

SeeBeyond™ eBusiness Integration Suite

Creating an End-to-End Scenario with e*Gate Integrator

Release 4.5.2



The information contained in this document is subject to change and is updated periodically to reflect changes to the applicable software. Although every effort has been made to ensure the accuracy of this document, SeeBeyond Technology Corporation (SeeBeyond) assumes no responsibility for any errors that may appear herein. The software described in this document is furnished under a License Agreement and may be used or copied only in accordance with the terms of such License Agreement. Printing, copying, or reproducing this document in any fashion is prohibited except in accordance with the License Agreement. The contents of this document are designated as being confidential and proprietary; are considered to be trade secrets of SeeBeyond; and may be used only in accordance with the License Agreement, as protected and enforceable by law. SeeBeyond assumes no responsibility for the use or reliability of its software on platforms that are not supported by SeeBeyond.

e*Gate, e*Insight, e*Way, e*Xchange, e*Xpressway, eBI, iBridge, Intelligent Bridge, IQ, SeeBeyond, and the SeeBeyond logo are trademarks and service marks of SeeBeyond Technology Corporation. All other brands or product names are trademarks of their respective companies.

© 1999–2002 by SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

This work is confidential and proprietary information of SeeBeyond and must be maintained in strict confidence.

Version 20020222144234.

Contents

List of Figures	6
------------------------	----------

List of Tables	8
-----------------------	----------

Chapter 1

Introduction	9
Document Purpose and Scope	9
Intended Audience	9
Organization of Information	9
Writing Conventions	10
Supporting Documents	11
Prerequisites	12
SeeBeyond Web Site	12

Chapter 2

Building a Java End-to-End Scenario	13
Java End-to-End Scenario Business Problem	13
Java End-to-End Scenario e*Gate Solution	13
Road Map For Setting Up the Scenario	14
Verify the e*Gate Installation	15
Create a New Schema	15
Create the Event Types and Java ETDs	16
Create the SysA.xsc ETD	16
Create the SysB.xsc ETD	18
Create the Collaboration Rules	20
Create the Java Pass Through Collaborations	21
Create the Java Collaboration Rule	22
Add the e*Ways and e*Way Connection	26
Add and Configure the e*Ways	27
Add the Multi-Mode e*Way	29
Configure the IQ Manager	30

Add the JMS e*Way Connection	30
Add the Collaborations that Route the Data	30
Add and Configure col_FileIn	31
Add and Configure col_JavaA2B	32
Add and Configure col_FileOut	32
Test the Scenario	33
Review the Complete Schema	33
Test the Schema	34
Start the Schema	34
Troubleshoot any problems	35
Java Troubleshooting	35

Chapter 3

e*Gate ELS End-to-End Scenario	36
ELS End-to-End Scenario Business Problem	36
ELS End-to-End Scenario e*Gate Solution	37
Road Map For Setting Up the Scenario	38
Verify the e*Gate Installation	38
Create a New Schema	39
Create the Event Types and Java ETDs	39
Create the Rec.xsc ETD	39
Create the Pur.xsc ETD	42
Create the Collaboration Rules	43
Create the Java Pass Through Collaboration Rules	44
Create the Java Collaboration Rule	45
Using the ELS Wizard	47
Edit the executeBusinessRules method	48
Add and Configure the e*Ways and e*Way Connection	51
Add and Configure the e*Ways	52
Add and Configure the ewRec file e*Way	52
Add and Configure the ewPur file e*Way	53
Add and Configure the ewELS Multi-Mode e*Way	54
Add the e*Way Connection	54
Add the Collaborations that Route the Data	55
Add and Configure colRec	55
Add and Configure colELS	56
Add and Configure colPur	57
Test the Scenario	57
Review the Complete Schema	57
Test the Schema	59
Start the Schema	59
Troubleshoot any problems	60
Java Troubleshooting	60

Chapter 4

Monk End-to-End Scenario	61
Monk End-to-End Scenario Business Problem	61
Road Map For Setting Up the Scenario	62
Verify the e*Gate Installation	63
Create a New Schema	63
Create the Event Types and Monk ETDs	63
Create the Monk ETD	64
Create the Event Types	65
Create the et_Valid and et_Invalid Event Types	66
Create the Collaboration Rules	66
Create the Pass Through Collaboration Rules	67
Create the Monk Collaboration Rule	68
Add the e*Ways and IQs	72
Add and Configure the e*Ways	72
Add the IQ	74
Add the Collaborations that Route the Data	75
Add and Configure col_Monk	76
Add and Configure col_Invalid	77
Add and Configure col_Valid	78
Test the Scenario	78
Review the Complete Schema	78
Test the Schema	80
Start the Schema	80
Step 10: Troubleshoot any problems	81

Chapter 5

Debugging and Log Files	83
Log File Locations	83
Generating Log Files	83
Index	85

List of Figures

Figure 1	Java End-to-End Business Problem	13
Figure 2	Java End-to-End Scenario Overview	14
Figure 3	Event Types and Java ETDs	16
Figure 4	SysA ETD Before Modifications	17
Figure 5	Completed SysA ETD	18
Figure 6	Completed SysB ETD	20
Figure 7	Collaboration Rules	21
Figure 8	Completed Collaboration Mapping Tab	22
Figure 9	JavaA2B Before Adding User-Defined Code	23
Figure 10	A Copy Rule is Added to cr_JavaA2B	24
Figure 11	Completed Rule	25
Figure 12	JavaA2B After Adding User-Defined Code	25
Figure 13	Completed cr_JavaA2B Properties	26
Figure 14	e*Ways and JMS e*Way Connection	27
Figure 15	ew_FileIn Configuration File	28
Figure 16	Collaborations Showing Publish and Subscribe Relationships	31
Figure 17	JavaE2Eoutput#.dat File	35
Figure 18	ELSE2E Business Problem	36
Figure 19	ELS Solution	37
Figure 20	ELS End-to-End Scenario Overview	37
Figure 21	Event Types and Java ETDs	39
Figure 22	Rec ETD Before Modifications	40
Figure 23	Completed Rec ETD	41
Figure 24	Completed Pur ETD	43
Figure 25	Collaboration Rules and Java Collaboration Rules Class	44
Figure 26	Completed Collaboration Mapping Tab	45
Figure 27	ELS Collaboration Rule Before Adding User-Defined Code	46
Figure 28	Java Collaboration Rules Editor After Enabling ELS	46
Figure 29	ELS Wizard Step 1	47
Figure 30	ELS Wizard Step 2	48
Figure 31	Copying to Repeating Node	49
Figure 32	After Editing the executeBusinessRules Method	50

List of Figures

Figure 33	Completed crELS Properties	51
Figure 34	e*Ways and e*Way Connection	52
Figure 35	ewRec Configuration File	53
Figure 36	Collaborations Showing Pub/Sub Relationships	55
Figure 37	ELS Output File	60
Figure 38	Monk End-to-End Scenario Overview	61
Figure 39	e*Gate Enterprise Manager	63
Figure 40	Event Types and Monk ETDs	64
Figure 41	New ETD dialog box	65
Figure 42	Collaboration Rules and Monk CRSs	67
Figure 43	Completed cr_Monk Publications tab	68
Figure 44	New Collaboration Rules Script Dialog Box	69
Figure 45	Monk Collaboration Rules Editor	69
Figure 46	Monk CRS After Adding User-Defined Code	71
Figure 47	Completed cr_Monk Properties	71
Figure 48	e*Ways and IQs	72
Figure 49	ew_Monk Configuration File	73
Figure 50	Collaborations Showing Pub/Sub Relationships	75
Figure 51	Collaboration - col_Monk Properties Dialog Box	77
Figure 52	MonkE2E Output Files in Windows Explorer	81
Figure 53	Invalid_output0.dat File	81
Figure 54	Valid_output0.dat File	81
Figure 55	Logging Options	84

List of Tables

Table 1	SysB ETD Fixed Node Properties	19
Table 2	JavaE2E Components	33
Table 3	Compiler Errors	35
Table 4	ELSE2E Components	57
Table 5	Compiler Errors	60
Table 6	JavaE2E Components	78

Introduction

This chapter introduces you to this guide, its general purpose and scope, and its organization. It also provides sources of related documentation and information.

1.1 Document Purpose and Scope

This document is a step-by-step guide to creating a simple working e*Gate schema. Using this document you should be able to set up and run the two end-to-end scenarios herein described. The scenarios are designed to be run on one machine on which e*Gate has been installed.

This document is not a GUI tutorial or a reference guide for basic e*Gate features. You should turn to the various e*Gate User's Guides and the e*Gate Primer for general information. To the extent that e*Gate features are discussed, only the features needed to create the end-to-end scenarios from scratch are discussed.

Three scenarios are described, two scenarios use a Java Collaboration environment and the third uses Monk (SeeBeyond's own scripting language) Collaboration environment.

1.2 Intended Audience

You must be familiar with how to use the e*Gate Enterprise Manager.

The reader of this guide is presumed to be an experienced PC user with some basic knowledge of e*Gate and to be able to navigate within the e*Gate Enterprise Manager GUI. This person must also have enough knowledge of Java, Windows NT/Windows 2000 or UNIX operations to run an e*Gate system and be thoroughly familiar with Windows-style GUI operations.

1.3 Organization of Information

This document is organized topically as follows:

- [Chapter 1 "Introduction" on page 9](#) — Gives a general preview of this document, its purpose, scope, and organization.

- **Chapter 2 “Building a Java End-to-End Scenario” on page 13** — Provides instructions on how to build a simple scenario that uses the Java Collaboration service to do data transformation.
- **Chapter 3 “e*Gate ELS End-to-End Scenario” on page 36** — Provides instructions on how to build a simple scenario that uses Event Linking and Sequencing to combine transactions that can arrive in many parts over time.
- **Chapter 4 “Monk End-to-End Scenario” on page 61**— Provides instructions on how to build a simple scenario that uses the Monk Collaboration service to do data routing.
- **Chapter 5 “Debugging and Log Files” on page 83**— Provides some introductory information on e*Gate troubleshooting.

1.4 Writing Conventions

The writing conventions listed in this section are observed throughout this document.

Hypertext Links

When you are using this guide online, cross-references are also hypertext links and appear in **blue text** as shown below. Click the **blue text** to jump to the section.

For information on these and related topics, see **“Parameter, Function, and Command Names” on page 11**.

Command Line

Text to be typed at the command line is displayed in a special font as shown below.

```
java -jar ValidationBuilder.jar
```

Variables within a command line are set in the same font and bold italic as shown below.

```
stcregutil -rh host-name -rs schema-name -un user-name  
-up password -ef output-directory
```

Code and Samples

Computer code and samples (including printouts) on a separate line or lines are set in Courier as shown below.

```
Configuration for BOB_Promotion
```

However, when these elements (or portions of them) or variables representing several possible elements appear within ordinary text, they are set in *italics* as shown below.

path and *file-name* are the path and file name specified as arguments to **-fr** in the **stcregutil** command line.

Notes and Cautions

Points of particular interest or significance to the reader are introduced with *Note*, *Caution*, or *Important*, and the text is displayed in *italics*, for example:

Note: *The Actions menu is only available when a Properties window is displayed.*

User Input

The names of items in the user interface such as icons or buttons that you click or select appear in **bold** as shown below.

Click **Apply** to save, or **OK** to save and close.

File Names and Paths

When names of files are given in the text, they appear in **bold** as shown below.

Use a text editor to open the **ValidationBuilder.properties** file.

When file paths and drive designations are used, with or without the file name, they appear in **bold** as shown below.

In the **Open** field, type **D:\setup\setup.exe** where **D:** is your CD-ROM drive.

Parameter, Function, and Command Names

When names of parameters, functions, and commands are given in the body of the text, they appear in **bold** as follows:

The default parameter **localhost** is normally only used for testing.

The Monk function **iq-put** places an Event into an IQ.

You can use the **stccb** utility to start the Control Broker.

Additional Conventions

Windows Systems — The e*Gate system is fully compliant with both Windows NT and Windows 2000 platforms. When this document refers to Windows, such statements apply to both Windows platforms.

UNIX and Linux Systems — This guide uses the backslash (“\”) as the separator within path names. If you are working on a UNIX or Linux system, please make the appropriate substitutions.

1.5 Supporting Documents

The following SeeBeyond documents provide additional information about the e*Gate Integrator system as explained in this guide:

- *e*Gate Integrator Collaboration Services Reference Guide*
- *e*Gate Integrator Installation Guide*
- *e*Gate Integrator Intelligent Queue Services Reference Guide*
- *e*Gate Integrator System Administration and Operations Guide*
- *e*Gate Integrator User’s Guide*
- *SeeBeyond eBusiness Integration Suite Primer*
- *SeeBeyond eBusiness Integration Suite Deployment Guide*
- *Monk Developer’s Reference*

- *Standard e*Way Intelligent Adapters User's Guide*

See the *SeeBeyond eBusiness Integration Suite Primer* for a complete list of e*Gate-related documentation. You can also refer to the appropriate Microsoft Windows or UNIX documents, if necessary.

1.6 Prerequisites

The following software components must be installed on the machine that runs this scenario. Refer to the *e*Gate Integrator Installation Guide* for instructions on how to install the e*Gate components. Refer to the documentation provided with the third-party software for instructions on how to install the other components.

- Registry Host
- Participating Host
- GUIs
 - ♦ Monitor
 - ♦ GUI

JDK 1.3.1 from Sun

JDK 1.3.1 (Java 2 SDK, Standard Edition, v 1.3.1) can be downloaded from the following source:

<http://java.sun.com/j2se/>

Internet Explorer 5.x or later

The Java Collaboration Editor requires files installed with IE 5 or later. IE 6.0 can be downloaded from the following source:

- ♦ <http://www.microsoft.com/windows/ie/>

1.7 SeeBeyond Web Site

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

<http://www.SeeBeyond.com/>

Building a Java End-to-End Scenario

This is a step-by-step description of how to set up and test a simple end-to-end e*Gate scenario using a Java Collaboration.

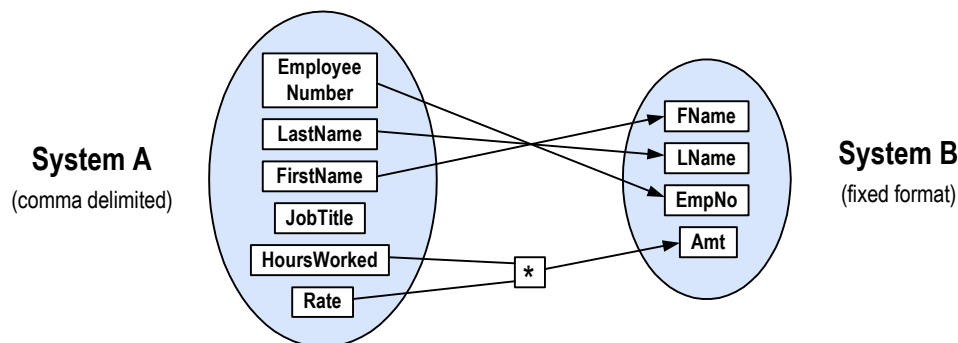
This scenario was developed under the Windows 2000 operating system.

Java End-to-End Scenario Business Problem

System A keeps track of the hours employees work per week; System B is responsible for paying them. The data from System A is in a delimited format and must be converted to a fixed format for System B. In addition, System B can only accept four fields: first name, last name, employee number, and amount. System A does not have an “amount” field, so the value for the amount field must be calculated before sending it to System B.

Figure 1 graphically depicts the data to be sent from a simple timecard system to a simple payroll system.

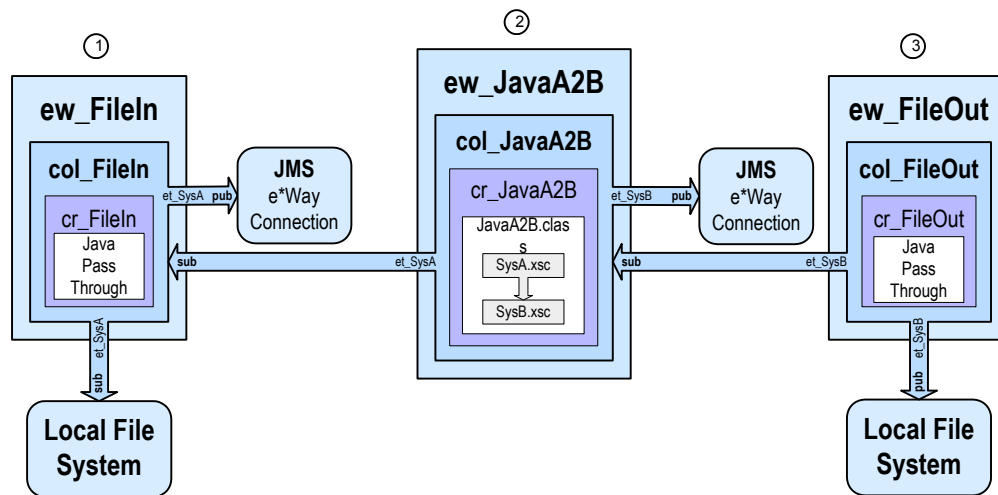
Figure 1 Java End-to-End Business Problem



Java End-to-End Scenario e*Gate Solution

The proposed e*Gate solution makes use of e*Gate Java Collaboration Service to transform the data from the System A format to the System B format. e*Gate is very flexible about where the actual transformation processing can occur as the data moves from System A to System B. The solution uses the Multi-Mode e*Way as the main transformation component and two Java Pass Through file e*Ways to bring data into and send data out from the e*Gate system. Figure 2 shows all the components and their relationships to one another in the complete e*Gate schema.

Figure 2 Java End-to-End Scenario Overview



Notes on the Java End-to-End Scenario Overview

- ① **ew_FileIn** brings data from System A into e*Gate.

The **col_FileIn** Collaboration in the **ew_FileIn** e*Way subscribes to a location on the local file system. It polls this location for a text file with extension ".fin" containing data from System A. Then it reads the message, packages the data as an **et_SysA** Event, and publishes the Event to the **JMS e*Way Connection**.

- ② **ew_JavaA2B** changes the data format and calculates the amount.

The **col_JavaA2B** Collaboration in the **ew_JavaA2B** e*Way subscribes to **et_SysA** Events published by **col_FileIn**. It uses the Java Collaboration Rule **cr_JavaA2B** to change **et_SysA** Events into **et_SysB** Events. This rule uses the **JavaA2B.class** which implements the transformation. **cr_JavaA2B** also computes the amount field in **et_SysB** by multiplying the hours and the rate from **et_SysA**. Finally, **col_JavaA2B** publishes the **et_SysB** Event to the **JMS e*Way Connection**.

- ③ **ew_FileOut** writes the transformed data out to local file system.

The **col_FileOut** Collaboration in the **ew_FileOut** e*Way subscribes to **et_SysB** Events published by **JMS e*Way Connection**. The **cr_FileOut** Collaboration Rule uses the Java Pass Through service to move the data without modifying it. When an **et_SysB** Event is retrieved, the e*Way packages it as a text file and writes it to the specified location on the local file system, completing the end-to-end scenario.

Road Map For Setting Up the Scenario

For the most part, you can create the components for an e*Gate scenario in any order you like. In cases where order is important, the GUI alerts you and gives you the opportunity to create the required component before proceeding. The following steps provide a useful guideline for creating e*Gate schemas.

The basic steps are:

- 1 Verify the e*Gate installation.
- 2 Create a new schema.
- 3 Create the Event Types and Java ETDs.
- 4 Create the Collaboration Rules.
- 5 Add and configure the e*Ways and the JMS e*Way Connection.
- 6 Add and define the Collaborations that route the data.
- 7 Review the complete schema.
- 8 Start the schema.
- 9 Test the schema.
- 10 Troubleshoot any problems.

By examining Figure 2 you notice that the road map works from the inside out when creating components. That is, the Event Types and Collaborations are created before creating the e*Ways that use them. This method has the advantage of letting you create all the components of the same type at the same time. It also ensures that the required components are available when you need them.

2.1 Verify the e*Gate Installation

This end-to-end scenario is designed to run on a single machine. Before beginning the configuration process, you must verify that you have all the required software installed on the target machine. Refer to the *e*Gate Integrator Installation Guide* for e*Gate system requirements and instructions to install the e*Gate components.

2.2 Create a New Schema

To create a new schema

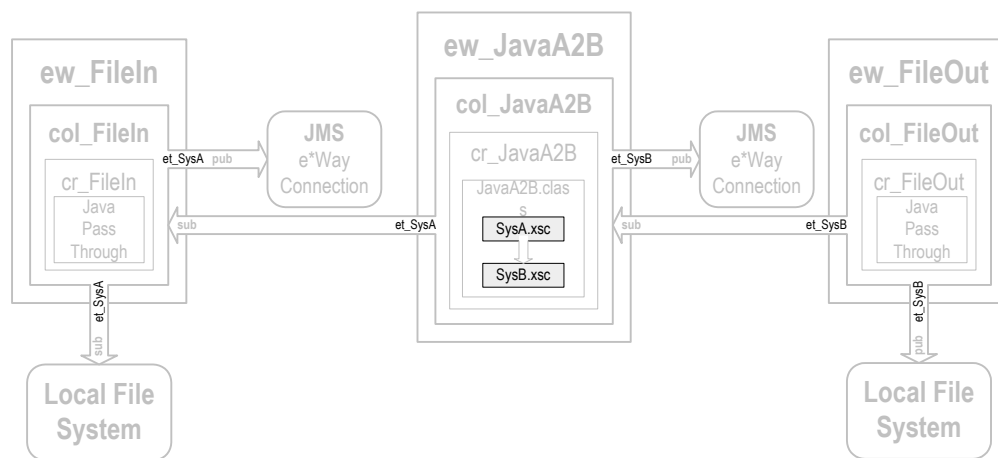
- 1 Start the e*Gate Enterprise Manager and log in as **Administrator** (or another user with administrator privileges) to the appropriate Registry Host.
- 2 In the **Open Schema on Registry Host** dialog box, click **New**.
- 3 In the **Enter New Schema Name** box, type **JavaE2E**, and then click **Open**.
The Enterprise Manager opens and displays the new **JavaE2E** schema.
- 4 At the bottom of the navigator (left) pane, click the **Components** tab.
You will perform all configuration steps in the **Components** tab.

2.3 Create the Event Types and Java ETDs

This scenario uses two Event Types. These Event Types require their own Event Type Definition (ETD). The first Event Type, **et_SysA**, models the comma-delimited format of the data received from System A as defined in **SysA.xsc**. The second Event Type, **et_SysB**, models the fixed-length data format required by System B as defined in **SysB.xsc**.

Figure 3 shows where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 3 Event Types and Java ETDs



2.3.1 Create the SysA.xsc ETD

Create the SysA Event Type

- 1 In the navigator pane of the Enterprise Manager, click the **Event Types** Folder.
- 2 On the **File** menu, point to **New**, and then click **Event Type**.
- 3 In **New Event Type Component** dialog box, type **et_SysA** for the Event Type name, and then click **OK**.
et_SysA is added to the list of Event Types.
- 4 In the list of Event Types, double-click **et_SysA**.
- 5 In the **Event Type - et_SysA Properties** dialog box, click **New**.
The Java ETD Editor displays.

Use the Standard ETD Wizard

- 1 On the **File** menu, click **New**.
- 2 In the **New Event Type Definition** dialog box, click the **Standard ETD** icon, and then click **OK**.

The Standard ETD Wizard appears.

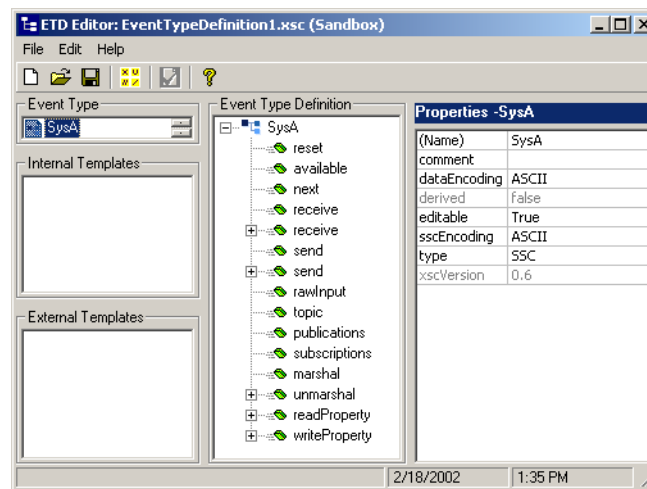
- 3 Read the **Introduction** window, and then click **Next**.
- 4 At the **Step 1** window, type **SysA** in the **Root Node Name** box.
- 5 Type **SysApackage** in the **Package Name** box, and then click **Next**.

The package name is the name you give to the collection of classes that make up a Java ETD. This group of Java classes is the result of the ETD creation process.

Important: The root node name *cannot* be the same as the package name entered in the Standard ETD Wizard.

- 6 At the **Step 2** screen, review the summary information, and then click **Finish**.
The new ETD is displayed in the Java ETD Editor as shown in Figure 4.

Figure 4 SysA ETD Before Modifications



Complete SysA.xsc in the Java ETD Editor

- 1 In the **Event Type Definition** area, right-click **SysA**, point to **Add Field**, and then click **As Child Node**.

A node with the default name **Field1** is added under the **SysA** root node.

- 2 Triple-click this new node, and then change its name to **EmployeeNumber**.

You are not required to make any other changes to the **EmployeeNumber** node's properties.

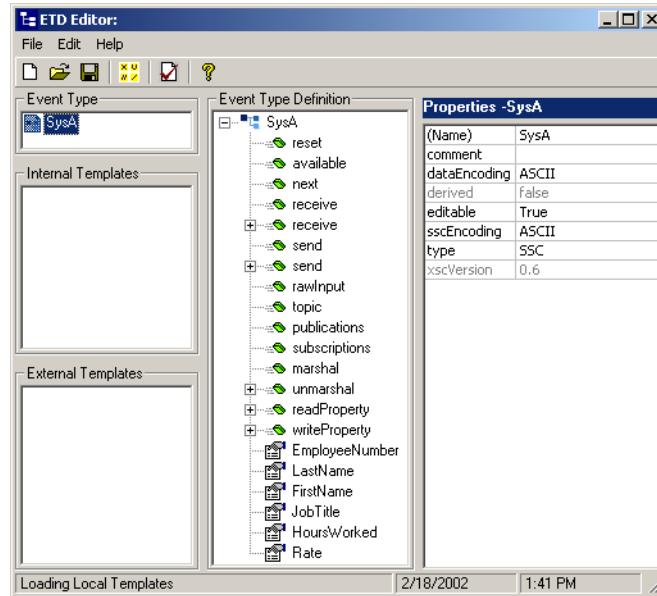
- 3 Repeat steps 1 through 3 for the other five nodes that must be added to this ETD. The additional node names are:


- ◆ LastName
- ◆ FirstName
- ◆ JobTitle
- ◆ HoursWorked

- ◆ Rate

The completed SysA ETD should look like the one shown in Figure 5.

Figure 5 Completed SysA ETD



- 4 Click the **Global Delimiter** button  from the toolbar.
 - A In the **Global Delimiter** dialog, click **Add Level**.
 - B Click **Add Single End Delimiter**.
 - C Click **Value** in the **Properties** pane and enter “,” (comma, no quotes).
- 5 Click **OK** to close the **Global Delimiter** dialog.
- 6 On the **File** menu, click **Compile and Save**.
- 7 In the **Save** dialog box, change the default name for the **.xsc** file, **EventTypeDefinition1.xsc**, to **SysA.xsc** and save to: **\etd\JavaE2E\SysA.xsc**.

- 8 Click **OK** to exit the **Save** dialog.

The SysA ETD is compiled and saved.
- 9 On the **File** menu, click **Close**.

The Java ETD Editor closes and the location of the file **SysA.xsc** is automatically entered in the **Event Type Definition** area of the **Event Type - et_SysA Properties** dialog box.

- 10 Click **OK** to close the **Event Type - et_SysA Properties** dialog box.

2.3.2 Create the SysB.xsc ETD

The SysB ETD corresponds to the fixed data structure used by System B.

Use the procedures described in [“Create the SysA Event Type” on page 16](#) and [“Use the Standard ETD Wizard” on page 16](#) to create a new Event Type named **et_SysB** and a corresponding Java ETD with Root Node name **SysB** and Package Name **SysBpackage**. Use the following procedure to complete the **SysB.xsc** in the Java ETD Editor.

Completing SysB.xsc in the Java ETD Editor

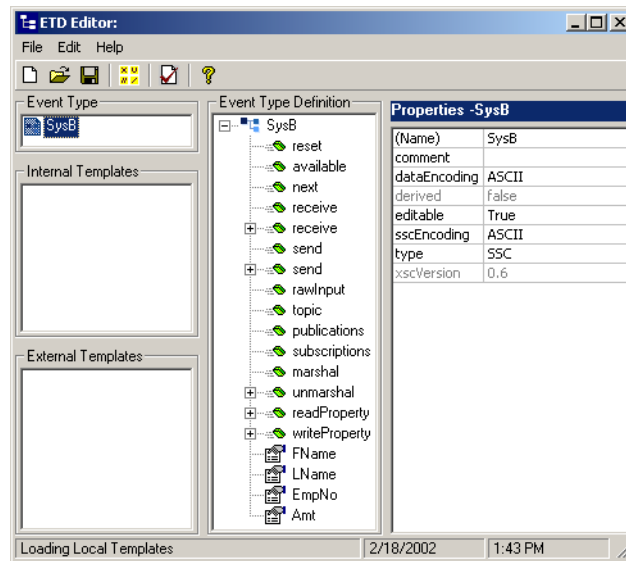
- 1 In the **Event Type Definition** area, select the **SysB** node, click **structure** in the **Properties** pane, and then click **fixed** in the list.
You are not required to make any other changes to the **SysB** root node’s properties.
- 2 Right-click **SysB**, point to **Add Field**, and then click **As Child Node**.
A node with the default name **Field1** is added to the ETD.
- 3 Right-click the **Field1** node and select **Rename**. Change its name to **FName**.
- 4 With the **FName** node still selected, click **length**, and then change the default value, **undefined**, to **25**.
- 5 Click **offset**, and then change the default value, **undefined**, to **0** (zero).
You are not required to make any other changes to the **FName** node’s properties.
- 6 Repeat steps 2 through 5 for the other 3 nodes that must be added to the SysB ETD.
Use the information in Table 1 to define these nodes.

Table 1 SysB ETD Fixed Node Properties

Node Name	structure	length	offset
FName	fixed	25	0
LName	fixed	25	25
EmpNo	fixed	10	50
Amt	fixed	10	60

When finished, the completed SysB ETD should look like the one shown in Figure 6.

Figure 6 Completed SysB ETD



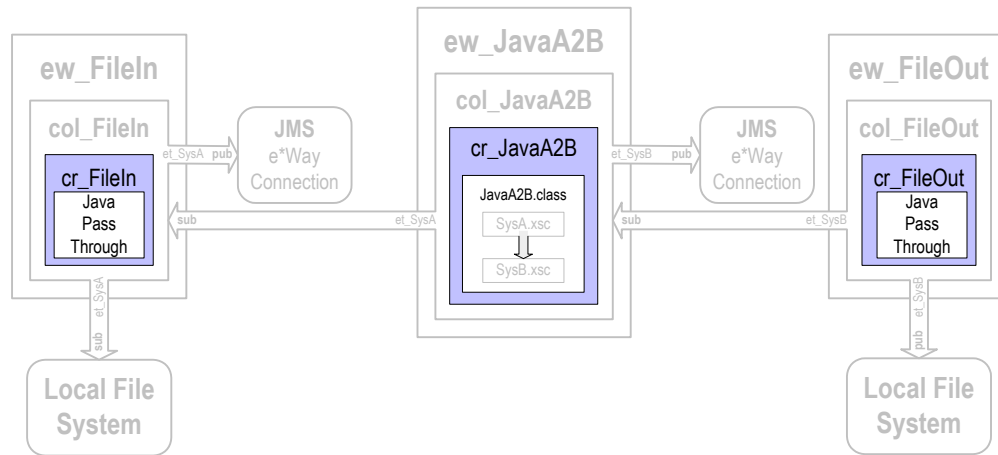
- 7 Compile and save the ETD as `\etd\JavaE2E\SysB.xsc`.
- 8 Close the Java ETD Editor.
- 9 Click **OK** to close the **Event Type - et_SysB Properties** dialog box.

2.4 Create the Collaboration Rules

This scenario uses three Collaboration Rules: two Java Pass Through rules and one Java Collaboration. The Java Pass Through rules, **cr_FileIn** and **cr_FileOut**, are used to route the Events through the e*Gate system and the Java Collaboration Rule **cr_JavaA2B** is used to transform the Event from Event Type **et_SysA** to Event Type **et_SysB**.

Figure 7 shows where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 7 Collaboration Rules



2.4.1 Create the Java Pass Through Collaborations

The Java Pass Through Collaborations are used to bring data into and take data away from the e*Gate system. The following procedure explains how to create the Java Pass Through Collaborations used in this scenario.

Create the cr_FileIn and cr_FileOut Collaboration Rules

- 1 In the navigator pane of the Enterprise Manager, click the **Collaboration Rules** folder.
- 2 On the **File** menu, point to **New**, and then click **Collaboration Rules**.
- 3 In the **New Collaboration Rules Component** dialog box, type **cr_FileIn** for the Collaboration Rule name, and then click **OK**.

cr_FileIn is added to the list of Collaboration Rules in the Enterprise Manager editor pane.

- 4 On the list of Collaboration Rules, double-click **cr_FileIn**.
- 5 In the **Collaboration Rules** section, click **Find** and explore to **collaboration_rules\STCLibrary** and double-click **STCJavaPassThrough.class**.

The path to **STCJavaPassThrough.class** displays in the **Collaboration Rules** section of the dialog, and the path to **STCJavaPassThrough.cfl** displays in the **Initialization File** section. The **STCJavaPassThrough.class** file configures the Collaboration Mapping Instances for you. You are not required to make any other changes to **cr_FileIn**.

- 6 Click **OK** to close the **Collaboration Rules - cr_FileIn Properties** dialog box.
- 7 Repeat steps 2 through 6 to create the **cr_FileOut** Collaboration Rule. Substitute **cr_FileOut** for the Collaboration Rule name.

2.4.2 Create the Java Collaboration Rule

The procedure for creating a Collaboration Rule that uses the Java Collaboration Service is different from creating other e*Gate Collaboration Rules. Use the following procedure to start the Java Collaboration Editor and create the Java Collaboration Rule used by this scenario.

Create cr_JavaA2B and Start the Java Collaboration Editor

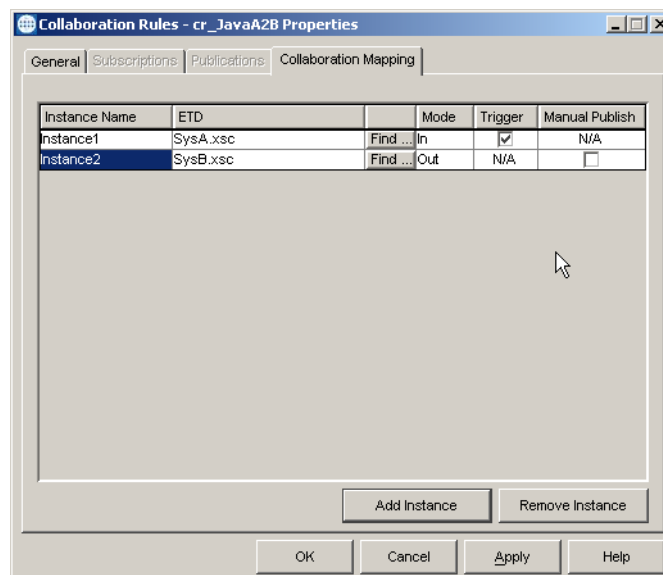
- 1 Use steps 1 through 5 from the procedure described in “[Create the cr_FileIn and cr_FileOut Collaboration Rules](#)” on page 21 to create a new Collaboration Rule named **cr_JavaA2B**.
- 2 Click the **Collaboration Mapping** tab, and then click **Add Instance**.
An instance row is added to the **Collaboration Mapping** tab.
- 3 In the **Instance Name** column, type **Instance1** for the instance name.
- 4 Click **Find** and explore to **etd\JavaE2E** and double-click **SysA.xsc**.
SysA.xsc is added to the **ETD** column of the instance row. You are not required to make any other changes to **Instance1**.
- 5 Add another ETD instance. Use **Instance2** for the instance name.

Important: The Java ETD instance names must be unique per schema.

- 6 Find and enter **SysB.xsc** as the ETD for **Instance2**.
- 7 Click in the **Mode** cell for **Instance2**, and then click **Out**.

You do not need to make any other changes to **Instance2**. The completed **Collaboration Mapping** tab looks like the one shown in Figure 8.

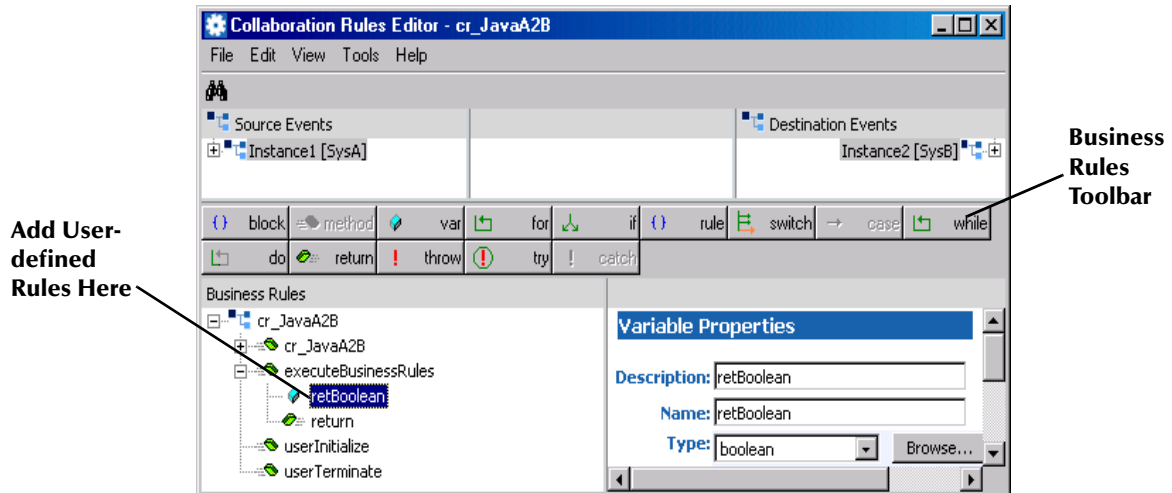
Figure 8 Completed Collaboration Mapping Tab



- 8 Click the **General** tab, and then in the **Collaboration Rules** area click **New**.

The Java Collaboration Editor opens a new Collaboration Rule with **Instance1 (SysA)** as the source Event and **Instance2 (SysB)** as the destination Event as shown in Figure 9.

Figure 9 JavaA2B Before Adding User-Defined Code



Create cr_JavaA2B.class in the Java Collaboration Rules Editor

- 1 On the **View** menu, click **Display Code**.

The **Business Rules** pane now displays the Java code in addition to the code labels.

- 2 In the **Source Events** and **Destination Events** panes, expand **Instance1** and **Instance2** to display the leaf nodes of the ETDs.
- 3 In the **Business Rules** pane, under the **executeBusinessRules** method, click the **retBoolean** variable.

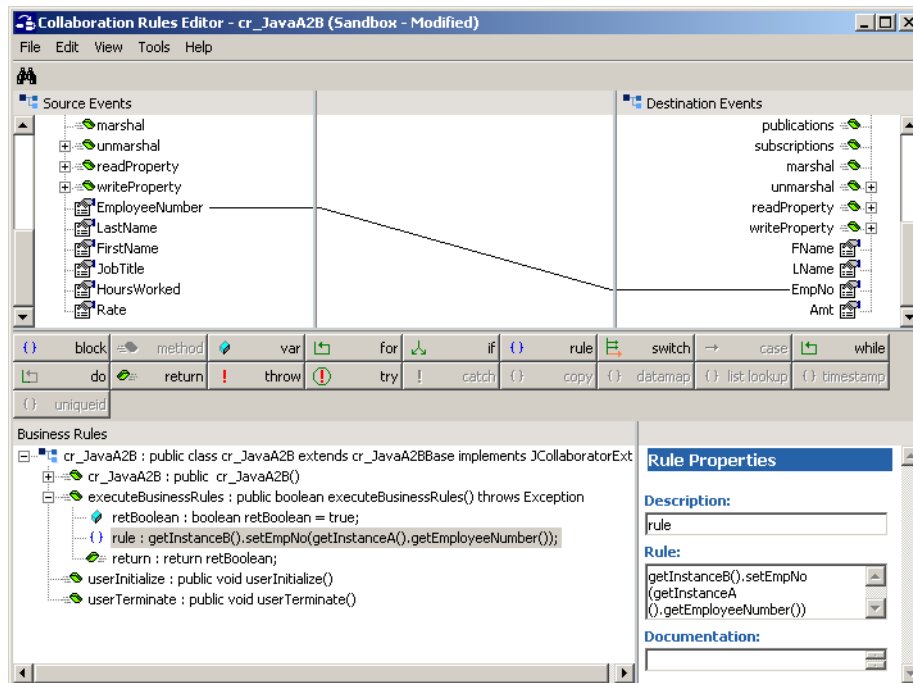
All the user-defined rules you add for this scenario are added within the **executeBusinessRules** method, and placed between the **retBoolean** variable and the **return** rule (see Figure 10).

- 4 With the **retBoolean** variable selected, drag the **EmployeeNumber** leaf node from the **Source Events** pane to the **EmpNo** leaf node in the **Destination Events** pane.

This results in the changes listed below. These changes are illustrated in Figure 10.

- ♦ A line is drawn connecting the two nodes.
- ♦ A rule that copies the contents of the **EmployeeNumber** node to the **EmpNo** node is added to the **executeBusinessRules** method between the **retBoolean** variable and the **return** rule.
- ♦ The text of the Java code is added to the Rule in the **Rule Properties** pane in the **Rule** box.

Figure 10 A Copy Rule is Added to cr_JavaA2B



- 5 Drag the **LastName** leaf node in the **Source Events** pane to the **LName** leaf node in the **Destination Events** pane.
- 6 Drag the **FirstName** leaf node in the **Source Events** pane to the **FName** leaf node in the **Destination Events** pane.
- 7 With the rule added in step 6 selected, click the **Rule** button on the **Business Rules** toolbar.

An empty rule placeholder is added to the **executeBusinessRules** method.

- 8 With the empty rule selected in the **Business Rules** pane, do the following:
 - A Drag the **Amt** leaf node in the **Destination Events** pane to the **Rule** field in the **Rule Properties** pane. The following text displays in the Rule field or your new rule:

```
getInstance2().setAmt
```

- B Position your cursor directly after the last letter of the **setAmt** method and type:

```
(" "+(Integer.parseInt(
```

- C Drag the **HoursWorked** leaf node in the **Source Events** pane to the **Rule** field in the **Rule Properties** pane.

- D Type:

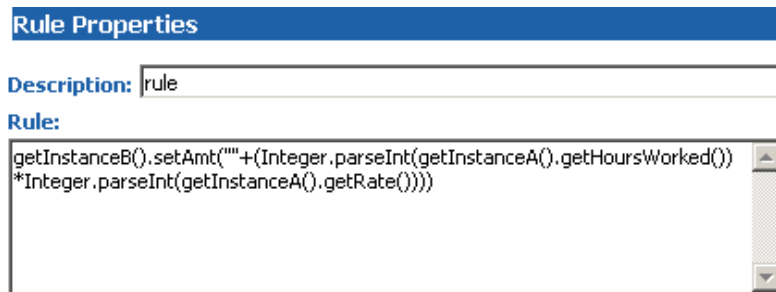
```
))*Integer.parseInt(
```

- E Drag the **Rate** leaf node in the **Source Events** pane to the **Rule** field in the **Rule Properties** pane.

- F Type:

```
)))
```


Figure 11 Completed Rule

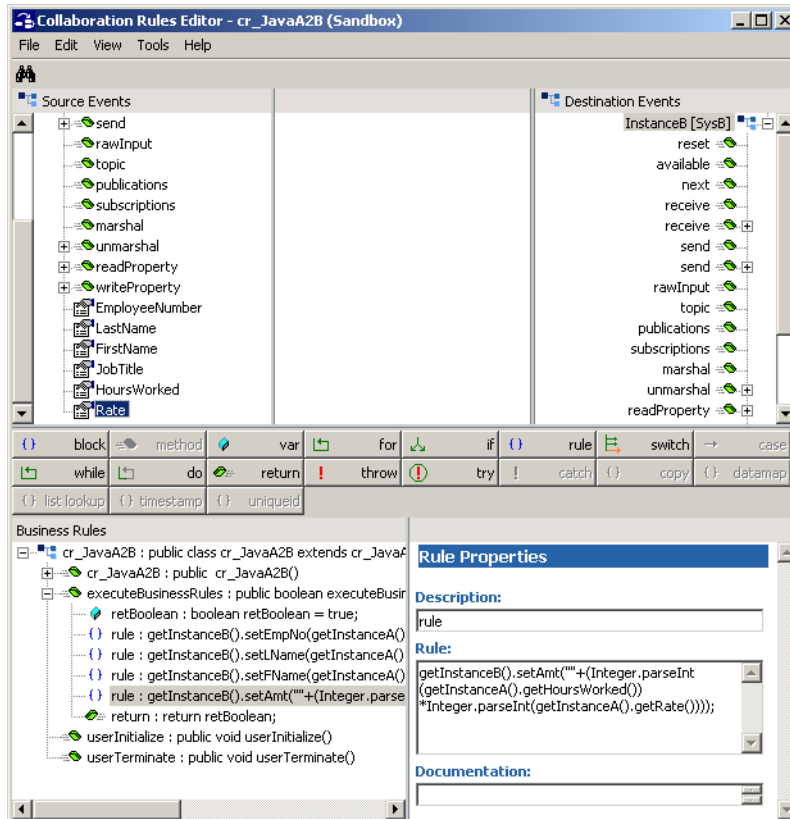


This rule multiplies the amounts from **HoursWorked** and **Rate** in the Source Event and copies the result to the **Amt** leaf node in the **Destination Event**.

- 9 On the **File** menu, click **Save**.
- 10 In the **Save** dialog box, navigate to the **collaboration_rules** folder, and then save the **cr_JavaA2B.xpr** file.
- 11 On the **File** menu, click **Compile**.

The Java source code is compiled. When the compiler is finished “Compile Completed” is displayed in the Compile/Debug pane. The Compile/Debug pane also displays any errors generated by the compilation process. Clear any errors before you continue.

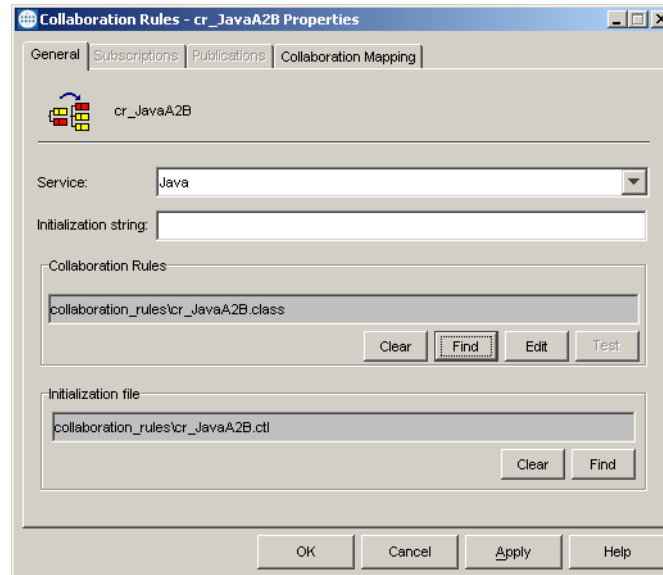
Figure 12 JavaA2B After Adding User-Defined Code



- 12 On the **File** menu, click **Exit**.

You may be prompted to save changes. The JCE closes and in the **Collaboration Rules - cr_JavaA2B Properties** dialog box, **collaboration_rules\cr_JavaA2B.class** is entered in the **Collaboration Rules** box and **collaboration_rules\cr_JavaA2B.ctl** is entered in the **Initialization file** box as shown in Figure 13.

Figure 13 Completed cr_JavaA2B Properties



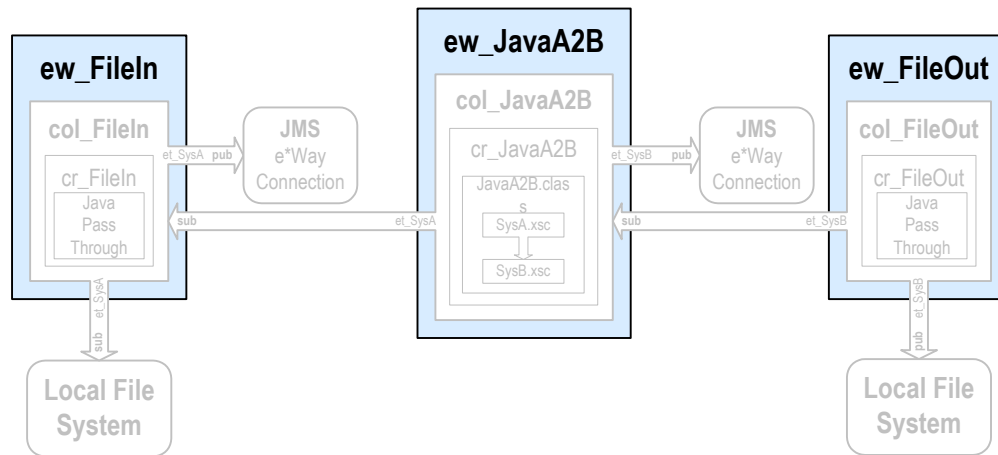
- 13 Click **OK** to close the **Collaboration Rules - cr_JavaA2B Properties** dialog box.

2.5 Add the e*Ways and e*Way Connection

After you have created your ETDs and Collaborations, you are ready to add and configure the e*Gate components that use these parts.

Figure 14 highlights the components added in this step.

Figure 14 e*Ways and JMS e*Way Connection



2.5.1 Add and Configure the e*Ways

Add and Configure the ew_FileIn file e*Way

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the Control Broker (*hostname_cb*).
- 2 On the **File** menu, point to **New**, point to **Module**, and then click **e*Way**.
- 3 In the **New e*Way Component** dialog box, type **ew_FileIn** for the e*Way name, and then click **OK**.

The **ew_FileIn** e*Way is added to the schema.

- 4 Right-click **ew_FileIn**, and then click **Properties**.
- 5 In the **e*Way - ew_FileIn Properties** dialog box, in the **Executable file** area, click **Find**.
- 6 In the **File Selection** dialog box, browse for and double-click the file **stcewfile.exe**.
The **bin\stcewfile.exe** file is added as the executable file, causing the component to become a file e*Way.

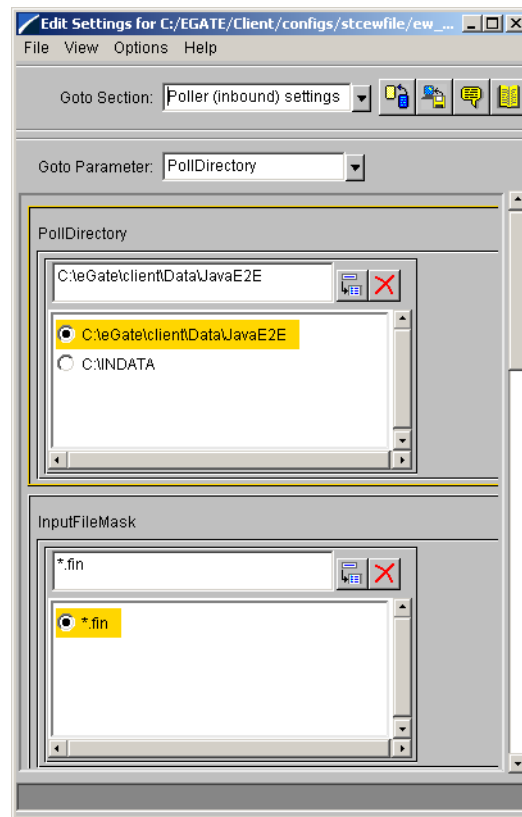
- 7 In the **Configuration file** area, click **New**.

The e*Way Configuration File Editor opens with a default file e*Way configuration file ready for editing.

- 8 In the **Goto Section** list, click **Poller (inbound) settings**.
- 9 In the **PollDirectory** box, type **C:\eGate\Client\Data\JavaE2E** and then press **ENTER**.

C:\eGate\Client\Data\JavaE2E is added as the directory to be polled to the **PollDirectory** list as shown in Figure 15. No other changes are necessary to the **ew_FileIn** e*Way's configuration file.

Figure 15 ew_FileIn Configuration File



- 10 On the **File** menu, click **Save**.
- 11 In the **Save As** dialog box, click **Save** to accept the default filename (**ew_FileIn.cfg**) and save the file.
- 12 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
The **configs\stcewfile\ew_FileIn.cfg** file is added to the **Configuration file** area in the **e*Way - ew_FileIn Properties** dialog box.
- 13 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 14 Click **OK** to close the **e*Way - ew_FileIn Properties** dialog box.

Add and Configure the ew_FileOut file e*Way

Adding the **ew_FileOut** e*Way follows the same general procedure as that outlined for adding the **ew_FileIn** e*Way above.

- 1 Use steps 1 through 7 from ["Add and Configure the ew_FileIn file e*Way" on page 27](#) to add another file e*Way named **ew_FileOut** and open its configuration file for editing.
- 2 In the e*Way Configuration File Editor, in **General Settings**, click **NO** for **AllowIncoming**, and **YES** for **AllowOutgoing**.
- 3 In the **Goto Section** list click **Outbound (send) settings**.
- 4 Add **C:\eGate\Client\Data\JavaE2E** as the default **OutputDirectory**.

- 5 Add **JavaE2Eoutput%d.dat** as the default **OutputFileName**.
No other changes are necessary to the **ew_FileOut** e*Way's configuration file.
- 6 On the **File** menu, click **Save**.
- 7 In the **Save As** dialog box, click **Save** to accept the default file name (**ew_FileOut.cfg**) and save the file.
- 8 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 9 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 10 Click **OK** to close the **e*Way - ew_FileOut Properties** dialog box.

2.5.2 Add the Multi-Mode e*Way

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the Control Broker (*hostname_cb*).
- 2 On the **File** menu, point to **New**, then point to **Module**, and then click **e*Way**.
- 3 In the **New e*Way Component** dialog box, type **ew_JavaA2B** for the e*Way name, and then click **OK**.

The **ew_JavaA2B** e*Way is added to the schema.

- 4 Right-click the **ew_JavaA2B** e*Way in the editor pane, and then click **Properties**.
- 5 In the **Configuration file** area, click **New**.

The e*Way Configuration File Editor opens with a default Multi-Mode e*Way configuration file.

- 6 Scroll to the bottom of the **JVM Settings** parameters and click **Remote debugging port number**.
- 7 In the **Remote debugging port number** box, type **8000**, and then press **ENTER**.

8000 is listed as the **Remote debugging port number**. No other changes are necessary to the **ew_JavaA2B** e*Way's configuration file.

Important: *In-schema debugging must be enabled on the participating host for this to work. See the e*Gate Integrator Installation Guide for more information.*

- 8 On the **File** menu, click **Save**.
- 9 In the **Save As** dialog box, click **Save** to accept the default filename (**ew_JavaA2B.cfg**) and save the file.
- 10 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 11 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 12 Click **OK** to close the **e*Way - ew_JavaA2B Properties** dialog box.

2.5.3 Configure the IQ Manager

- 1 In the e*Gate Enterprise Manager, in the navigation pane, double-click the IQ manager (*hostname_iqmgr*).
- 2 In the **Configuration File** area, click **New**.
- 3 From the **File** menu click **Save** and then click **Save** again to accept the default name.
- 4 Click the **Start Up** tab, then select the **Start automatically** checkbox, and then click **OK**.

2.5.4 Add the JMS e*Way Connection

- 1 In the navigator pane of the Enterprise Manager, click the **e*Way Connections** folder.
- 2 In the editor pane, right-click and point to **New**, and then click **e*Way Connection**.
- 3 In the **New e*Way Connection Component** dialog box, type **JMS** for the e*Way Connection name, and then click **OK**.

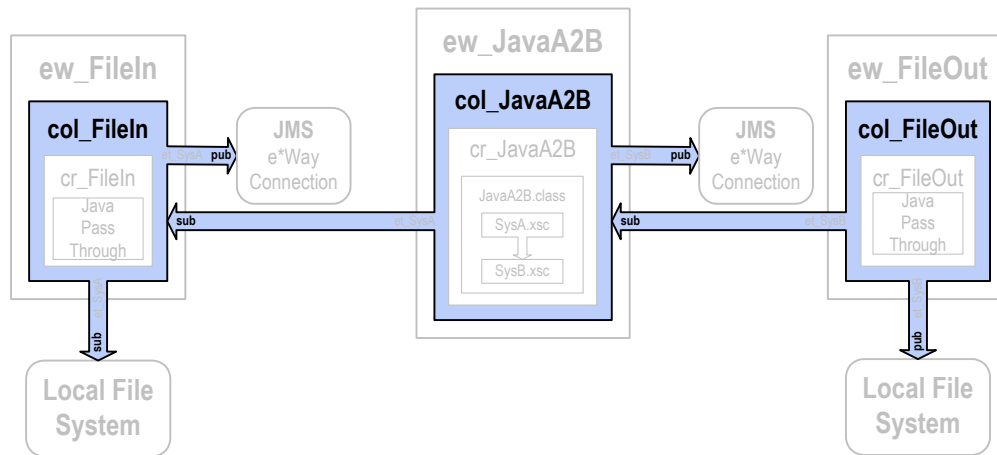
JMS is added to the list of e*Way Connections.

- 4 In the editor pane, double-click **JMS**.
The **e*Way Connection - JMS Properties** dialog box displays.
- 5 From the **e*Way Connection Type** drop-down list, select **SeeBeyond JMS**.
- 6 In the **Configuration File** area, click **New**.
- 7 From the **JMS IQ Manager** drop-down list, select your IQ Manger.
- 8 Click **OK** to save your configuration file.
- 9 Click **OK** to close the **e*Way Connection - JMS Properties** dialog box.

2.6 Add the Collaborations that Route the Data

The Collaborations are used by the e*Ways to route the data through the e*Gate system. Typically, the collaborations are configured in upstream to downstream order. Figure 16 shows the relationships of the collaborations to the remainder of the parts that make up the complete schema.

Figure 16 Collaborations Showing Publish and Subscribe Relationships



2.6.1 Add and Configure col_FileIn

The **col_FileIn** Collaboration brings the data into the e*Gate system. Use the following procedure to add and configure **col_FileIn**.

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the **ew_FileIn** e*Way.
- 2 On the **File** menu, point to **New**, click **Collaboration**.
- 3 In the **New Collaboration Component** dialog box, type **col_FileIn** for the Collaboration name, and then click **OK**.
- 4 In the editor pane, double-click **col_FileIn**.
The **Collaboration - col_FileIn Properties** dialog box displays.
- 5 In the **Collaboration Rules** list, click **cr_FileIn**.
- 6 In the **Subscriptions** area, click **Add**.
A row is added to the **Subscriptions** box.
- 7 In the **Instance Name** column, select **JavaPassThroughIn**. In the **Event Type** column, click **et_SysA** on the list, and then in the **Source** column, select **<EXTERNAL>** from the list.
- 8 In the **Publications** area, click **Add**.
A row is added to the **Publications** box.
- 9 In the **Instance Name** column, select **JavaPassThroughOut**. In the **Event Type** column, click **et_SysA** on the list, and then in the **Destination** column, select **JMS** from the list.
- 10 Click **OK** to close the **Collaboration - col_FileIn Properties** dialog box.

2.6.2 Add and Configure col_JavaA2B

The **col_JavaA2B** Collaboration changes the data from the **et_SysA** Event Type to the **et_SysB** Event Type. Use the following procedure to add and configure **col_JavaA2B**.

- 1 Use steps 1 through 4 from “[Add and Configure col_FileIn](#)” on page 31 to add a Collaboration to the **ew_JavaA2B** e*Way named **col_JavaA2B** and open its properties dialog box.
- 2 In the **Collaboration Rules** list, click **cr_JavaA2B**.
- 3 In the **Subscriptions** area, click **Add**.
A row is added to **Subscriptions** box.
- 4 Double-click in the **Instance Name** column and click **Instance1** on the list.
- 5 Double-click in the **Event Type** column and click **et_SysA** on the list.
- 6 Double-click in the **Source** column and click **JMS** on the list.
- 7 In the **Publications** area, click **Add**.
A row is added to **Publications** area.
- 8 Double-click in the **Instance Name** column, and then click **Instance2** on the list.
- 9 Double-click in the **Event Type** column, and then click **et_SysB** on the list.
- 10 Double-click in the **Destination** column, and then click **JMS** on the list.
- 11 Click **OK** to close the **Collaboration - col_JavaA2B Properties** dialog box.

2.6.3 Add and Configure col_FileOut

The **col_FileOut** Collaboration sends the transformed data out of the e*Gate system. Use the following procedure to add and configure **col_FileOut**.

- 1 Use steps 1 through 4 from “[Add and Configure col_FileIn](#)” on page 31 to add a Collaboration to the **ew_FileOut** e*Way named **col_FileOut** and open its properties dialog box.
- 2 In the **Collaboration Rules** list, click **cr_FileOut**.
- 3 In the **Subscriptions** area, click **Add**.
- 4 In the **Instance Name** column, select **JavaPassThroughIn**. In the **Event Type** column, click **et_SysB** on the list, and then in the **Source** column, select **JMS** from the list.
- 5 In the **Publications** area, click **Add**.
- 6 In the **Instance Name** column, select **JavaPassThroughOut**. In the **Event Type** column, click **et_SysB** on the list, and in the **Destination** column, select **<EXTERNAL>** from the list.
- 7 Click **OK** to close the **Collaboration - col_FileOut Properties** dialog box.

2.7 Test the Scenario

The following road map steps are covered in this section:

Step 7: Review the complete schema.

Step 8: Test the schema.

Step 9: Start the schema.

Step 10: Troubleshoot any problems.

2.7.1 Review the Complete Schema

Table 2 lists all the components for the schema. Check all the settings. Substitute the name of the machine running the schema for hostname where applicable.

Table 2 JavaE2E Components

Component	Logical Name	Settings
Schema	JavaE2E	
Control Broker	hostname_cb	
IQ Manager	hostname_iqmgr	Service = SeeBeyond JMS Config file = hostname_iqmgr.cfg Start Up = Auto
Event Type	et_SysA	SysA.xsc
	et_SysB	SysB.xsc
Java ETD	SysA.xsc	Package Name = SysApackage
	SysB.xsc	Package Name = SysBpackage
Collaboration Rule	cr_FileIn	Service = Java JavaPassThroughIn GenericInEvent.ssc Trigger JavaPassThroughOut GenericOutEvent.ssc
	cr_JavaA2B	Service = Java Instance1 SysA.xsc In Trigger Instance2 SysB.xsc Out
	cr_FileOut	Service = Java JavaPassThroughIn GenericInEvent.ssc Trigger JavaPassThroughOut GenericOutEvent.ssc
Java Collaboration Rule	cr_JavaA2B.class	Source = Instance1 Destination = Instance2
Inbound e*Way	ew_FileIn	Executable = stcewfile.exe Config file = ew_FileIn.cfg Start Up = Auto Collaboration = col_FileIn

Table 2 JavaE2E Components

Component	Logical Name	Settings
Outbound e*Way	ew_FileOut	Executable = stcewfile.exe Config file = ew_FileOut.cfg Start Up = Auto Collaboration = col_FileOut
Multi-Mode e*Way	ew_JavaA2B	Executable = stceway.exe Config file = ew_JavaA2B.cfg Start Up = Auto Collaboration = col_JavaA2B
JMS e*Way Connection	JMS	Service = SeeBeyond JMS Config file = jms.cfg
Collaboration	col_FileIn	Collab Rule = cr_FileIn Subscription = et_SysA from <EXTERNAL> Publication = et_SysA to JMS
	col_JavaA2B	Collab Rule = cr_JavaA2B Subscription = et_SysA from JMS Publication = et_SysB to JMS
	col_FileOut	Collab Rule = cr_FileOut Subscription = et_SysB from JMS Publication = et_SysB to <EXTERNAL>

2.7.2 Test the Schema

Test the scenario by sending data into the system and verifying the output.

Create the Input Data File

- 1 Use a text editor such as Notepad to create a text file that contains the following data.


```
10000,Contrary,Mary,President,60,100
10001,Horner,Jack,Line Operator,40,20
10002,Seashells,Sally,Consultant,20,75
10003,Dumpty,Humpty,Manager,50,40
10004,Winky,William,Marketing VP,40,60
```
- 2 Save the file as `c:\eGate\client\data\JavaE2E\JavaE2Einput.txt`.

2.7.3 Start the Schema

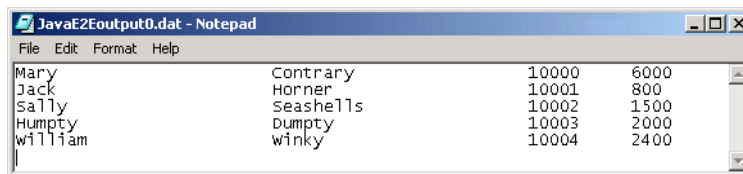
- 1 Use the following command to start the Control Broker from a command line.


```
stccb.exe -rh hostname -rs JavaE2E -ln hostname_cb -un username -up password
```
- 2 Start the e*Gate Monitor.
- 3 Verify that all the components in the schema are running.

Testing in Windows 2000

- 1 Once all the scenario components have been started successfully, use Windows Explorer to navigate to `c:\eGate\client\data\JavaE2E\`.
- 2 Change the file extension on the input file `JavaE2Einput.txt` to `.fin`.
- 3 Click **Yes** to confirm this choice.
- 4 Verify that the extension changes to `.~in` indicating that the `ew_FileIn e*Way` has retrieved the file.
- 5 Almost immediately, the output file, `JavaE2Eoutput#.dat`, should appear in the directory, indicating a successful conclusion to the test.
- 6 Verify that the output looks like that shown in Figure 17.

Figure 17 JavaE2Eoutput#.dat File



2.7.4 Troubleshoot any problems

Java Troubleshooting

If you are unable to find the problem by reviewing the table [“JavaE2E Components” on page 33](#), try using [“Debugging and Log Files” on page 83](#) to start the trace or error log component in your schema.

Table 3 Compiler Errors

Error	Possible Solution
unreachable statement	In the <code>executeBusinessRules</code> method, the <code>return retBoolean</code> rule needs to be placed below the user-defined rules.

e*Gate ELS End-to-End Scenario

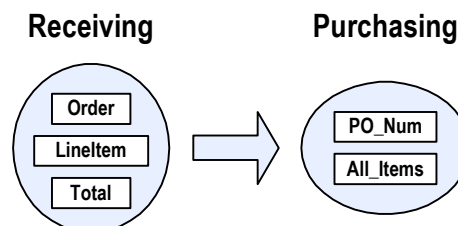
This is a step-by-step description of how to set up and test a simple end-to-end e*Gate scenario using the e*Gate Event Linking and Sequencing (ELS) feature.

This scenario was developed under the Windows 2000 operating system.

ELS End-to-End Scenario Business Problem

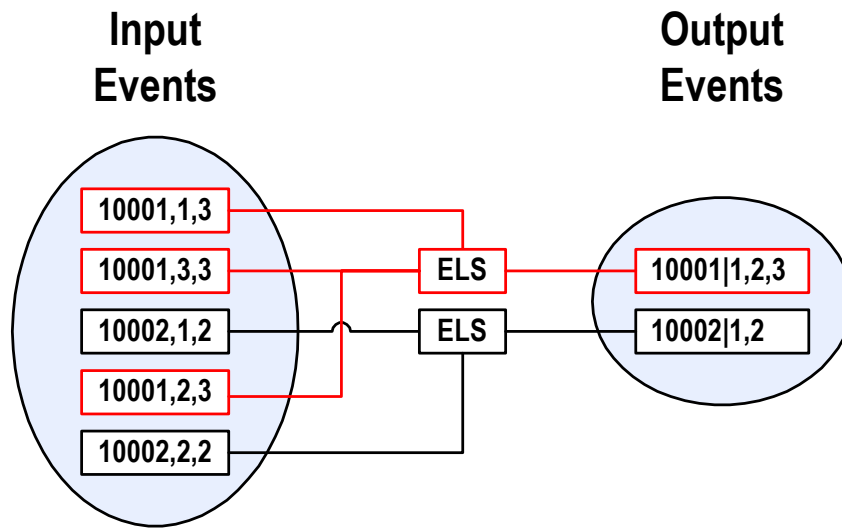
Purchasing buys office supplies for the company. The orders generated by the purchasing system have multiple line items. Receiving processes the items as they are delivered. Typically, only individual line items are delivered, not entire purchase orders. Purchasing only pays purchase orders that are completely fulfilled. The company needs a way to track whether an entire purchase order has been filled.

Figure 18 ELSE2E Business Problem



This scenario is designed to show how e*Gate's ELS feature can be used to correlate the individual line items taken in by Receiving so that when all the line items for a specific purchase order have all been received, a combined Event can be sent to Purchasing. Purchasing uses the receipt of this Event to trigger payment of the purchase order.

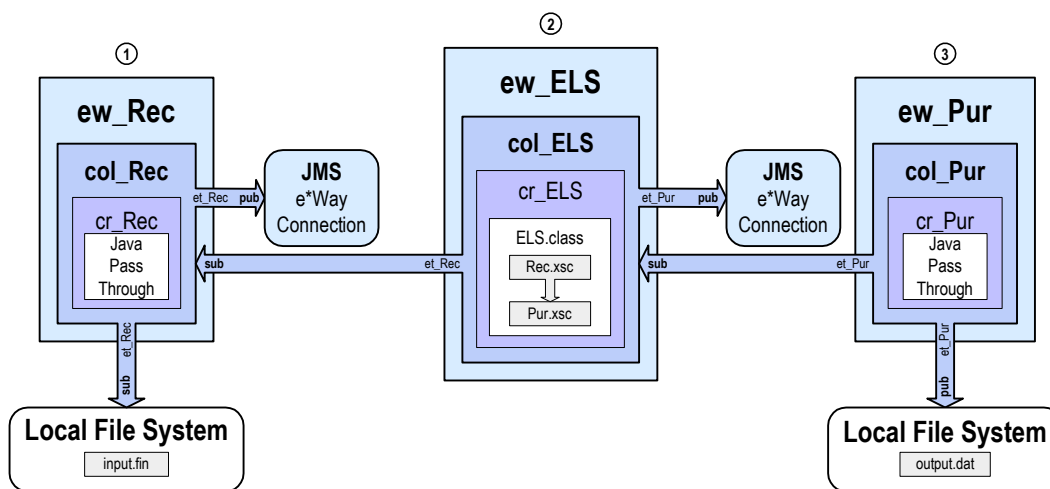
Figure 19 ELS Solution



ELS End-to-End Scenario e*Gate Solution

The proposed e*Gate solution makes use of e*Gate enhanced Java Collaboration Service and the ELS methods added to the Java Collaboration to combine the data from Events that have the same order number into a single Event. e*Gate is very flexible about where the actual ELS processing can occur as the data moves from the receiving system to the purchasing system. Our solution uses a Java Collaboration as the main transformation component and two simple file e*Ways to bring data into and send data out from the e*Gate system. Figure 20 shows all the components and their relationships to one another in the complete e*Gate schema.

Figure 20 ELS End-to-End Scenario Overview



Notes on the ELS End-to-End Scenario Overview

- ① ewRec brings data from the Receiving System into e*Gate.

The **colRec** Collaboration in the **ewRec** e*Way subscribes to a location on the local file system. It polls this location for a text file with extension **".fin"** containing data from the Receiving System, packages the data as an **etRec** Event, and then publishes the Event to the **JMS e*Way Connection**.

- ② **ewELS** combines Events with matching order numbers into a single Event.

The **colELS** in the **ewELS** subscribes to **etRec** Events published by the **JMS e*Way Connection**. It uses an ELS-enabled Java Collaboration Rule **crELS** to temporarily store **etRec** Events. Once all the Events associated with a specific order are received, **crELS** publishes a combined Event, **etPur**, to the **JMS e*Way Connection**. This Collaboration Rule uses the **ELS.class** file that implements the business logic.

- ③ **ewPur** writes the combined Event out to local file system.

The **colPur** Collaboration in the **ewPur** e*Way subscribes to **etPur** Events published by the **JMS e*Way Connection**. The **crPur** Collaboration Rule uses the Java Pass Through service to move the data without modifying it. When a **etPur** Event is retrieved, the e*Way packages it as a text file and writes it to the specified location on the local file system, completing the end-to-end scenario.

Road Map For Setting Up the Scenario

Use the following general steps to set up an end-to-end scenario using Java and ELS:

- 1 Verify the e*Gate installation.
- 2 Create a new schema.
- 3 Create the Event Types and Java ETDs.
- 4 Create the Collaboration Rules.
- 5 Add and configure the e*Ways and e*Way Connection.
- 6 Add and define the Collaborations that route the data.
- 7 Review the complete schema.
- 8 Start the schema.
- 9 Test the schema.
- 10 Troubleshoot any problems.

3.1 Verify the e*Gate Installation

This ELS end-to-end scenario is designed to be run on a single machine. Before beginning the configuration process, you must verify that you have all the required software installed on the target machine. See **"Prerequisites" on page 12** to review the prerequisites.

3.2 Create a New Schema

To create a new schema

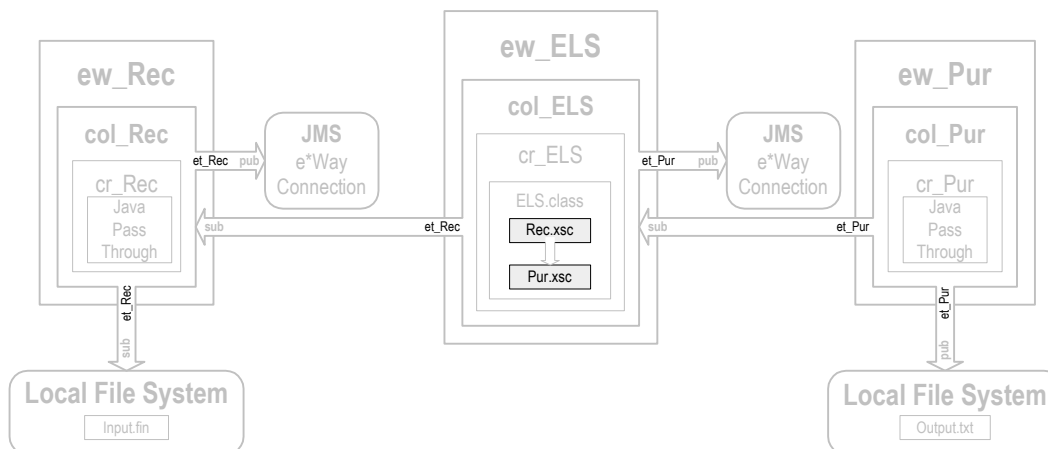
- 1 Start the e*Gate Enterprise Manager and log in as **Administrator** (or another user with administrator privileges) to the appropriate Registry Host.
- 2 In the **Open Schema on Registry Host** dialog box, click **New**.
- 3 In the **Enter New Schema Name** box, type **ELSE2E**, and then click **Open**.
The Enterprise Manager opens and displays the new **ELSE2E** schema.
- 4 At the bottom of the navigator (left) pane, click the **Components** tab.
All schema configuration steps are performed with the **Components** tab selected.

3.3 Create the Event Types and Java ETDs

This scenario uses two Event Types, each with its own Event Type Definition (ETD). The first Event Type, **etRec**, models the format of the data sent by the Receiving System. The second Event Type, **etPur**, models the format used by the Purchasing System.

Figure 21 shows where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 21 Event Types and Java ETDs



3.3.1 Create the Rec.xsc ETD

Select the Default ETD Editor

- 1 On the Enterprise Manager **Options** menu, click **Default Editor**.
- 2 Click **Java**, and then click **OK**.

Java is the default ETD editor on install.

Start the Java ETD Editor

- 1 In the navigator pane of the Enterprise Manager, click the **Event Types** Folder.
- 2 On the **File** menu, point to **New**, and then click **Event Type**.
- 3 In **New Event Type Component** dialog box, type **etRec** for the Event Type name, and then click **OK**.

etRec is added to the list of Event Types in the Enterprise Manager editor pane.

- 4 On the list of Event Types, double-click **etRec**.
- 5 In the **Event Type - etRec Properties** dialog box, click **New**.

The Java ETD Editor appears.

Use the Standard ETD Wizard

- 1 On the **File** menu, click **New**.
- 2 In the **New Event Type Definition** dialog box, click the **Standard ETD** icon, and then click **OK**.

The Standard ETD Wizard appears.

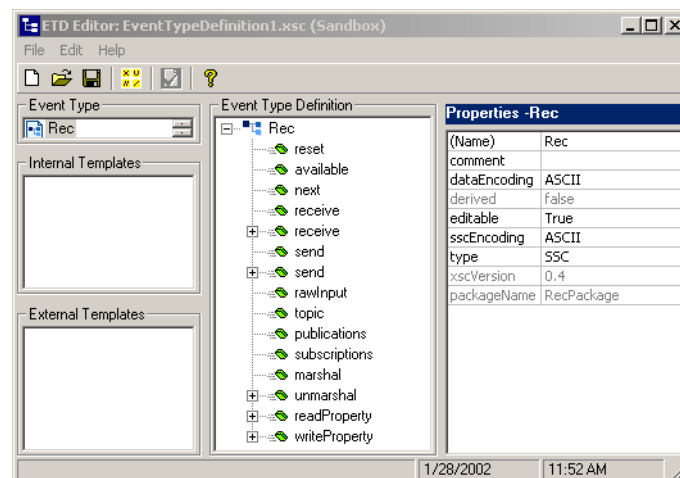
- 3 Read the **Introduction** screen, and then click **Next**.
- 4 On the **Step 1** window, type **Rec** in the **Root Node Name** box.
- 5 Type **RecPackage** in the **ETD Package Name** box, and then click **Next**.

The package name is the name you give to the collection of classes that make up a Java ETD. This group of Java classes is the result of the ETD creation process.


- 6 On the **Step 2** screen, review the summary information, and then click **Finish**.

The new ETD is displayed in the Java ETD Editor as shown in Figure 22.

Figure 22 Rec ETD Before Modifications

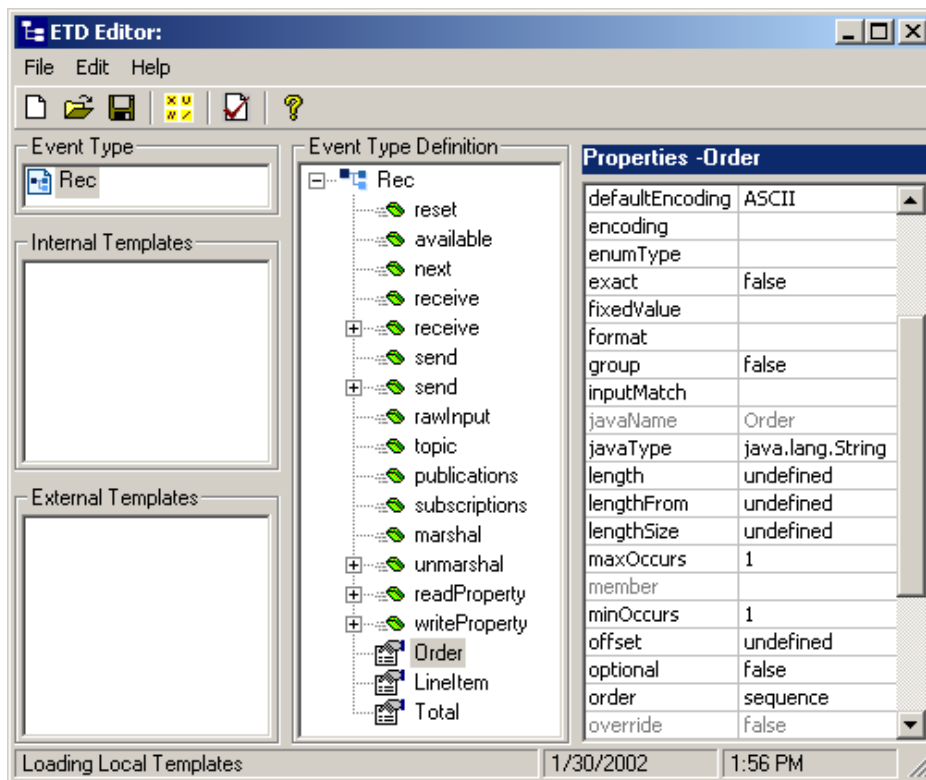


Complete Rec.xsc in the Java ETD Editor

- 1 Right-click **Rec**, point to **Add Field**, and then click **As Child Node**.
A node with the default name **Field#** is added under the **Rec** root node.
- 2 Right-click this new node, click **Rename**, and then change its name to **Order**.
- 3 With the **Order** node still selected, click **local delimiter** in the **Properties** pane, and then click the ellipses  button.
 - A In the **Local Delimiter** dialog, click **Add Single End Delimiter**.
 - B Click **Value** in the **Properties** pane and enter **“,”** (comma, no quotes).
 Click OK to close the **Local Delimiter** dialog. No other changes are necessary to the **Order** node’s properties.
- 4 Repeat steps 1 through 3 for the other two nodes that must be added to this ETD.
The additional node names are:
 - LineItem
 - Total
- 5 With the **Total** node still selected, click **javaType** in the **Properties** pane, and select **“int”** from the list.

When finished the completed Rec ETD should look like the one shown in Figure 23.

Figure 23 Completed Rec ETD



- 6 On the **File** menu, click **Compile and Save**.
- 7 In the **Save** dialog box, change the default name for the **.xsc** file, **EventTypeDefinition#.xsc**, to **Rec.xsc** and then click **Save**.

The Rec ETD is compiled and saved.

- 8 On the **File** menu, click **Close**.

The Java ETD Editor closes and the location of the file **Rec.xsc** is automatically entered in the **Event Type Definition** area of the **Event Type - etRec Properties** dialog box.

- 9 Click **OK** to close the **Event Type - etRec Properties** dialog box.

3.3.2 Create the Pur.xsc ETD

This ETD corresponds to the fixed data structure used by the Purchasing System.

Use the procedures described in [“Start the Java ETD Editor” on page 40](#) and [“Use the Standard ETD Wizard” on page 40](#) to create a new Event Type named **etPur** and a corresponding Java ETD named **Pur.xsc** with a package name of **PurPackage**. Use the following procedure to complete the **Pur.xsc** in the Java ETD Editor.

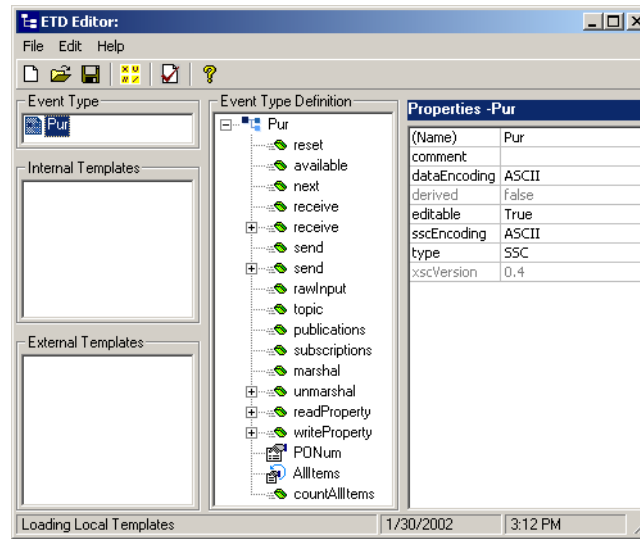
Completing Pur.xsc in the Java ETD Editor

- 1 Right-click **Pur**, point to **Add Field**, and then click **As Child Node**.
A node with the default name **Field#** is added under the **Pur** root node.
- 2 Triple-click the **Field1** node, and then change its name to **PONum**.
- 3 With the **PONum**, enter “|” (pipe character, no quotes) as the ending delimiter.
No other changes are necessary to the **PONum** node’s properties.
- 4 Right-click **Pur**, point to **Add Field**, and then click **As Child Node**.
- 5 Change the node name to **AllItems**.
- 6 With the **AllItems** node still selected, enter “,” (comma, no quotes) as the ending delimiter.
- 7 Click **maxOccurs**, and then enter “-1” (minus one, no quotes) to change **AllItems** to a repeating node.

Minus one for **maxOccurs** is the indicator for a repeating node.

No other changes are necessary to the **Pur** ETD’s properties. The **Pur** ETD should look like the one shown in Figure 24.

Figure 24 Completed Pur ETD



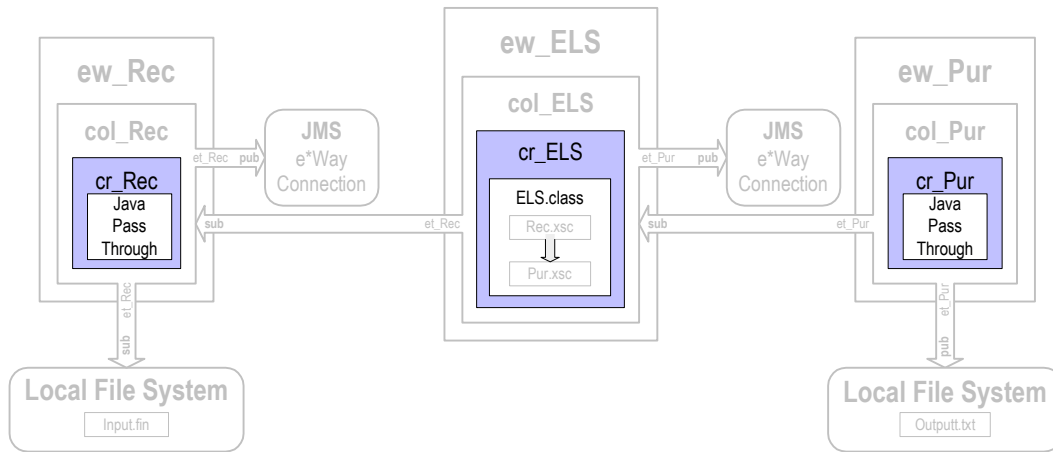
- 8 Compile and save the ETD as `\etd\Pur.xsc`.
- 9 Close the Java ETD Editor.
- 10 Click **OK** to close the **Event Type - etPur Properties** dialog box.

3.4 Create the Collaboration Rules

This scenario uses three Collaboration Rules: two Java Pass Through rules and one that uses a Java Collaboration Rule. The Java Pass Through rules, **crRec** and **crPur**, are used to route the Events through the e*Gate system and the Java Collaboration Rule **crELS** is used to combine the individual line item Events into a single complete purchase order Event.

Figure 25 shows where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 25 Collaboration Rules and Java Collaboration Rules Class



3.4.1 Create the Java Pass Through Collaboration Rules

The Java Pass Through Collaboration Rules are used to bring data into and take data away from the e*Gate system. They do not change the data. The following procedure explains how to add the Java Pass Through Collaborations used in this scenario.

Create the crRec and crPur Collaboration Rules

- 1 In the navigator pane of the Enterprise Manager, click the **Collaboration Rules** folder.
- 2 On the **File** menu, point to **New**, and then click **Collaboration Rules**.
- 3 In the **New Collaboration Rules Component** dialog box, type **crRec** for the Collaboration Rule name, and then click **OK**.
crRec is added to the list of Collaboration Rules in the Enterprise Manager editor pane.
- 4 On the list of Collaboration Rules, double-click **crRec**.
- 5 In the **Collaboration Rules - crRec Properties** dialog box, click **Find in the Collaboration Rules section**.
- 6 Navigate to `collaboration_rules\STCLibrary` and select `STCJavaPassThrough.class`.
- 7 Click **OK** to close the **Collaboration Rules - crRec Properties** dialog box.
- 8 Repeat steps 2 through 6 to create the **crPur** Collaboration Rule. Substitute **crPur** for the Collaboration Rule name and substitute **etPur** for the Event Type.

Selecting the `STCJavaPassThrough.class` file automatically configures the collaboration mapping tab for you.

3.4.2 Create the Java Collaboration Rule

The procedure for creating a Collaboration Rule that uses the Java Collaboration Service is different from creating other e*Gate Collaboration Rules. Use the following procedure to start the Java Collaboration Editor and create the Java Collaboration Rule used by this scenario.

Create crELS and Start the Java Collaboration Editor

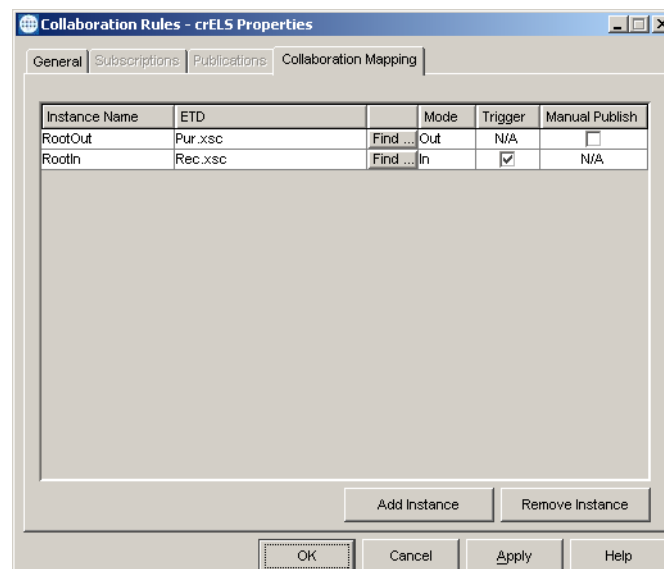
- 1 Use steps 1 through 4 from the procedure described in **“Create the crRec and crPur Collaboration Rules” on page 44** to create a new Collaboration Rule named **crELS**.
- 2 Click the **Collaboration Mapping** tab, and then click **Add Instance**.
An instance row is added to the **Collaboration Mapping** tab.
- 3 In the **Instance Name** column, type **RootIn** for the instance name.
- 4 Click **Find**, navigate to **etd** and double-click **Rec.xsc**.
Rec.xsc is added to the **ETD** column of the instance row. No other changes are necessary to **RootIn**.
- 5 Add another ETD instance. Use **RootOut** for the instance name.

Important: The Java ETD instance names must be unique per schema.

- 6 Find and enter **Pur.xsc** as the ETD for **RootOut**.
- 7 Click in the **Mode** cell for **RootOut**, and then click **Out** on the **Mode** list.

No other changes are necessary to **RootOut**. The completed Collaboration Mapping tab looks like the one shown in Figure 26.

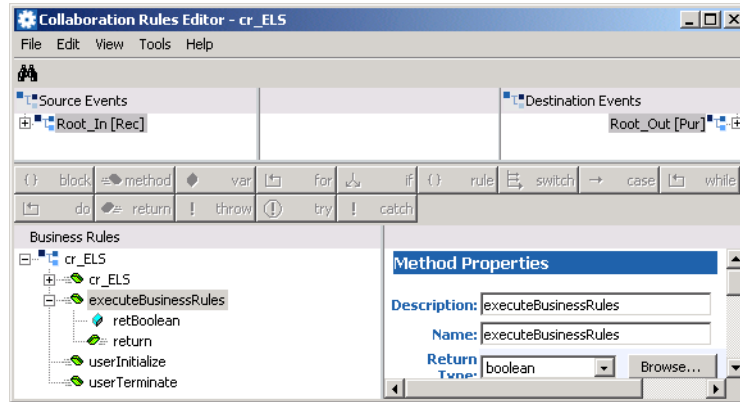
Figure 26 Completed Collaboration Mapping Tab



- 8 Click the **General** tab, and then in the **Collaboration Rules** area click **New**.

The Java Collaboration Editor opens a new Collaboration Rule with **RootIn (Rec)** as the source ETD and **RootOut (Pur)** as the destination ETD as shown in Figure 27.

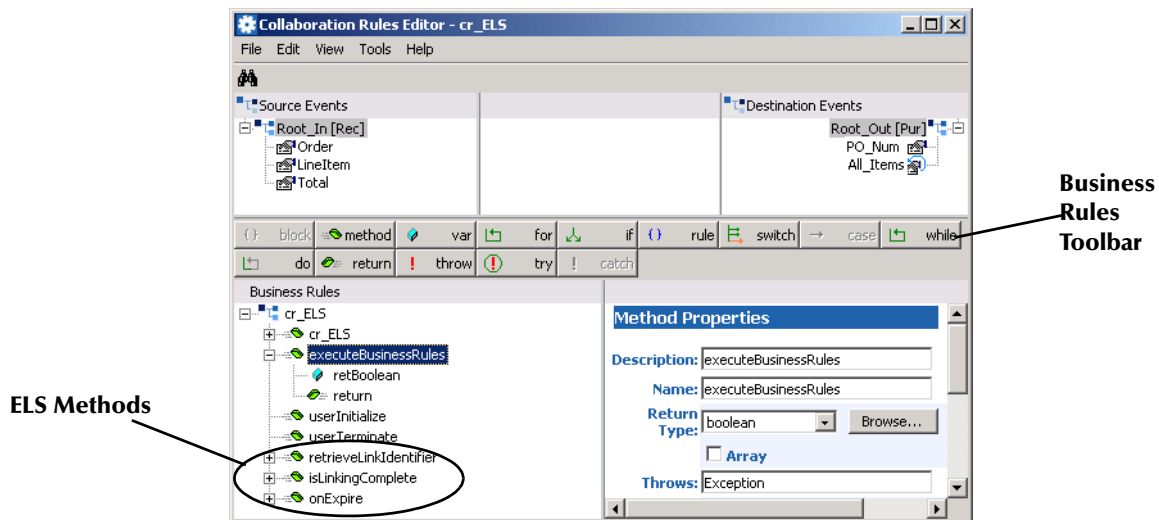
Figure 27 ELS Collaboration Rule Before Adding User-Defined Code



Prepare the Java Collaboration Editor to Add Business Rules

- 1 On the **File** menu, click **Enable ELS**.
The ELS methods, **retrieveLinkIdentifier**, **isLinkingComplete**, and **onExpire** are added to the business rules.
- 2 In the **Source Events** and **Destination Events** panes, expand **RootIn** and **RootOut** to display the leaf nodes of the ETDs.

Figure 28 Java Collaboration Rules Editor After Enabling ELS



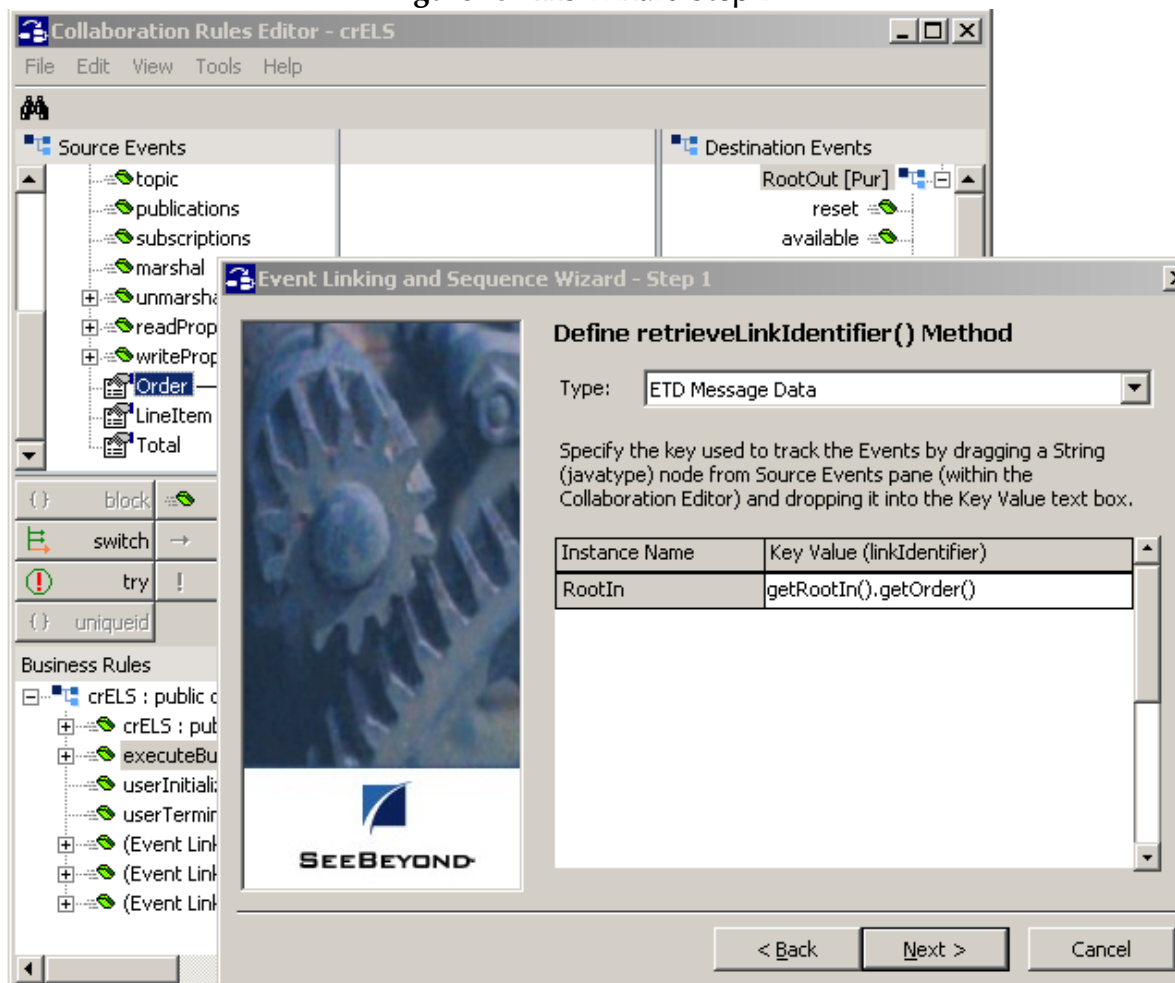
Using the ELS Wizard

The ELS Wizard will guide you through the process of setting up an ELS enabled schema. The ELS Wizard allows you to drag nodes directly from your Collaboration Rule, into the input fields within the ELS Wizard interface.

Define ELS rules with the ELS Wizard

- 1 On the **Tools** menu, click **ELS Wizard**.
The ELS Wizard displays.
- 2 Read the Welcome screen, then click **Next** to continue.
- 3 Select and drag the **Order** field from the Source Events pane of the Collaboration Rules editor into the Key Value (linkidentifier) input field of the ELS Wizard screen.

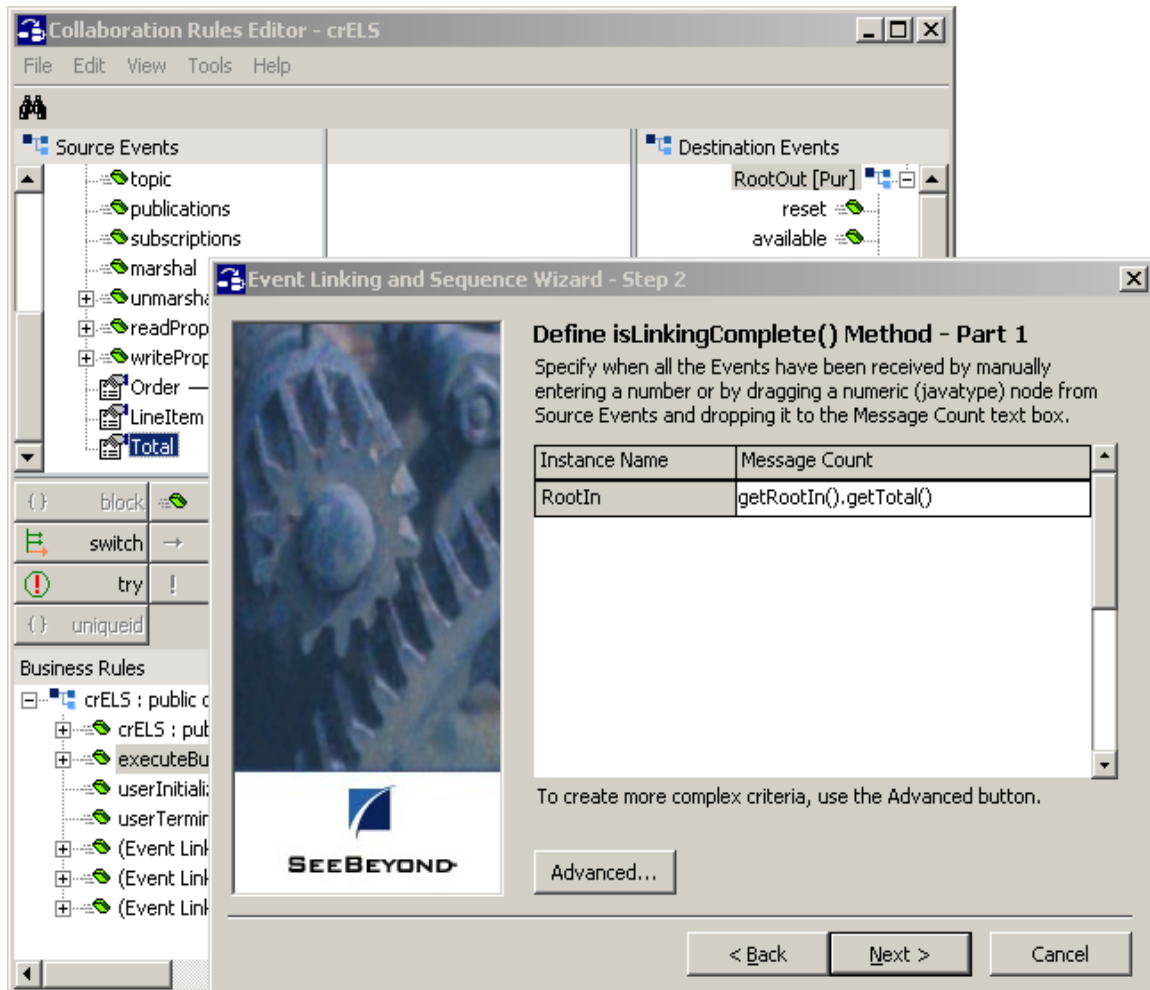
Figure 29 ELS Wizard Step 1



- 4 Select and drag the **Total** field from the Source Events pane of the Collaboration Rules editor into the Message Count input field of the ELS Wizard screen.
- 5 Click **Next** to continue.
- 6 Enter **10000** in the Expiration field of the ELS Wizard screen.

- Click Finish to exit the ELS Wizard.

Figure 30 ELS Wizard Step 2



Edit the executeBusinessRules method

The **executeBusinessRules** method is called whenever the **isLinkingComplete** method returns a Boolean true. It is used to build the destination Events.

Use the following procedure to edit the **executeBusinessRules** method for this scenario:

- Expand the **executeBusinessRules** method.
- Click the **retBoolean** variable.
- Add a variable using the following information.

Description	Repeating node counter
Name	count
Type	int

Initial Value 1

This variable is used to copy the line item data to the **AllItems** repeating node in the destination Event.

- 4 With the **Repeating node counter** variable selected, add a rule using the following information.

Description Copy order number
Rule `getRootOut().setPONum(getRootIn().getOrder())`

This rule is used to copy the purchase order number to the destination Event.

Note: You can also enter this rule by dragging the **Order** node in the **Source Events** pane to the **PONum** node in the **Destination Events** pane.

- 5 With the **Copy order number** rule selected, add a rule using the following information.

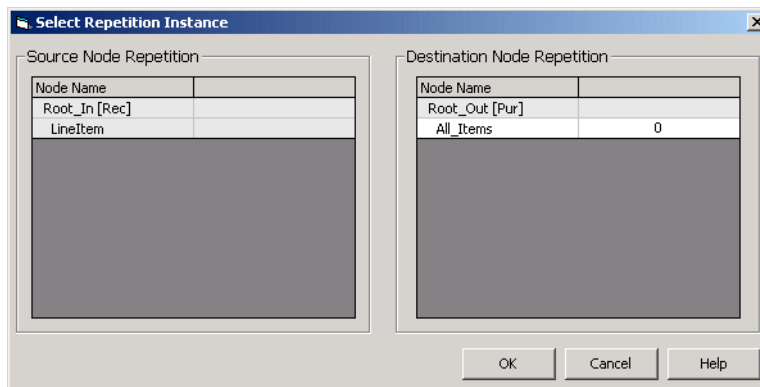
Description Copy first line item number
Rule `getRootOut().setAllItems(0,getRootIn().getLineItem())`

This rule is used to copy the first line item number to the **AllItems** repeating node in the destination Event. Subsequent line item numbers are copied to this node from within the following while rule.

Because **AllItems** is a repeating node, you must use the syntax “**#,value**” when setting its value. # is an integer that specifies where in the repetition the value falls, 0 = first, 1 = second, 2 = third, and so on.

Note: You can also enter this rule by dragging the **LineItem** node in the **Source Events** pane to the **AllItems** node in the **Destination Events** pane, and then changing the repetition instance for the **AllItems** node to “0” as shown in Figure 31.

Figure 31 Copying to Repeating Node



- With the **Copy first line item number** rule selected, add a **while** rule using the following information.

Description	Copy additional line item numbers
Condition	getRootIn().next()

The while rule processes as long as there is another matching Event to retrieve.

- With the **Copy additional line item numbers** while rule selected, add a rule as a child using the following information.

Description	Copy next line item number
Rule	getRootOut().setAllItems(count,getRootIn().getLineItem())

This rule is used to copy all the remaining line item numbers to the **AllItems** repeating node in the destination Event.

Note: You can also enter this rule by dragging the **LineItem** node in the **Source Events** pane to the **AllItems** node in the **Destination Events** pane, and then changing the repetition instance for the **AllItems** node to “count”.

- With the **Copy next line item number** rule selected, add a rule using the following information.

Description	Add one to repeating node counter
Rule	count++

This rule increments the repeating node counter. The count variable is used in the **Copy next line item number** rule to set values in the **AllItems** repeating node.

See Figure 32 for the completed executeBusinessRules Method.

Figure 32 After Editing the executeBusinessRules Method

```

executeBusinessRules : public boolean executeBusinessRules() throws Exception
{
    retBoolean : boolean retBoolean = true;
    Repeating node counter : int count = 1;
    Copy order number : getRoot_Out().setPO_Num(getRoot_In().getOrder());
    Copy first line item number : getRoot_Out().setAll_Items(0,getRoot_In().getLineItem());
    Copy additional line item numbers : while (getRoot_In().next())
    {
        Copy next line item number : getRoot_Out().setAll_Items(count,getRoot_In().getLineItem());
        Add one to repeating node counter : count++;
    }
    return : return retBoolean;
}
    
```

Compile and Save the Java Collaboration Rules

- On the **File** menu, click **Save**.
- In the **Save** dialog box, navigate to the **collaboration_rules** folder, and then save the **crELS.xpr** file.
- On the **File** menu, click **Compile**.

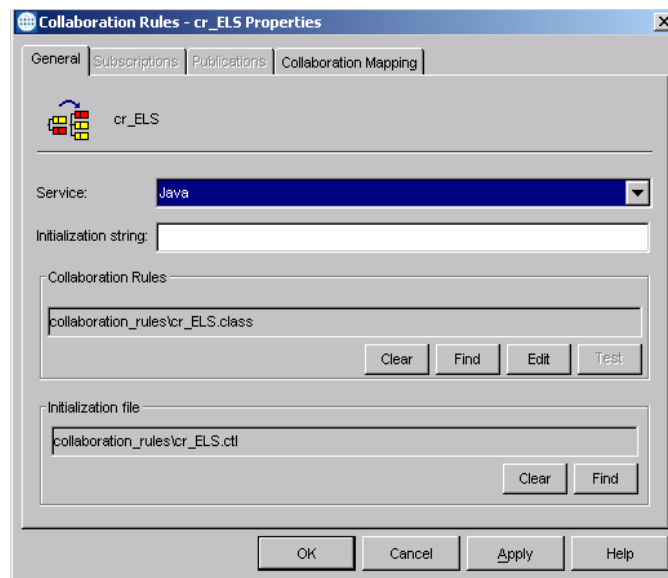
The compile pane is displayed at the bottom of the Java Collaboration Editor.

The Java source code is compiled. When the compiler is finished “**Compile Completed.**” is displayed in the compile pane. The compile pane also displays any errors generated by the compilation process.

- 4 On the **File** menu, click **Exit**.
- 5 If prompted to, click **Yes** to save your changes.

The JCE closes and in the **Collaboration Rules - crELS Properties** dialog box, **collaboration_rules\crELS.class** is entered in the **Collaboration Rules** area and **collaboration_rules\crELS.ctl** is entered in the **Initialization file** area as shown in Figure 33.

Figure 33 Completed crELS Properties



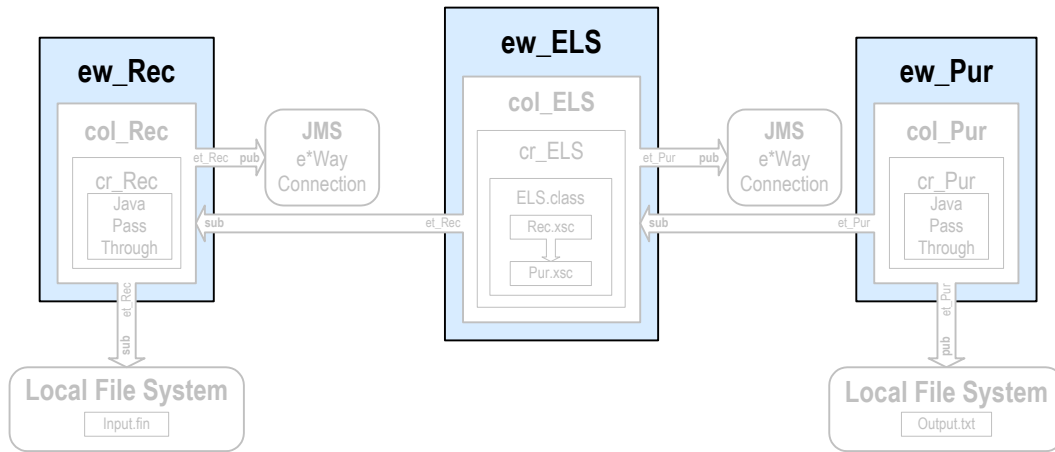
- 6 Click **OK** to close the **Collaboration Rules - crELS Properties** dialog box.

3.5 Add and Configure the e*Ways and e*Way Connection

Once you have created your ETDs and Collaborations, you are ready to add and configure the e*Gate components that use these parts.

Figure 34 highlights the components added in this step.

Figure 34 e*Ways and e*Way Connection



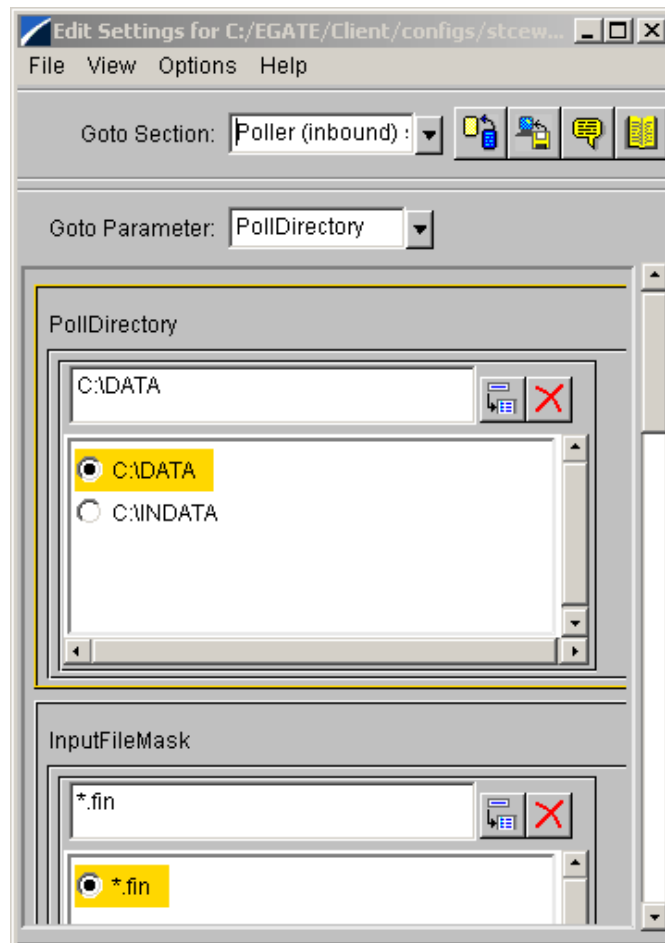
3.5.1 Add and Configure the e*Ways

Add and Configure the ewRec file e*Way

- 1 In the e*Gate Enterprise Manager, in the navigation pane, expand the **Participating Hosts** folder, and then expand the host (*hostname*) to which you are going to add the components.
 - 2 In the navigation pane, click the Control Broker (*hostname_cb*).
 - 3 On the **File** menu, point to **New**, point to **Module**, and then click the **e*Way**.
 - 4 In the **New e*Way Component** dialog box, type **ewRec** for the e*Way name, and then click **OK**.
- The **ewRec** e*Way is added to the schema.
- 5 Right-click the **ewRec** e*Way in the editor pane, and then click **Properties**.
 - 6 In the **e*Way - ewRec Properties** dialog box, in the **Executable file** area, click **Find**.
 - 7 In the **File Selection** dialog box, browse for and double-click the file **stcewfile.exe**. **bin\stcewfile.exe** is added as the **ewRec** e*Way's executable file, causing the component to become a file e*Way.
 - 8 In the **Configuration file** area, click **New**.
- The e*Way Configuration File Editor opens with a default file e*Way configuration file ready for editing.
- 9 In the **Goto Section** list, click **Poller (inbound) settings**.
 - 10 In the **PollDirectory** box, change the default value (**c:\INDATA**) to **c:\Data** and then press **ENTER**.

c:\Data is added as the directory to be polled to the **PollDirectory** list as shown in Figure 35. No other changes are necessary to the **ewRec** e*Way's configuration file.

Figure 35 ewRec Configuration File



- 11 On the **File** menu, click **Save**.
- 12 In the **Save As** dialog box, click **Save** to accept the default filename (**ewRec.cfg**) and save the file.
- 13 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
configs\stcewfile\ewRec.cfg is added to the **Configuration file** area in the e*Way - **ewRec Properties** dialog box.
- 14 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 15 Click **OK** to close the e*Way - **ewRec Properties** dialog box.

Add and Configure the ewPur file e*Way

Adding the **ewPur** e*Way follows the same general procedure as that outlined for adding the **ewRec** e*Way above.

- 1 Use steps 1 through 8 from **“Add and Configure the ewRec file e*Way” on page 52** to add another file e*Way named **ewPur** and open its configuration file for editing.
- 2 In the e*Way Configuration File Editor, on the **General Settings** screen, click **NO** for **AllowIncoming**, and **YES** for **AllowOutgoing**.

- 3 In the **Goto Section** list, click **Outbound (send) settings**.
- 4 Change the default **OutputFileName** (output%d.dat) to **ELSE2Eoutput%d.dat**.
No other changes are necessary to the **ewPur** e*Way's configuration file.
- 5 On the e*Way Configuration File Editor's **File** menu, click **Save**.
- 6 In the **Save As** dialog box, click **Save** to accept the default file name (**ewPur.cfg**) and save the file.
- 7 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 8 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 9 Click **OK** to close the **e*Way - ewPur Properties** dialog box.

Add and Configure the ewELS Multi-Mode e*Way

- 1 In the navigation pane, click the Control Broker (*hostname_cb*).
- 2 On the **File** menu, point to **New**, point to **Module**, and then click the **e*Way**.
- 3 In the **New e*Way Component** dialog box, type **ewELS** for the e*Way name, and then click **OK**.
The **ewELS** e*Way is added to the schema.
- 4 Right-click the **ewELS** e*Way in the editor pane, and then click **Properties**.
- 5 In the **e*Way - ewELS Properties** dialog box, in the **Executable file** area, click **Find**.
- 6 In the **Configuration file** area, click **New**.
The e*Way Configuration File Editor opens with a default Multi-Mode e*Way configuration file ready for editing.
- 7 Scroll to the bottom of the **JVM Settings** parameters.
- 8 In the **Remote Debugging Port Number** box, type **8000**, and then press **ENTER**.
8000 is listed as the **Remote Debugging Port Number**. No other changes are necessary to the **ewELS** e*Way's configuration file.
- 9 On the **File** menu, click **Save**.
- 10 In the **Save As** dialog box, click **Save** to accept the default filename (**ewELS.cfg**) and save the file.
- 11 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 12 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 13 Click **OK** to close the **e*Way - ewELS Properties** dialog box.

3.5.2 Add the e*Way Connection

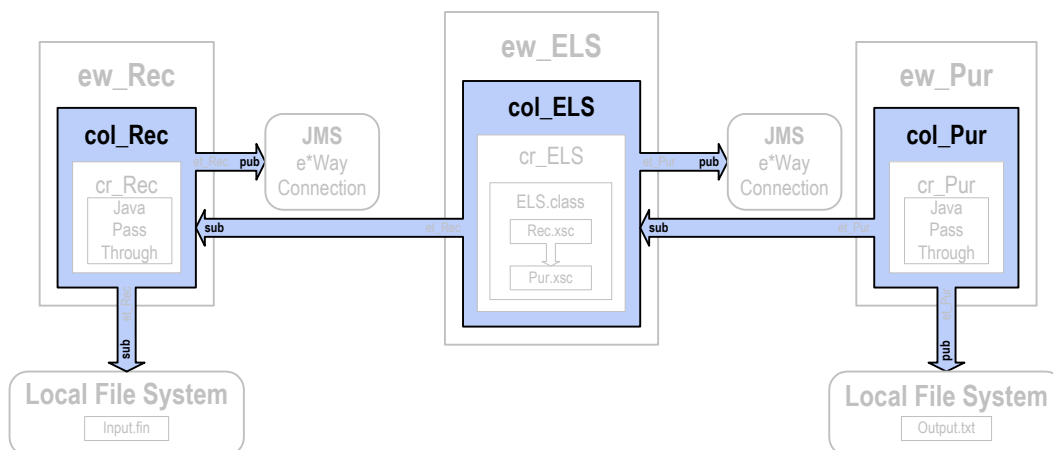
- 1 In the e*Gate Enterprise Manager navigation pane, select the **e*Way Connections** folder.
- 2 In the contents pane, right-click and select **New e*Way Connection**.

- 3 In the **New e*Way Connection** dialog box, type **JMS** for the e*Way Connection name, and then click **OK**.
- 4 Double-click the **JMS e*Way Connection**.
The **e*Way Connection - JMS properties** displays.
- 5 In the **e*Way Connection Type** section, select **SeeBeyond JMS**.
- 6 In the **e*Way Configuration File** section, click **New**.
- 7 Select **hostname-iqmgr** from the **JMS IQ Manager** list.
Server Name, Hostname and Port Number populate automatically.
- 8 Click **OK** to exit.
- 9 Click **OK** to exit **e*Way Connection - JMS properties**.

3.6 Add the Collaborations that Route the Data

Collaborations are used by the e*Ways to route the data through the e*Gate system. Typically, the collaborations are configured in upstream to downstream order. Figure 36 shows the relationships of the collaborations to the remainder of the parts that make up the complete schema.

Figure 36 Collaborations Showing Pub/Sub Relationships



3.6.1 Add and Configure colRec

The **colRec** Collaboration brings the data into the e*Gate system. Use the following procedure to add and configure **colRec**.

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the **ewRec** e*Way.
- 2 On the **File** menu, point to **New**, click **Collaboration**.

- 3 In the **New Collaboration Component** dialog box, type **colRec** for the Collaboration name, and then click **OK**.
- 4 In the editor pane, double-click **colRec**.
The **Collaboration - colRec Properties** dialog box appears.
- 5 In the **Collaboration Rules** list, click **crRec**.
- 6 In the **Subscriptions** area, click **Add**.
A row is added to the **Subscriptions** box.
- 7 In the **Event Type** column, click **etRec (Required)** on the list, and then in the **Source** column, click **<EXTERNAL>** on the list.
- 8 In the **Publications** area, click **Add**.
A row is added to the **Publications** box.
- 9 In the **Event Type** column, click **etRec** on the list, and then in the **Destination** column, click **JMS** on the list.
No further configuration is required for **colRec**.
- 10 Click **OK** to close the **Collaboration - colRec Properties** dialog box.

3.6.2 Add and Configure colELS

The **colELS** Collaboration changes the data from the **etRec** Event Type to the **etPur** Event Type. Use the following procedure to add and configure **colELS**.

- 1 Use steps 1 through 4 from **"Add and Configure colRec" on page 55** to add a Collaboration to the **ewELS** e*Way named **colELS** and open its properties dialog box.
- 2 In the **Collaboration Rules** list, click **crELS**.
- 3 In the **Subscriptions** area, click **Add**.
A row is added to **Subscriptions** box.
- 4 Double-click in the **Instance Name** column and click **RootIn** on the list.
- 5 Double-click in the **Event Type** column and click **etRec** on the list.
- 6 Double-click in the **Source** column and click **JMS** on the list.
- 7 In the **Publications** area, click **Add**.
A row is added to **Publications** area.
- 8 Double-click in the **Instance Name** column, and then click **RootOut** on the list.
- 9 Double-click in the **Event Type** column, and then click **etPur** on the list.
- 10 Double-click in the **Destination** column, and then click **JMS** on the list.
- 11 Click **OK** to close the **Collaboration - colELS Properties** dialog box.

3.6.3 Add and Configure colPur

The **colPur** Collaboration sends the transformed data out of the e*Gate system. Use the following procedure to add and configure **colPur**.

- 1 Use steps 1 through 4 from **“Add and Configure colRec” on page 55** to add a Collaboration to the **ewPur** e*Way named **colPur** and open its properties dialog box.
- 2 In the **Collaboration Rules** list, click **crPur**.
- 3 In the **Subscriptions** area, click **Add**.
- 4 In the **Event Type** column, click **etPur (Required)** on the list, and then in the **Source** column, click **JMS** on the list.
- 5 In the **Publications** area, click **Add**.
- 6 In the **Event Type** column, click **etPur** on the list, and then in the **Destination** column, click **<EXTERNAL>** on the list.
- 7 Click **OK** to close the **Collaboration - colPur Properties** dialog box.

3.7 Test the Scenario

The following road map steps are covered in this section:

- Step 7: Review the complete schema.
- Step 8: Start the schema.
- Step 9: Test the schema.
- Step 10: Troubleshoot any problems.

3.7.1 Review the Complete Schema

The following table lists all the components for the schema. Double-check all the settings. Substitute the name of the machine running the schema for hostname where applicable.

Table 4 ELSE2E Components

Component	Logical Name	Settings
Schema	ELSE2E	
Control Broker	hostname_cb	
IQ Manager	hostname_iqmgr	Start Up = Auto
Event Type	etRec	
	etPur	

Table 4 ELSE2E Components

Component	Logical Name	Settings
Java ETD	Rec.xsc	Package Name = RecPackage
	Pur.xsc	Package Name = PurPackage
Collaboration Rule	crRec	Service = Java Sub = etRec Pub = etRec
	crELS	Service = Java
		Instance Name = RootIn ETD = Rec.xsc Mode = In Trigger = Yes Manual Publish = No
crPur	Instance Name = RootOut ETD = Pur.xsc Mode = Out Trigger = No Manual Publish = No	
Java Collaboration Rule Class	ELS.class	Source = RootIn (Rec) Destination = RootOut (Pur)
Inbound e*Way	ewRec	Executable = stcewfile.exe Config file = ewRec.cfg Start Up = Auto Collaboration = colRec
Outbound e*Way	ewPur	Executable = stcewfile.exe Config file = ewPur.cfg Start Up = Auto Collaboration = colPur
Multi-Mode e*Way	ewELS	Executable = stceway.exe Config file = ewELS.cfg Start Up = Auto Collaboration = colELS
e*Way Connection	JMS	Type: SeeBeyond JMS

Table 4 ELSE2E Components

Component	Logical Name	Settings
Collaboration	colRec	Collab Rule = crRec Sub = etRec from <EXTERNAL> Pub = etRec to JMS
	colELS	Collab Rule = crELS Sub = etRec from JMS Pub = etPur to JMS
	colPur	Collab Rule = crPur Sub = etPur from JMS Pub = etPur to <EXTERNAL>

3.7.2 Test the Schema

Test the scenario by sending data into the system and verifying the output.

Create the Input Data File

- 1 Use a text editor such as Notepad to create a text file that contains the following data.

```
10000,1,3
10000,3,3
10000,2,3
10001,1,2
10002,1,1
10001,2,2
```

- 2 Save the file as Save the file as **c:\data\ELSE2Einput.txt**.

3.7.3 Start the Schema

- 1 Use the following command to start the Control Broker from a command line.

```
stccb.exe -rh hostname -rs ELSE2E -ln hostname_cb -un username -up password
```

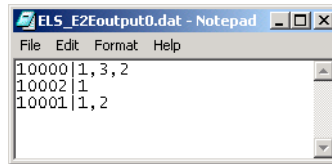
- 2 Start the e*Gate Monitor.
- 3 Verify that all the components in the schema are running.

Testing in Windows 2000

- 1 Once all the scenario components have been started successfully, use Windows Explorer to navigate to **c:\data**
- 2 Change the file extension on the input file **ELSE2Einput.txt** to **.fin**.
- 3 Click **Yes** to confirm this choice.
- 4 Verify that the extension changes to **.~in** indicating that the **ewRec** e*Way has retrieved the file.

- 5 After a few moments, the output file, **ELSE2Eoutput#.dat**, should appear in the directory, indicating a successful conclusion to the test.
- 6 Verify that the output looks like the following.

Figure 37 ELS Output File



3.7.4 Troubleshoot any problems

Java Troubleshooting

If you are unable to find the problem by reviewing the table [“ELSE2E Components” on page 57](#), try using [“Debugging and Log Files” on page 83](#) to start the trace or error log component in your schema.

Table 5 Compiler Errors

Error	Possible Solution
unreachable statement	In the executeBusinessRules method, the return retBoolean rule needs to be placed below the user-defined rules.

Monk End-to-End Scenario

This chapter describes the basic steps required to create a simple e*Gate 4.5.2 end-to-end scenario using a Monk Collaboration to do the data transformation.

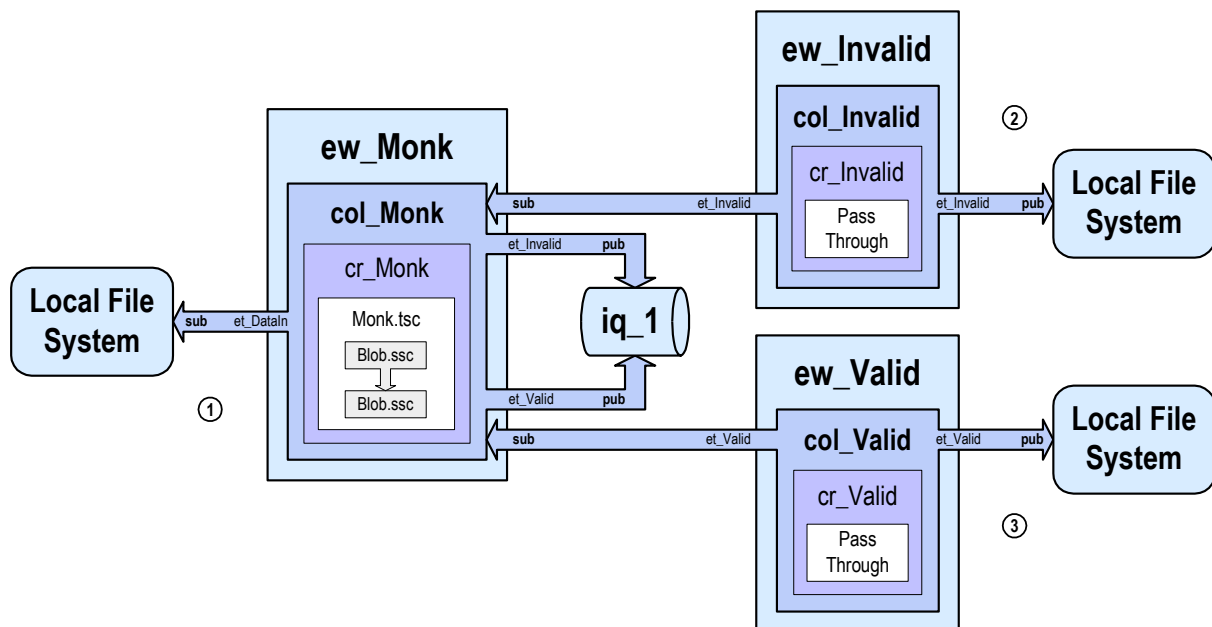
*Note: For information on building more complex scenarios, see the e*Gate Integrator User's Guide.*

Monk End-to-End Scenario Business Problem

The business needs a way to determine if incoming data is valid or not and route the data accordingly. Valid data is sent to one system while invalid data is sent to another.

The e*Gate solution uses a Monk Collaboration in the inbound e*Way to separate and route the data. Figure 38 shows the component relationships of the e*Gate system that implements this business logic.

Figure 38 Monk End-to-End Scenario Overview



Notes on the Monk End-to-End Scenario Overview

① **ew_Monk** brings the data into e*Gate and separates it into valid and invalid types.

The **col_Monk** Collaboration in the **ew_Monk** e*Way subscribes to a location on the local file system containing data to process. The data is packaged as Event Type (**et_DataIn**) and processed by the Collaboration Rules Script (**Monk.tsc**). The Monk script determines whether the data is valid or invalid. Invalid data is repackaged as Event Type **et_Invalid** and published to **iq_1**. Valid data is repackaged as Event Type **et_Valid** and also published to **iq_1**.

② **ew_Invalid** retrieves the invalid data and publishes it to a location on the local file system.

The **col_Invalid** in the **ew_Invalid** subscribes to **et_Invalid** Events published by **col_Monk**. The **cr_Invalid** Collaboration Rule uses the Pass Through service to move the data without modifying it. When an **et_Invalid** Event is retrieved, the **ew_Invalid** file e*Way packages it as a text file and writes it to the specified location on the local file system.

③ **ew_Valid** retrieves the valid data and publishes it to a location on the local file system.

The **col_Valid** in the **ew_Valid** subscribes to **et_Valid** Events published by **col_Monk**. When an **et_Valid** Event is retrieved, the **ew_Valid** file e*Way packages it as a text file and writes it to the specified location on the local file system.

Road Map For Setting Up the Scenario

Setting up an end-to-end scenario using Monk follows the same basic steps as setting up the Java end-to-end scenario. However, instead of using the SeeBeyond Java Editors to create the ETDs and Collaboration Rules, you use the Monk editors to create these components.

The basic steps are:

- 1 Verify the e*Gate installation.
- 2 Create a new schema.
- 3 Create the Event Types and Monk ETDs.
- 4 Create the Collaboration Rules and Monk Collaboration Rules scripts.
- 5 Add and configure the e*Ways and IQs.
- 6 Add and define the Collaborations that route the data.
- 7 Review the complete schema.
- 8 Start the schema.
- 9 Test the schema.
- 10 Troubleshoot any problems.

4.1 Verify the e*Gate Installation

This end-to-end scenario is designed to be run on a single machine. Before beginning the configuration process, you must verify that you have all the required software installed on the target machine. Refer to the *e*Gate Integrator Installation Guide* for instructions on how to install the e*Gate components and the e*Gate system requirements.

4.2 Create a New Schema

To create a new schema:

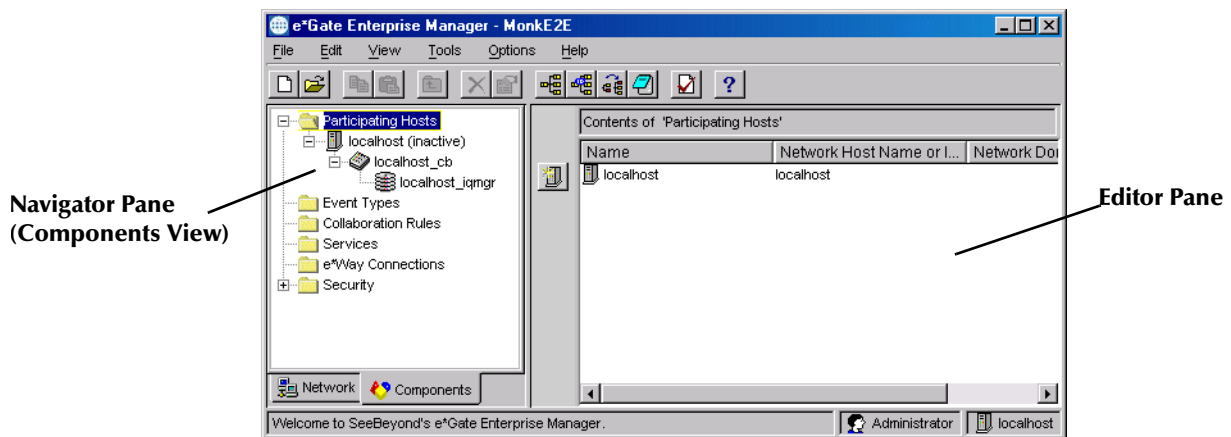
- 1 Start the e*Gate Enterprise Manager and log in as **Administrator** (or another user with administrator privileges) to the appropriate Registry Host.
- 2 In the **Open Schema on Registry Host** dialog box, click **New**.
- 3 In the **Enter New Schema Name** box, type **MonkeE2E**, and then click **Open**.

The Enterprise Manager opens and displays the new **MonkeE2E** schema.

- 4 At the bottom of the navigator (left) pane, click the **Components** tab.

All schema configuration steps are performed with the **Components** tab selected as shown in Figure 39.

Figure 39 e*Gate Enterprise Manager



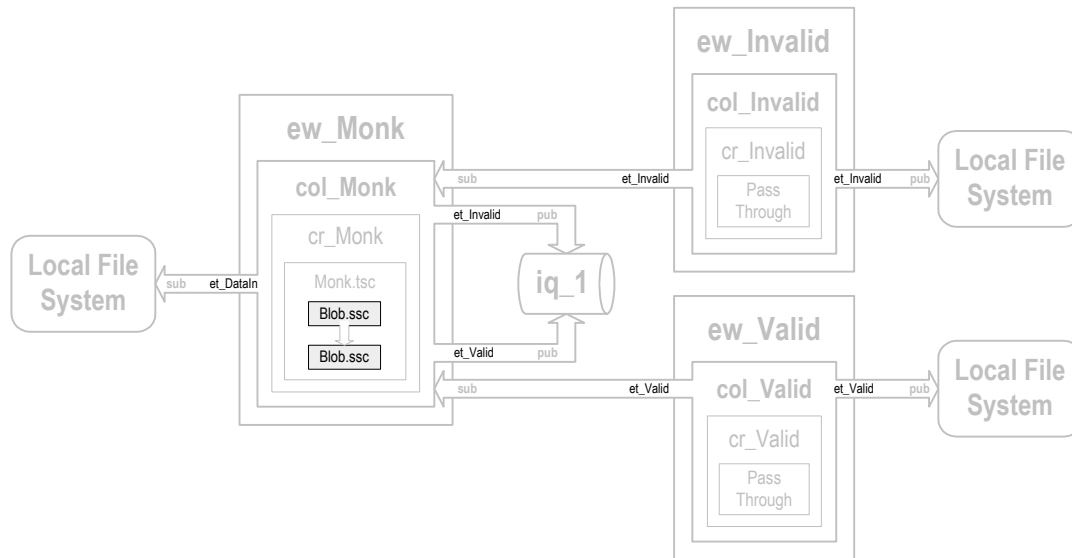
4.3 Create the Event Types and Monk ETDs

This scenario uses three different Event Types that share a common Event Type Definition (ETD).

The ETD is created first using the Monk ETD Editor and then the three Event Types are created in the e*Gate Enterprise Manager.

The darkened portions of Figure 40 show where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 40 Event Types and Monk ETDs



4.3.1 Create the Monk ETD

ETD structures can be very complicated, having many multiple nodes and levels depending on the type of data they model. The data used for this scenario does not have a complicated structure, in fact, an ETD with only one node is all that is required. Use the following procedure to create the **Blob.ssc** ETD.

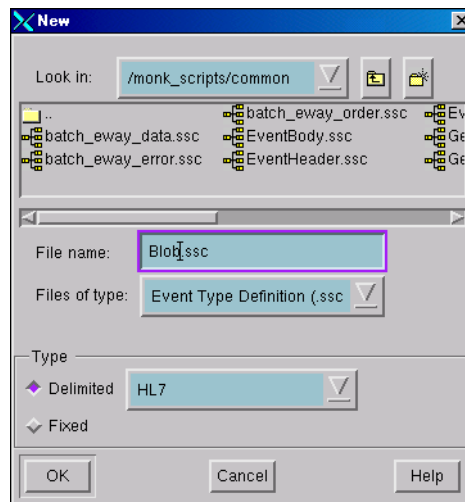
Select the Default ETD Editor

- 1 On the Enterprise Manager **Options** menu, click **Default Editor**.
- 2 Click **Monk**, and then click **OK**.

Start the Monk ETD Editor

- 1 On the **Tools** menu, click **ETD Editor**.
The ETD Editor opens.
- 2 On the **File** menu, click **New**.
- 3 In the **File name** box, type **Blob.ssc**, as shown in Figure 41.

Figure 41 New ETD dialog box



- 4 Click **OK**.
- 5 On the **Edit** menu, click **Add Node**.

A single root node named Blob is added to the ETD work space. No other changes are necessary to the ETD.

- 6 On the **File** menu, click **Save**.
- 7 On the **File** menu, click **Close**.

The ETD Editor closes.

4.3.2 Create the Event Types

Event Types group together similar Events. In this scenario the similarity between the Events named by a particular Event Types is their routing. For example, all the Events routed to the **ew_Valid** e*Way use the Event Type **et_Valid**.

Create the **et_DataIn** Event Type

This Event Type is used to bring the data into the e*Gate system. Use the following procedure to create the **et_DataIn** Event Type:

- 1 In the Enterprise Manager, in the Navigator pane, click the **Event Types** folder.
- 2 On the **File** menu, point to **New**, and then click **Event Type**.
- 3 In the **New Event Type Component** dialog box, type **et_DataIn** and click **OK**.
The **et_DataIn** Event Type is added to the list of Event Types in the Editor pane.
- 4 In the Editor pane, in the **Name** column, double-click **et_DataIn**.
- 5 In the **Event Type - et_DataIn Properties** dialog box, click **Find**.
- 6 In the **Event Type Definition Selection** dialog box, browse for and double-click the ETD created in [“Create the Monk ETD” on page 64](#).

In the **Event Type - et_DataIn Properties** dialog `monk_scripts\common\Blob.ssc` is added to the Event Type Definition area.

- 7 Click **OK** to close the **Event Type - et_DataIn Properties** dialog box.

Create the `et_Valid` and `et_Invalid` Event Types

The `et_Valid` Event Type is used to route data to the `ew_Valid` e*Way. The `et_Invalid` Event Type is used to route data to the `ew_Invalid` e*Way.

Use the procedure described in [“Create the et_DataIn Event Type” on page 65](#), substituting the appropriate Event Type names in step 3 to add the `et_Valid` and `et_Invalid` Event Types.

4.4 Create the Collaboration Rules

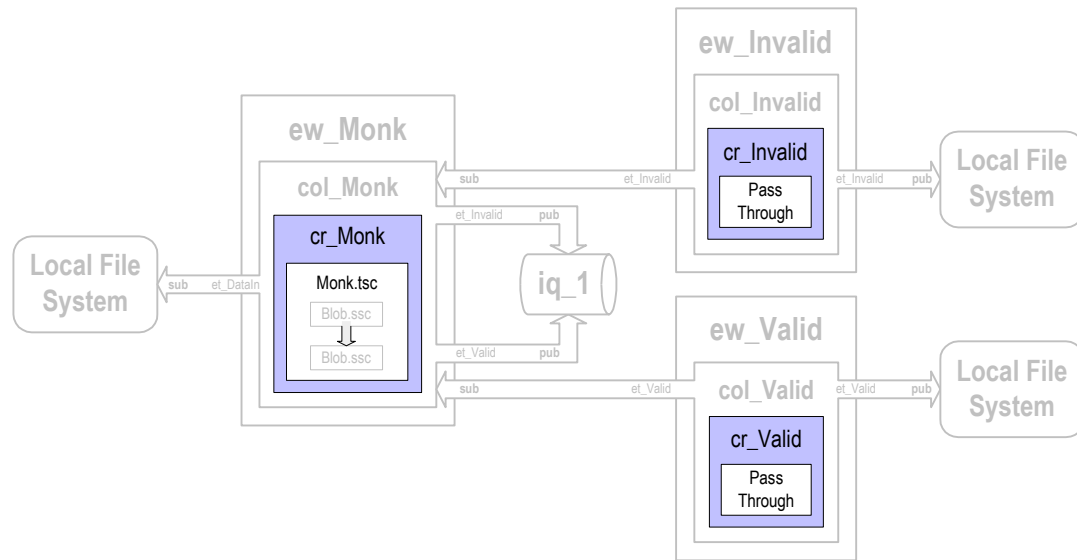
Collaboration Rules are used by the Collaborations that route the data. Collaborations that route data without change use Pass Through Collaboration Rules. Data that must be transformed in addition to being routed must use Collaboration Rules. These rules in turn use a Collaboration Service (programming environment) that allows data transformation.

e*Gate has several Collaboration Services such as Java, C, Monk and others. The choice of a Collaboration Service depends on what you want to do with the data, as well as your available programming resources.

The Monk end-to-end scenario uses a Monk Collaboration to sort the data into Valid and Invalid Event Types. It also uses two Pass Through Collaborations.

Figure 42 shows where these parts fit into the collection of interrelated components that make up the finished schema.

Figure 42 Collaboration Rules and Monk CRSs



4.4.1 Create the Pass Through Collaboration Rules

Pass Through Collaboration Rules do not change the data and therefore do not need the overhead of a sophisticated Collaboration environment such as Java or Monk. The following procedure explains how to create the Pass Through Collaborations used in this scenario.

Create the cr_Invalid and cr_Valid Collaboration Rules

- 1 In the navigator pane of the Enterprise Manager, click the **Collaboration Rules** folder.
- 2 On the **File** menu, point to **New**, and then click **Collaboration Rules**.
- 3 In the **New Collaboration Rules Component** dialog box, type **cr_Invalid** for the Collaboration Rule name, and then click **OK**.
cr_Invalid is added to the list of Collaboration Rules in the Enterprise Manager editor pane.
- 4 On the list of Collaboration Rules, double-click **cr_Invalid**.
- 5 In the **Collaboration Rules - cr_Invalid Properties** dialog box, click the **Subscriptions** tab.
- 6 Use the arrow buttons to move the **et_Invalid** Event Type from the list of **Available Input Event Types** to the list of **Selected Input Event Types**.
- 7 Click the **Publications** tab.
- 8 Use the arrow buttons to move the **et_Invalid** Event Type from the list of **Available Output Event Types** to the list of **Selected Output Event Types**.
- 9 Click **OK** to close the **Collaboration Rules - cr_Invalid Properties** dialog box.

- 10 Repeat steps 2 through 9 to create the **cr_Valid** Collaboration Rule. Substitute **cr_Valid** for the Collaboration Rule name and substitute **et_Valid** for the Event Type.

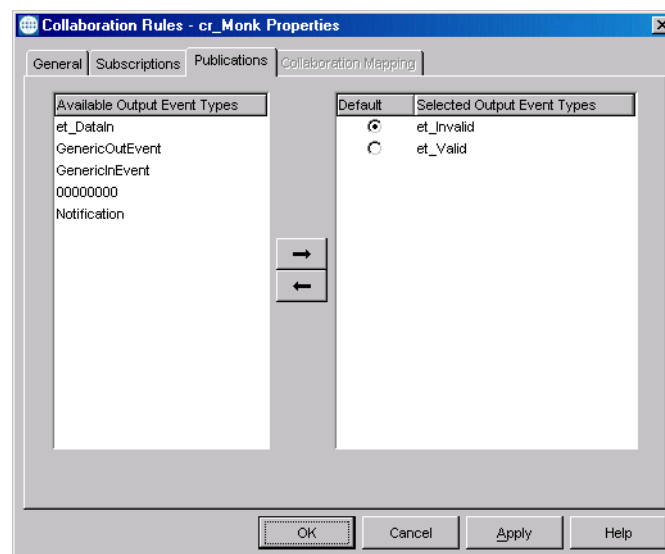
4.4.2 Create the Monk Collaboration Rule

The procedure for creating a Collaboration Rule that uses the Monk Collaboration Service requires that you also create a Collaboration Rules script (CRS) used by the Collaboration Rule. The CRS contains the programming that implements the business logic of the data transformation you desire. The Monk CRS is created using the Monk Collaboration Rules Editor.

Create cr_Monk and Start the Monk Collaboration Rules Editor

- 1 Use steps 1 through 4 from the procedure described in **“Create the cr_Invalid and cr_Valid Collaboration Rules” on page 67** to create a new Collaboration Rule named **cr_Monk**.
- 2 In the **Collaboration Rules - cr_Monk** dialog box, on the **General** tab, in the **Service** list, click **Monk**.
- 3 In the **Collaboration Rules - cr_Monk Properties** dialog box, click the **Subscriptions** tab.
- 4 Use the arrow buttons to move the **et_DataIn** Event Type from the list of **Available Input Event Types** to the list of **Selected Input Event Types**.
- 5 Click the **Publications** tab.
- 6 Use the arrow buttons to move the **et_Invalid** and **et_Valid** Event Types from the list of **Available Output Event Types** to the list of **Selected Output Event Types** as shown in Figure 43.

Figure 43 Completed cr_Monk Publications tab

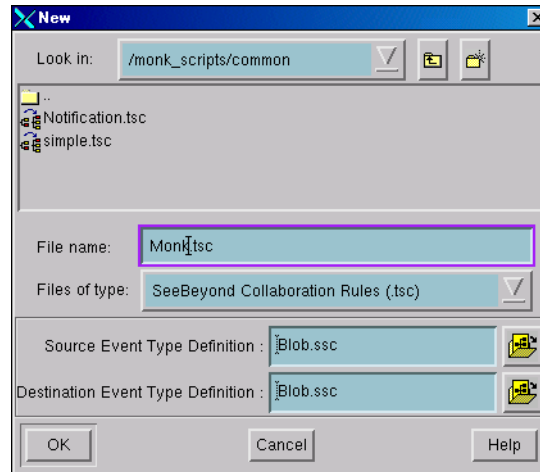


- 7 Click the **General** tab, and then in the **Collaboration Rules** area click **New**.

The Monk Collaboration Editor opens.

- 8 In the **New** dialog box, type **Monk.tsc** as shown in Figure 44.

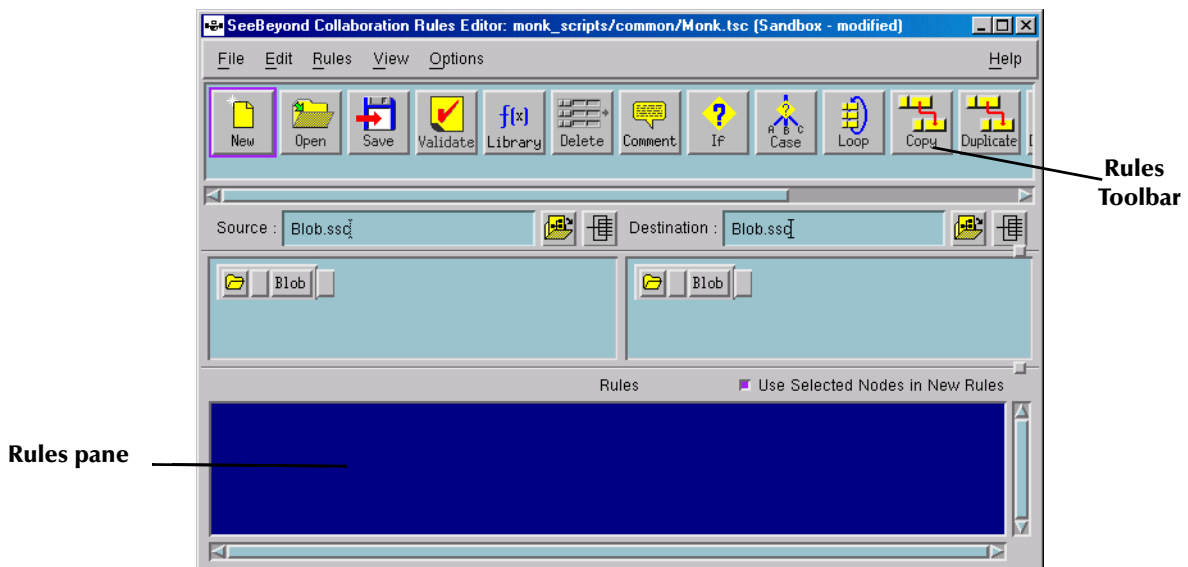
Figure 44 New Collaboration Rules Script Dialog Box



- 9 Also in the **New** dialog box, verify that **SeeBeyond Collaboration Rules (.tsc)** is listed as the file type and that **Blob.ssc** is listed as the Source ETD and Destination ETD, then click **OK**.

The file `monk_scripts\common\Monk.tsc` is now open and the Monk Collaboration Rules Editor should appear similar (after resizing) to Figure 45.

Figure 45 Monk Collaboration Rules Editor



Create Monk.tsc CRS in the Monk Collaboration Rules Editor

The CRS is where you implement the business logic that validates the data. The data is valid if it is a number and invalid if it is not. The programming language used is Monk,

SeeBeyond's Scheme based scripting language. Use the following procedure to create the **Monk.tsc** CRS:

- 1 On the **Rules** menu, click **Add If**.

An **If...Else** rule appears in the **Rules** pane of the Editor.

- 2 Replace **<test>** in the condition portion of the **IF** rule with the following.

```
(number? (get ~input%Blob))
```

This tests whether the data in the input Event is a number. If it is, the statements immediately following the condition line (the **IF** block) are executed; otherwise, the statements under the **ELSE** (the **ELSE** block) are executed.

- 3 Click the empty **IF** block immediately under the **IF** test line test.

- 4 On the **Rules** menu, click **Add Function**.

A generic (non-specific) function line is added to the **IF** block.

- 5 In the text box, replace **<Functions>** with the following.

```
(iq-put "et_Valid" (get ~input%Blob) (list "et_DataIn") 0 0 0)
```

This code retrieves the input data and repackages it as an **et_Valid** Event Type and publishes it to an IQ using **iq-put**.

- 6 Click the empty **ELSE** block immediately below the **ELSE** label.

- 7 On the **Rules** menu, click **Add Function**.

- 8 In the text box replace **<Functions>** with the following.

```
(iq-put "et_Invalid" (get ~input%Blob) (list "et_DataIn") 0 0 0)
```

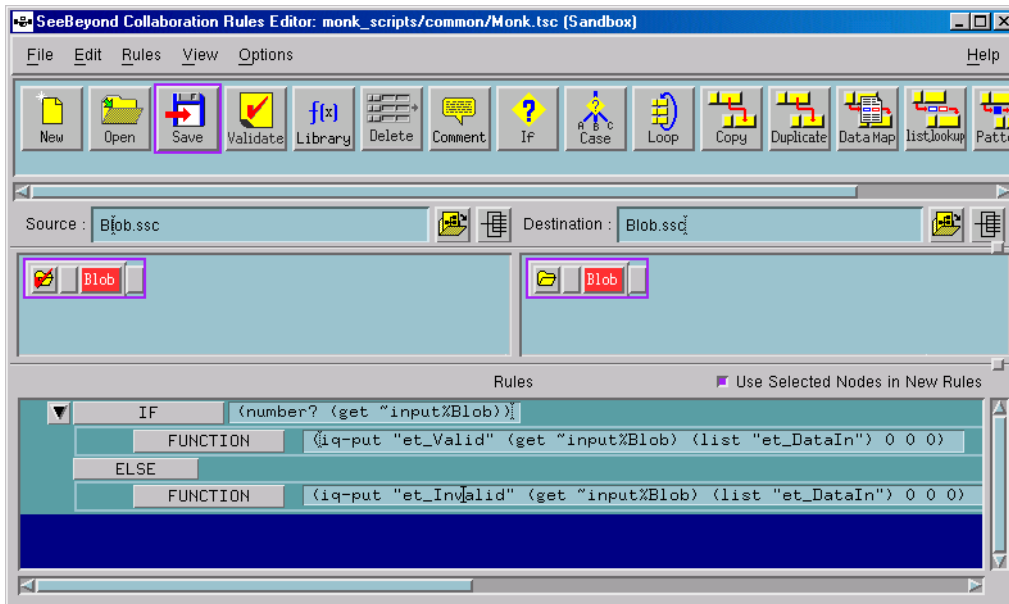
This code retrieves the input data and repackages it as an **et_Invalid** Event Type and publishes it to an IQ using **iq-put**.

- 9 On the **File** menu, click **Validate**.

- 10 Correct any errors, then click **OK**.

The Monk Collaboration Rules Editor should appear as shown in Figure 46.

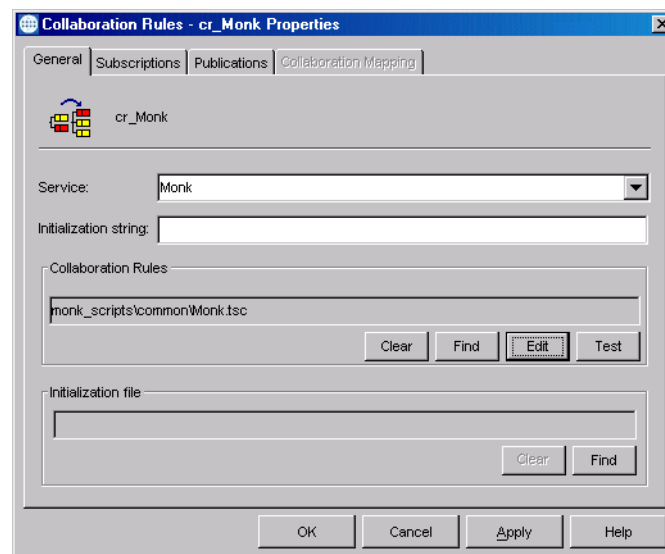
Figure 46 Monk CRS After Adding User-Defined Code



- 11 On the **File** menu, click **Save**.
- 12 On the **File** menu, click **Close**.

The Monk Collaboration Rules Editor closes and in the **Collaboration Rules - cr_Monk Properties** dialog box, **monk_scripts\common\Monk.tsc** is entered in the **Collaboration Rules** area as shown in Figure 47.

Figure 47 Completed cr_Monk Properties



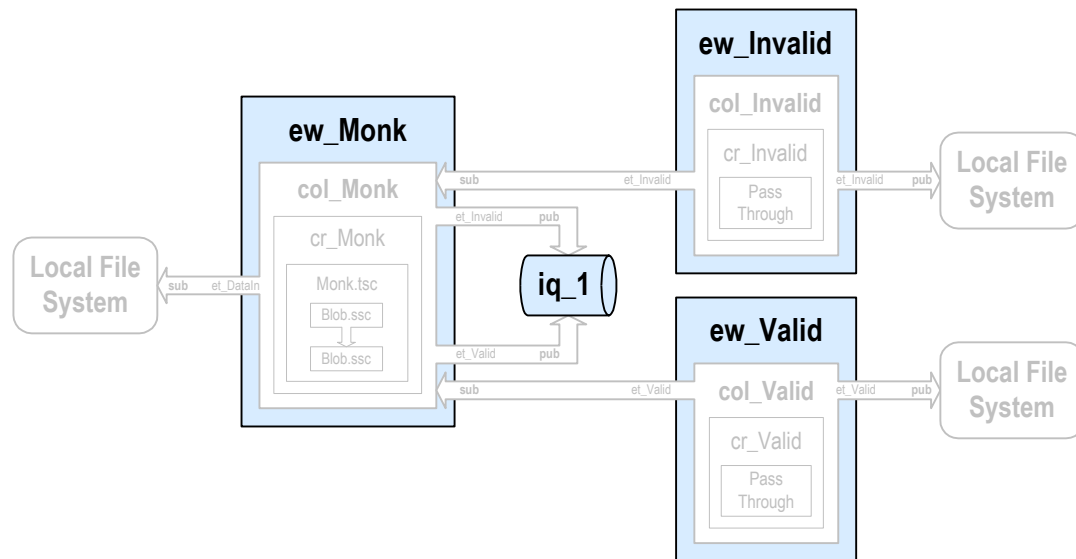
- 13 Click **OK** to close the **Collaboration Rules - cr_Monk Properties** dialog box.

4.5 Add the e*Ways and IQs

Once you have created your ETDs and Collaboration Rules, you are ready to add and configure the e*Gate components that use these parts.

Figure 48 highlights the components added in this step.

Figure 48 e*Ways and IQs



4.5.1 Add and Configure the e*Ways

e*Ways are used to connect to external systems in order to exchange data. They can also perform data transformation as needed.

All of the e*Ways in the Monk end-to-end scenario are file e*Ways, meaning that they are designed to connect to a local file system in order to retrieve and write files.

Add and Configure the ew_Monk file e*Way

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the Control Broker (*hostname_cb*).
- 2 On the **File** menu, point to **New**, then point to **Module**, and then click **e*Way**.
- 3 In the **New e*Way Component** dialog box, type **ew_Monk** for the e*Way name, and then click **OK**.

The **ew_Monk** e*Way is added to the schema.

- 4 Right-click the **ew_Monk** e*Way in the editor pane, and then click **Properties**.
- 5 In the **e*Way - ew_Monk Properties** dialog box, in the **Executable file** area, click **Find**.
- 6 In the **File Selection** dialog box, browse for and double-click the file **stcewfile.exe**.

`bin\stcewfile.exe` is added as the `ew_Monk` e*Way's executable file, causing the component to become a file e*Way.

- 7 In the **Configuration file** area, click **New**.

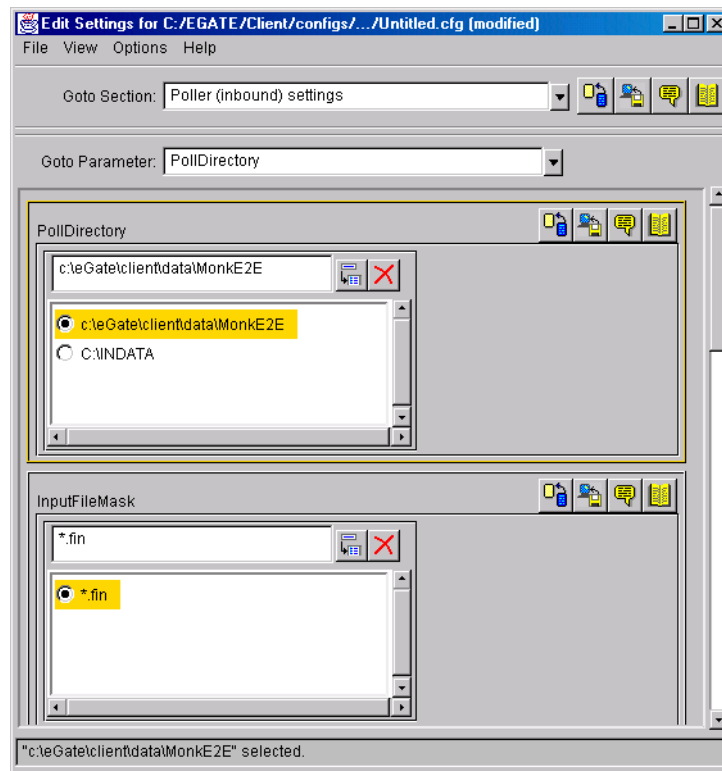
The e*Way Configuration File Editor opens with a default file e*Way configuration file ready for editing.

- 8 In the **Goto Section** list, click **Poller (inbound) settings**.

- 9 In the **PollDirectory** box, change the default value (`c:\INDATA`) to `c:\eGate\client\data\MonkE2E` and then press **ENTER**.

`c:\eGate\client\data\MonkE2E` is added as the directory to be polled to the **PollDirectory** list as shown in Figure 49. No other changes are necessary to the `ew_Monk` e*Way's configuration file.

Figure 49 ew_Monk Configuration File



- 10 On the **File** menu, click **Save**.
- 11 In the **Save As** dialog box, click **Save** to accept the default filename (`ew_Monk.cfg`) and save the file.
- 12 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
`configs\stcewfile\ew_Monk.cfg` is added to the **Configuration file** area in the **e*Way - ew_Monk Properties** dialog box.
- 13 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 14 Click **OK** to close the **e*Way - ew_Monk Properties** dialog box.

Add and Configure the ew_Invalid e*Way

- 1 Use steps 1 through 7 from “[Add and Configure the ew_Monk file e*Way](#)” on [page 72](#) to add another file e*Way named **ew_Invalid** and open its configuration file for editing.
- 2 In the e*Way Configuration File Editor, on the **General Settings** screen, click **NO** for **AllowIncoming**, and **YES** for **AllowOutgoing**.
- 3 In the **Goto Section** list, click **Outbound (send) settings**.
- 4 Add **c:\eGate\client\data\MonkE2E** as the default **OutputDirectory**.
- 5 Add **Invalid_output%d.dat** as the default **OutputFileName**.
No other changes are necessary to the **ew_Invalid** e*Way’s configuration file.
- 6 On the e*Way Configuration File Editor’s **File** menu, click **Save**.
- 7 In the **Save As** dialog box, click **Save** to accept the default file name (**ew_Invalid.cfg**) and save the file.
- 8 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 9 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 10 Click **OK** to close the **e*Way - ew_Invalid Properties** dialog box.

Add and Configure the ew_Valid e*Way

- 1 Use steps 1 through 7 from “[Add and Configure the ew_Monk file e*Way](#)” on [page 72](#) to add another file e*Way named **ew_Valid** and open its configuration file for editing.
- 2 In the e*Way Configuration File Editor, on the **General Settings** screen, click **NO** for **AllowIncoming**, and **YES** for **AllowOutgoing**.
- 3 In the **Goto Section** list, click **Outbound (send) settings**.
- 4 Add **c:\eGate\client\data\MonkE2E** as the default **OutputDirectory**.
- 5 Add **Valid_output%d.dat** as the default **OutputFileName**.
No other changes are necessary to the **ew_Valid** e*Way’s configuration file.
- 6 On the e*Way Configuration File Editor’s **File** menu, click **Save**.
- 7 In the **Save As** dialog box, click **Save** to accept the default file name (**ew_Valid.cfg**) and save the file.
- 8 On the **File** menu, click **Close** to quit the e*Way Configuration File Editor.
- 9 Click the **Start Up** tab, and then select the **Start automatically** check box.
- 10 Click **OK** to close the **e*Way - ew_Valid Properties** dialog box.

4.5.2 Add the IQ

- 1 In the e*Gate Enterprise Manager, in the navigation pane, double-click the IQ manager (*hostname_iqmgr*).

- 2 Click the **General** tab, then from the **IQ Manager Type** drop-down list, select **SeeBeyond Standard**.
- 3 Click the **Start Up** tab, then select the **Start automatically** checkbox, and then click **OK**.
- 4 On the **File** menu, point to **New**, and then click **IQ**.
- 5 In the **New IQ Component** dialog box, type **iq_1** for the IQ name, and then click **OK**.

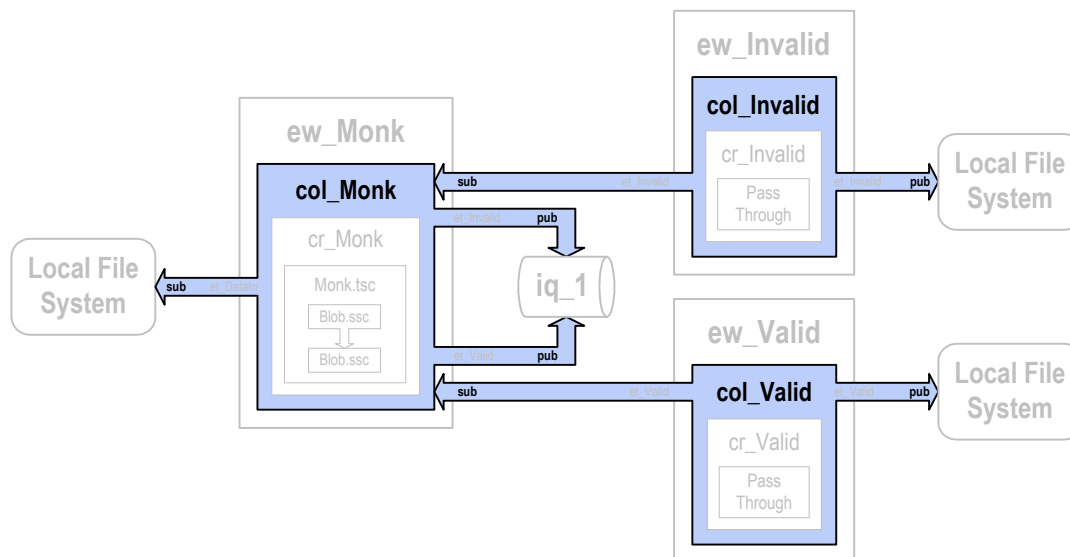
No further configuration is required for **iq_1**.

4.6 Add the Collaborations that Route the Data

The Collaborations are used by the e*Ways to route the data through the e*Gate system. Typically, the Collaborations are configured in upstream to downstream order. That is, the **col_Monk** Collaboration must be configured before configuring either the **col_Invalid** or **col_Valid** Collaborations.

Figure 50 shows the relationships of the collaborations to the remainder of the parts that make up the complete schema.

Figure 50 Collaborations Showing Pub/Sub Relationships



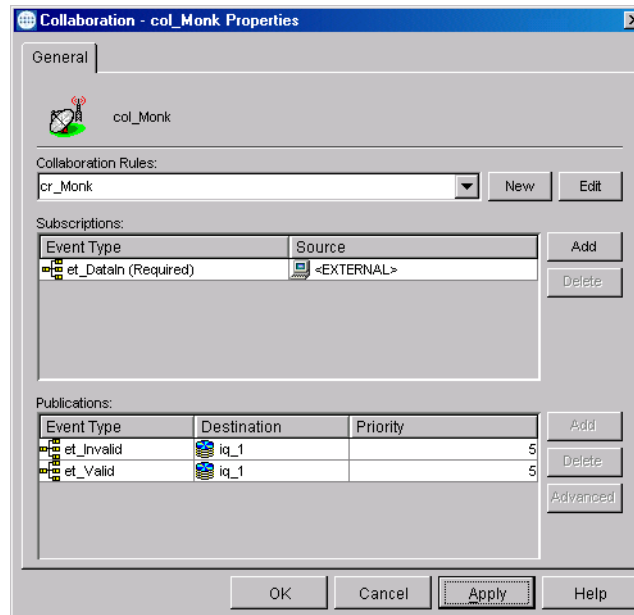
4.6.1 Add and Configure col_Monk

The **col_Monk** Collaboration brings the data into the e*Gate system and routes it depending on the whether the data is valid or invalid. Use the following procedure to add and configure **col_Monk**.

- 1 In the e*Gate Enterprise Manager, in the navigation pane, click the **ew_Monk e*Way**.
- 2 On the **File** menu, point to **New**, click **Collaboration**.
- 3 In the **New Collaboration Component** dialog box, type **col_Monk** for the Collaboration name, and then click **OK**.
- 4 In the editor pane, double-click **col_Monk**.
The **Collaboration - col_Monk Properties** dialog box appears.
- 5 In the **Collaboration Rules** list, click **cr_Monk**.
- 6 In the **Subscriptions** area, click **Add**.
A row is added to the **Subscriptions** box.
- 7 In the **Event Type** column, click **et_DataIn (Required)** on the list, and then in the **Source** column, click **<EXTERNAL>** on the list.
- 8 In the **Publications** area, click **Add**.
A row is added to the **Publications** box.
- 9 In the **Event Type** column, click **et_Invalid** on the list, and then in the **Destination** column, click **iq_1** on the list.
- 10 In the **Publications** area, click **Add**.
- 11 In the **Event Type** column, click **et_Valid** on the list, and then in the **Destination** column, click **iq_1** on the list.

No further configuration is required for **col_Monk**. The **Collaboration - col_Monk Properties** dialog box should look like Figure 51.

Figure 51 Collaboration - col_Monk Properties Dialog Box



- 12 Click **OK** to close the **Collaboration - col_Monk Properties** dialog box.

4.6.2 Add and Configure col_Invalid

The **col_Invalid** Collaboration subscribes to **et_Invalid** Events published by **col_Monk**. It publishes them to **<EXTERNAL>**. Use the following procedure to add and configure **col_Invalid**.

- 1 Use steps 1 through 4 from **"Add and Configure col_Monk"** on page 76 to add a Collaboration to the **ew_Invalid** e*Way named **col_Invalid** and open its properties dialog box.
- 2 In the **Collaboration Rules** list, click **cr_Invalid**.
- 3 In the **Subscriptions** area, click **Add**.
- 4 In the **Event Type** column, click **et_Invalid (Required)** on the list, and then in the **Source** column, click **col_Monk** on the list.
- 5 In the **Publications** area, click **Add**.
- 6 In the **Event Type** column, click **et_Invalid** on the list, and then in the **Destination** column, click **<EXTERNAL>** on the list.

No further configuration is required for **col_Invalid**.

- 7 Click **OK** to close the **Collaboration - col_Invalid Properties** dialog box.

4.6.3 Add and Configure col_Valid

The **col_Valid** Collaboration subscribes to **et_Valid** Events published by **col_Monk**. It publishes them to <EXTERNAL>. Use the following procedure to add and configure **col_Valid**.

- 1 Use steps 1 through 4 from “[Add and Configure col_Monk](#)” on page 76 to add a Collaboration to the **ew_Valid** e*Way named **col_Valid** and open its properties dialog box.
 - 2 In the **Collaboration Rules** list, click **cr_Valid**.
 - 3 In the **Subscriptions** area, click **Add**.
 - 4 In the **Event Type** column, click **et_Valid (Required)** on the list, and then in the **Source** column, click **col_Monk** on the list.
 - 5 In the **Publications** area, click **Add**.
 - 6 In the **Event Type** column, click **et_Valid** on the list, and then in the **Destination** column, click <EXTERNAL> on the list.
- No further configuration is required for **col_Valid**.
- 7 Click **OK** to close the **Collaboration - col_Valid Properties** dialog box.

4.7 Test the Scenario

The following road map steps are covered in this section:

- Step 7: Review the complete schema.
- Step 8: Start the schema.
- Step 9: Test the schema.
- Step 10: Troubleshoot any problems.

4.7.1 Review the Complete Schema

The following table lists all the components for the schema. Double-check all the settings. Substitute the name of the machine running the schema for *hostname* where applicable.

Table 6 JavaE2E Components

Component	Logical Name	Settings
Schema	MonkE2E	
Control Broker	<i>hostname_cb</i>	
IQ Manager	<i>hostname_iqmgr</i>	Start Up = Auto

Table 6 JavaE2E Components

Component	Logical Name	Settings
Event Type	et_DataIn	ETD = Blob.ssc
	et_Invalid	ETD = Blob.ssc
	et_Valid	ETD = Blob.ssc
ETD	Blob.ssc	
Collaboration Rule	cr_Monk	Service = Monk CRS = Monk.tsc Sub = et_DataIn Pub = et_Invalid, et_Valid
	cr_Invalid	Service = Pass Through Sub = et_Invalid Pub = et_Invalid
	cr_Valid	Service = Pass Through Sub = et_Valid from col_Monk Pub = et_Valid to <EXTERNAL>
Collaboration Rules script	Monk.tsc	Source = Blob.ssc Destination = Blob.ssc
Inbound e*Way	ew_Monk	Executable = stcewfile.exe Config file = ew_Monk.cfg Start Up = Auto Collaboration = col_Monk
Outbound e*Ways	ew_Invalid	Executable = stcewfile.exe Config file = ew_Invalid.cfg Start Up = Auto Collaboration = col_Invalid
	ew_Valid	Executable = stceway.exe Config file = ew_Valid.cfg Start Up = Auto Collaboration = col_Valid
IQ	iq_1	Service = STC_Standard
Collaboration	col_Monk	Collab Rule = cr_Monk Sub = et_DataIn from <EXTERNAL> Pub = et_Invalid, et_Valid to iq_1
	col_Invalid	Collab Rule = cr_Invalid Sub = et_Invalid from col_Monk Pub = et_Invalid to <EXTERNAL>
	col_Valid	Collab Rule = cr_Valid Sub = et_Valid from col_Monk Pub = et_Valid to <EXTERNAL>

4.7.2 Test the Schema

Test the scenario by sending data into the system and verifying the output.

Create the Input Data File

- 1 Use a text editor such as Notepad to create a text file that contains the following data.

```
1
2
3
error
```

The Collaboration Rule used by the **ew_Monk e*Way** tests the data to see whether it is numeric. The above data contains both “valid” and “invalid” data for the test.

- 2 Save the file as Save the file as
c:\eGate\client\data\MonkE2E\MonkE2Einput.txt.

4.7.3 Start the Schema

- 1 Use the following command to start the Control Broker from a command line.

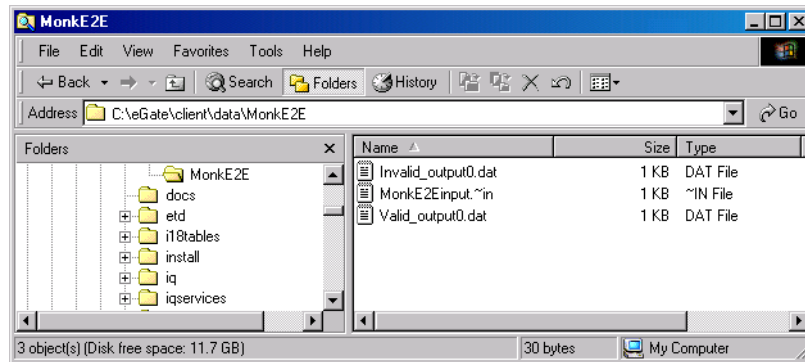
```
stccb.exe -rh hostname -rs MonkE2E -ln hostname_cb -un username -  
up password
```

- 2 Start the e*Gate Monitor.
- 3 Verify that the all the components in the schema are running.

Testing in Windows 2000

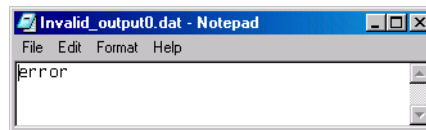
- 1 Once all the scenario components have been started successfully, use Windows Explorer to navigate to **c:\eGate\client\data\MonkE2E**.
- 2 Change the file extension on the input file **MonkE2Einput.txt** to **.fin**.
- 3 Click **Yes** to confirm this choice.
- 4 Verify that the extension changes to **.~in** indicating that the **ew_Monk e*Way** has retrieved the file.
- 5 Almost immediately, two output files (**Invalid_output0.dat** and **Valid_output0.dat**) appear in the directory as shown in Figure 52.

Figure 52 MonkE2E Output Files in Windows Explorer



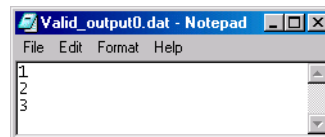
- 6 Verify that the output for **Invalid_output0.dat** is like that shown in Figure 53.

Figure 53 Invalid_output0.dat File



- 7 Verify that the output for **Valid_output0.dat** is like that shown in Figure 54.

Figure 54 Valid_output0.dat File



4.7.4 Step 10: Troubleshoot any problems

If you do not get the expected results, do the following:

- Double-check your schema. Use the table at [“Review the Complete Schema” on page 78](#). Make sure that you created and configured all the components as directed in the instructions.
- Use the e*Gate Monitor to confirm that all of the components are running; manually start any that may not have started correctly.
- If a component starts successfully but halts immediately, the most likely cause is misconfiguration. Check the following:
 - ♦ Does each e*Way have an executable file and a configuration file associated with it?
 - ♦ Does each Event Type have an ETD file assigned?
 - ♦ Do the validating Collaboration Rules have the Monk service assigned, and the proper Collaboration Rules file both created and assigned?

- ♦ Is the IQ Manager running, and did you configure both IQs?
- If all of the components start and stay running, the most likely problem is that the validation logic in the **Inbound** e*Way's Collaboration Rules script is not working properly. Use the Collaboration Rules Editor and check that you have created the file in accordance with the procedure in **“Create the Monk Collaboration Rule” on page 68**.

Debugging and Log Files

One of the easiest ways to debug your e*Gate configuration is through the use of log files. All executable components—BOBs, e*Ways, IQ Managers, and Control Brokers—have the ability to create log files that contain whatever level of debugging information you select.

5.1 Log File Locations

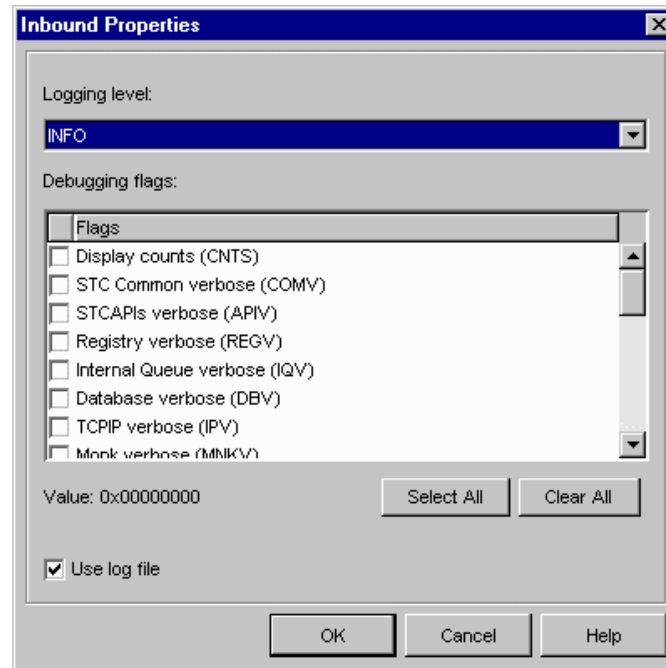
All log files are stored in the `\eGate\client\logs` directory on the Participating Host running the elements that generate the log entries. Logs are named after the component that creates them; for example, the component **Inbound_eWay** creates a log file called **Inbound_eWay.log**.

5.2 Generating Log Files

To configure a component to generate a log file:

- 1 In the e*Gate Enterprise Manager window, select the component that you want to configure and display its properties.
- 2 Select the **Advanced** tab, then click **Log**.
- 3 Select the desired logging options (see [Figure 55 on page 84](#)).

Figure 55 Logging Options



You can view a component’s log using any text editor, and you can view the log while the component is still running. However, depending on the editor, you may need to re-read the file to “refresh” your view of the log data. You cannot get log updates “on the fly.”

The most common error most first-time e*Gate developers will find in a log file is, “Unable to load module configuration.” This message means that you have created an e*Gate component but not assigned both an executable file and a configuration file to it.

For more information about logging and debugging options, see the *e*Gate Integrator System Administration and Operations Guide*.

Index

C

collaboration
 CRS 68
 java 13, 14
 java pass through 21
 Monk 61, 62
 conventions, writing in document 10

D

debugging 83
 default value 19
 destination event 23
 document
 conventions 11

E

e*Way
 JMS connection 14, 15, 30
 multi-mode 13
 enterprise manager 16
 ETD
 standard 16
 ETD editor
 java 17, 19, 20
 Monk 64
 ETD Wizard 17
 event type
 definition 16
 java 16
 event type definition 17

I

installation
 Windows NT/2000 10

J

java
 collaboration rule 20
 pass through 13, 14, 21
 JMS Connection 14

M

Monk
 editor 62

N

node names 17
 node property
 offset 19
 structure 19

P

package name 17
 poller 27

R

root node name 17

S

SeeBeyond Web site 12
 service
 java collaboration 13, 22
 source event 23
 supporting documents 11

V

variable
 length 19

W

Windows 2000 13
 Windows NT/2000
 installation 10