_FLD_SALUTATION1
      PIN_FLD_LAST_NAME1
      PIN_FLD_FIRST_NAME1
      PIN_FLD_MIDDLE_NAME1
      PIN_FLD_TITLE1
      PIN_FLD_COMPANY1
      PIN_FLD_ADDRESS1
      PIN_FLD_CITY1
      PIN_FLD_STATE1
      PIN_FLD_ZIP1
      PIN_FLD_COUNTRY1
      PIN_FLD_EMAIL_ADDR1
      PIN_FLD_CONTACT_TYPE1
      PIN_FLD_PHONES1
      getPIN_FLD_PHONES1
      reset
    PIN_FLD_LOCALES0
      PIN_FLD_LOCALE1
      reset
    getPIN_FLD_NAMEINFO0
    getPIN_FLD_LOCALES0
    reset
  OutputFlist
    toString
    PIN_FLD_POID0
    reset
  opcode_execute
  toString
  toFlist
  reset
  available
  next
  receive
  receive
```

Properties -PCM_OP_CREATE_O

(Name)	PCM_OP_CREATE_C
anchored	false
array	false
beginAnchored	false
beginDelim	
childMax	undefined
childMin	undefined
comment	opcode object node.
defaultValue	
defaultBytes	
defaultEncoding	ASCII
encoding	
endAnchored	false
endDelim	
endOfRec	false
fixedValue	
format	
inputMatch	
javaName	PCM_OP_CREATE_C
javaType	java.lang.String
length	undefined
maxOccurs	1
member	
minOccurs	1
order	sequence
precedence	child
public	false
readOnly	false
reference	
required	false
scavOutput	false
scavenger	
separator	false
structure	
type	CLASS
optional	false

An ETD built this way contains all the object structure required. When the builder invokes the OPCODEs, the data is passed to Portal server.

5.1.8 Collaborations

Collaborations are the components that receive and process Event Types, then forward the output to other e*Gate components or an external. Collaborations consist of the Subscriber, which “listens” for Events of a known type (sometimes from a given source), and the Publisher, which distributes the transformed Event to a specified recipient.

Create the InboundFileeWay collaboration, subscribe from External (which is our file containing the data corresponding to “admintool”), and publish to JMS as follows:

1 In the e*Gate Schema Designer, select the Navigator's **Components** tab.

2 Open the host on which you want to create the Collaboration.

3 Select a **Control Broker.**

4 Select the **InboundFileeWay** to assign the Collaboration.

5 On the palette, click the icon.

6. Enter the name of the new Collaboration, then click **OK.** (For the sample, "inboundCollab".)
7. Select the new **Collaboration**, then right-click to edit its properties.
8. From the **Collaboration Rules** list, select the necessary **Collaboration Rules** file that you created previously. (For the sample, "javaCollabPassThrough".)
9. In the **Subscriptions** area, click **Add** to define the input Event Types to which this Collaboration will subscribe.
   - A From the **Instance Name** list, select the Instance Name that you previously defined **JavaPassThroughIn.**
   - B From the **Event Type** list, select the **Event Type** that you previously defined (GenericInEvent).
   - C Select the **Source** from the **Source** list. In this case, it should be <External>.
10. In the **Publications** area, click **Add** to define the output **Event Types** that this Collaboration will publish.
    - A From the **Instance Name** list, select the **Instance Name** that you previously defined **JavaPassThroughOut.**
    - B From the **Event Types** list, select the **Event Type** that you previously defined (GenericInEvent).
    - C Select the publication destination from the **Destination** list. In this case, it should be <jmsConn>.
11. Click **OK** to exit.

Create the PortalClientConnectioneWay collaboration that subscribes from JMS, sends to Portal, receives a response back from Portal, and sends back to JMS for the Outbound e*Way to pick up as follows:

1. In the e*Gate Schema Designer, select the Navigator's **Components** tab.
2. Open the host on which you want to create the Collaboration.
3. Select a **Control Broker.**
4. Select the **PortalClientConnectioneWay** to assign the Collaboration.
5. On the palette, click the icon.
6. Enter the name of the new Collaboration, then click **OK.** (For the sample, "GetPlanList_CR".)
7. Select the new **Collaboration**, then right-click to edit its properties.
8. From the **Collaboration Rules** list, select the **Collaboration Rules** file that you created previously. (For the sample, "GetPlanList".)
9. In the **Subscriptions** area, click **Add** to define the input Event Types to which this Collaboration will subscribe.
   - A From the **Instance Name** list, select the Instance Name that you previously defined **inBlob.**

B From the **Event Type** list, select the **Event Type** that you previously defined (GenericInEvent).

C Select the **Source** from the **Source** list. In this case, it should be <jmsConn>.

10 In the **Publications** area, click **Add** to define the output **Event Types** that this Collaboration will publish.

A From the **Instance Name** list, select the Instance Name that you previously defined **outBlob.**

B From the **Event Type** list, select the **Event Type** that you previously defined (GenericOutEvent).

C Select the publication destination from the **Destination** list. In this case, it should be <jmsConn>.

11 In the **Publications** area, click **Add** to define the output **Event Types** that this Collaboration will publish.

A From the **Instance Name** list, select the Instance Name that you previously defined **outPlanList.**

B From the **Event Type** list, select the **Event Type** that you previously defined (GenericOutEvent).

C Select the publication destination from the **Destination** list. In this case, it should be <portalConn>.

Create the OutboundFileeWay collaboration as follows:

1 In the e*Gate Schema Designer, select the Navigator's **Components** tab.

2 Open the host on which you want to create the Collaboration.

3 Select a **Control Broker.**

4 Select the **OutboundFileeWay** to assign the Collaboration.

5 On the palette, click the icon.

6 Enter the name of the new Collaboration, then click **OK.** (For the sample, “outboundCollab”.)

7 Select the new **Collaboration**, then right-click to edit its properties.

8 From the **Collaboration Rules** list, select the **Collaboration Rules** file that you created previously. (For the sample, “javaCollabPassThrough”.)

9 In the **Subscriptions** area, click **Add** to define the input Event Types to which this Collaboration will subscribe.

A From the **Instance Name** list, select the Instance Name that you previously defined **JavaPassThroughIn.**

B From the **Event Type** list, select the **Event Type** that you previously defined (GenericOutEvent).

C Select the **Source** from the **Source** list. In this case, it should be <jmsConn>.

10 In the **Publications** area, click **Add** to define the output **Event Types** that this Collaboration will publish.

A From the **Instance Name** list, select the Instance Name that you previously defined **JavaPassThroughIn.**

B From the **Event Type** list, select the **Event Type** that you previously defined (GenericOutEvent).

C Select the **Source** from the **Source** list. In this case, it should be <External>.

5.2 Importing the Sample Schema

The previous sections provided the basics for creating a Portal Client e*Way schema. This section describes how to import and execute the sample schema for the Portal Client e*Way included on the Installation CD-ROM. It is assumed that the Portal Client e*Way has been installed properly, and that all of the necessary files and scripts are located in the default location.

Two versions of the Portal Client e*Way sample schema are provided with the Installation CD-ROM:

- **TestPortal.zip**: for use with Portal Infranet Application Server version 6.1
- **PortalSchema65.zip**: for Portal Infranet Application Server version 6.5, SP 2

Note: *PortalSchema65.zip requires J2SE v 1.4.X JRE (Sun Microsystems™ Java Runtime Environment). See the Readme included in the ewportal sample folder for more information.*

The sample schema consist of two file-based e*Ways, one Multi-Mode e*Way, two Collaboration Rules, two e*Way Connections and three Collaborations, as follows:

- **InboundFileeWay** - This e*Way will receive input from an external source, apply java pass through Collaboration Rules, and publish the information to a JMS Connection.
- **PortalClientConnectioneWay** - This Multi-Mode e*Way applies extended Java Collaboration Rules to an inbound Event to perform the desired business logic, in this case sets the database ID, copies a specified field, executes an OPCODE, and returns the data to a specified location.
- **OutboundFileeWay -** This e*Way will receive information from the JMS Connection and publish to the external system.
- **javaCollabPassThrough** - This Collaboration Rule is associated with both the *InboundFileeWay* and the *OutboundFileeWay* and is used for passing the Event without modification.
- **GetPlanList** - The Collaboration Rule is associated with the *PortalClientConnectioneWay* (a Multi-Mode e*Way), and is used to perform the transformation process.
- **portalConn** - This e*Way Connection is used to establish communication with the Infranet system.

- **jmsConn**- This JMS Connection is used to pass data from a Java-based collaboration within e*Gate.

5.2.1 Importing a Schema

The sample schema is located on the Installation CD in the following directory:

```
samples\ewportal
```

To load a sample schema:

To load a sample schema:

Start e*Gate Schema Designer

Create a new schema

File->Import Definitions from file

Choose the sample schema found on the distribution media:

```
TestPortal.zip or PortalSchema65.zip
```

The additional **opchtml.zip** file contains corrected versions of OPCODE specifications provided by Portal in the Infranet 6.1 or 6.5 Documentation installation subdirectory Infranet 6.1 or 6.5 Documentation\Documentation\html\flist.specs. See the Readme included in the ewportal sample folder for more information.

5.2.2 Execute the Schema

To execute the sample schema, do the following:

1. Go to the command line prompt, and enter the following:

   ```
   stccb -rh hostname -rs portalSample -un username -up user password
   -ln hostname_cb
   ```

   Substitute ***hostname***, ***username*** and ***user password*** as appropriate.
2. Exit from the command line prompt, and start the Schema Manager GUI.
3. When prompted, specify the ***hostname*** which contains the Control Broker you started in Step 1 above.
4. Select the sample schema.
5. After you verify that the Control Broker is connected (the message in the Control tab of the console will indicate command *succeeded* and status as *up*), highlight the IQ Manager, ***hostname***_igmgr, then click on the right button of the mouse, and select **Start**.
6. Select each of the e*Ways, right-click the mouse, and select **Start**.
7. To view the output, copy the output file (specified in the OutboundFileeWay e*Way configuration file). Save to a convenient location, open.

Note: *While the schema is running, opening the destination file, will cause errors.*

Index