SeeBeyond ICAN Suite

e*Way Intelligent Adapter for MSMQ User's Guide

Release 5.0.5 for Schema Run-time Environment (SRE)

Java Version

SEEBEYOND®

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.

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

Version 20050406012212.

Contents

Chapter 4

Chapter 5

Chapter 1

Introduction

This chapter introduces you to SeeBeyond™ Technology Corporation's (SeeBeyond™) e*Way Intelligent Adapter for the MSMQ (Microsoft Message Queuing). It includes an overview of this manual and the e*Way, and a list of system requirements for installation.

1.1 MSMQ Introduction

MSMQ is "middleware" that provides commercial messaging and queuing services. Messaging enables programs to communicate with each other via messages instead of direct connection.

Placing these messages in queues for temporary storage frees up programs to continue to work independently. This process also allows communication across a network of dissimilar components, processors, operating systems, and protocols.

1.2 Intended Reader

The reader of this guide is presumed:

- To be a developer or system administrator with responsibility for maintaining the e*Gate system
- To have moderate to high-level knowledge of Windows operations and administration
- To be thoroughly familiar with MSMQ, Java, and Windows-style operations

1.3 MSMQ e*Way Overview

The Java-enabled MSMQ e*Way enables the e*Gate system to exchange data with Microsoft Message Queue (MSMQ) Service version 2.0. MSMQ is a client-server message broker supporting an open application programming interface (API), and it is available on a variety of operating systems, including Windows.

1.3.1 e*Way Operation

The MSMQ e*Way is configurable and transparently integrates existing systems and databases to MSMQ through e*Gate. This document explains how to install and configure the MSMQ Java-enabled e*Way.

1.3.2 Intelligent Queues and IQ Managers

A key component within e*Gate is the Intelligent Queue (IQ), which provides storage for data within the e*Gate system. The MSMQ IQ allows e*Gate to leverage existing MSMQ operational and management infrastructure for its underlying persistent storage. The MSMQ IQ makes use of queuing capabilities.

Queuing allows applications to run independently of one another, at different speeds and times. Applications can send messages to a queue and get messages from a queue at any time. The MSMQ e*Way allows e*Gate to seamlessly exchange data with applications that are MSMQ enabled.

1.3.3 e*Way Components

The MSMQ e*Way is made up of the following components:

- Multi-Mode e*Way, a core e*Gate component, executable file, **stceway.exe** (see **Chapter 3** for details)
- Configuration files that the e*Gate Schema Designer’s e*Way Configuration Editor uses to define configuration parameters (see **Chapter 4** for details)
- Additional files necessary for operation, as shown in **Table 1** (provides a complete list of installed files)

1.4 Supported Operating Systems

The MSMQ e*Way is supported on the following operating systems:

- Windows 2000, Windows XP, and Windows Server 2003

1.5 System Requirements

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

- An e*Gate Participating Host
- 4 MB of free disk space

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

- A TCP/IP network connection

The client components of MSMQ have their own requirements. See that system's documentation for more details.

1.6 External System Requirements

To enable the e*Way to communicate properly with the MSMQ system, the following are required:

- Microsoft Messaging Queues, version 1.0 or 2.0
- SQL server version 6.5 or 7.0 (Windows only)

Chapter 2

Installation

This chapter explains how to install the e*Way Intelligent Adapter for MSMQ.

2.1 Installation on Windows Systems

2.1.1 Pre-installation

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

2.1.2 Installation Procedure

To install the Java-enabled MSMQ e*Way on Windows systems

1. Log in as an Administrator on the work station on which you want to install the e*Way.
2. Insert the e*Way installation CD-ROM into the CD-ROM drive.
3. Insert the installation CD into the CD-ROM drive.

   If Autorun is enabled, the setup program automatically starts. Otherwise:

   - On the task bar, click the **Start** button, then click **Run**.
   - In the **Open** field, type **D:\setup\setup.exe** where **D:** is your CD-ROM drive.
4. The InstallShield setup application launches. Follow the installation instructions until you come to the **Please choose the product to install** dialog box.
5. Select **e*Gate Integrator**, then click **Next**.
6. Follow the on-screen instructions until you come to the second **Please choose the product to install** dialog box.
7. Clear the check boxes for all selections except **Add-ons**, and then click **Next**.
8. Follow the on-screen instructions until you come to the **Select Components** dialog box.

9 Highlight (but do not check) **e*Ways**, and then click the **Change** button. The **Select Sub-components** dialog box appears.

10 Select **MSMQ e*Way**. Click the **Continue** button to return to the **Select Components** dialog box, then click **Next**.

11 Follow the rest of the on-screen instructions to install the Java-enabled MSMQ e*Way. For details of e*Gate installation, refer to the ***SeeBeyond ICAN Suite Installation Guide***.
Be sure to install the e*Way files in the suggested client installation directory. The installation utility detects and suggests the appropriate installation directory. **Unless you are directed to do so by SeeBeyond support personnel, do not change the suggested installation directory setting**.

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

2.2 Files/Directories Created by the Installation

The MSMQ e*Way installation process installs the files shown in Table 1 within the e*Gate **client** directory tree. Files are installed within the **eGate\client\** tree on the Participating Host and committed to the “default” schema on the Registry Host.

Table 1 Files Created by Installation

Directories	Files
configs\msmq	msmq.def
bin\win32	stcmsmqapi.dll
classes\	stcmsmqapi.jar

Note: *You must manually add the **stcmsmqapi.dll file** to the classpath of your Collaboration Rules. You do not have to take this step, however, for the Java Pass Through Collaboration Rules.*

Chapter 3

Multi-Mode e*Way Configuration

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

3.1 Multi-Mode e*Way Properties

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

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

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

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

   The e*Way Configuration Editor window opens.
8. When the **Settings** page opens, set the configuration parameters for this e*Way's configuration file (see **"JVM Settings" on page 11** and **"General Settings" on page 14** for details).
9. After selecting the desired parameters, click **Save** on the **File** menu to save the configuration (**.cfg**) file.
10. Close the **.cfg** file and e*Way Configuration Editor.
11. Set the properties for the e*Way in the **e*Way Properties** dialog box.
12. Click **OK** to close the dialog box and save the properties.

3.2 JVM Settings

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

JNI DLL Absolute Pathname

Description

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

C:\eGate\client\bin\Jre or C:\jdk\jre\bin\server

This parameter is *mandatory*.

Required Values

A valid path name.

Additional Information

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

OS	Java 2 JNI DLL Name
Windows systems	jvm.dll

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

```
%MY_JNIDLL%
```

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

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

CLASSPATH Prepend

Description

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

Required Values

An absolute path or an environmental variable.

Additional Information

If left unset, no paths are prepended to the CLASSPATH environment variable unless CLASSPATH Append From Environment Variable is set to yes, in which case the classpath is appended to the variable.

Existing environment variables may be referenced in this parameter by enclosing the variable name in a pair of “%” symbols. For example:

```
%MY_PRECLASSPATH%
```

CLASSPATH Override

Description

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

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

Required Values

An absolute path or an environmental variable.

Additional Information

Existing environment variables may be referenced in this parameter by enclosing the variable name in a pair of “%” symbols. For example:

```
%MY_CLASSPATH%
```

CLASSPATH Append From Environment Variable

Description

Specifies the classpath append from the environmental variable.

Required Values

YES or **NO**. Enter Yes to append.

Initial Heap Size

Description

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

Required Values

An integer between 0 and 2147483647.

Maximum Heap Size

Description

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

Required Values

An integer between 0 and 2147483647.

Maximum Stack Size for Native Threads

Description

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

Required Values

An integer between 0 and 2147483647.

Maximum Stack Size for JVM Threads

Description

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

Required Values

An integer between 0 and 2147483647.

Disable JIT

Description

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

Required Values

YES or **NO**.

Note: *This parameter is not supported for Java release 1.*

Allow Remote Debugging of JVM

Description

Specifies whether to allow remote debugging of the JVM.

Required Values

YES or **NO**.

3.3 General Settings

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

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

3.3.1 Rollback Wait Interval

Description

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

Required Values

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

3.3.2 Standard IQ FIFO

Description

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

Required Values

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

Chapter 4

e*Way Connection Configuration

This chapter explains how to configure the e*Way Intelligent Adapter for MSMQ Connection configuration.

4.1 Configuring e*Way Connections

e*Way Connections are set using the Schema Designer.

To create and configure e*Way Connections

1. In the Schema Designer's **Component** editor, select the **e*Way Connections** folder.
2. On the palette, click on the **Create a New e*Way Connection** button.
3. The **New e*Way Connection Component** dialog box opens, enter a name for the **e*Way Connection**. For the purposes of the sample implementation enter **cp_msmq** as the name.
4. Double-click on the new **e*Way Connection**. The **e*Way Connection Properties** dialog box opens.
5. From the **e*Way Connection Type** drop-down box, select **MSMQ**.
6. Enter the **Event Type "get" interval** in the dialog box provided. 100 milliseconds is the configured default. The "get interval is the intervening period at which, when subscribed to, the e*Way connection is polled.
7. From the **e*Way Connection Configuration File**, click **New** to create a new Configuration File for this e*Way Connection. (To use an existing file, click **Find**.)

The MSMQ e*Way Connection configuration parameters are organized into the following sections.

- **General Settings** on page 16

4.1.1 General Settings

This section contains a set of top level parameters:

Connection Type

Description

String-set. Specifies the JMS Messaging Model. There are two connection types.

- Queue (point to point)

Required Values

Value is Queue. Queue is the configured default.

Transaction Type

Description

String-set. Specifies the Transaction Type.

- **Non-Transactional**. Provides the highest level of performance, with the minimum level of message protection. No rollback is available during the send and receive period, causing the possible loss of data in the case of a system error. When the **send()** method is called the transaction is immediate.

Required Values

Non-Transactional type.

Delivery Mode

Description

String-set. Specifies the Message Delivery Mode. Marking the message as:

- **Persistent**: Provides the highest level of protection, ensuring that the message is saved to a reliable persistent store by the Message Server before the publish() method returns.

This setting must match the setting in the MSMQ queue manager

Required Values

Persistent. Persistent is the only delivery mode supported at this time.

Default Outgoing Message Type

Description

String-set. Specifies the Message Type to creating during publish/send. The outgoing message type is published within the message header. This is only relevant to sending, providing information for the receiver.

Required Values

Text. The supported type default is Bytes. Binary data is not currently supported.

Factory Class Name

Description

String-set. Specifies the Factory class to be used to connect to the JMS IQ Manager. This is advanced configuration to be utilized in future development, and should not be changed from the default.

Required Values

The valid factory class name. The configured default is **com.stc.eways.stcmsmqapi.JMSMSMQFactoryAdapter**. Retain the default setting.

Chapter 5

Implementation

This chapter explains a sample schema that helps you understand how to implement the e*Way Intelligent Adapter for MSMQ in a production environment.

5.1 Implementation Overview

This section explains how to implement the MSMQ e*Way using e*Gate Integrator schema sample included on your installation CD-ROM. Find this sample on the CD-ROM at the following path location:

\samples\ewmsmq\Java

This sample allows you to observe an end-to-end data-exchange scenario involving e*Gate, the e*Way, and sample operations. This chapter explains how to implement the this sample schema that uses the MSMQ e*Way.

You can also use the procedures given in this chapter to create your own schemas based on the samples provided. It is recommended that you use a combination of both methods, creating your own schema like each sample, then importing the samples into e*Gate to check your results.

Before Importing or Running a Sample Schema

To import and/or run a sample schema, the MSMQ e*Way must be installed, MSMQ must be available to the schema, and all appropriate configuration of MSMQ must have been done.

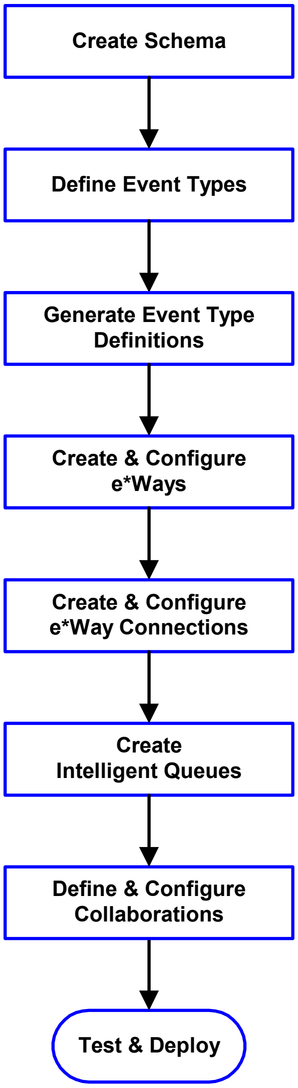
To import a sample schema

1. Copy the desired .**zip** file, for example, **msmqSample.zip,** from the **samples\ewmsmq\Java** directory in the install CD-ROM to your desktop or to a temporary directory, then unzip the file.
2. Start the e*Gate Schema Designer.
3. On the **Open Schema from Registry Host** dialog box, click **New**.
4. On the **New Schema** dialog box, click **Create from export**, and then click **Find**.
5. On the **Import from File** dialog box, browse to the directory that contains the sample schema.
6. Click the **.zip** file then click **Open**.

   The schema is installed.

To create the sample schema

Use the following implementation sequence:


Create Schema
Define Event Types
Generate Event Type Definitions
Create & Configure e*Ways
Create & Configure e*Way Connections
Create Intelligent Queues
Define & Configure Collaborations
Test & Deploy

1 The first step is to create a new schema. The rest of these steps apply only to this schema.

2 The second step is to create and define the Event Types you are transporting and processing within the schema.

3 Third, you need to associate the Event Types created in the previous step with Event Type Definitions (ETDs) you want to use in the schema.

4 The fourth step is to create and configure the required e*Ways.

5 The fifth step is to configure the e*Way Connections.

6 Now you need to create Intelligent Queues (IQs) and IQ Managers (if necessary) to hold published Events.

7 You need to create the desired Collaboration Rules for your schema, along with their associated Business Rules.

8 Next, you need to define and configure the Collaborations between Event Types.

9 Finally, you must check and test your Schema. Once you have verified that it is working correctly, you can deploy it to your production environment.

Chapter Organization

This chapter is set up sequentially, and you can use it as a tutorial to teach you how to implement the MSMQ e*Way.

5.2 Sample Schema: msmqSample

This section explains how to implement the sample schema for the MSMQ e*Way. The schema demonstrates how to configure the essential features of the e*Way in a typical e*Gate environment.

5.2.1 Sample Schema Overview

This section provides a general overview of the sample schema, its configuration, and how it operates. The name of this schema is msmqSample, and it is contained in the import file **msmqSample.zip**.

Schema Operation

This sample schema has the following input/output setup:

- **Input**: A file from an external local system, to be sent to an MSMQ queue.
- **Output**: The response file, to an external local system.

The schema uses a Pass Through Multi-Mode e*Way to pick up a file from an external system. This file is then transferred to a Java Messaging Service (JMS) IQ Manager via an e*Way Connection.

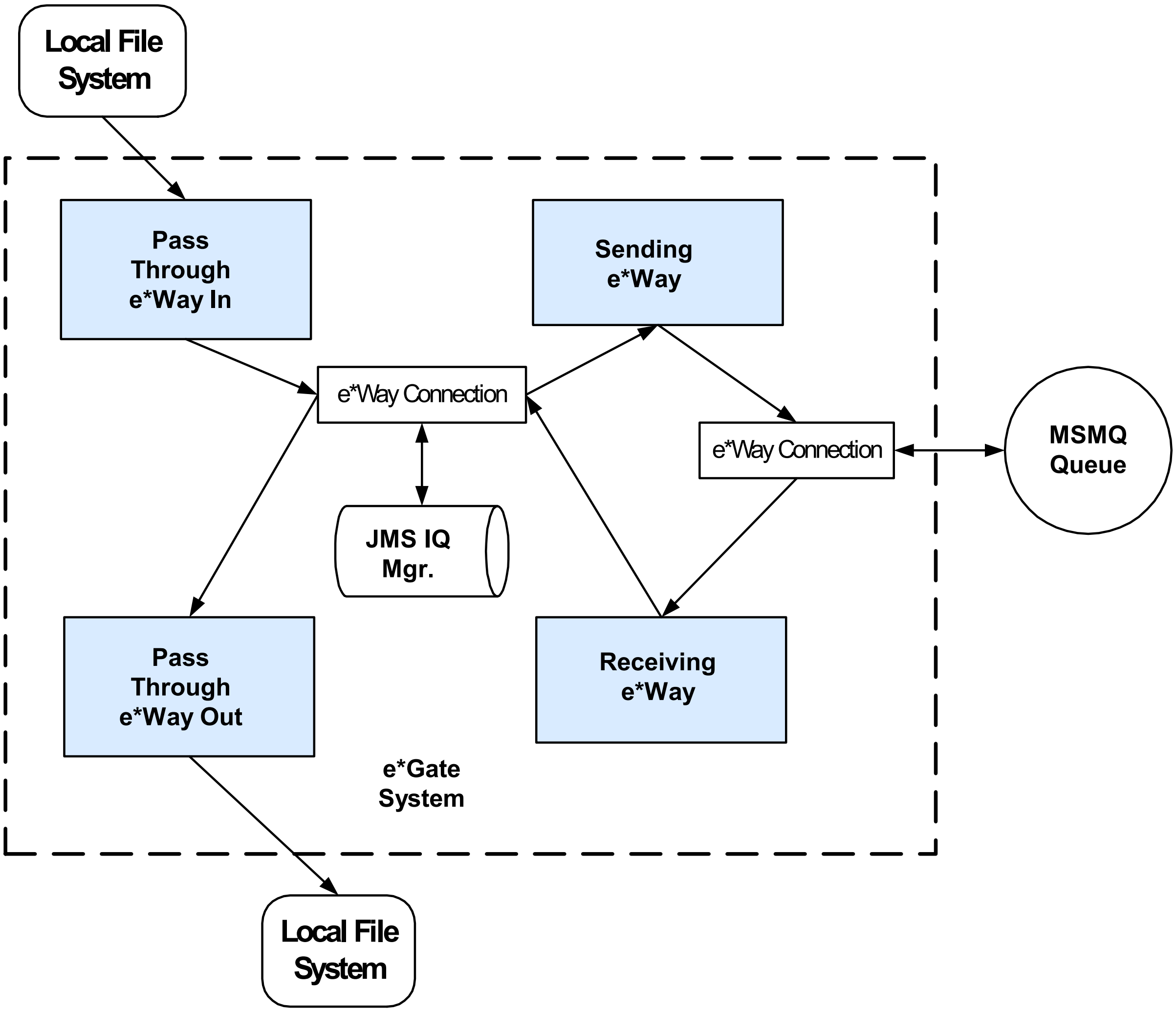
The file is then sent out via another e*Way Connection, as a message to an external MSMQ queue. The same e*Way Connection receives a response message.

Finally, the response (via the initial e*Way Connection and IQ Manager again) is sent out as a file through another Pass Through Multi-Mode e*Way, to a local external system. You can read the file from that location.

Schema Setup

Figure 1 on page 21 shows a diagram of the schema’s general architecture. The arrows show the direction of data flow.

Figure 1 msmqSample Schema Diagram



Schema Components

The msmqSample schema consists of the following main e*Gate components:

- **input**: Pass Through Multi-Mode e*Way that brings the local file into e*Gate.
- **output**: Pass Through Multi-Mode e*Way that outputs the file to the local system.
- **msmq_rcv**: MSMQ Multi-Mode e*Way that sends the file as a message to an MSMQ queue.
- **msmq_send**: MSMQ Multi-Mode e*Way that receives the file as a response message from the MSMQ queue.
- **collab_input**: Collaboration for the **input** e*Way.
    - **input**: Collaboration Rule for **collab_input**.

- **collab_output**: Collaboration for the **output** e*Way.
  - **cr_pass**: Collaboration Rule for **collab_output**.
- **cp_msmqSend**: Collaboration for the **msmq_send** e*Way.
  - **sendtomsmq**: Collaboration Rule for **cp_msmqSend**.
- **cp_msmqRcv**: Collaboration for the **msmq_rcv** e*Way.
  - **cr_pass**: Collaboration Rule for **cp_msmqRcv**.
- **localhost_iqmgr**: SeeBeyond JMS IQ Manager.
- **cp_jmsqueue**: e*Way Connection for the JMS IQ Manager.
- **cp_msmq**: e*Way Connection for the MSMQ e*Ways.

5.2.2 Creating the Sample Schema

This section explains the basic steps for how to create the schema msmqSample.

Note: *For complete information on how to set up an e*Gate schema, see the* ***e*Gate Integrator User's Guide*** *and* ***Creating and End-to-end Scenario with e*Gate Integrator****.*

Creating a New Schema

The first task in deploying the schema example is to create a new schema name. While it is possible to use the default schema for this example implementation, it is recommended that you create a separate schema for testing purposes.

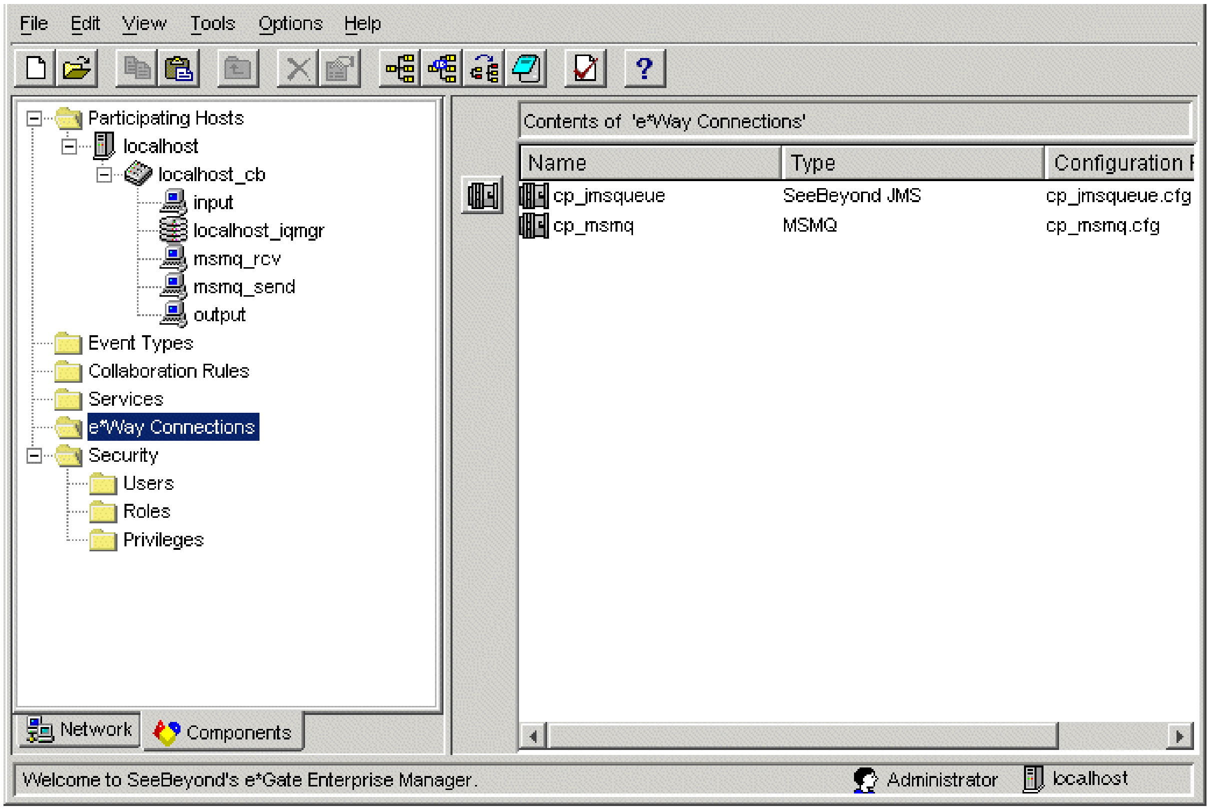
To create a new schema

1. Start the e*Gate Schema Designer.
2. When the Schema Designer prompts you to log in, select the host that you specified during installation, and enter your password.
3. You are then be prompted to select a schema. Click on **New**.
4. Enter a name for the new schema. You can enter **msmqSample** or any name desired.

The Schema Designer opens under your new schema, with many of the schema's basic components already created. From the Schema Designer, you can access the ETD Editor and Collaboration Rules Editor features. You are now ready to begin creating the necessary components for this sample schema.

Figure 2 on page 23 shows an example of the Schema Designer window with this schema already created.

Figure 2 Schema Designer Main Window for msmqSample



Contents of 'e*Way Connections'

Name	Type	Configuration F
cp_jmsqueue	SeeBeyond JMS	cp_jmsqueue.cfg
cp_msmq	MSMQ	cp_msmq.cfg

Welcome to SeeBeyond's e*Gate Enterprise Manager.

Creating Event Types and ETDs

The e*Way installation includes one **.xsc** file for this implementation. You must create is as a user-defined or Custom ETD.

Creating Event Types

Using the Schema Designer, you create the following Event Types:

- **et_output**
- **qa-msmq.msmq.com\testmsmq1** (for the user-defined ETD)

User-defined ETD

In this sample schema, you use the following user-defined ETD:

- **msmqevent.xsc**

To create the Event Types and ETDs

1. Highlight the **Event Type** folder on the **Components** tab of the e*Gate Schema Designer.
2. On the palette, click the icon to create a new Event Type.
3. Enter the name of the Event Type (**et_output**), then click **OK**.

4 Select the new **Event Type**, then right-click to edit its properties.

5 The **Event Type Properties** dialog box appears (see Figure 3).

Figure 3 Event Type Properties Dialog Box: et_output

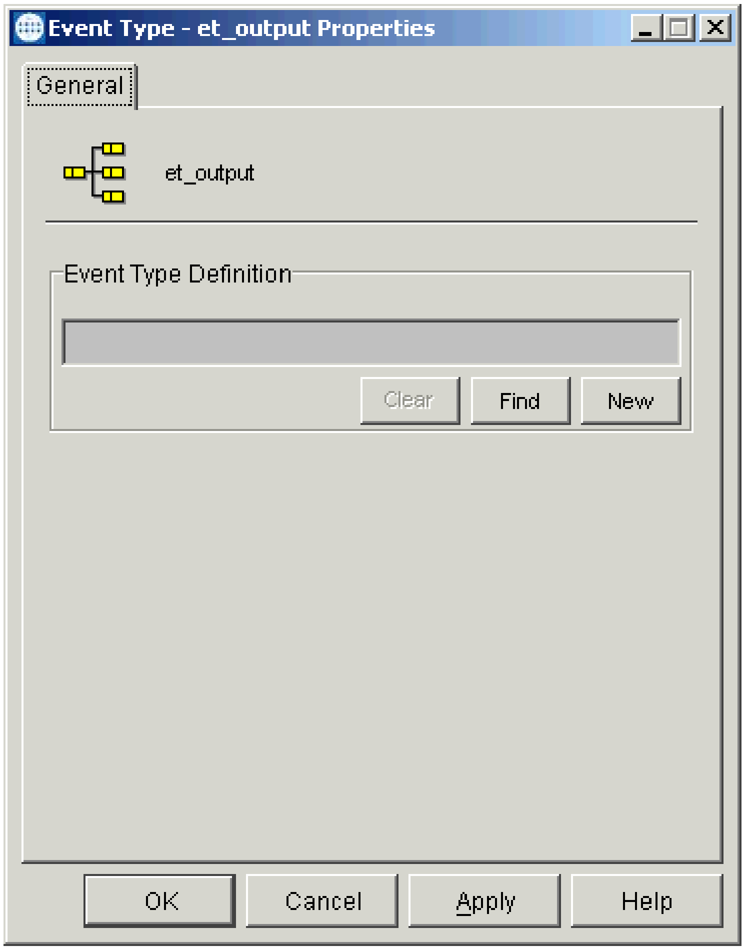


Figure 3 shows the ETD already selected. You do not need to select an ETD for this Event Type.

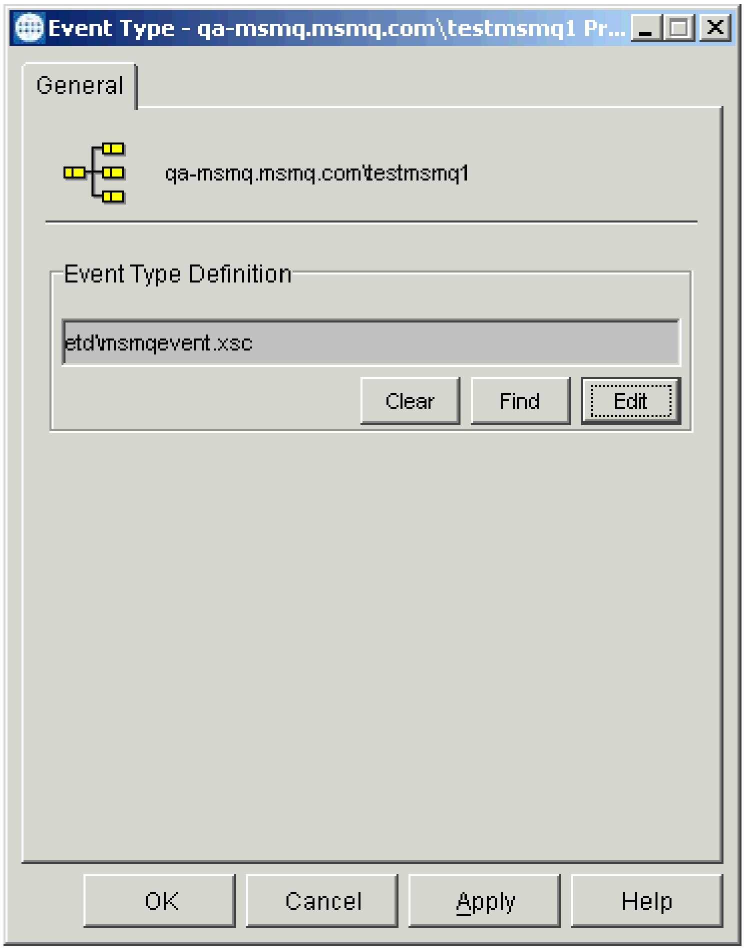
6 Click **OK** to save the Event Type and return to the Schema Designer.

7 Create another new Event Type named **qa-msmq.msmq.com\testmsmq1**.

8 Select the new **Event Type** again, then right-click to edit its properties.

9 The **Event Type Properties** dialog box appears (see Figure 4).

Figure 4 Event Type Properties Dialog Box: qa-msmq.msmq.com\testmsmq1



10 Click **OK** to save the Event Type.

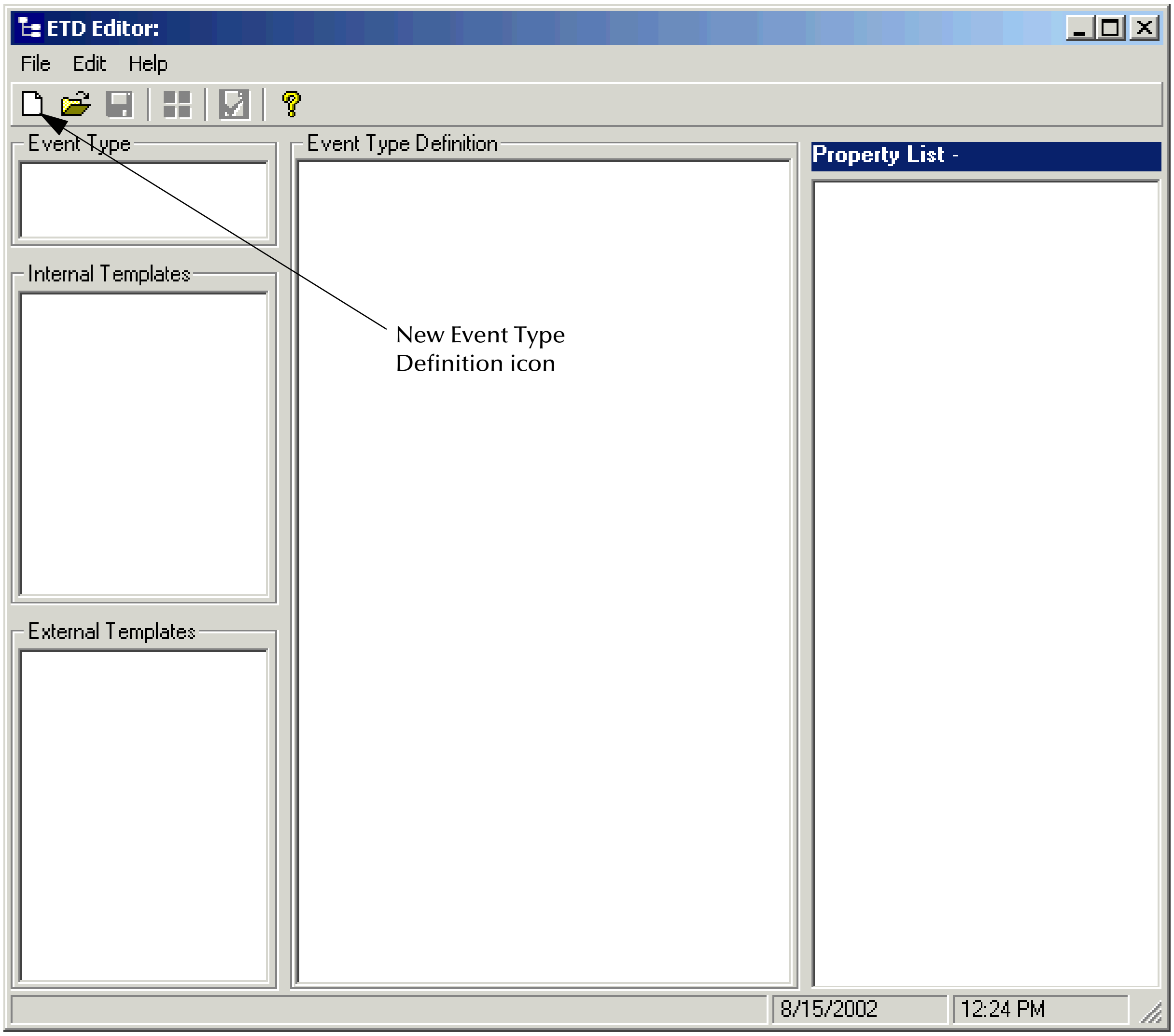
For this Event Type, you must create a user-defined, Custom ETD. To do so, you must use the ETD Editor feature of the Schema Designer.

11 From the Schema Designer menu bar, select **Options** and click **Default Editor**. For this schema, set the default to Java.

12 Click the icon in the toolbar for the ETD Editor.

The ETD Editor Main window appears (see **Figure 5 on page 26**)

Figure 5 ETD Editor Main Window

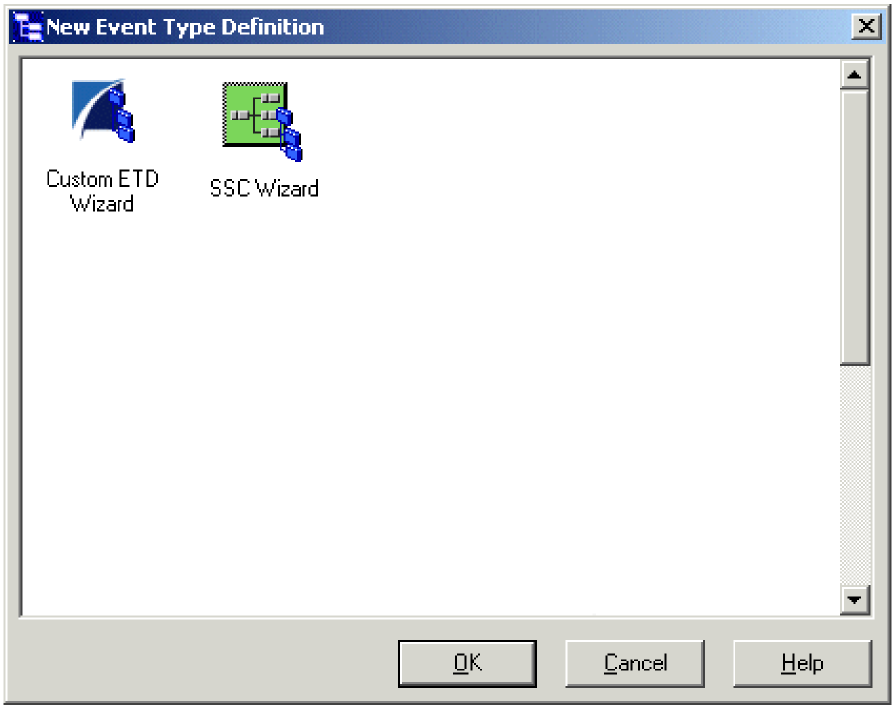


New Event Type Definition icon

13 Click the **New Event Type Definition** icon in the window's tool bar.

The **New Event Type Definition** dialog box appears (see **Figure 6 on page 27**).

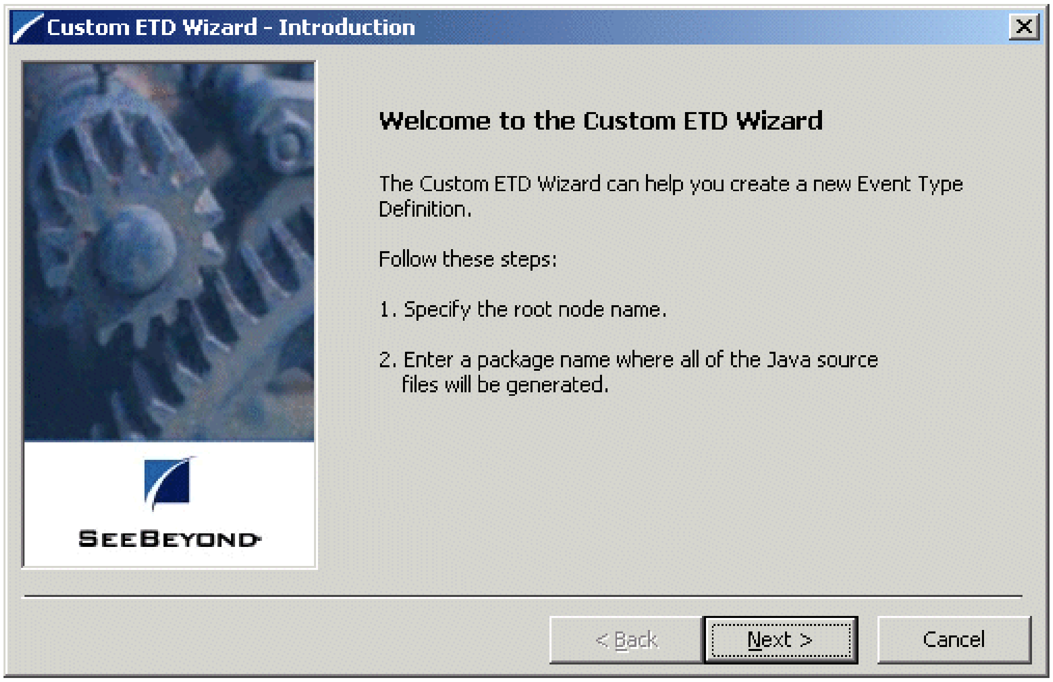
Figure 6 New Event Type Definition Dialog Box



14 Double-click the **Custom ETD Wizard** icon.

The **Introduction** dialog box for the Custom ETD wizard appears (see Figure 7).

Figure 7 Custom ETD Wizard: Introduction



Welcome to the Custom ETD Wizard

The Custom ETD Wizard can help you create a new Event Type Definition.

Follow these steps:

1. Specify the root node name.

2. Enter a package name where all of the Java source files will be generated.

15 Follow the instructions given by the wizard to create a new ETD and name it **msmqevent.xsc**. Be sure to create the ETD in the **client\etd** directory.

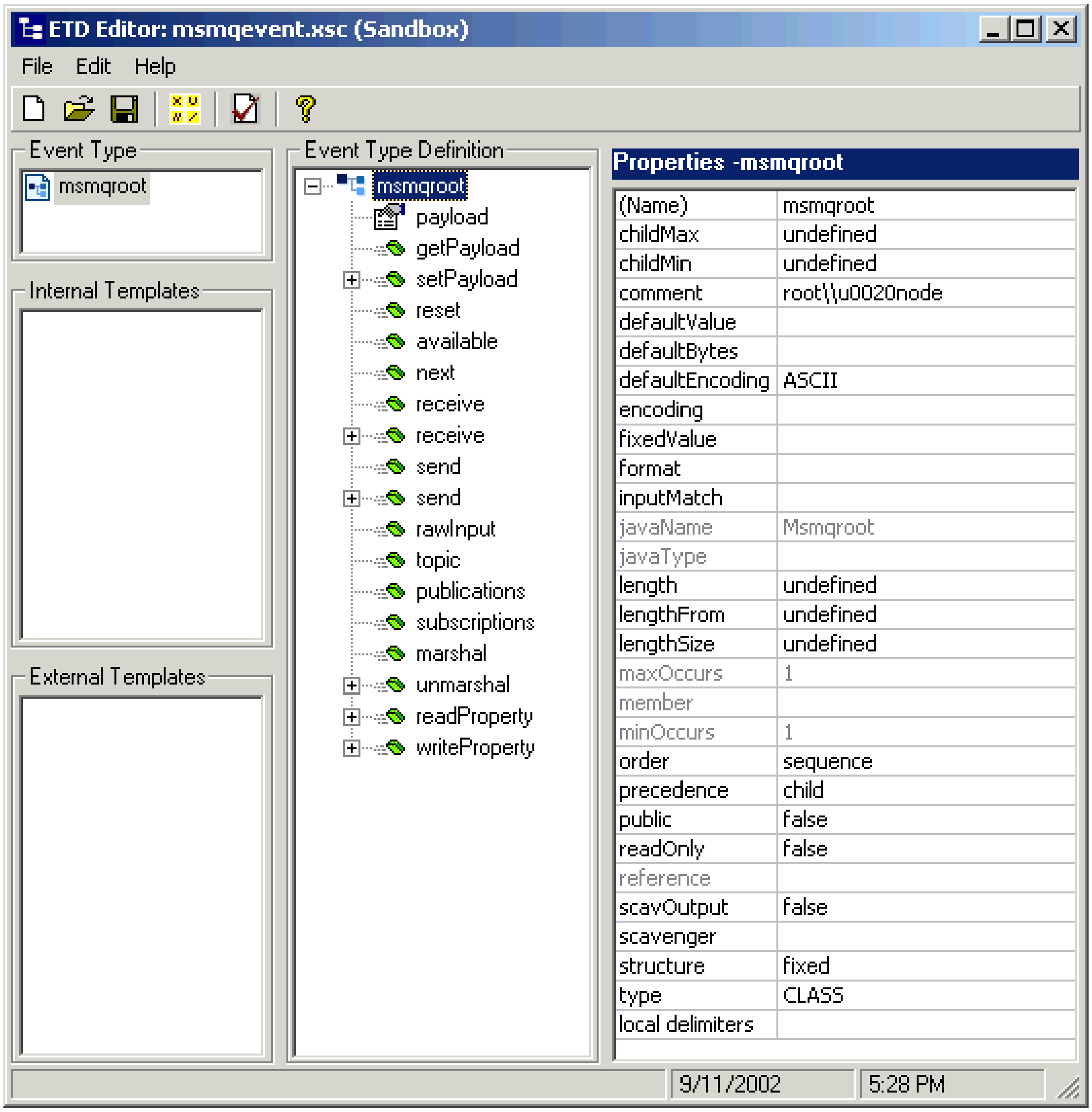
Note: *For complete instructions on how to use the Custom ETD wizard to create a new ETD, see the* ***e*Gate Integrator User's Guide****.*

16 From the ETD Editor, click the **Open** icon. Using the **Open** dialog box, double-click the **msmqevent.xsc** file.

The contents of the file appear in the ETD Editor window.

17 Name the root node **msmqroot** and assign it the properties shown in Figure 8.

Figure 8 msmqevent.xsc in ETD Editor

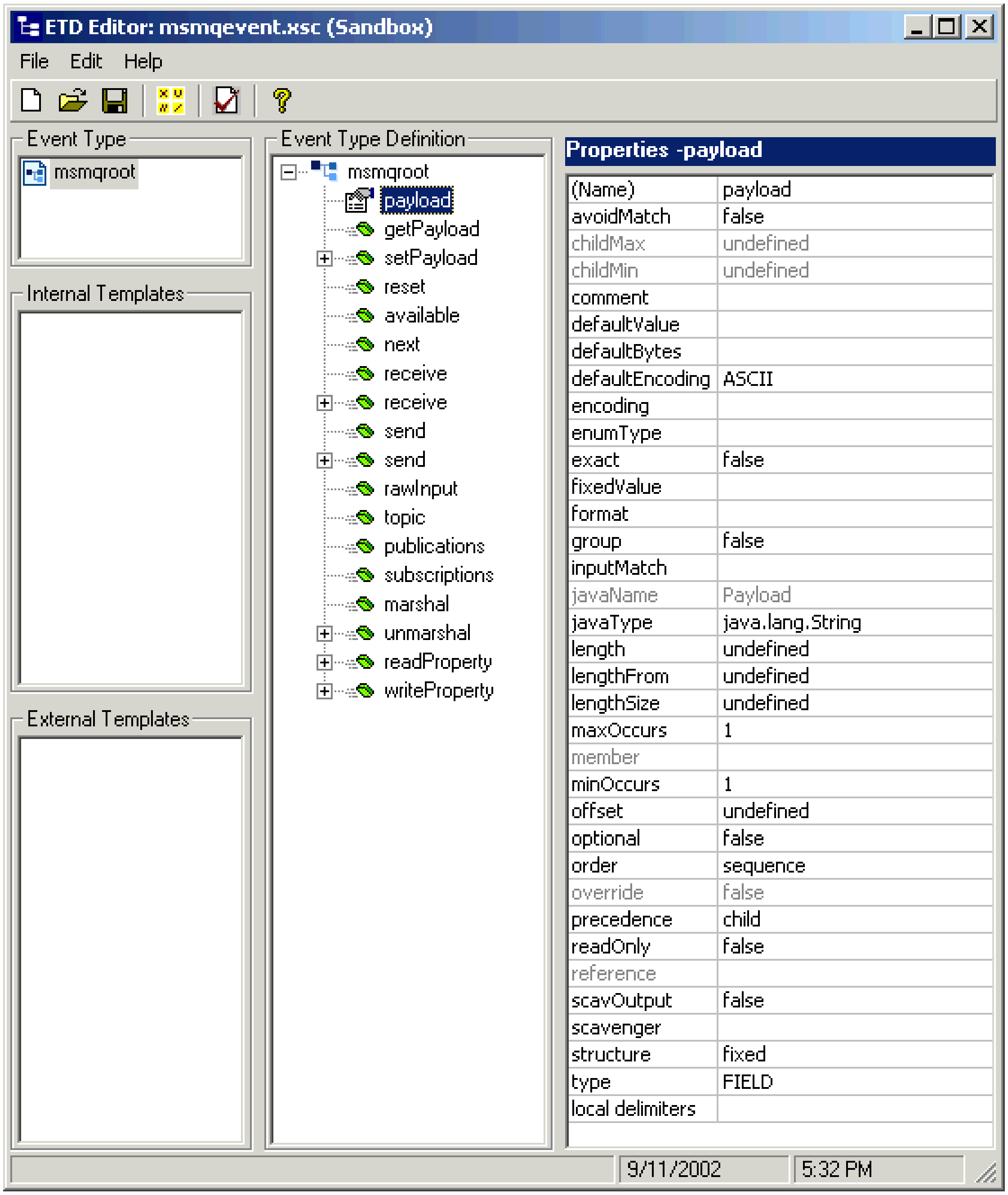


(Name)	msmqroot
childMax	undefined
childMin	undefined
comment	root\\u0020node
defaultValue	
defaultBytes	
defaultEncoding	ASCII
encoding	
fixedValue	
format	
inputMatch	
javaName	Msmqroot
javaType	
length	undefined
lengthFrom	undefined
lengthSize	undefined
maxOccurs	1
member	
minOccurs	1
order	sequence
precedence	child
public	false
readOnly	false
reference	
scavOutput	false
scavenger	
structure	fixed
type	CLASS
local delimiters	

*Note: For complete instructions on how to use the ETD Editor, see the **e*Gate Integrator User's Guide**.*

18 Add **payload** as a child node Be sure that you give this node the properties shown in Figure 9.

Figure 9 msmqevent.xsc in ETD Editor: payload Node



Properties -payload

(Name)	payload
avoidMatch	false
childMax	undefined
childMin	undefined
comment	
defaultValue	
defaultBytes	
defaultEncoding	ASCII
encoding	
enumType	
exact	false
fixedValue	
format	
group	false
inputMatch	
javaName	Payload
javaType	java.lang.String
length	undefined
lengthFrom	undefined
lengthSize	undefined
maxOccurs	1
member	
minOccurs	1
offset	undefined
optional	false
order	sequence
override	false
precedence	child
readOnly	false
reference	
scavOutput	false
scavenger	
structure	fixed
type	FIELD
local delimiters	

19 Finish building the ETD until it looks like the one shown in Figure 9.

20 When you are finished editing the ETD, save your changes and close the ETD Editor.

21 From the Schema Designer, open the **Event Type Properties** dialog box for the **qa-msmq.msmq.com\testmsmq1** Event Type and associate it with the **msmqevent.xsc** ETD you created.

22 When you are finished with the dialog box, click **OK** to close it and save your changes.

Creating and Configuring e*Ways

You must create the following Multi-Mode e*Ways:

- Pass Through In: **input**
- Pass Through Out: **output**
- MSMQ Sending: **msmq_rcv**
- MSMQ Receiving: **msmq_send**

Note: *For detailed information on the Multi-Mode e*Way, see the **Standard e*Way Intelligent Adapter User's Guide**.*

To create the Pass Through In and Out Multi-Mode e*Ways

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

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

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

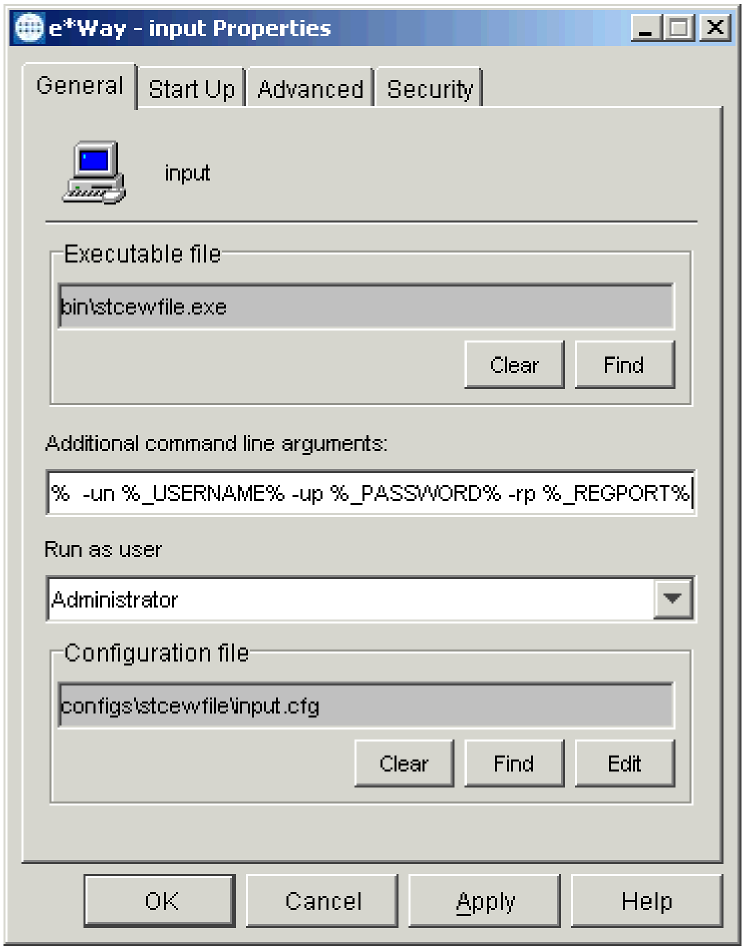
4 On the palette, click the icon to create a new e*Way.

5 Enter the name of the new e*Way (**input**), then click **OK**.

6 Select the new component, then double-click to edit its properties.

7 When the **e*Way Properties** dialog box appears, use the default executable file, **stceway.exe** (see **Figure 10 on page 31**).

Figure 10 input e*Way Properties Dialog Box



input

Executable file

bin\stcewfile.exe

Additional command line arguments:

% -un %_USERNAME% -up %_PASSWORD% -rp %_REGPORT%

Run as user

Administrator

Configuration file

configs\stcewfile\input.cfg

Configure the **input** e*Way properties as shown in the previous figure. Use the default configuration parameters.

8 To edit the JVM Settings, select **New** under the **Configuration File** text box. Use the default configuration parameters, as shown in the e*Way Configuration Editor.

Note: *See* **Chapter 3** *for more information on how to configure the Multi-Mode e*Way.*

9 Save the **.cfg** file, and exit the e*Way Configuration Editor, returning to the **e*Way Properties** dialog box..

10 Use the **Startup**, **Advanced**, and **Security** tabs to modify the default settings for each.

A Use the **Startup** tab to specify whether the Multi-Mode e*Way starts automatically, restarts after abnormal termination or due to, for example, scheduling.

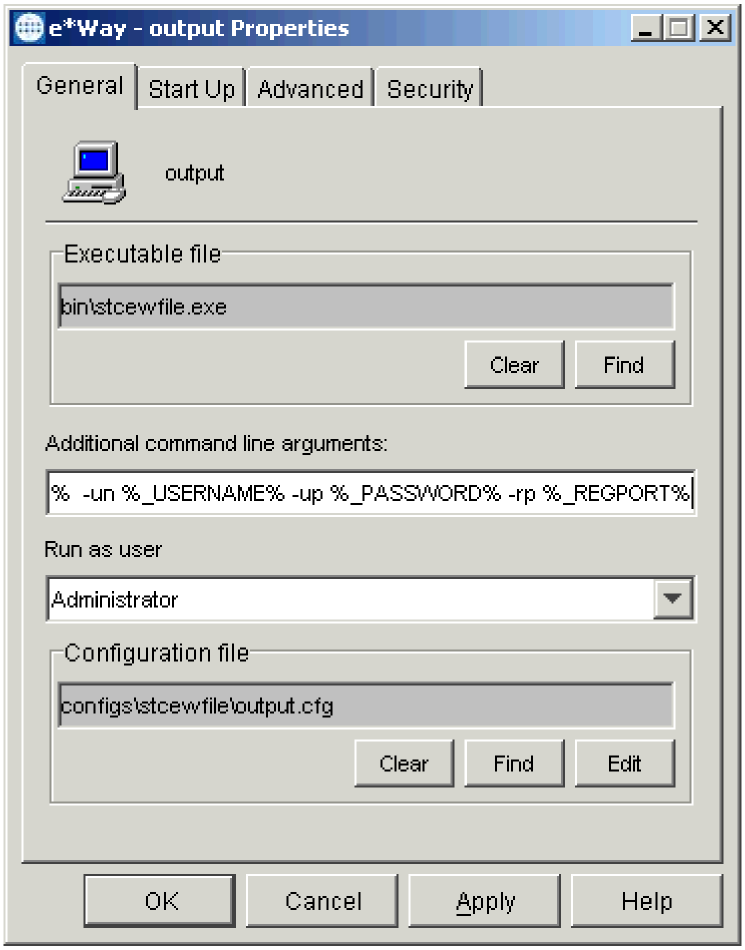
B Use the **Advanced** tab to specify or view the activity and error logging levels, as well as the Event threshold information.

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

11 Select **OK** to close the **e*Way Properties** dialog box and save your settings.

12 Repeat steps 4 through 11 for the **output, msmq_rcv,** and **msmq_send** e*Ways (see Figure 11 through **Figure 13 on page 34**).

Figure 11 output e*Way Properties Dialog Box



e*Way - output Properties

output

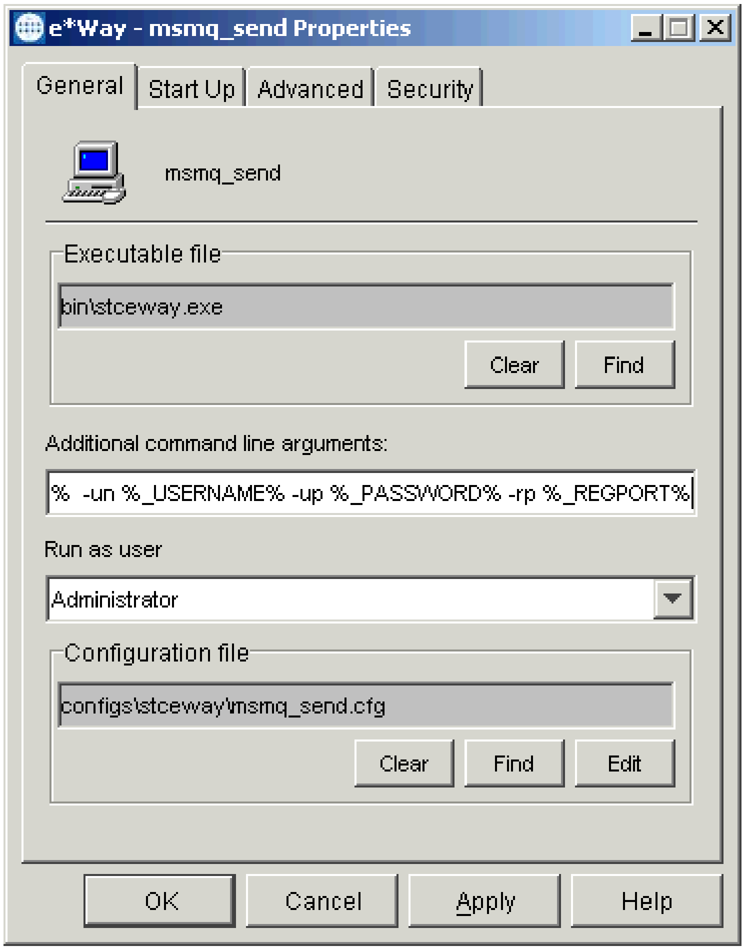
Executable file: bin\stcewfile.exe

Additional command line arguments: % -un %_USERNAME% -up %_PASSWORD% -rp %_REGPORT%

Run as user: Administrator

Configuration file: configs\stcewfile\output.cfg

Figure 12 msmq_rcv e*Way Properties Dialog Box



msmq_send

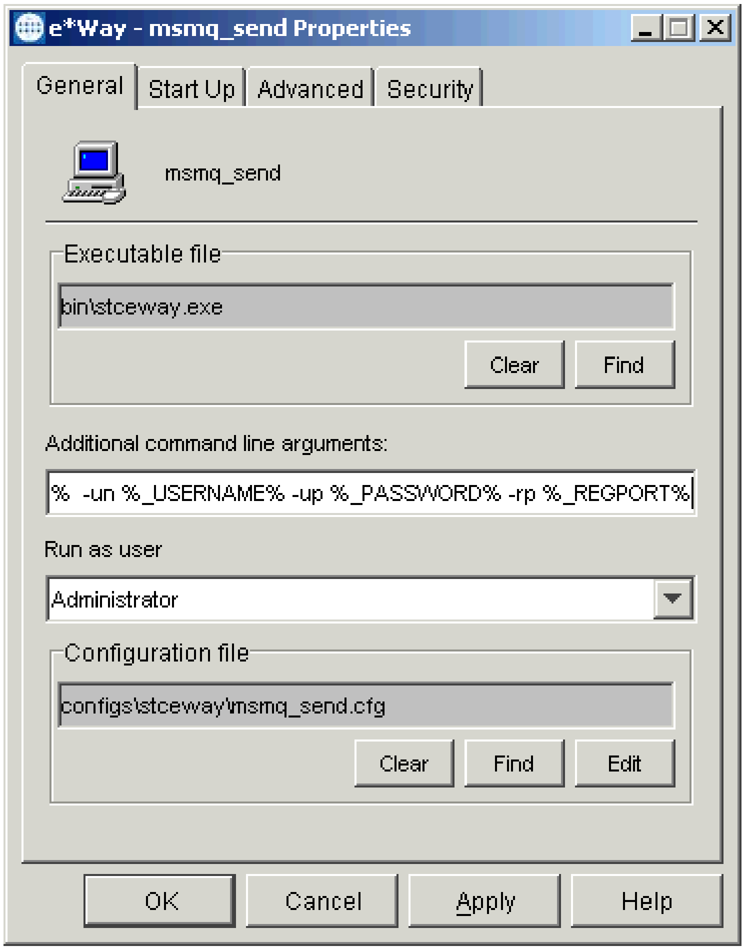
Executable file: bin\stceway.exe

Additional command line arguments: % -un %_USERNAME% -up %_PASSWORD% -rp %_REGPORT%

Run as user: Administrator

Configuration file: configs\stceway\msmq_send.cfg

Figure 13 msmq_send e*Way Properties Dialog Box



msmq_send

Executable file

bin\stceway.exe

Additional command line arguments:

% -un %_USERNAME% -up %_PASSWORD% -rp %_REGPORT%

Run as user

Administrator

Configuration file

configs\stceway\msmq_send.cfg

Configure the properties for these three e*Ways as shown in the previous figures. Use the default configuration parameters.

Creating and Configuring e*Way Connections

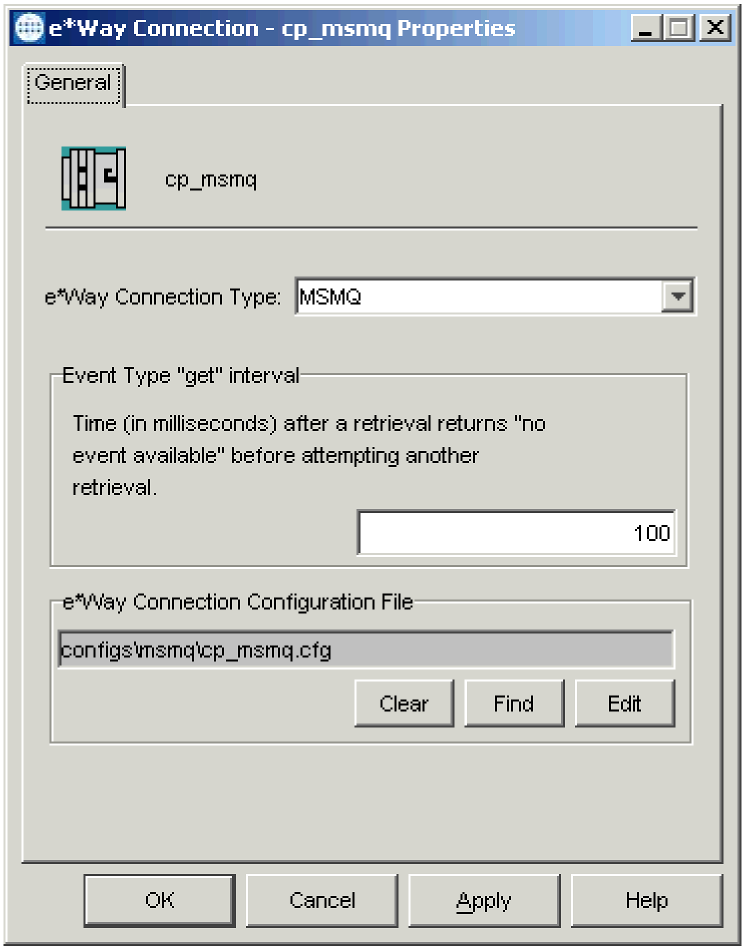
The e*Way Connection configuration file contains the connection information needed to communicate with the local file system, the JMS, and the MSMQ queue.

To create and configure the cp_msmq e*Way Connection

1. Highlight the **e*Way Connection** folder on the **Components** tab of the e*Gate Schema Designer.
2. On the palette, click the icon to create a new e*Way Connection.
3. Enter the name of the e*Way Connection (**cp_msmq**), then click **OK**.
4. Select the new **e*Way Connection**, then right-click to edit its properties.

5 When the **e*Way Connection Properties** dialog box opens, select **MSMQ** from the **e*Way Connection Type** drop-down menu (see Figure 14).

Figure 14 cp_msmq e*Way Connection Properties Dialog Box



e*Way Connection - cp_msmq Properties

General

cp_msmq

e*Way Connection Type: MSMQ

Event Type "get" interval

Time (in milliseconds) after a retrieval returns "no event available" before attempting another retrieval.

100

e*Way Connection Configuration File

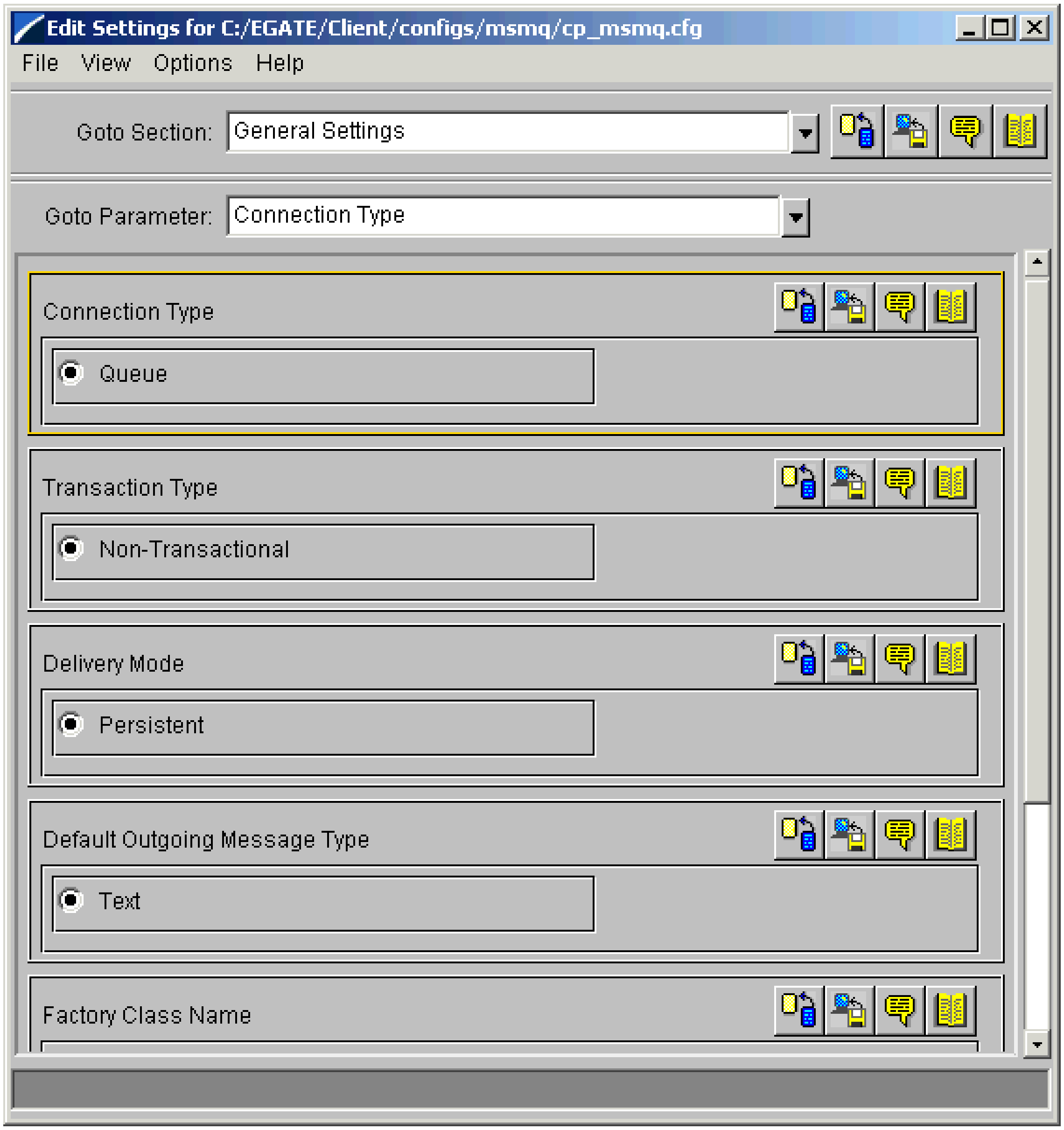
configs\msmq\cp_msmq.cfg

Configure the e*Way Connection properties as shown in the previous figure.

6 Under **e*Way Connection Configuration File**, click **New** (where the **Edit** button is in the previous figure).

The e*Way Configuration Editor Main window opens (see **Figure 15 on page 36**).

Figure 15 e*Way Configuration Editor: cp_msmq General Settings



Edit Settings for C:/EGATE/Client/configs/msmq/cp_msmq.cfg

Goto Section: General Settings

Goto Parameter: Connection Type

Connection Type
● Queue

Transaction Type
● Non-Transactional

Delivery Mode
● Persistent

Default Outgoing Message Type
● Text

Factory Class Name

7 Select the desired parameters. In most cases, you can use the defaults. See Chapter 4 for details.

Note: *See the* ***e*Gate Integrator User's Guide*** *for complete information on how to use the e*Way Configuration Editor.*

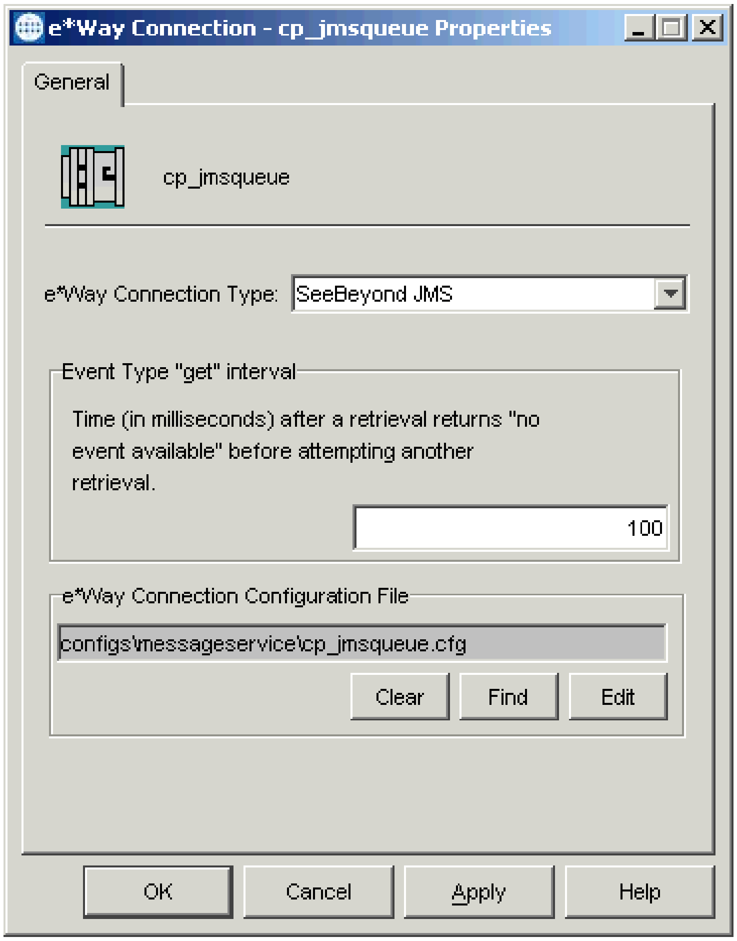
8 When you are finished, save the **.cfg** file, close the e*Way Configuration Editor, and promote the file to run time.

9 Click **OK** to close the **e*Way Connection Properties** dialog box.

To create and configure the cp_jmsqueue e*Way Connection

1 Repeat steps 2 through 6 under the **procedure on page 34** for the **cp_jmsqueue** e*Way Connection (see Figure 16).

Figure 16 cp_jmsqueue e*Way Connection Properties Dialog Box



e*Way Connection - cp_jmsqueue Properties

cp_jmsqueue

e*Way Connection Type: SeeBeyond JMS

Event Type "get" interval

Time (in milliseconds) after a retrieval returns "no event available" before attempting another retrieval. 100

e*Way Connection Configuration File

configs\messageservice\cp_jmsqueue.cfg

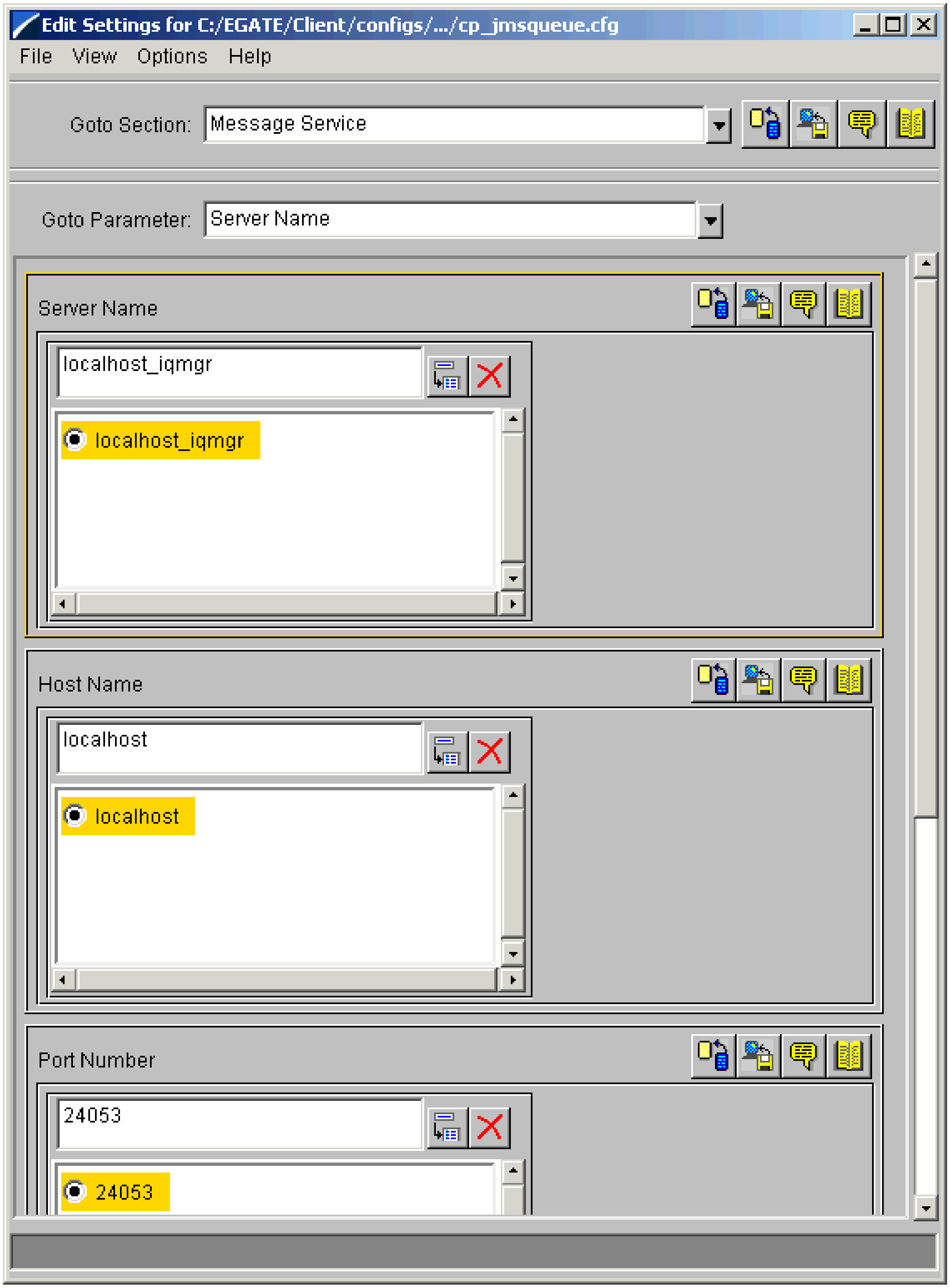
Configure the e*Way Connection properties as shown in the previous figure. Be sure to select **SeeBeyond JMS** as the **e*Way Connection Type**.

2 Using the e*Way Configuration Editor accept the default settings for all parameters except for:

- **Message Service:**
  - **Server Name**
  - **Host Name**
  - **Port Number**

You must enter your system settings for these parameters (see **Figure 17 on page 38**).

Figure 17 e*Way Configuration Editor: Necessary cp_jmsqueue Settings



Edit Settings for C:/EGATE/Client/configs/.../cp_jmsqueue.cfg

Goto Section: Message Service

Goto Parameter: Server Name

Server Name

localhost_iqmgr

● localhost_iqmgr

Host Name

localhost

● localhost

Port Number

24053

● 24053

3 When you are finished, save the **.cfg** file, close the e*Way Configuration Editor, and promote the file to run time.

4 Click **OK** to close the **e*Way Connection Properties** dialog box.

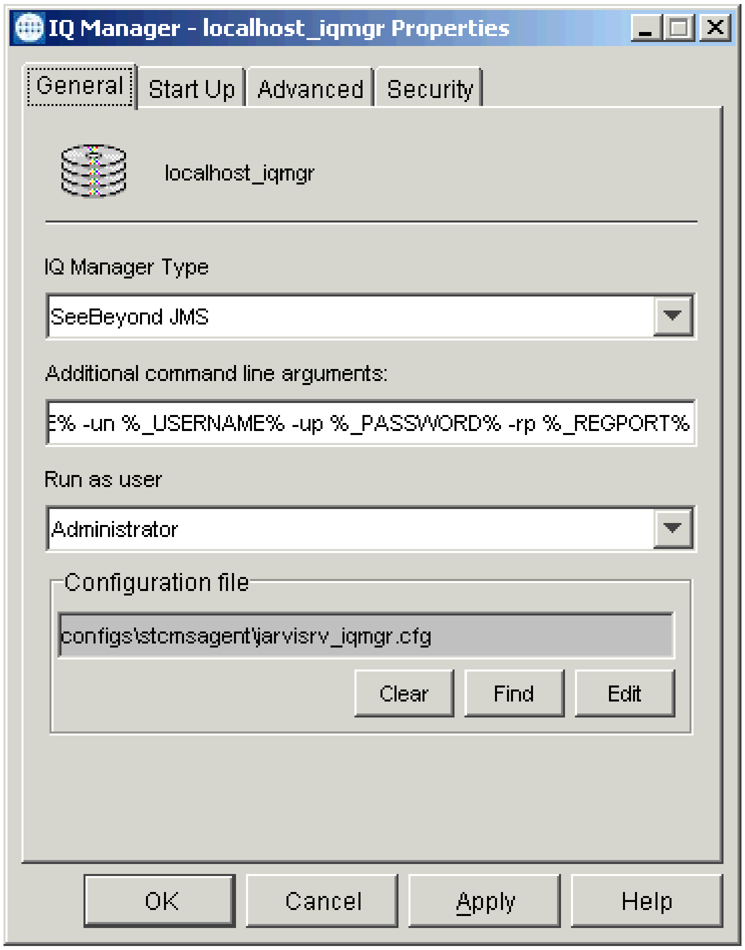
Checking the IQ Manager

By default, the IQ Manager in your schema is configured to use the SeeBeyond Java Messaging Service (JMS) IQ Service. The system has already named the IQ Manager for you, so you can keep that name if desired. In the sample, the component is named **localhost_iqmgr**.

IQs use IQ Services to transport data. IQ Services provide the mechanism for moving Events between IQs, handling the low-level implementation of data exchange (such as system calls to initialize or reorganize a database).

For an illustration of the IQ Manager Properties dialog box for this schema, see Figure 18.

Figure 18 IQ Manager Properties Dialog Box



IQ Manager - localhost_iqmgr Properties

localhost_iqmgr

IQ Manager Type: SeeBeyond JMS

Additional command line arguments: E% -un %_USERNAME% -up %_PASSWORD% -rp %_REGPORT%

Run as user: Administrator

Configuration file: configs\stcmsagent\jarvisrv_iqmgr.cfg

Note: *See the **SeeBeyond JMS Intelligent Queue User's Guide** for more information on the JMS IQ Manager feature.*

Creating Collaboration Rules

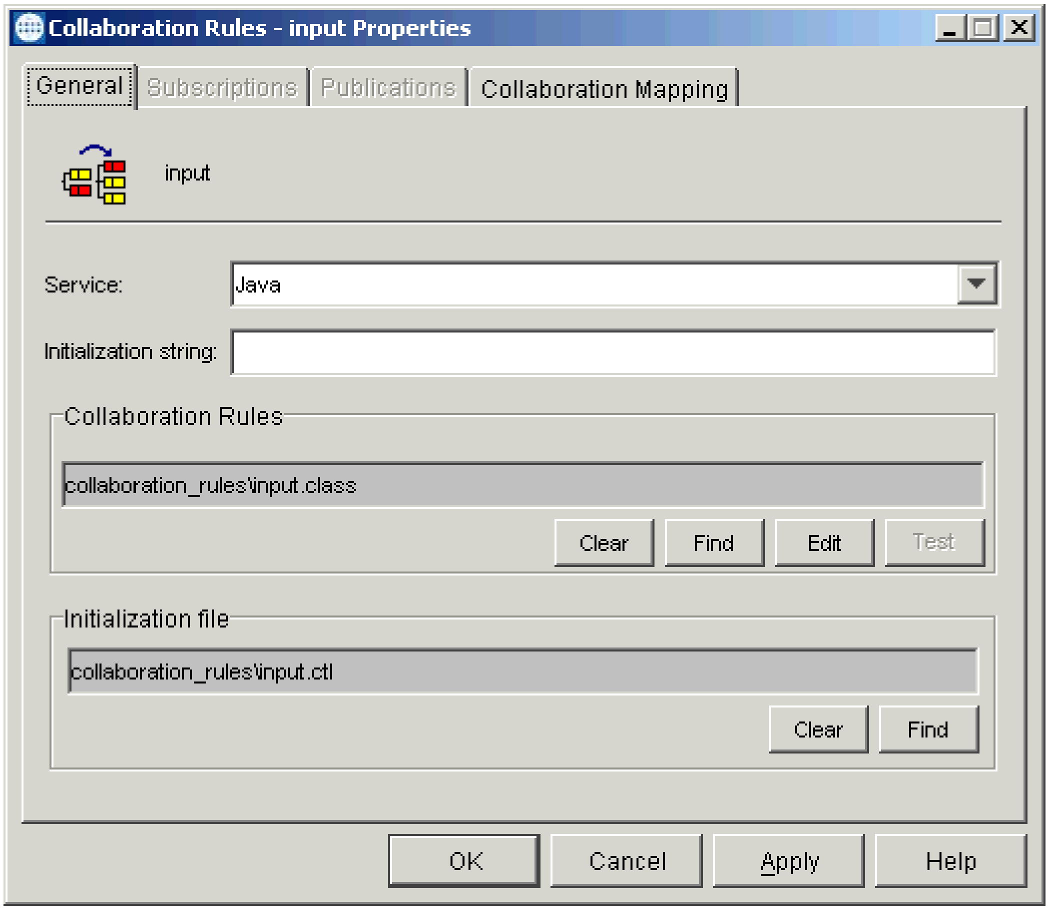
The next step is to create the Collaboration Rules that extract and process selected information from the source Event Type defined previously, according to its associated Collaboration Service.

From the Schema Designer menu bar, select **Options** and click **Default Editor**. For this schema, the **Default Editor** must be set to the Java programming language.

To create the input Collaboration Rules file

1. Select the **Components** tab in the e*Gate Schema Designer.
2. In the Navigation pane, select the **Collaboration Rules** folder.
3. On the palette, click the **Collaboration Rules** icon.
4. Enter the name of the new Collaboration Rule, then click **OK.** Use **input** for this example, for the **input** e*Way's Collaboration, **collab_input**.
5. Select the new **Collaboration Rule**, then right-click to edit its properties.
6. The **Collaboration Rules Properties** dialog box appears (see Figure 19).

Figure 19 Collaboration Rules Properties Dialog Box for input: General Tab



Collaboration Rules - input Properties

input

Service: Java

Initialization string:

Collaboration Rules: collaboration_rules\input.class

Initialization file: collaboration_rules\input.ctl

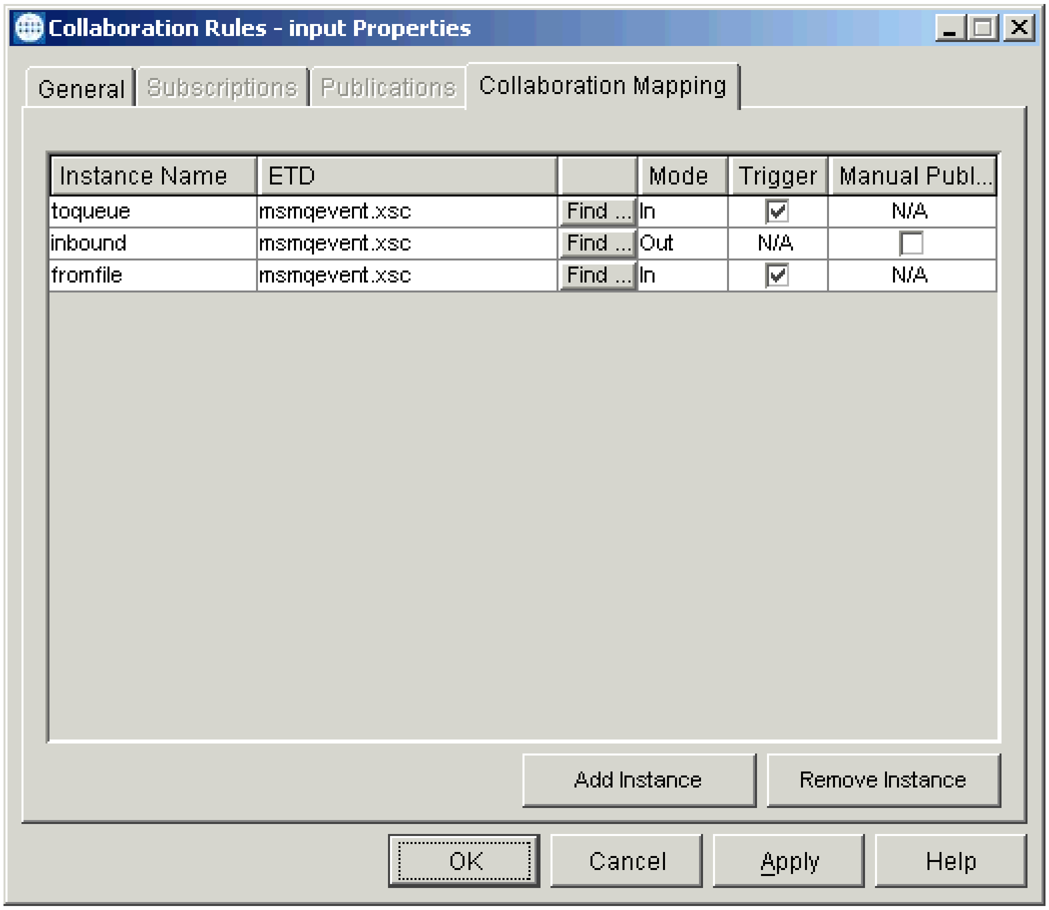
7 On the **General** tab in the dialog box select the **Java** Collaboration Service. In this example, the Collaboration Rules use the e*Gate Java Collaboration Service to manipulate Events or Event data.

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

9 Click the **Collaboration Mapping** tab.

10 Using the **Add Instance** button, create instances to coincide with the Event Types (see Figure 20).

Figure 20 Collaboration Rules Properties Dialog Box for input: Mapping Tab



Instance Name	ETD		Mode	Trigger	Manual Publ...
toqueue	msmqevent.xsc	Find ...	In	☑	N/A
inbound	msmqevent.xsc	Find ...	Out	N/A	□
fromfile	msmqevent.xsc	Find ...	In	☑	N/A

Configure the rest of **input** as shown in the previous figure.

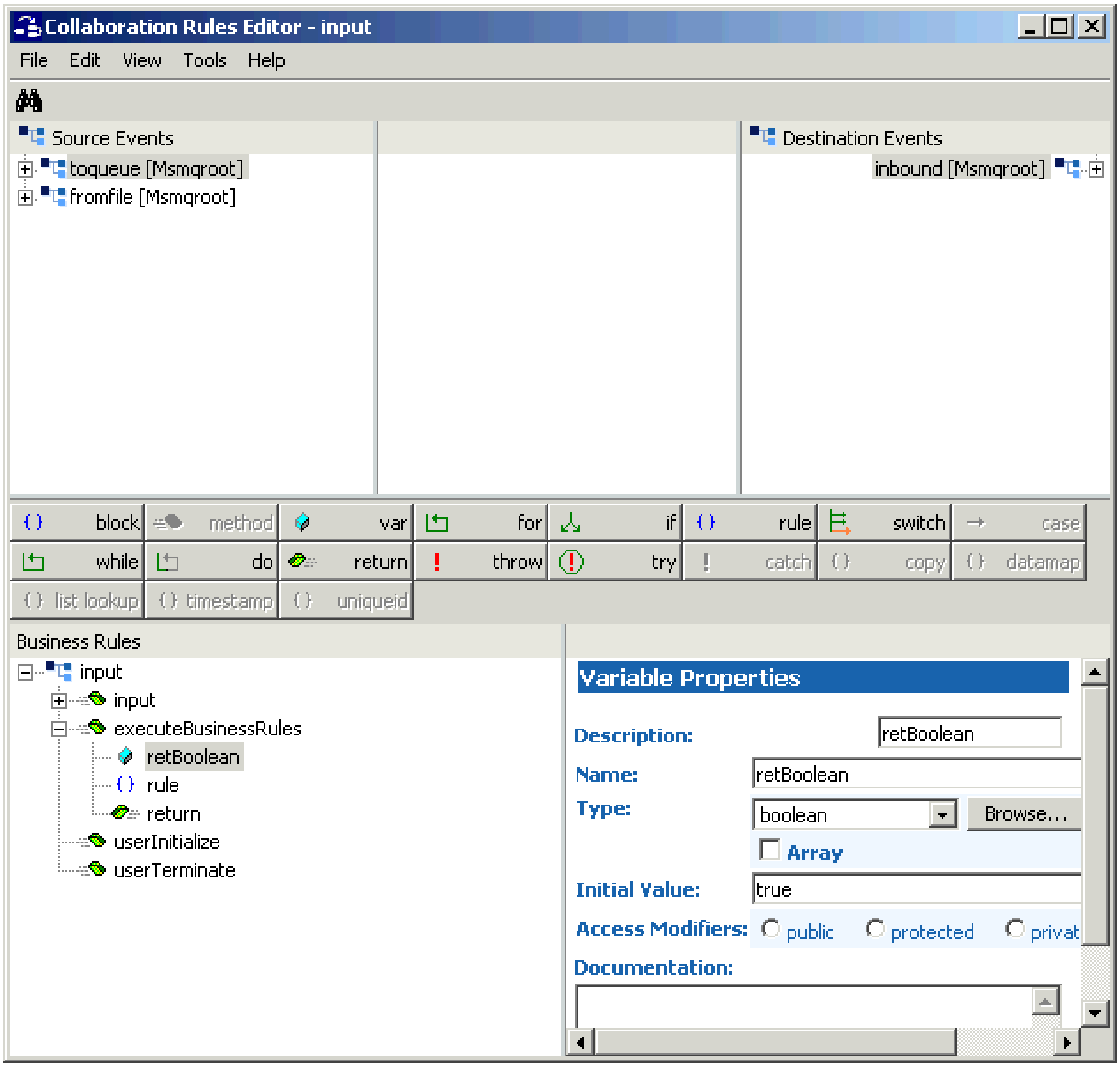
Note: *You can use the **msmqevent.xsc** ETD from the sample for your user-defined ETD or create one of your own that fits the example.*

11 Select the **General** tab again, then click **New**.

The Collaboration Rules Editor Main window opens.

12 Expand the window to full size for optimum viewing (see Figure 21).

Figure 21 Collaboration Rules Editor: input Start



Description: retBoolean

Name: retBoolean

Type: boolean

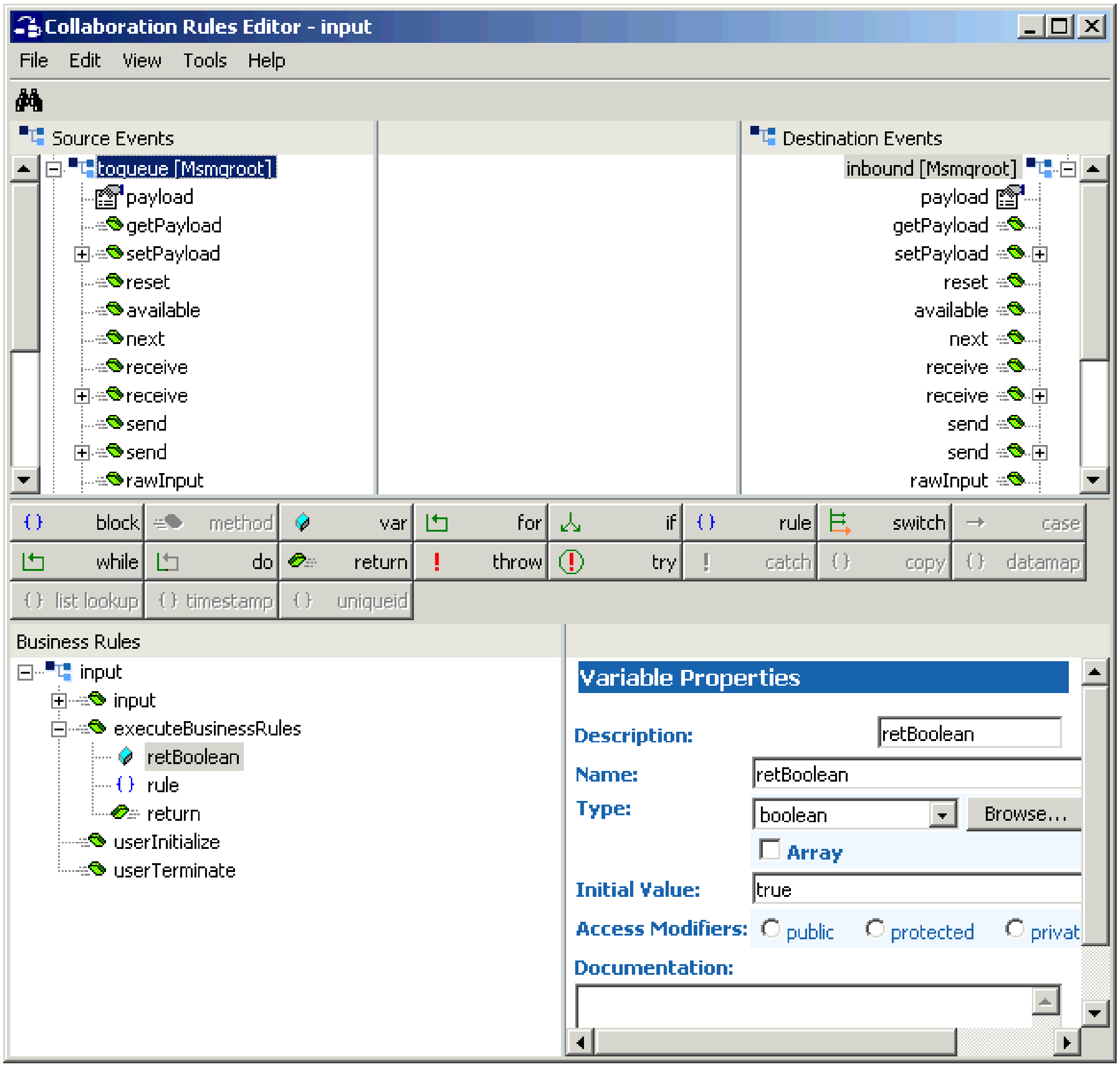
□ Array

Initial Value: true

Access Modifiers: ○ public ○ protected ○ privat

13 Expand the source and destination Events, as well as the Business Rules. Figure 22 shows the results. Note Business Rules associated with each ETD.

Figure 22 Collaboration Rules Editor: input Expanded



Description: retBoolean

Name: retBoolean

Type: boolean

□ Array

Initial Value: true

Access Modifiers: ○ public ○ protected ○ privat

Documentation:

14 Use the Collaboration Rules in the sample schema, and create the Business Rules for your **input.** Set them up in the same way as they are for **input** in the sample schema.

Note: *For complete information on how to use the Collaboration Rules Editor, see the* ***e*Gate Integrator User's Guide****.*

15 You must create a Collaboration Rules class or use one from the sample.

16 To save the Collaboration Rules file, click **Save** on the **File** menu. The **Save** dialog box appears.

17 Provide a name for the **.xpr** file (for this example, use **input.xpr**) then click **Save**.

18 When you have finished defining all the desired business logic, compile the Java code by selecting **Compile** from the **File** menu.

If the code compiles successfully, the message **Compile Completed** appears. If the outcome is unsuccessful, a Java Compiler error message appears. If there are any Java errors, be sure to correct them.

19 Once the compilation is complete, you can exit the Collaboration Rules Editor.

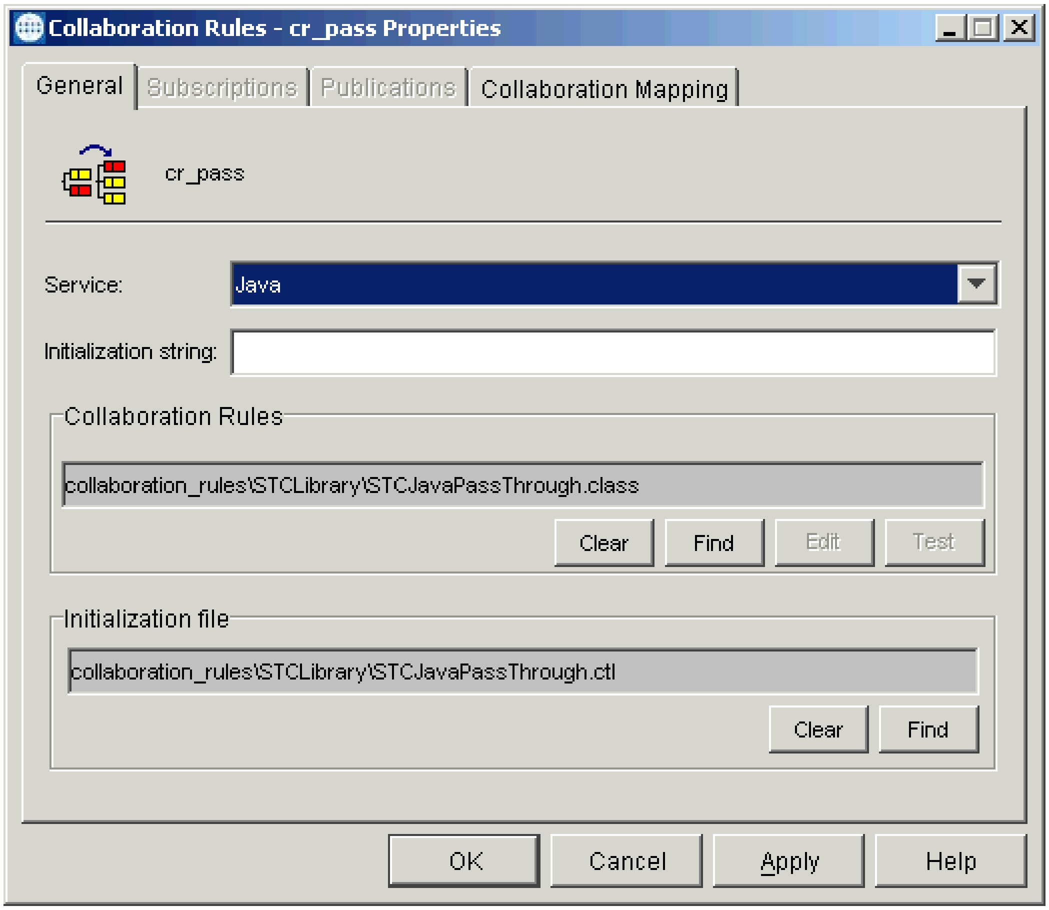
To create the cr_pass Collaboration Rules file

1 Repeat steps 3 through 5 under the **procedure on page 40** to create the next Collaboration Rule.

Use **cr_pass** as the name for this example, for the **output** e*Way's Collaboration, **collab_output**.

2 Double-click on the **cr_pass** icon. The **Collaboration Rules Properties** dialog box appears (see Figure 23).

Figure 23 Collaboration Rules Properties Dialog Box for cr_pass: General Tab



Collaboration Rules - cr_pass Properties

cr_pass

Service: Java

Initialization string:

Collaboration Rules

collaboration_rules\STCLibrary\STCJavaPassThrough.class

Initialization file

collaboration_rules\STCLibrary\STCJavaPassThrough.ctl

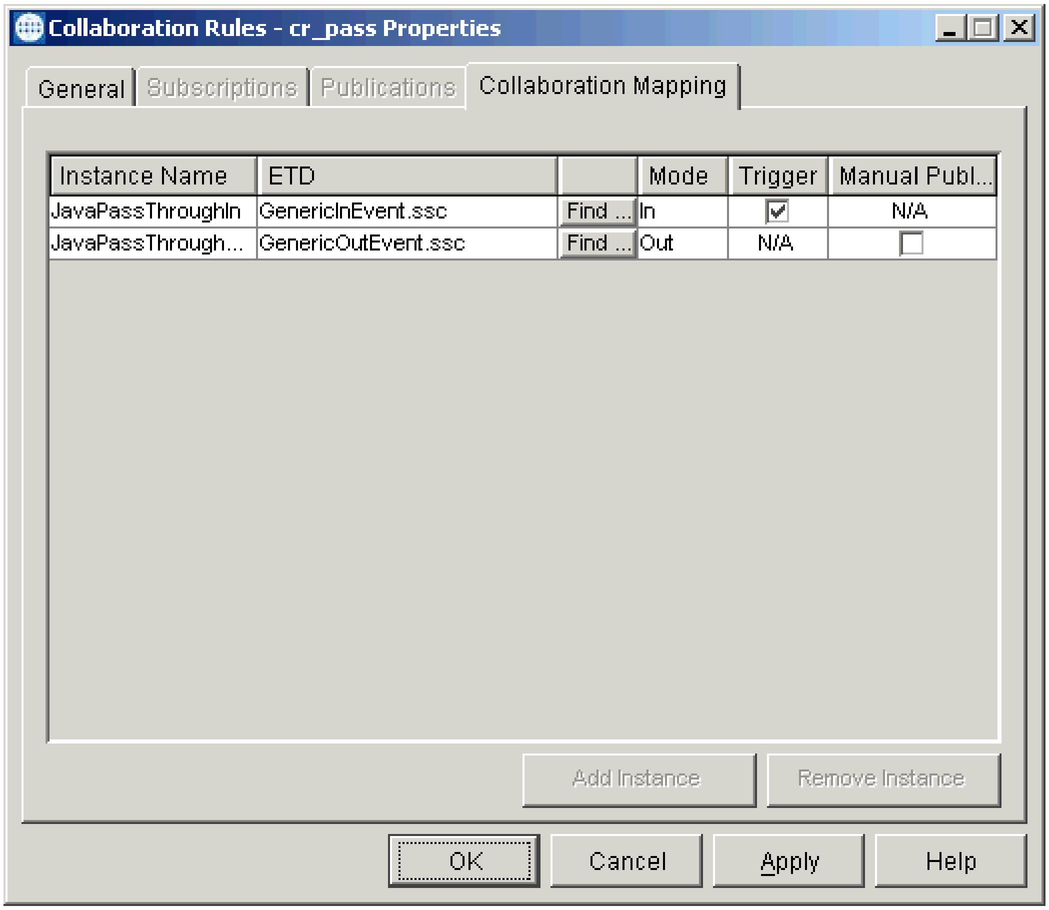
On the **General** tab in the dialog box select the **Java** Collaboration Service.

3 In the **Initialization String** text box (optional), enter any required initialization string that the Collaboration Service may require.

4 Click the **Collaboration Mapping** tab.

5 Using the **Add Instance** button, create instances to coincide with the Event Types (see Figure 24).

Figure 24 Collaboration Rules Properties Dialog Box for cr_pass: Mapping Tab



Instance Name	ETD		Mode	Trigger	Manual Publ...
JavaPassThroughIn	GenericInEvent.ssc	Find ...	In	☑	N/A
JavaPassThrough...	GenericOutEvent.ssc	Find ...	Out	N/A	□

Configure the rest of **cr_pass** as shown in the previous figure.

6 Select the **General** tab again, then under **Collaboration Rules**, click **Find**.

7 From the resulting dialog box, select the file **STCJavaPassThrough.class**.

8 Under **Initialization File**, click **Find**.

9 From the resulting dialog box, select the file **STCJavaPassThrough.ctl**.

10 Click **OK** to save your Pass Through Collaboration Rules file.

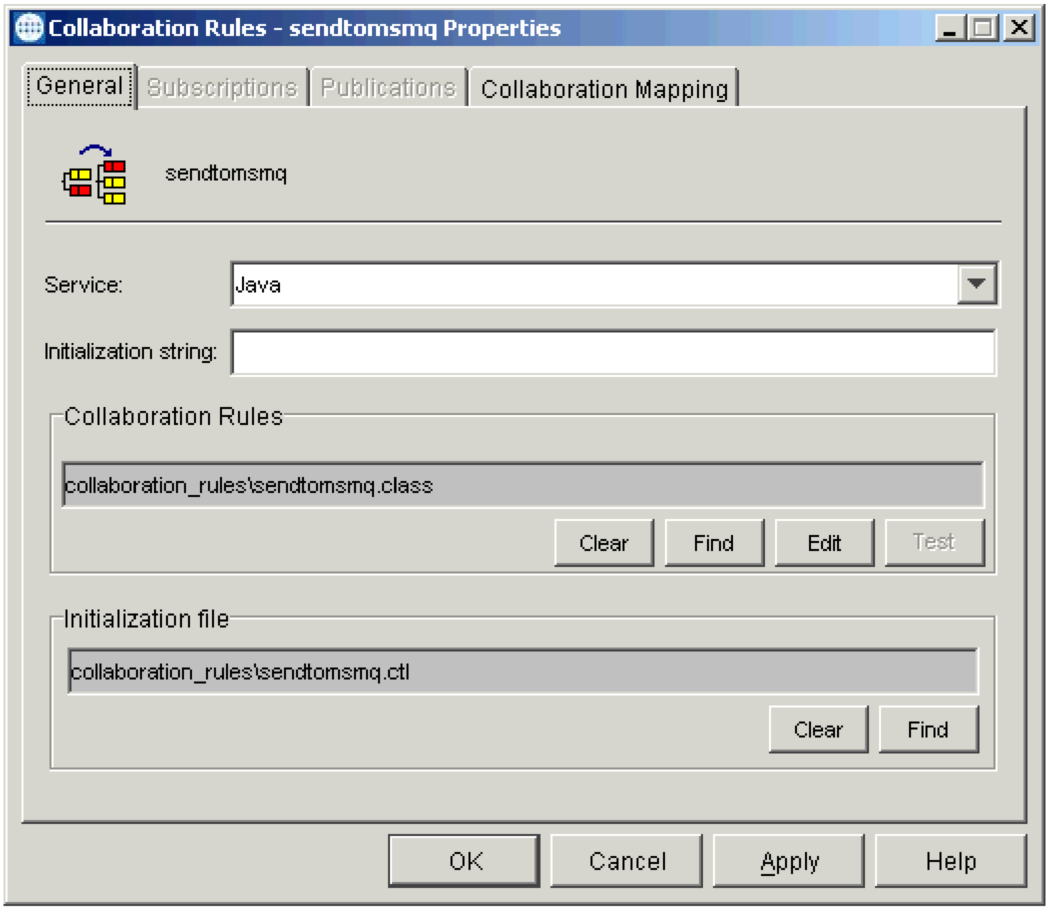
To create the sendtomsmq Collaboration Rules file

1 Repeat steps 3 through 5 under the **procedure on page 40** to create the next Collaboration Rule.

Use **sendtomsmq** as the name for this example, for the **msmq_send** e*Way's Collaboration, **cp_msmqsend**.

2 Double-click on the **sendtomsmq** icon. The **Collaboration Rules Properties** dialog box appears (see Figure 25).

Figure 25 Collaboration Rules Properties Dialog Box for sendtomsmq: General Tab



Collaboration Rules - sendtomsmq Properties

sendtomsmq

Service: Java

Initialization string:

Collaboration Rules: collaboration_rules\sendtomsmq.class

Initialization file: collaboration_rules\sendtomsmq.ctl

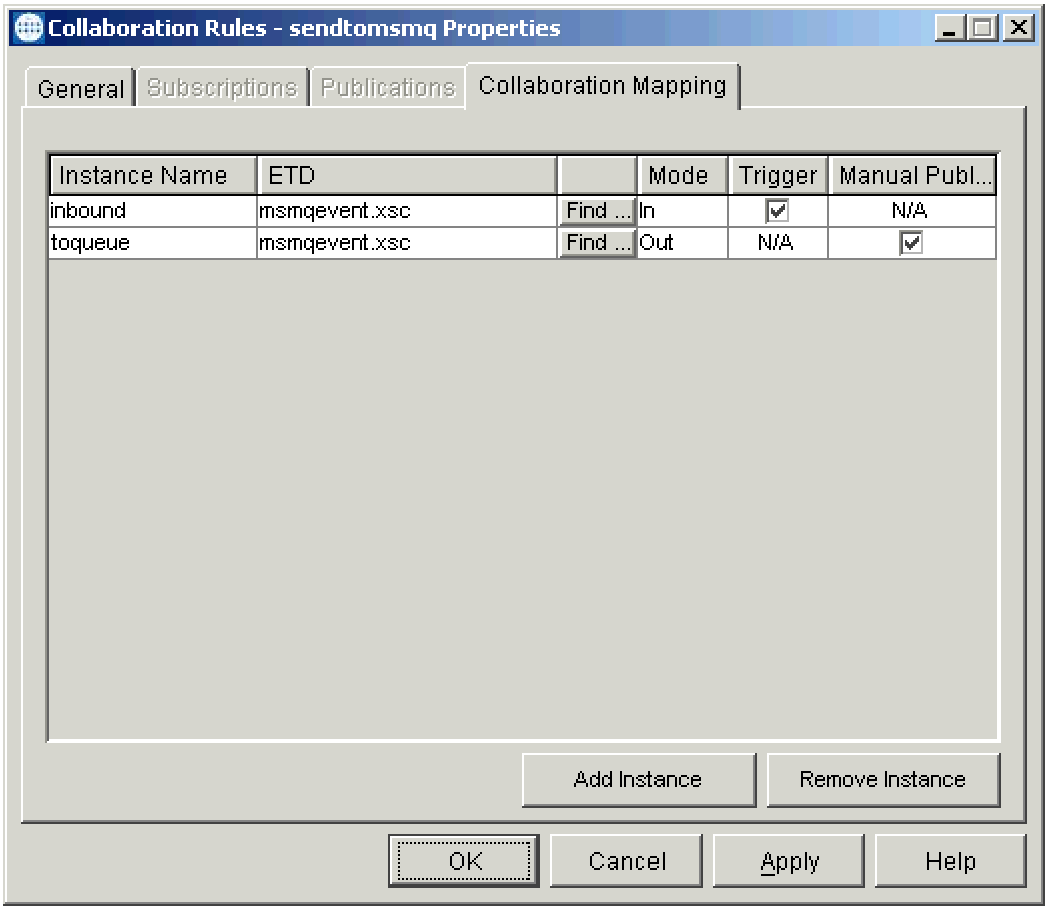
3 On the **General** tab in the dialog box select the **Java** Collaboration Service. In this example, the Collaboration Rules use the e*Gate Java Collaboration Service to manipulate Events or Event data.

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

5 Click the **Collaboration Mapping** tab.

6 Using the **Add Instance** button, create instances to coincide with the Event Types (see Figure 26).

Figure 26 Collaboration Rules Properties Dialog Box for sendtomsmq: Mapping Tab



Instance Name	ETD		Mode	Trigger	Manual Publ...
inbound	msmqevent.xsc	Find ...	In	☑	N/A
toqueue	msmqevent.xsc	Find ...	Out	N/A	☑

Configure the rest of **sendtomsmq** as shown in the previous figure.

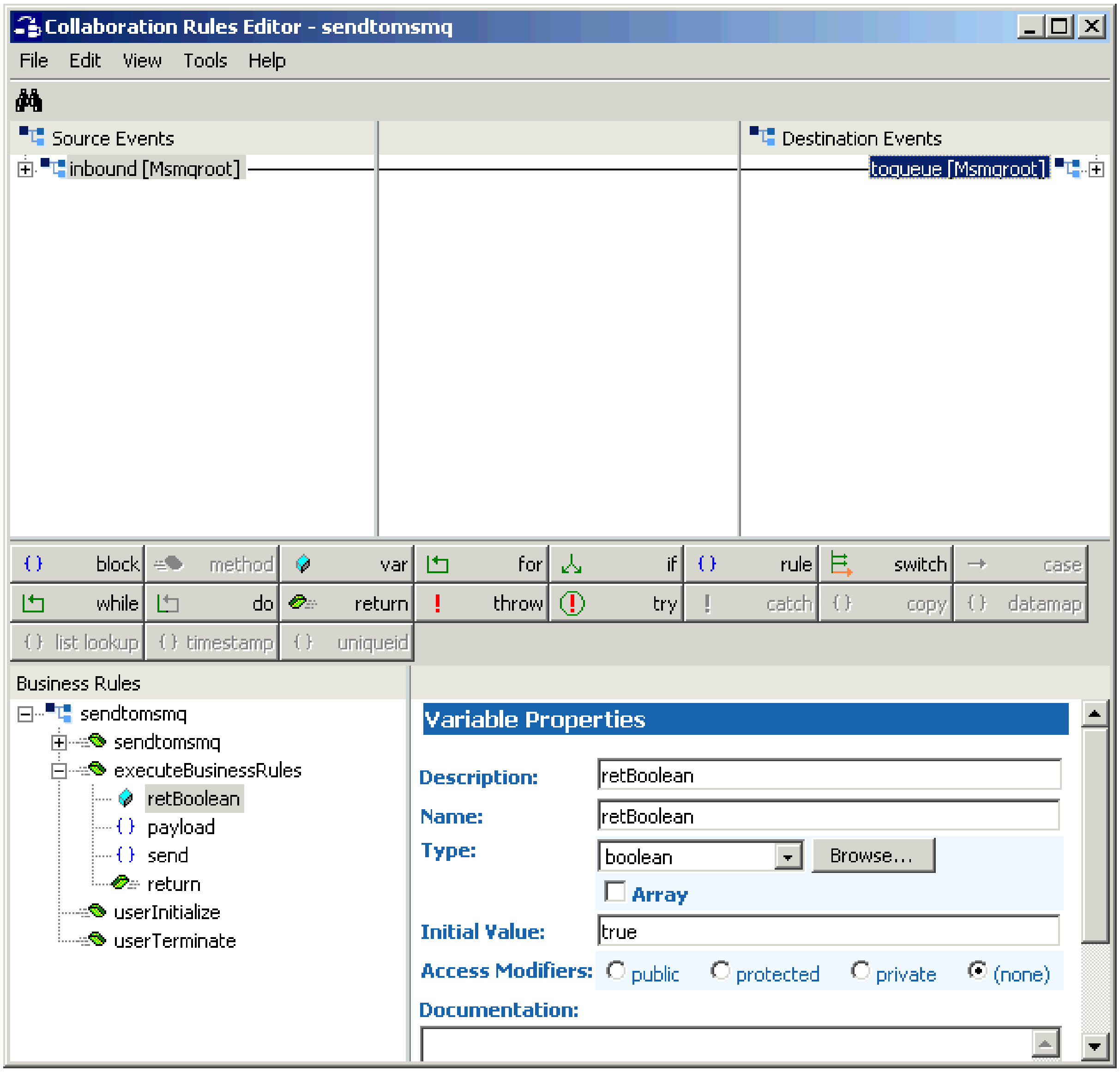
Note: *You can use the **msmqevent.xsc** ETD from the sample for your user-defined ETD or create one of your own that fits the example.*

7 Select the **General** tab again, then click **New**.

The Collaboration Rules Editor Main window opens.

8 Expand the window to full size for optimum viewing (see Figure 27).

Figure 27 Collaboration Rules Editor: sendtomsmq Start



Description: retBoolean

Name: retBoolean

Type: boolean

□ **Array**

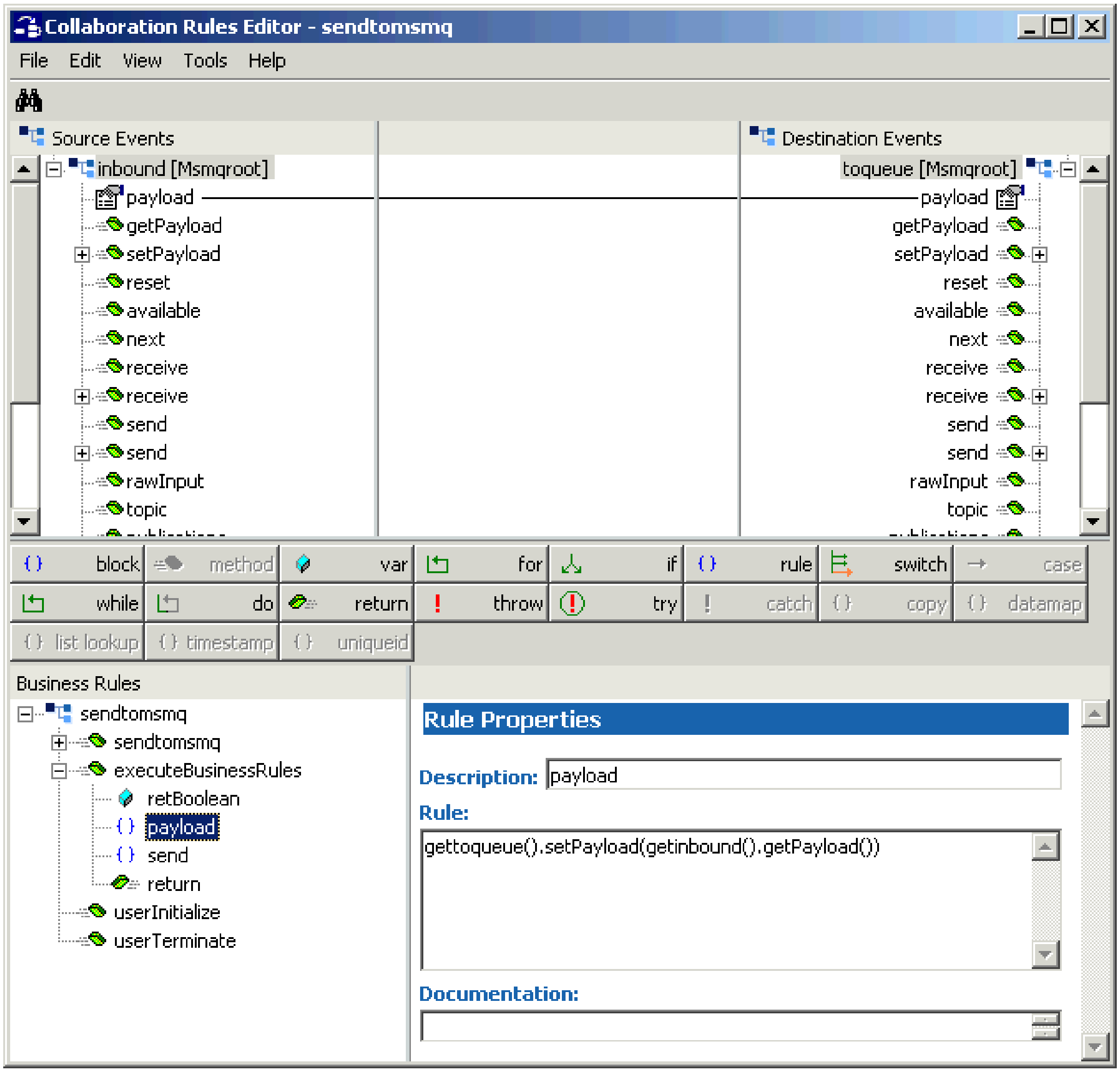
Initial Value: true

Access Modifiers: ○ public ○ protected ○ private ● (none)

Documentation:

9 Expand the source and destination Events, as well as the Business Rules. Figure 22 shows the results. Note Business Rules associated with each ETD.

Figure 28 Collaboration Rules Editor: sendtomsmq Expanded



Description: payload

Rule:

```
gettoqueue().setPayload(getinbound().getPayload())
```

Documentation:

10 Use the Collaboration Rules in the sample schema, and create the Business Rules for your **sendtomsmq.** Set them up in the same way as they are for **sendtomsmq** in the sample schema.

Note: For complete information on how to use the Collaboration Rules Editor, see the ***e*Gate Integrator User's Guide****.*

11 You must create a Collaboration Rules class or use one from the sample.

12 To save the Collaboration Rules file, click **Save** on the **File** menu. The **Save** dialog box appears.

13 Provide a name for the **.xpr** file (for this example, use **sendtomsmq.xpr**) then click **Save**.

14 When you have finished defining all the desired business logic, compile the Java code by selecting **Compile** from the **File** menu.

If the code compiles successfully, the message **Compile Completed** appears. If the outcome is unsuccessful, a Java Compiler error message appears. If there are any Java errors, be sure to correct them.

15 Once the compilation is complete, you can exit the Collaboration Rules Editor.

Creating Collaborations

Collaborations are the components that receive and process Event Types, then forward the output to other e*Gate components or an external system.

Collaborations consist of the subscriber, which receives Events of a known type (sometimes from a given source), and the publisher, which distributes transformed Events to a specified recipient.

To create the Collaborations

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

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

3 Select the Control Broker for this schema.

4 Select the **input** e*Way to assign the Collaboration.

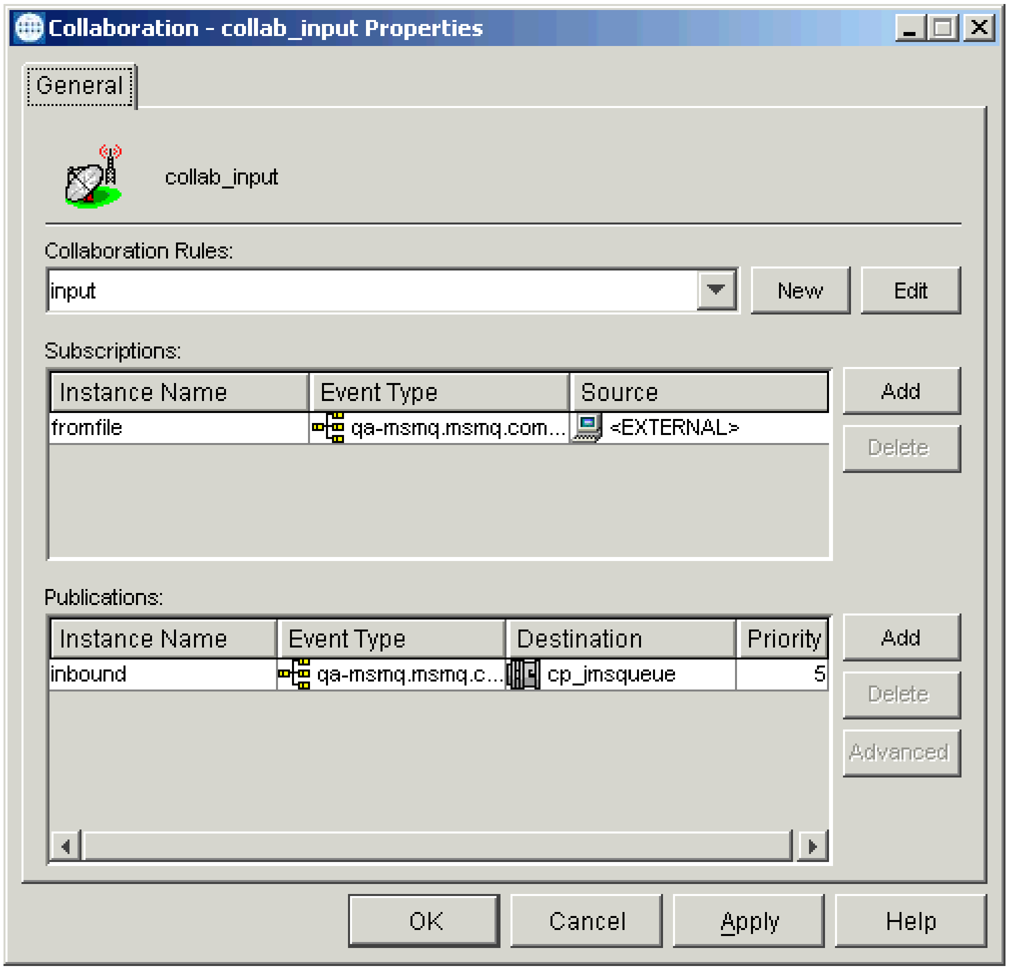
5 On the palette, click the **Collaboration** icon.

6 Enter the name (**collab_input**) of the new Collaboration, then click **OK**.

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

8 The **Collaboration Properties** dialog box appears (see Figure 29).

Figure 29 collab_input Properties Dialog Box



Collaboration - collab_input Properties

General

collab_input

Collaboration Rules: input

Subscriptions:

Instance Name	Event Type	Source
fromfile	qa-msmq.msmq.com...	<EXTERNAL>

Publications:

Instance Name	Event Type	Destination	Priority
inbound	qa-msmq.msmq.c...	cp_jmsqueue	5

Configure the appropriate Collaboration properties as shown in the previous figure.

9 From the **Collaboration Rules** list, select the first Collaboration Rule that you created previously (**input**) for this Collaboration.

10 In the **Subscriptions** area, click **Add** to define the input Event Type to which this Collaboration subscribes to (**qa-msmq.msmq.com\testmsmq1**). Also, select **fromfile** as the **Instance Name** and **<EXTERNAL>** for the **Source**.

11 In the **Publications** area, click **Add** to define the output Event Type that this Collaboration publishes to (**qa-msmq.msmq.com\testmsmq1**). Also select **inbound** as the **Instance Name** and **cp_jmsqueue** as the **Destination**.

12 Click **OK** to close the dialog box and save your changes.

13 Select the **output** e*Way to assign the next Collaboration.

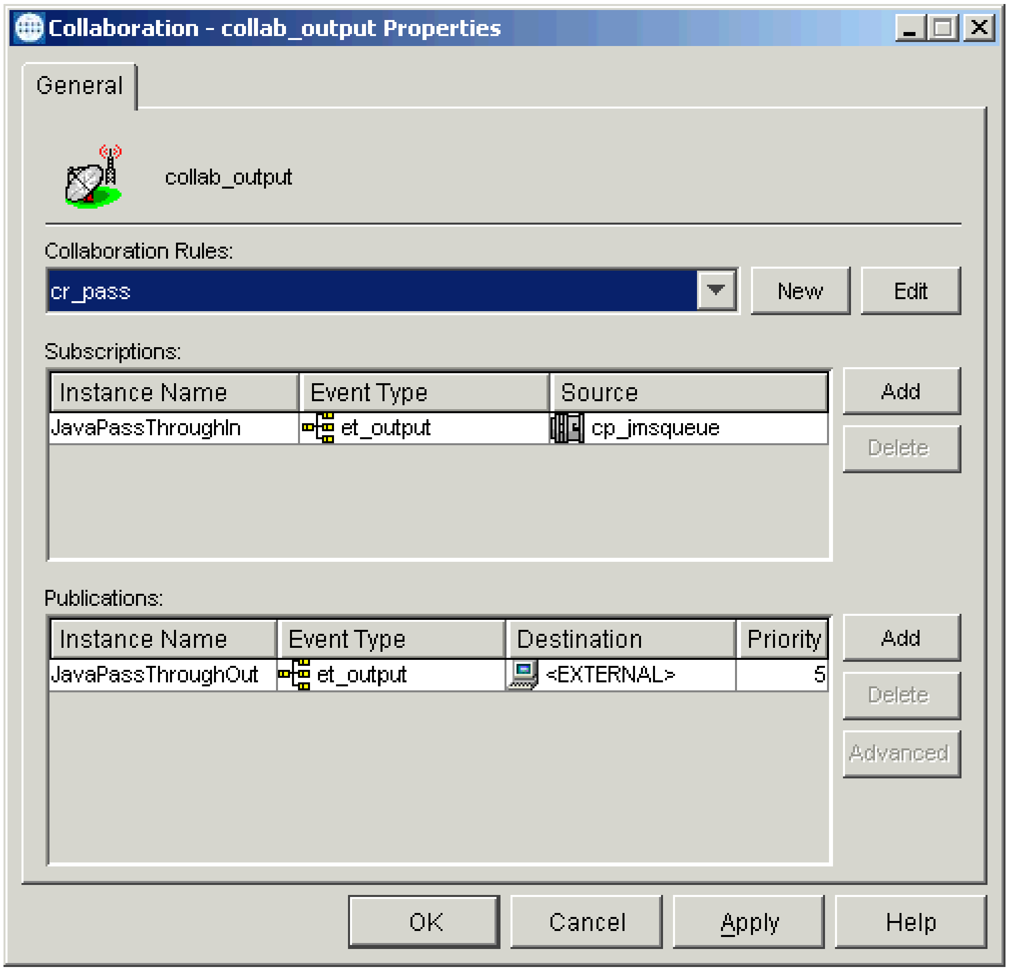
14 On the palette, click the **Collaboration** icon.

15 Enter the name (**collab_output**) of the new Collaboration, then click **OK**.

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

17 The **Collaboration Properties** dialog box appears (see Figure 30).

Figure 30 collab_output Properties Dialog Box



Collaboration - collab_output Properties

General

collab_output

Collaboration Rules: cr_pass

Subscriptions:

Instance Name	Event Type	Source
JavaPassThroughIn	et_output	cp_jmsqueue

Publications:

Instance Name	Event Type	Destination	Priority
JavaPassThroughOut	et_output	<EXTERNAL>	5

Configure the appropriate Collaboration properties as shown in the previous figure.

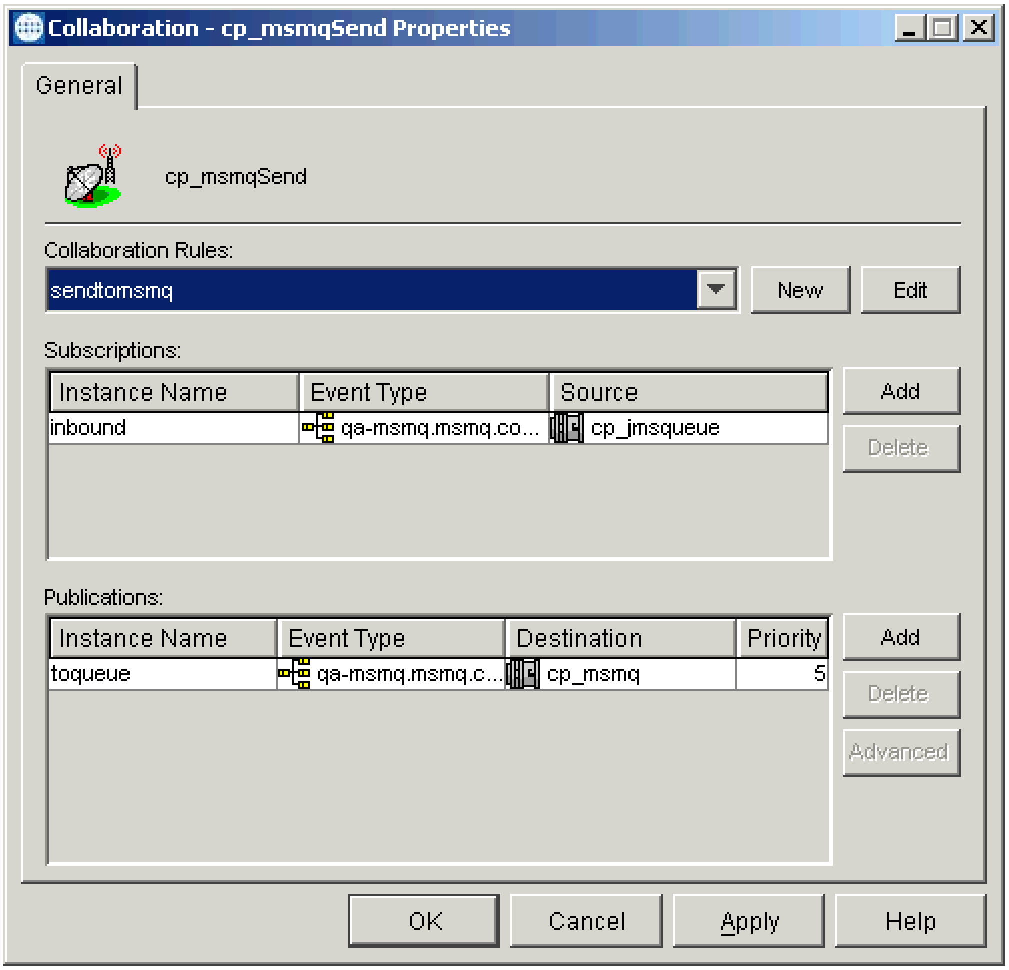
18 From the **Collaboration Rules** list, select the Collaboration Rule that you created previously (**cr_pass**) for this Collaboration.

19 Finish configuring the Collaboration as shown in Figure 30 then click **OK** to close the dialog box and save your changes.

20 Create two more Collaborations, following the steps 1 through 5 in these procedures. Name them **cp_msmqSend** (for the **msmq_send** e*Way) and **cp_msmqRcv** (for the **msmq_rcv** e*Way).

21 Configure these Collaboration as shown in Figure 31 and **Figure 32 on page 54**.

Figure 31 cp_msmqSend Properties Dialog Box



cp_msmqSend

Collaboration Rules: sendtomsmq

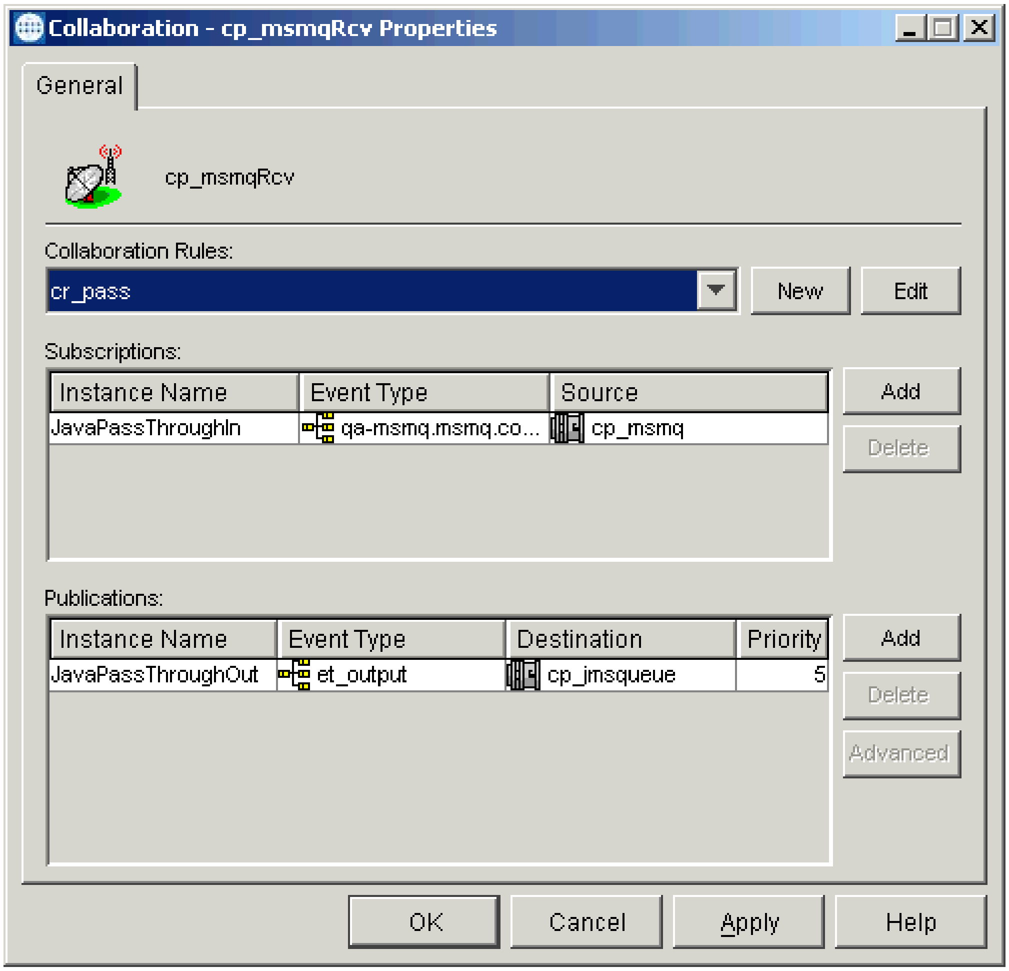
Subscriptions:

Instance Name	Event Type	Source
inbound	qa-msmq.msmq.co...	cp_jmsqueue

Publications:

Instance Name	Event Type	Destination	Priority
toqueue	qa-msmq.msmq.c...	cp_msmq	5

Figure 32 cp_msmqRcv Properties Dialog Box



Collaboration - cp_msmqRcv Properties

General

cp_msmqRcv

Collaboration Rules: cr_pass

Subscriptions:

Instance Name	Event Type	Source
JavaPassThroughIn	qa-msmq.msmq.co...	cp_msmq

Publications:

Instance Name	Event Type	Destination	Priority
JavaPassThroughOut	et_output	cp_jmsqueue	5

22 When you are finished with each dialog box, click **OK** to close them and save your changes.

Running the Schema

To run the schema

- From the command line prompt, enter on a single line:

```
stccb -rh hostname -rs schemaname -un username
     -up user_password -ln localhost_cb
```

Substitute the appropriate names for ***hostname***, ***username, schemaname,*** and ***user_password*** as appropriate.

The schema components start automatically. When there are no more run-time messages, check the output file. If the schema is operating correctly, the response message from the MSMQ queue appears as a file in the directory you specified on a local external system.

Index