SeeBeyond™ eBusiness Integration Suite

e-Mail e*Way Intelligent Adapter User's Guide

Release 4.5.2

Java Version

Version 20021011142354.

Contents

Chapter 1

Introduction 6

Overview 6
Intended Reader 6
Components 7

Operational Overview 7

Korean and Japanese e-mail Text Support 7

Supported Operating Systems 8

System Requirements 8

Chapter 2

Installation 9

Windows NT or Windows 2000 9
Pre-installation 9
Installation Procedure 9

UNIX 10
Pre-installation 10
Installation Procedure 10

Files/Directories Created by the Installation 11

Chapter 3

Multi-Mode e*Way Configuration 12

Multi-Mode e*Way 12
JVM Settings 12
JNI DLL Absolute Pathname 12
CLASSPATH Prepend 13
CLASSPATH Override 13
CLASSPATH Append From Environment Variable 14
Initial Heap Size 14
Maximum Heap Size 14
Maximum Stack Size for Native Threads 15
Maximum Stack Size for JVM Threads 15
Class Garbage Collection 15
Garbage Collection Activity Reporting 15

Asynchronous Garbage Collection 15
Report JVM Info and all Class Loads 16
Disable JIT 16
Remote Debugging Port Number 16
Suspend option for debugging 16

Chapter 4

e*Way Connection Configuration 17

e-Mail e*Way Connections 17

e-Mail e*Way Connection Configuration Parameters 18

Connector 19

Type 19
Class 19
Property.Tag 19

Connection Settings 19

HostSend 20
PortSend 20
UserSend 20
PasswordSend 20
HostRecv 21
PortRecv 21
UserRecv 21
PasswordRecv 21
SessionAuth 22

SeeBeyond JMS e*Way Connections 22

JMS e*Way Connection Configuration Parameters 23

General Settings 23

Connection Type 24
Transaction Type 24
Delivery Mode 24
Maximum Number of Bytes to Read 24
Default Outgoing Message Type 25
Factory Class Name 25

Message Service 25

Server Name 25
Host Name 25
Port Number 26
Maximum Message Cache Size 26

Chapter 5

Implementation 27

Implementation Overview 27

Sample Schema—Overview 28
Implementing the Sample Schema on e*Gate 4.5.0 28

Creating and Configuring Event Types 29

Creating Event Types with the Custom ETD Wizard 29
Creating Event Types from an Existing DTD 30
Creating Event Types from an Existing XSC 32

Creating and Configuring the e*Ways 32

Creating the Inbound e*Way 32

Creating the Outbound e*Way 33
Creating the Multi-Mode e*Ways 34

Collaboration Rules 34
Creating the cr_inputJava Collaboration Rules 35
Creating the cr_output_java Collaboration Rules 37
Creating the Java Collaboration Rules for cr_send_mail 37
Creating the Java Collaboration Rules for cr_recv_mail 39
Creating Business Rules using the Collaboration Rules Editor 40
Creating the cr_recv_mail Collaboration Rules 41
Creating the cr_send_mail Collaboration Rules Class 47

Enabling Japanese and Korean Character Support 51

Creating the JMS Queue Server (IQ Manager) 52

Creating e*Way Connections 53
To create and configure cp_rcvEmail e*Way Connection 53
To create and configure cp_sendEmail e*Way Connection 54
To create and configure JMS e*Way Connections 54

Collaborations 55
Creating the Input Collaboration 55
Creating the Output Collaboration 57
Creating the Receive Collaboration 57
Creating the Send Collaboration 58

Executing the Schema 59

Chapter 6

Java Methods 60

The EmailAddress Class 60
EmailAddress 60

The EmailAttachmentFile Class 61
EmailAttachmentFile 61
save 61

The EmailMessage Class 62
sendMessage 62
receiveMessage 63

Index 64

Chapter 1

Introduction

This document describes how to install and configure the Java™ e-Mail e*Way Intelligent Adapter.

1.1 Overview

The Java e-Mail e*Way allow e*Gate users to automate many of what were typically manual email operations. Email can be sent and received programatically without human intervention. Business rules and data mapping are defined using standard e*Gate tools which include Java Collaborations and the ETD Editor.

How does the e*Way connect?

The e-Mail e*Way takes advantage of widely used standard protocol.,

- POP3 (Post Office Protocol) which can be thought of as a store and forward service. POP3 is used for retrieving email from a mail account. POP3 requires the mail server name, the TCP/IP Port, the email account name, and the email account password.
- SMTP (Simple Mail Transfer Protocol) which is almost a mirror image of POP3, and is used for sending email to a mail account. SMTP, like POP3 requires the mail server name, the TCP/IP Port, the email account name, and the email account password. Some mail servers also require POP3 authentication to send messages.
- MIME (Multi-purpose Internet Mail Extensions) protocol. Servers insert the MIME header at the beginning of any Web transmission. Clients use the header to select appropriate applications for the type of data the header indicates. Multipart MIME messages (alternate text/HTML) and multiple attachments (other than nested MIME objects) are supported.

The e*Way provides a standard ETD (mailclient.xsc) for managing email content and for all sending and retrieving of email. The ETD also allows for dynamic configuration of the connection fields within a Collaboration (a connection that has been setup can be changed from within the Collaboration).

1.1.1. 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 working knowledge of Java Programming.
- To be familiar with SMTP and POP3 mail server protocols.
- To thoroughly familiar with Windows and UNIX operations and administration.

1.1.2. Components

The following components comprise the Java e-Mail e*Way:

- **stcemail.jar**: Contains the logic required by the e*Way to gain access to the e-Mail Server etc.
- **mailclient.xsc**: Allows the user to create hierarchical Event Type Definitions manually to be used in conjunction with the parsing engine contained within the extended Java Collaboration Service.
- **e*Way Connections**: The e-Mail e*Way Connections provide access to the information necessary for connection to a specified external system.

A complete list of installed files appears in **Table 1 on page 11**.

1.2 Operational Overview

The Java version of the e-Mail e*Way uses Java Collaborations to interact with one or more external systems. The Java Collaboration Service enables a variety of e*Gate components—such as e*Way Intelligent Adapters (e*Ways) to connect to external systems and execute business rules written entirely in Java.

An e*Gate component is defined as Java-enabled based on the selection of the Java Collaboration Service in the Collaboration Rule setup.

1.3 Korean and Japanese e-mail Text Support

The Java e-Mail e*Way supports Korean (Hangul) and Japanese character encoding in both the address and subject headers and text content of the e-mail message (for both text/plain and text/HTML). File attachment names are converted to “B” encoding. The e-Mail e*Way conforms to RFC2047 standards for Multipurpose Internet Mail Extensions (MIME).

In order for the email to be in encoded in ISO-2022-KR (Korean characters) or ISO-2022-JP (Japanese characters), the value for CharSet must be set as "ISO-2022-KR" or "ISO-2022-JP" (see **Enabling Japanese and Korean Character Support** on page 51).

1.4 Supported Operating Systems

The the e-Mail e*Way is available on the following operating systems:

- Windows 2000, Windows 2000 SP1, Windows 2000 SP2, and Windows 2000 SP3
- Windows NT 4.0 SP6a
- Solaris 2.6, 7, and 8
- AIX 4.3.3
- HP-UX 11.0 and HP-UX 11i
- Japanese Windows 2000, Windows 2000 SP1, Windows 2000 SP2, and Windows 2000 SP3 (Java only)
- Japanese Windows NT 4.0 SP6a (Java Only)
- Japanese Solaris 2.6, 7, and 8 (Java Only)
- Japanese HP-UX 11.0 (Java Only)
- Korean Windows 2000, Windows 2000 SP1, Windows 2000 SP2, and Windows 2000 SP3 (Java Only)
- Korean Windows NT 4.0 SP6a (Java Only)
- Korean Solaris 8 (Java Only)

Note: *Open and review the **Readme.txt** for the e-Mail e*Way for any additional information or requirements, prior to installation. The Readme.txt is located on the Installation CD_ROM at setup\addons\ewemail.*

1.5 System Requirements

To use the e-Mail e*Way, you need the following:

- An e*Gate Participating Host, version 4.5.1 or later.
- A TCP/IP network connection.
- Additional disk space for e*Way executable, configuration, library, and script files. The disk space is required on both the Participating and the Registry Host. 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.

Important: *The E-mail e*Way does not currently support sending/receiving messages containing multi-level nested MIME parts.*

Chapter 2

Installation

This chapter describes how to install the e-Mail e*Way. Both the Monk and Java-enabled versions are installed.

2.1 Windows NT or Windows 2000

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 e-Mail e*Way on a Windows NT/ Windows 2000 system

1. Log in as an Administrator to the workstation on which you are installing the e*Way.
2. Insert the e*Way installation CD into the CD-ROM drive.
3. If the CD-ROM drive's "Autorun" feature is enabled, the setup application should launch automatically; skip ahead to step 4. Otherwise, use the Windows Explorer or the Control Panel's **Add/Remove Applications** feature to launch the file **setup.exe** on the CD-ROM drive.
4. The InstallShield setup application launches. Follow the installation instructions until you come to the **Please choose the product to install** dialog box.
5. Select **e*Gate Integrator**, then click **Next**.
6. Follow the on-screen instructions until you come to the second **Please choose the product to install** dialog box.
7. Clear the check boxes for all selections except **Add-ons**, and then click **Next**.
8. Follow the on-screen instructions until you come to the **Select Components** dialog box.
9. Highlight (but do not check) **e*Ways**, and then click the **Change** button. The **SelectSub-components** dialog box appears.

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

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

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

2.2 UNIX

2.2.1. Pre-installation

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

2.2.2. Installation Procedure

To install the e-Mail e*Way on a UNIX system

1 Log in on the workstation containing the CD-ROM drive, and insert the CD-ROM into the drive.

2 If necessary, mount the CD-ROM drive.

3 At the shell prompt, type

cd /cdrom

4 Start the installation script by typing

setup.sh

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

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

6 After installation is complete, exit the installation utility and launch the Enterprise Manager.

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

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

2.3 Files/Directories Created by the Installation

The e-Mail e*Way installation process will install the following files within the e*Gate directory tree. Files will be 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 the installation of the Java-enabled e-Mail e*Way

e*Gate Directory	Files(s)
client\ThirdParty\jaf-1.0.1\classes\	activation.jar
client\ThirdParty\jaxp1.0.1\classes	jaxp.jar
client\ThirdParty\javamail-1.2\classes\	imap.jar pop3.jar mail.jar mailapi.jar smtp.jar
client\classes\	stcemail.jar stcutil.jar
configs\mailclient\	email.def
etd\mailclient\	mailclient.xsc

Note: *Please see the e-Mail e*Way Intelligent Adapter User's Guide (Monk-enabled) for information regarding the files installed by the Monk-enabled version of the e*Way.*

Chapter 3

Multi-Mode e*Way Configuration

This chapter describes how to create and configure a Multi-Mode e*Way.

3.1 Multi-Mode e*Way

Multi-Mode e*Way properties are set using the Enterprise Manager.

To create and configure a New Multi-Mode e*Way

1. Select the Navigator's Components tab.
2. Open the host on which you want to create the e*Way.
3. On the Palette, click the **Create a New e*Way** button.
4. The New e*Way Component window appears. Enter the name of the new e*Way, then click **OK**.
5. Right-click the new e*Way and select **Properties**. The e*Way Properties dialog box appears.
6. The Executable File field contains **stceway.exe** as the default.
7. Under the Configuration File field, click on the **New** button. When the Settings page opens, set the configuration parameters for this configuration file.
8. After selecting the desired parameters, save the configuration file. Close the **.cfg** file and select **OK** to close the e*Way Properties Window.

Multi-Mode e*Way Configuration Parameters

3.1.1. JVM Settings

The JVM Settings control basic Java Virtual Machine settings.

JNI DLL Absolute Pathname

Description

Specifies the absolute pathname to where the JNI DLL installed by the Java 2 SDK 1.3.1_02 is located on the Participating Host.

Required Values

A valid pathname.

Additional Information

The JNI dll name varies on different O/S platforms:

OS	Java 2 JNI DLL Name
NT 4.0/ Windows 2000	jvm.dll
Solaris 2.6, 2.7, 2.8	libjvm.so
Linux 6	libjvm.so
HP-UX	libjvm.sl
AIX 4.3	libjvm.a

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

```
%MY_JNIDLL%
```

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

To ensure that the JNI DLL loads successfully, the Dynamic Load Library search path environment variable must be set appropriately to include all the directories under the Java 2 SDK (or JDK) installation directory that contain shared libraries (UNIX) or DLLs (NT).

CLASSPATH Prepend

Description

Specifies the paths to be prepended to the CLASSPATH environment variable for the Java VM.

Required Values

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

Additional Information

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

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

```
%MY_PRECLASSPATH%
```

CLASSPATH Override

Description

Specifies the complete CLASSPATH variable to be used by the Java VM. This parameter is optional. If left unset, an appropriate CLASSPATH environment variable

(consisting of required e*Gate components concatenated with the system version of CLASSPATH) will be set.

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

Required Values

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

Additional Information

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

```
%MY_CLASSPATH%
```

CLASSPATH Append From Environment Variable

Description

Specifies the classpath append from the environmental variable.

Required Values

YES or **NO**.

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 will be used.

Required Values

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

Maximum Heap Size

Description

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

Required Values

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

Maximum Stack Size for Native Threads

Description

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

Required Values

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

Maximum Stack Size for JVM Threads

Description

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

Required Values

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

Class Garbage Collection

Description

Specifies whether the Class Garbage Collection will be done automatically by the JVM. The selection affects performance issues. Reserved for future use. Do not change from default value.

Required Values

YES or **NO**.

Garbage Collection Activity Reporting

Description

Specifies whether garbage collection activity will be reported for debugging purposes. Reserved for future use. Do not change from default value.

Required Values

YES or **NO**.

Asynchronous Garbage Collection

Description

Specifies whether asynchronous garbage collection activity will be reported for debugging purposes.

Required Values

YES or **NO**.

Report JVM Info and all Class Loads

Description

Specifies whether the JVM information and all class loads will be reported for debugging purposes. Reserved for future use. Do not change from default value.

Required Values

YES or **NO**.

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.*

Remote Debugging Port Number

Description

Specifies the port number used for remote debugging of the JVM.

Required Values

An integer between 2000 and 65536.

Suspend option for debugging

Description

Specifies whether debugging occurs during JVM startup.

Required Values

YES or **NO**.

Chapter 4

e*Way Connection Configuration

The e*Way Connection configuration file contains the connection information along with the information needed to communicate via e-mail. This chapter describes how to configure the following e*Way Connections:

- **e-Mail e*Way Connections** on page 17
- **SeeBeyond JMS e*Way Connections** on page 22

Note: *Having multiple connection points accessing the same e-mail account will result in receiving the same e-mail message from more than one connection point. Therefore to avoid duplicates, only one Java E-mail e*Way connection point should be active per e-mail account.*

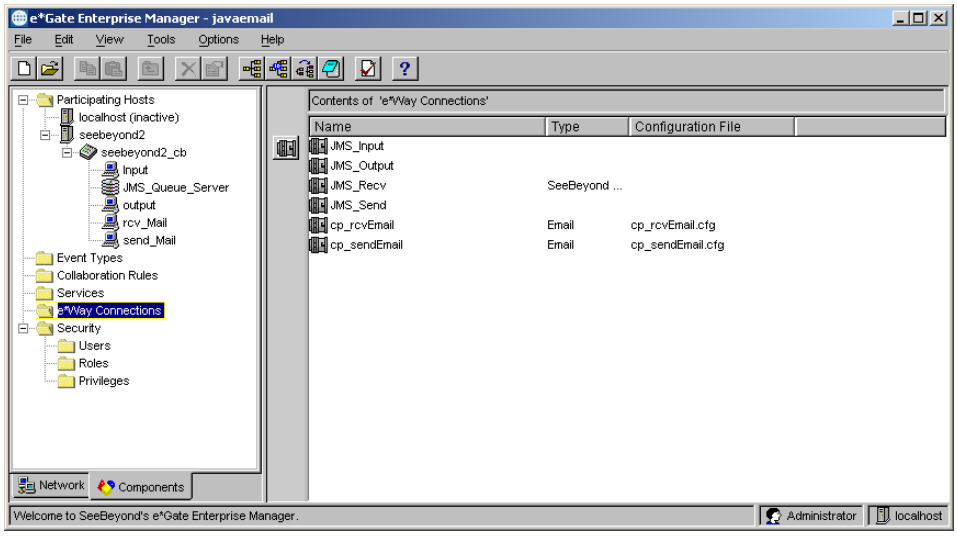
4.1 e-Mail e*Way Connections

e*Way Connections are set using the Enterprise Manager.

To create and configure e*Way Connections

1 In the Enterprise Manager's **Component** editor, select the **e*Way Connections** folder.

Figure 1 Enterprise Manager Component Editor



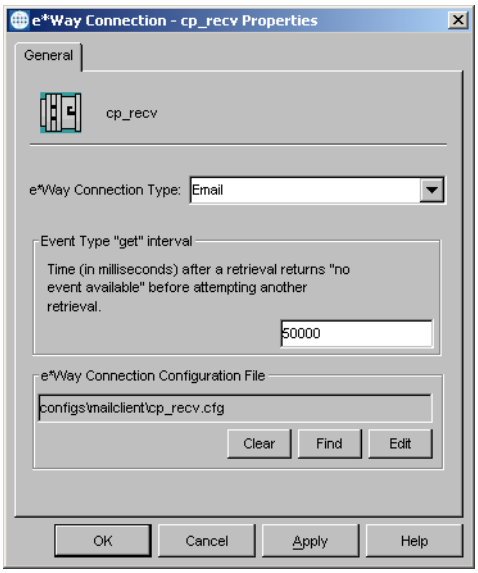
2 On the palette, click the **Create a New e*Way Connection** button.

3 The New e*Way Connection Component dialog box appears. Enter a name for the **e*Way Connection** and click **OK**.

4 Double-click on the new e*Way Connection to open the e*Way Connections Properties dialog box (see Figure 3).

5 From the **e*Way Connection Type** drop-down box, select Email

Figure 2 e*Way Connection Properties dialog box



6 Clear the **Event Type "get" interval** field and enter 600000 for time in milliseconds before attempting another retrieval after a retrieval returns "no event available."

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 e*Way Editor appears.

Note: *If changes are made to an existing e*Way Connection file, any e*Ways using the revised e*Way Connection must be restarted.*

4.2 e-Mail e*Way Connection Configuration Parameters

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

- **Connector** on page 19
- **Connection Settings** on page 19

4.2.1. Connector

This section contains a set of top level parameters:

- Type
- Class
- Property.Tag

Type

Description

Specifies the type of connection to be used as the default for e-mail.

Required Values

String-set. The configured default is always **email** for email.

Class

Description

Specifies the class name of the e-mail connector object.

Required Values

String-set. A valid e-mail connector object class name. The configured default is **com.stc.eways.email.EmailConnector**.

Property.Tag

Description

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

Required Values

String-set. A valid data source.

4.2.2. Connection Settings

This section contains a set of top-level parameters

- HostSend
- PortSend
- UserSend
- PasswordSend
- HostRecv

- PortRecv
- UserRecv
- PasswordRecv
- SessionAuth

HostSend

Description

Specifies the host name of the server used to send messages. This is required for "sending" e*Way connections.

Required Values

String-set. A valid server name.

PortSend

Description

Specifies the port number to connect to when sending messages. This is required for "sending" e*Way connections.

Required Values

Integer-set. An integer in the range of 1 to 65535. The configured default is 25.

UserSend

Description

Specifies the user name used when sending messages. This is required for "sending" e*Way connections.

Required Values

String-set. A valid user login name.

PasswordSend

Description

Specifies the password used when sending messages. This is required for "sending" e*Way connections.

Required Values

String-encrypt. A valid user login password.

HostRecv

Description

Specifies the host name of the server used to receive messages. This is required for "receiving" e*Way connections. This is also required for "sending" e*Way connections when the SessionAuth parameter is set to Yes (for POP3 login).

Required Values

String-set. A valid server name.

PortRecv

Description

Specifies the port number to connect to when receiving messages. This is required for "receiving" e*Way connections.This is also required for "sending" e*Way connections when the SessionAuth parameter is set to Yes (for POP3 login).

Required Values

Integer-set. An integer in the range of 1 to 65535. The configured default is 110.

UserRecv

Description

Specifies the user name used when receiving messages. This is required for "receiving" e*Way connections. This is also required for "sending" e*Way connections when the SessionAuth parameter is set to Yes (for POP3 login).

Required Values

String-set. A valid user login name.

PasswordRecv

Description

Specifies the password used when receiving messages. This is required for "receiving" e*Way connections. This is also required for "sending" e*Way connections when the SessionAuth parameter is set to Yes (for POP3 login).

Required Values

String-encrypt. A valid user login password.

SessionAuth

Description

Determines whether a POP3 session authentication is performed before attempting an SMTP connection. This is required by some e-mail services. Set the value to **YES** only when necessary. Yes requires that settings for HostRecv, PortRecv, UserRecv, and PasswordRecv are entered for the "sending" e*Way connection.

Required Values

String-set. **YES** or **NO**. Yes indicates that POP3 session authentication will be performed before attempting an SMTP connection.

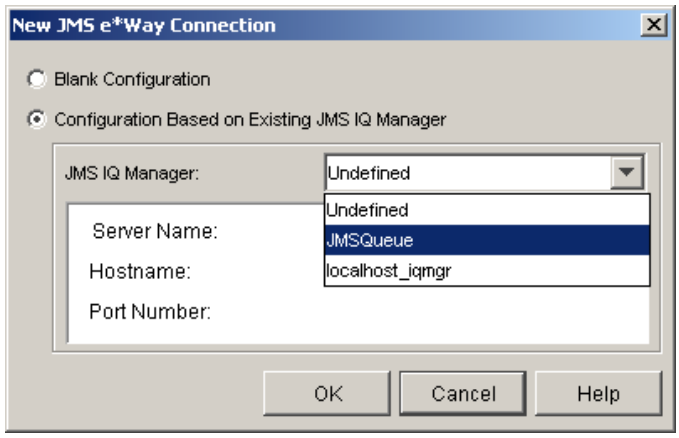
4.3 SeeBeyond JMS e*Way Connections

The SeeBeyond JMS e*Way Connections are used to connect to the JMSQueue Manager. For further information on the JMS IQ Manager and configuration parameters for JMS e*Way Connections see the **e*Gate Integrator User's Guide**.

To create and configure e*Way Connections

1. In the Enterprise Manager'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 appears. Enter a name for the **e*Way Connection**. For the purposes of the sample enter **JMS_Input** as the name.
4. Double-click on the new **e*Way Connection**. The **e*Way Connection Properties** dialog box appears.
5. From the **e*Way Connection Type** drop-down box, select **SeeBeyond JMS**.
6. Enter the **Event Type "get"** interval in the dialog box provided. 10000 milliseconds is the configured default.
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**.) When the **New** button is selected the **New JMS e*Way Connection** dialog box appears.

Figure 3 New JMS e*Way Connection dialog box



8 Do one of the following:

- To create an entirely new set of configuration parameters, click the Blank Configuration option, click OK, and then use the Edit Settings dialog box to set the configuration parameters for this configuration file.
- To inherit pre-existing Message Service configuration parameters, click Configuration Based on Existing JMS IQ Manager, and then select a JMS IQ Manager from the list (its server name, hostname, and IP address are displayed) and click OK. A configuration file is created; if you want to view or modify its configuration parameters or add user notes, click Edit.

9 After using the **Edit Settings** dialog box to set configuration parameters and add user notes, save the .cfg file (using the same name as the e*Way Connection, and accepting the default location), close the Edit Settings dialog box, and then click OK to close the e*Way Connection Properties dialog box.

4.3.1. JMS e*Way Connection Configuration Parameters

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

- **General Settings** on page 23
- **Message Service** on page 25

4.3.2. General Settings

This section contains a set of top level parameters:

- Connection Type
- Transaction Type
- Delivery Mode
- Maximum Number of Bytes to Read
- Default Outgoing Message Type

- Factory Class Name

Connection Type

Description

Specifies the JMS Messaging Model. Publish-and-subscribe (Topic) or

Point-to-point (Queue). The connection type must be specified.

Required Values

String-set. Value is Queue. Queue is the configured default.

Transaction Type

Description

Specifies the Transaction Type. There are three transaction types. One-phase transactional behavior (**Internal**). Two-phase transactional behavior (**XA-compliant**). No transactional behavior (**Non-Transactional**). The transaction type must be specified.

Required Values

String-set. A valid transaction type. One of three provided: Internal, Non-Transactional or XA-compliant. Non-Transactional is the configured default

Delivery Mode

Description

Specifies the Message Delivery Mode. Marking the message as persistent ensures that the message will be saved to a reliable persistent store by the Message Server before the publish() method returns. The delivery mode must be specified.

Required Values

String-set. Non-Persistent or Persistent. Persistent is the configured default.

Maximum Number of Bytes to Read

Description

Specifies the maximum number of bytes to read at a time from the received Bytes Message. The maximum number of bytes must be specified.

Required Values

Integer-set. An integer in the range of 1 to 200,000,000. The configured default is 5000.

Default Outgoing Message Type

Description

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

Required Values

String-set. Bytes or Text. The configured default is Bytes.

Factory Class Name

Description

Specifies the Factory class to be used to connect to the JMS IQ Manager. This is advanced configuration to be utilized in future development.

Required Values

String-set. The valid factory class name. The configured default is com.stc.common.collabService.SBYNJMSFactory. Retain the default setting.

4.3.3. Message Service

This section contains a set of top level parameters:

- Server Name
- Host Name
- Port Number
- Maximum Message Cache Size

Server Name

Description

Specifies the name of the JMS IQ Manager. The queue manager name must be specified.

Required Values

String-set. Enter the name of the JMS queue manager.

Host Name

Description

Specifies the name of the host on which the JMS Queue Manager resides.

Required Values

String-set. The valid host name.

Port Number

Description

Specifies the number of the port on which the JMS Queue Manager is running.

Required Values

Integer-set. An integer in the range of 1000 to 65536.

Maximum Message Cache Size

Description

Specifies the maximum cache size

Required Values

String-set. A valid cache size.

Chapter 5

Implementation

This chapter contains information pertinent to implementing the Java-enabled e-Mail e*Way in a production environment. A complete sample schema is included on the CD. This chapter gives directions for executing the sample schema, giving examples of how those components were created and configured. It is assumed that the e-Mail e*Way package has been successfully installed.

Important: *Closely monitor the "sending" email account for any errors in delivery. All error messages for errors in delivery that occur after the initial mail server will be sent only to the sending mail account.*

Note: *After a successful call to* ***EmailMessage.receiveMessage()*** *and before a successful call to* ***send()****, the received e-mail message is contained ONLY in the EmailMessage instance in the Collaboration Rule. Failure to process and/or store it properly will result in message loss.*

5.1 Implementation Overview

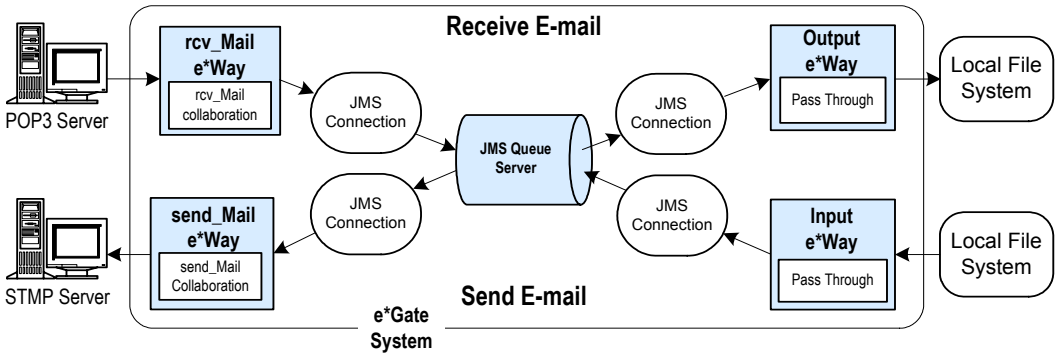
During installation, the Host and Control Broker are automatically created and configured. The names of the Host and Control Broker default to the name of the host machine on which the e*Gate Enterprise Manager GUI is installed.

Creating the sample schema

- Make sure that the Control Broker is activated.
- In the e*Gate Enterprise Manager, define and configure the following:
  - The Multi-Mode e*Way Components, as described in **Chapter 3**
  - The Inbound and Outbound e*Ways using **stcewfile.exe**
  - The e*Way Connections, as described in **Chapter 4**.
  - Event Type Definitions used to package the data being exchanged with the external system.
  - Collaboration Rules to process Events.
  - Configure Collaborations within the e*Way component to apply the required Collaboration Rules.

The following sections describe how to define and associate the above components for a sample schema.

Figure 4 The e-Mail e*Way sample schema—Overview



Sample Schema—Overview

Receive e-mail

1. The rcv_Mail e*Way reads the inbound information from the POP3 Server, transforms the Event, and publishes it as an Event to the JMS Connection
2. The JMS Connection subscribes to the rcv_Mail e*Way and publishes to the JMS_Queue Server.
3. The Output e*Way subscribes to the inbound Event from the rcv_Mail e*Way, and publishes it to the local file system.

Send e-mail

1. The Input e*Way reads the outbound information from the local file system and publishes it to the JMS Connection
2. The JMS Connection subscribes to the Input e*Way and publishes to the JMS_Queue Server.
3. The send_Mail e*Way subscribes to the outbound information from the Input e*Way, transforms the Event, and publishes it to the STMP Server.

Implementing the Sample Schema on e*Gate 4.5.0

The sample schema for the Java-enabled e-Mail e*Way was created using e*Gate 4.5.1. When attempting to implementing the sample using e*Gate 4.5.0, make note of the following:

- It is necessary to create the schema manually and not simply import the schema for e*Gate 4.5.0. When imported, the sample will display in the editors but will not work with e*Gate 4.5.0.
- In e*Gate 4.5.0 implementations the receiving Collaboration should use a closed loop where you set the number of messages to retrieve at any one run of the Collaboration. 4.5.1 does not impose this limitation. This is due to the send() method in the receiving Collaboration not sending until the Collaboration has

exited successfully. Until that time all received messages are held in system memory.

- The subscribers must be started before starting the publishers to JMS.

These conditions only apply to implementation on e*Gate 4.5.0.

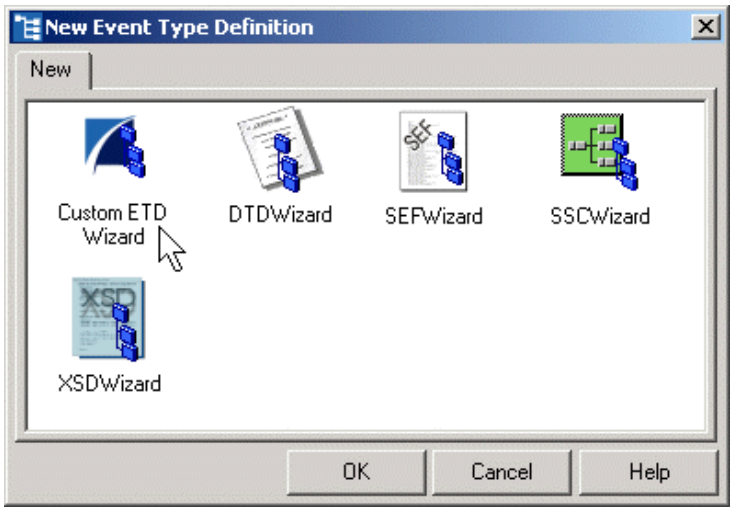
5.2 Creating and Configuring Event Types

An Event Type Definition is a graphical representation of the layout of data in an Event. The following are examples of how Event Types are created and configured.

Creating Event Types with the Custom ETD Wizard

1. Select the **Event Types** folder on the Components tab of the e*Gate Navigator.
2. On the palette, click the **Create a New Event Type** button.
3. Enter the name of the Event Type, then click **OK** (for this example, **node**).
4. Double-click the new Event Type to edit its properties. The **Properties** window appears.
5. Click the **New** button. The ETD Editor appears.
6. Select **New** from the **File** menu on Task Manager. The **New Event Type Definition** window appears.
7. Select the Custom ETD Wizard and click OK (see Figure 5).

Figure 5 Event Type Definition Wizard

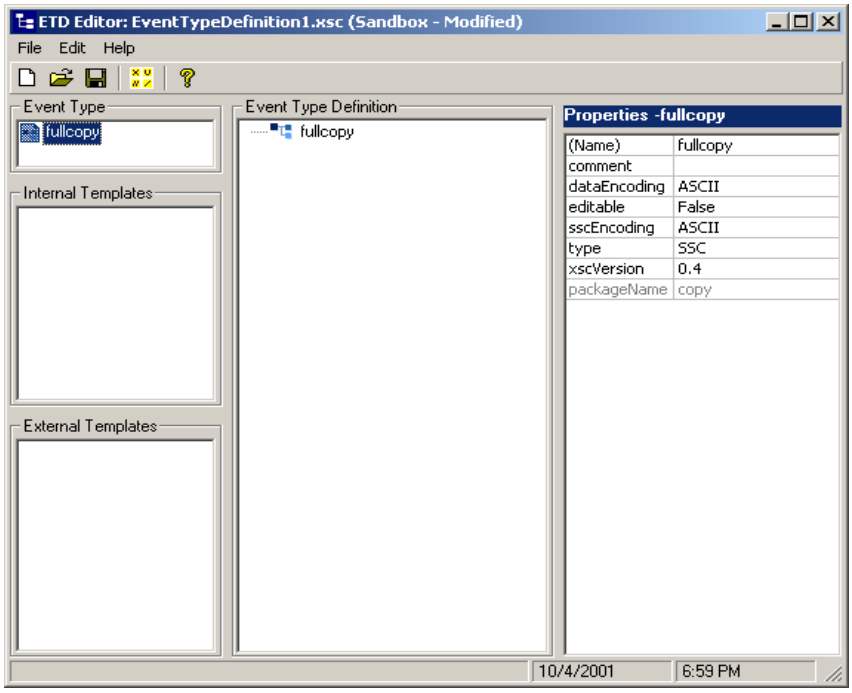


8. Enter the Root Node Name (for this example, **fullcopy**).
9. Enter the Package Name where the ETD builder can place all the generated Java classes associated with the created ETD (for this example, **copy**).

10 Click **Next**, review the entries, and click **Finish**.

11 In the **Event Type Definition** pane, right-click **fullcopy**, and select **Add Field**, **As Child Node** (see Figure 6).

Figure 6 ETD Editor



12 Triple-click **Field1** and change the name to **emaildata**.

13 From the **File** menu, click **Compile and Save**. Name the file (for this example, **emailCopy**).

14 From the **File** menu click **Promote to RunTime**

Creating Event Types from an Existing DTD

The following procedures show how to create an Event Type Definition (ETD) from an existing Document Type Definition (DTD) using **EmailMessage.dtd** as the input file.

1 Select the **Event Types** folder on the Components tab of the e*Gate Navigator.

2 On the palette, click the **Create a New Event Type** button.

3 Enter the name of the **Event**, then click **OK** (for the example, **et_Input**).

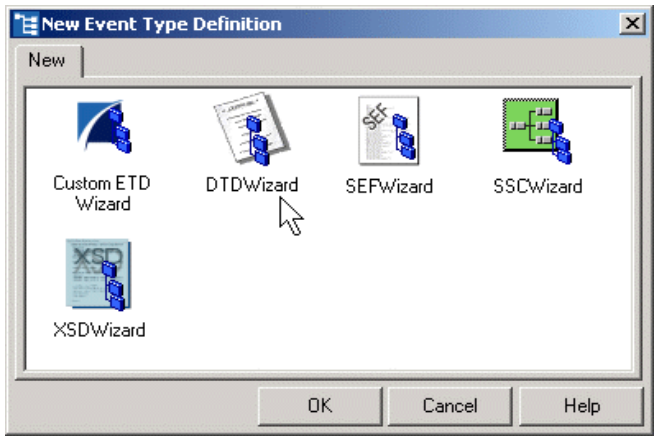
4 Double-click the new Event Type to edit its properties. The **Properties** window appears.

5 Click the **New** button. The ETD Editor appears.

6 Select **New** from the **File** menu on Task Manager. The **New Event Type Definition** window appears.

7 Select the **DTD Wizard** and click **OK** (see Figure 7).

Figure 7 Event Type Definition Wizard



8 Enter a package name where the DTD builder can place all the generated Java classes associated with the created ETD. (For this example, **etdEmailMessage.**)

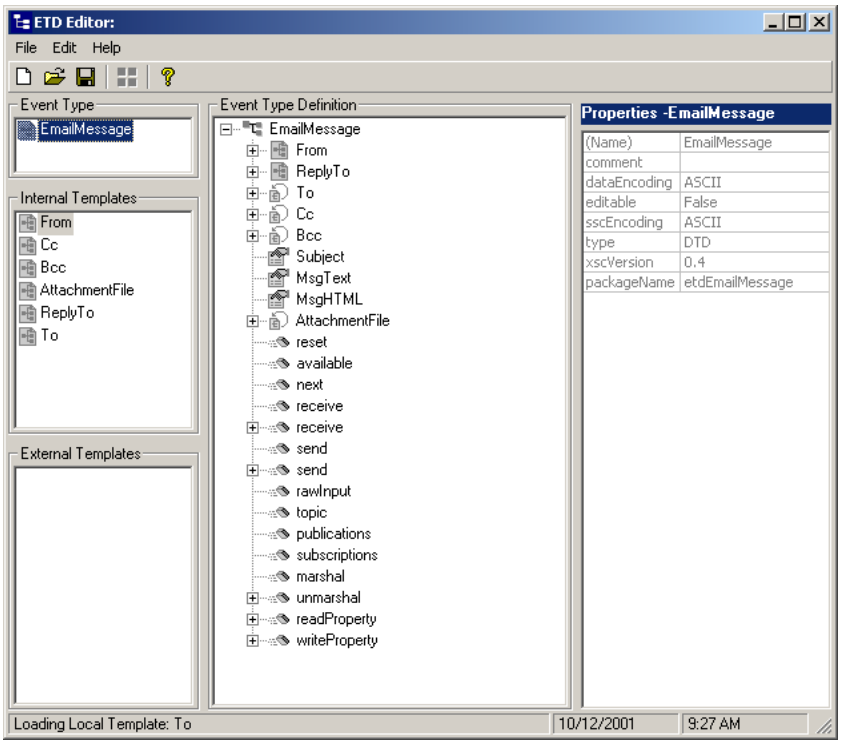
9 Select a DTD file to be used by the DTD builder to generate an ETD file. Using Browse, navigate to an existing DTD (for this example, use **EmailMessage.dtd**).

10 Enter a root node name (for this example, **EmailMessage**).

11 Click **Next**, review the summary information, and click **Finish**.

12 The editor displays the newly converted **.xsc** file. From the **File** menu, click **Compile and Save**. Once saved, the completed ETD is displayed (see Figure 8).

Figure 8 Event Type Editor



13 From the **File** menu, click **Promote to Run Time**.

14 Click **OK** to close the Event Type Properties dialog box.

15 Repeat steps 1-15 to create an additional Event Type, changing the name in step 3 to **et_output**.

Creating Event Types from an Existing XSC

The following procedures show how to create an Event Type Definition from an existing .xsc file using **mailclient.xsc**. The **mailclient.xsc** file is used when creating all Schemas.

1 Select the **Event Types** folder on the **Components** tab of the e*Gate Navigator.

2 On the palette, click the **Create a New Event Type** button.

3 Enter the name of the **Event Type** in the **New Event Type Component** window, then click **OK**. (For this sample, the Event Type is defined as **et_Send**.)

4 Double-click the new **Event Type** to edit its properties. The **Event Type Properties** dialog box appears.

5 Click the **Find** button under the **Event Type Definition** field.

6 Browse to and select **mailclient.xsc**.

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

8 Repeat steps 1-7 to create an additional Event Type changing the name in step 3 to **et_Rcv**.

Note: *For more information on the creating and modifying Java-enabled ETDs, see the "e*Gate Integrator User's Guide" or consult e*Gate's Online Help.*

5.3 Creating and Configuring the e*Ways

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

- Input e*Way
- Output e*Way
- Multi-Mode (send_Mail and rcv_Mail) e*Ways

These are the only e*Ways necessary for purposes of this schema. The following sections provide instructions for creating each e*Way.

Creating the Inbound e*Way

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

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

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

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

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

6 Right-click the new e*Way and select Properties. The e*Way Properties dialog box appears.

7 Click the **Find** button under the **Executable File** field, and select **stcewfile.exe** as the executable file.

8 Under the **Configuration File** field, click the **New** button. The **Edit Settings** dialog box appears. Change settings to match the following:

Table 2 Configuration Parameters for the Input e*Way

Parameter	Value
General Settings (unless otherwise stated, leave settings as default)	
AllowIncoming	YES
AllowOutgoing	NO
Outbound Settings	*Take default settings*
Poller Inbound Settings	
PollDirectory	*Directory to which your e-mail is sent*
InputFileMask	*Currently used extension*
RemoveEOL	YES
MultipleRecordsPerFile	YES
Performance Testing	*Take default settings*

9 Click **File**, **Save** and **File**, **Promote To Run Time** to save the .cfg file close the **Edit Settings** dialog box. This will save the .**cfg** file, promote it to the e*Gate runtime environment and close the Edit Settings dialog box.

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

Creating the Outbound e*Way

1 Repeat steps 1 through 7 above, changing the name in step 5 to **output**, to create the Output e*Way.

2 Under the **Configuration File** field, click the **New** button. The **Edit Settings** dialog box appears. Change settings to match the following:

Table 3 Configuration Parameters for the Output e*Way

Parameter	Value
General Settings (unless otherwise stated, leave settings as default)	
AllowIncoming	NO
AllowOutgoing	YES
Outbound Settings	

Table 3 Configuration Parameters for the Output e*Way

Parameter	Value
OutputDirectory	*Directory to which outbound e-mail is sent*
OutputFileName	*Preferred file name*
MultipleRecordsPerFile	NO
Poller Inbound Settings	*Take default settings*
Performance Testing	*Take default settings*

3 Save the .cfg file and Promote to Run Time.

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

Creating the Multi-Mode e*Ways

To create and configure the send_Mail Multi-Mode e*Way

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

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

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

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

5 Enter the name of the new Multi-Mode e*Way (in this case, **send_Mail**).

6 Right-click the new component and select **Properties**. The **e*Way Properties** window appears.

7 The **Executable File** field defaults to the **stceway.exe**.

8 Under the **Configuration File** field, click the **New** button. The **Edit Settings** window appears. See **"Multi-Mode e*Way Configuration" on page 29** for details on the parameters associated with the Multi-Mode e*Way, JVM Settings.

9 Go to **File**, **Save** to save the **.cfg** file. (For the sample save the file as **send_Mail.cfg**.)

10 Save the .cfg file and Promote to Run Time.

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

To create and configure the rcv_Mail Multi-Mode e*Way

Repeat steps 1-11 above, changing the name in step 5 and 9 to **rcv_Mail**, to create the rcv_Mail e*Way.

5.4 Collaboration Rules

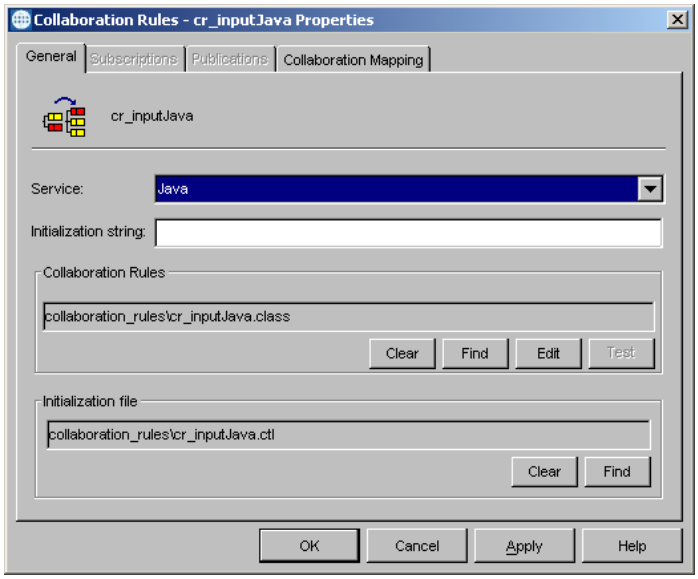
The next step is to create the Collaboration Rules that will extract selected information from the source Event Type defined earlier. The Collaboration Rules will then process the extracted information according to its associated Collaboration Service. The **Default**

Editor can be set to either **Monk** or **Java**. From the **Enterprise Manager Task Bar,** select **Options** and click on **Default Editor**. Set the Default Editor to **Java**.

5.4.1. Creating the cr_inputJava Collaboration Rules

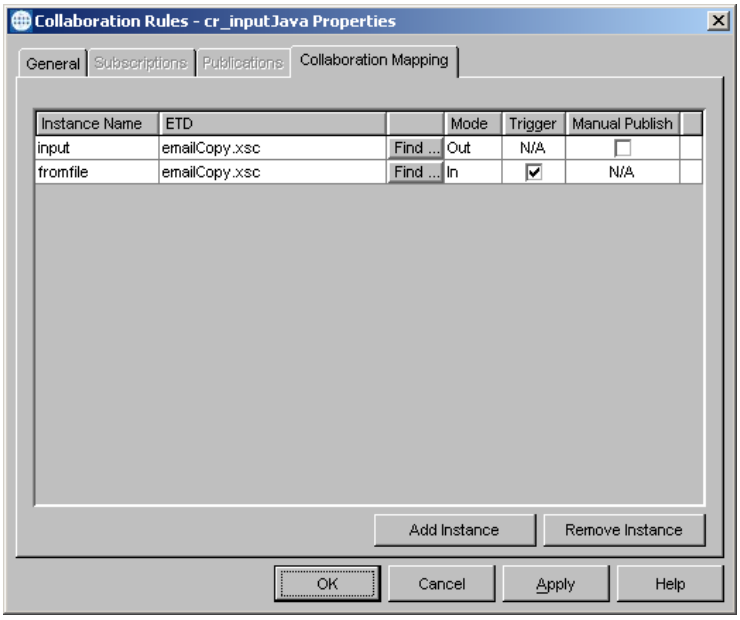
1. Select the Navigator's **Components** tab in the e*Gate Enterprise Manager.
2. In the **Navigator**, select the **Collaboration Rules** folder.
3. On the palette, click the **Create New Collaboration Rules** button.
4. Enter the name of the new Collaboration Rule (for this case, **cr_inputJava**).
5. Double-click the new Collaboration Rule to edit its properties. The **Collaboration Rules Properties** dialog box appears.
6. The Service defaults to **Java**. The Collaboration Rules will use the Java Service to manipulate Events or Event data. The **Subscriptions** and **Publications** tabs are disabled and the **Collaboration Mapping** tab is enabled.

Figure 9 Collaboration Rules Properties dialog box



7. In the **Initialization string** field, enter any required initialization string that the Collaboration Service may require. This field can be left blank.
8. Click on the **Collaboration Mapping** tab (see Figure 10).
9. Click on the **Add Instance** button to create instances that correspond to the event types. In the **Instance Name** column type **input** as the instance name.
10. Click the **Find** button and navigate to and select **emailCopy.xsc**. This adds **emailCopy.xsc** to the **ETD** column of the instance row.
11. In the **Mode** column, click the right side of the field to open a drop down list box. Select **Out** as the Mode.
12. Make sure the checkbox under the **Manual Publish** column is cleared.

Figure 10 Collaboration Rules Properties, Collaboration Mapping



13 Click on the **Add Instance** button again to create another instance.

14 In the **Instance Name** column, enter **fromfile** as the instance name.

15 For the ETD column, click the **Find** button, navigate to, and select **emailCopy.xsc,** to add **emailCopy.xsc** to the **ETD** column.

16 In the Mode column, click the right side of the field to open a drop down box. Select **In** from the drop–down menu.

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

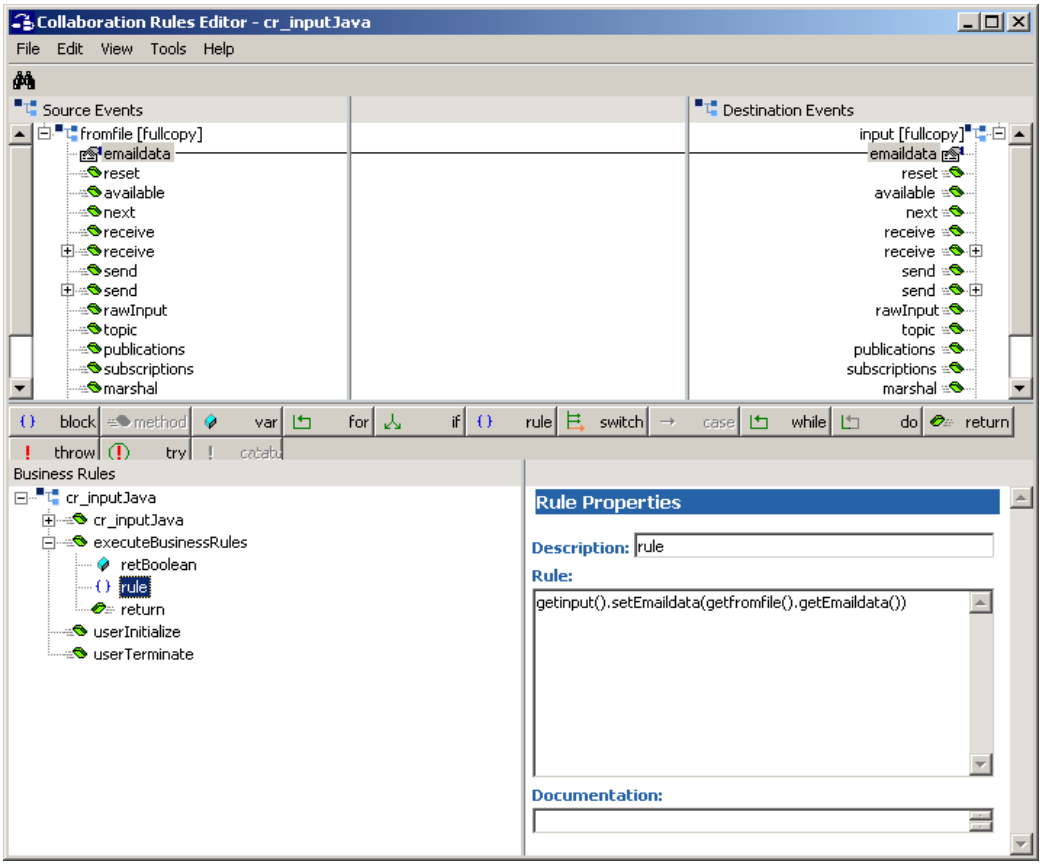
18 Click on the **General** tab. Click **New** under the Collaboration Rules Field. The Collaboration Rules Editor appears.

19 Expand the **Source Events** fields and the **Destination Events** fields to view components. In the **Business Rules** pane, select **retBoolean**.

20 Drag-and-drop **emaildata** from the Source Events pane onto **emaildata** under Destination Events pane. A line is displayed between the two components. A **rule** is added in the Business Rules pane under retBoolean and the Java code is displayed in the **Rule Properties** pane in the **Rules** field. The following code is created:

```
getinput().setEmaildata(getfromfile().getEmaildata())
```

Figure 11 Collaboration Rules Editor, cr_inputJava



21 From the **File** menu, click **Compile**. The Compile pane appears at the bottom of the Collaboration Rules Editor, displaying whether compile is successful or unsuccessful due to errors. If the compile is successful, save and promote the file, and exit the Collaboration Rules Editor.

Creating the cr_output_java Collaboration Rules

1 To create the **cr_output_java** Collaboration Rules, repeat steps 1-21 under **Creating the cr_inputJava Collaboration Rules** on page 35, with the following changes:

- Replacing the name in step 3 with **cr_output_java**.
- Replace the Instance Name in step 9 with **tofile**.
- Replace the second Instance Name in step 14 with **eater**.

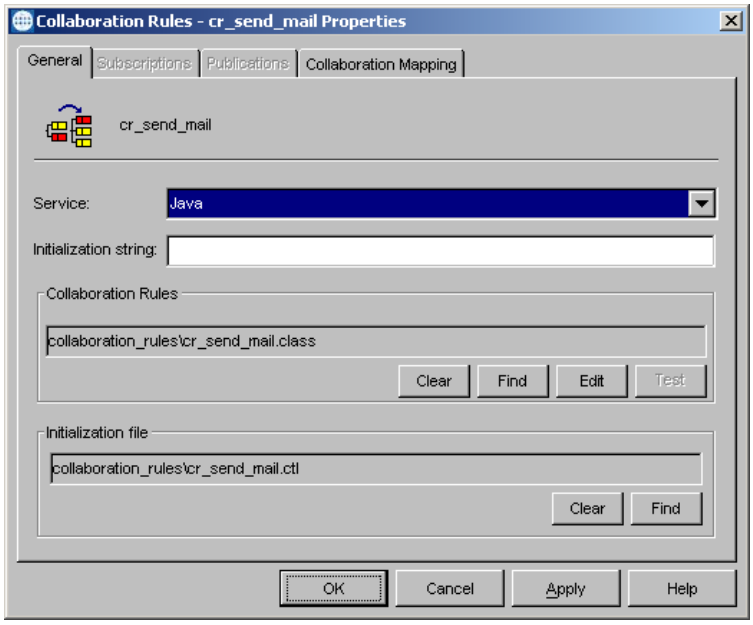
Creating the Java Collaboration Rules for cr_send_mail

1 On the e*Gate palette, click the **Create New Collaboration Rules** button.

2 Enter the name of the new Collaboration Rule, then click **OK** (for this sample, **cr_send_mail**).

3 Double-click the new Collaboration Rule to edit its properties. The **Collaboration Rules Properties** dialog box appears (see Figure 12).

Figure 12 Collaboration Rules Properties dialog box



4 In the Service field select **Java**. The Collaboration Rules use the Java Service to manipulate Events or Event data. The **Subscriptions** and **Publications** tabs are disabled and the **Collaboration Mapping** tab is enabled.

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

6 Click on the **Collaboration Mapping** tab (see Figure 13).

7 Click the **Add Instance** button to create instances that correspond to the event types. In the **Instance Name** column, enter **output** for the instance name.

8 Click the **Find** button and navigate to and select **mailclient.xsc**. **mailclient.xsc** is added to the **ETD** column of the instance row.

9 In the **Mode** column, click the right side of the field to open a drop down list box. Select **Out** as the Mode.

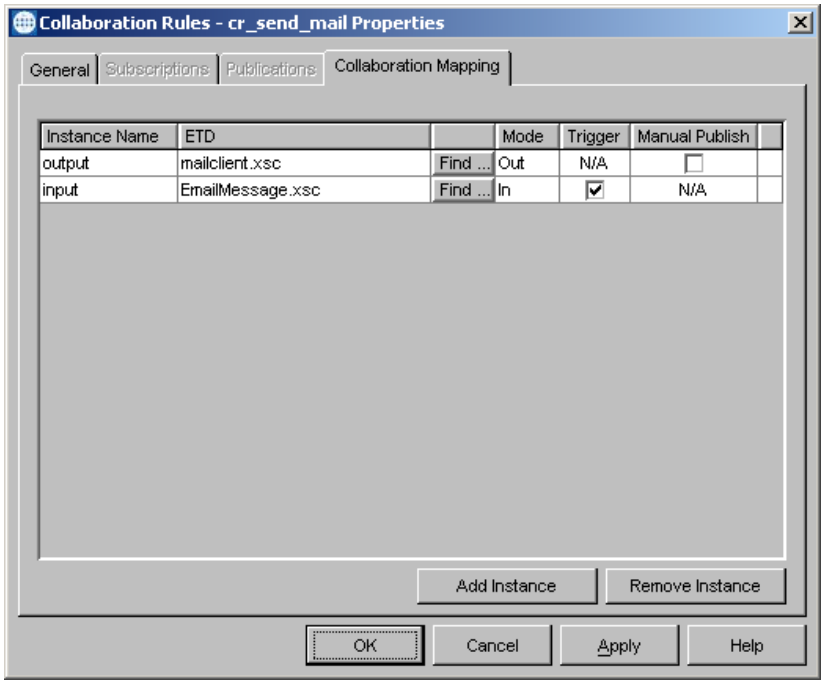
10 Make sure the checkbox under the **Manual Publish** column is cleared.

11 Click on the **Add Instance** button again to create another instance.

12 In the **Instance Name** column, enter **input** for the instance name.

13 Click the **Find** button and navigate to **EmailMessage.xsc**. Double-click **EmailMessage.xsc**. to add it to the **ETD** column of the instance row.

Figure 13 Collaboration Rules Properties, Collaboration Mapping



14 In the Mode column, click the right side of the field to open a drop down box. Select **In** from the drop–down menu.

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

16 Click on the **General** tab. Under the **Collaboration Rules** field click **Find** and navigate to **cr_send_mail.class**. This places **cr_send_mail.class** in the **Collaboration Rules** field and **cr_send_mail.ctl** in the **Initialization file** field

17 Click the **Edit** button under the **Collaboration Rules** field to open the **Collaboration Rules Editor** for **cr_send_mail** or click **OK** to close the Collaboration Rules Properties dialog box. See **Creating the cr_send_mail Collaboration Rules Class** on page 47 for information on using the Collaboration Rules Editor to create the business logic for the cr_send_mail Collaboration.

Creating the Java Collaboration Rules for cr_recv_mail

1 To create the cr_recv_mail Collaboration Rules, repeat steps 1-8 under **Creating the Java Collaboration Rules for cr_send_mail** on page 37, replacing the name in step 4 with **cr_recv_mail**.

2 Click on the **Add Instance** button to create instances that correspond to the event types. In the **Instance Name** column, enter **eater** for the instance name.

3 Click the **Find** button, navigate to and select **EmailMessage.xsc**. **EmailMessage.xsc** is added to the **ETD** column of the instance row.

4 In the **Mode** column, click the right side of the field to open a drop down list box. Select **Out** as the Mode.

5 Make sure the checkbox under the **Manual Publish** column is selected.

6 Click on the **Add Instance** button again to create another instance.

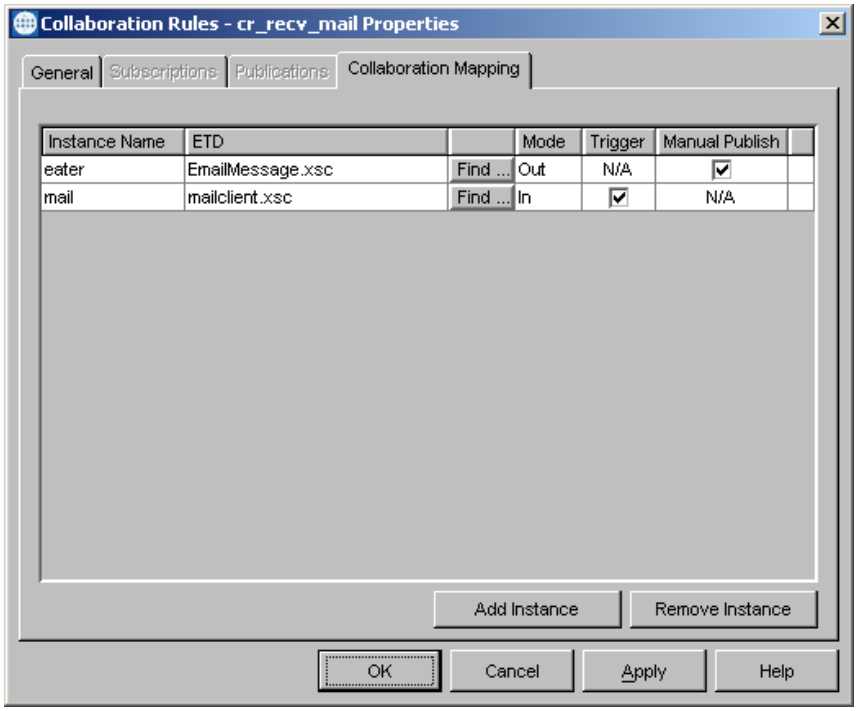
7 In the **Instance Name** column, enter **mail** for the instance name.

8 Click the **Find** button and navigate to **mailclient.xsc**. Double-click **mailclient.xsc**. to add it to the **ETD** column of the instance row.

9 In the Mode column, click the right side of the field to open a drop down box. Select **In** from the drop–down menu.

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

Figure 14 Collaboration Rules Properties, Collaboration Mapping



11 Click on the **General** tab. Click **New** under the Collaboration Rules field. This opens the **Collaboration Rules Editor** for **cr_recv_mail**.

5.4.2. Creating Business Rules using the Collaboration Rules Editor

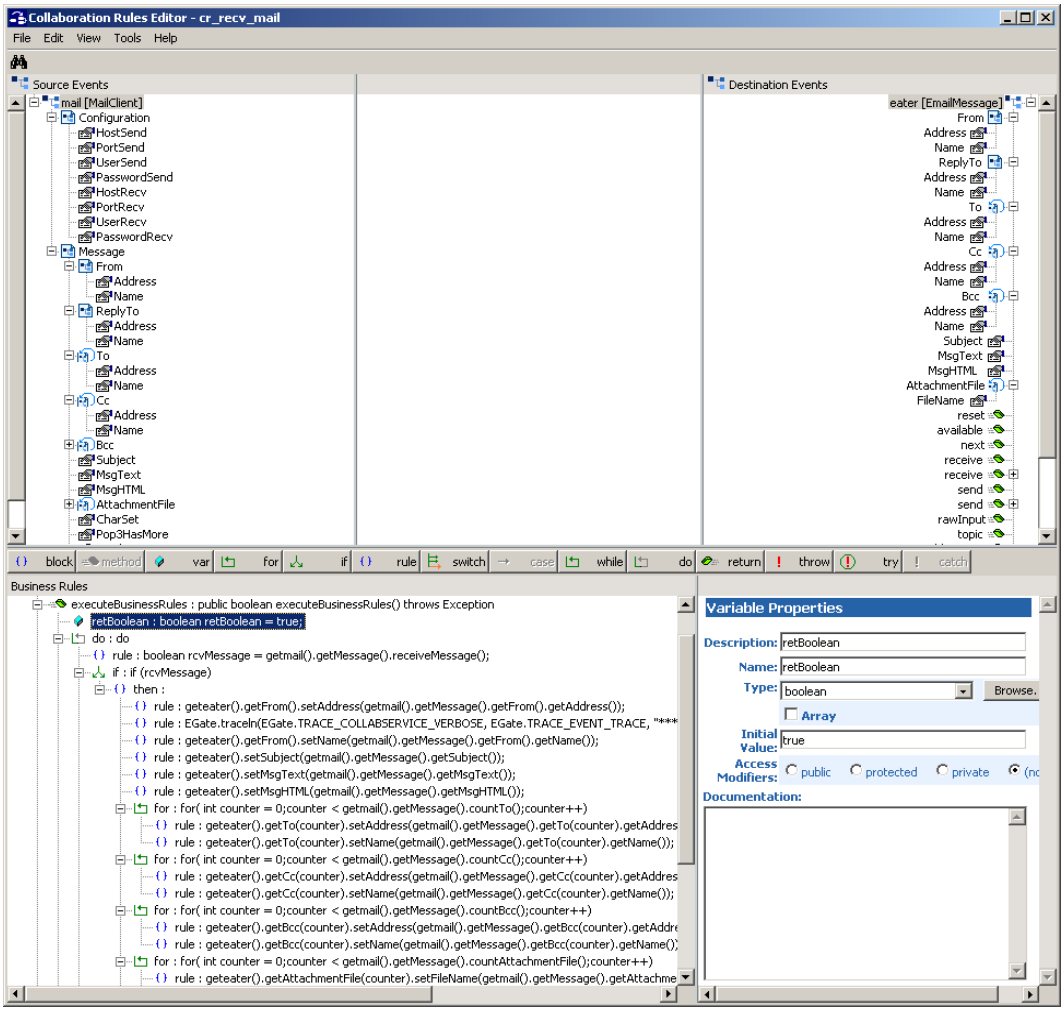
The business logic (see Figure 20) of the Collaboration is defined using the Java Collaboration Rules Editor (see Figure 15). A Collaboration Rule is created by designating one or more Source Events and one or more Destination Events and then setting up rules to govern the relationship between fields in the Event instances.

Each **rule** is created by clicking the **rule** button on the Business Rules toolbar or by "dragging and dropping" a node or method from the Source Events pane onto a node or method in the Destination Events pane. For more information on using the Java Collaboration Rules Editor, see the *e*Gate Integrator User's Guide.*

Creating the cr_recv_mail Collaboration Rules

1 To open the Collaboration Rules Editor for **cr_recv_mail**, open the Collaboration Rules Properties for **cr_recv_mail**. Click the **New** (new Collaboration) or **Edit** (edit an existing Collaboration) button under the Collaboration Rules field. The Collaboration Rules Editor appears.

Figure 15 Collaboration Rules Editor



2 Expand the Collaboration Editor it to full size for optimum viewing, expanding the Source and Destination Events as well.

3 In the Business Rules pane, select **retBoolean**. Click **do** on the Collaboration Rules Editor **toolbar**. A **do** expression appears under **retBoolean**. Select the **do** expression and add a **rule** (as a child).

Note: *For 4.5.0 a* ***for*** *loop should be used in place of the* ***do*** *expression so that they can exit after getting "x" number of messages, or use the following:*

```
Runtime mem = Runtime.getRuntime();
if (mem.freeMemory()>1000) {
      continue processing
```

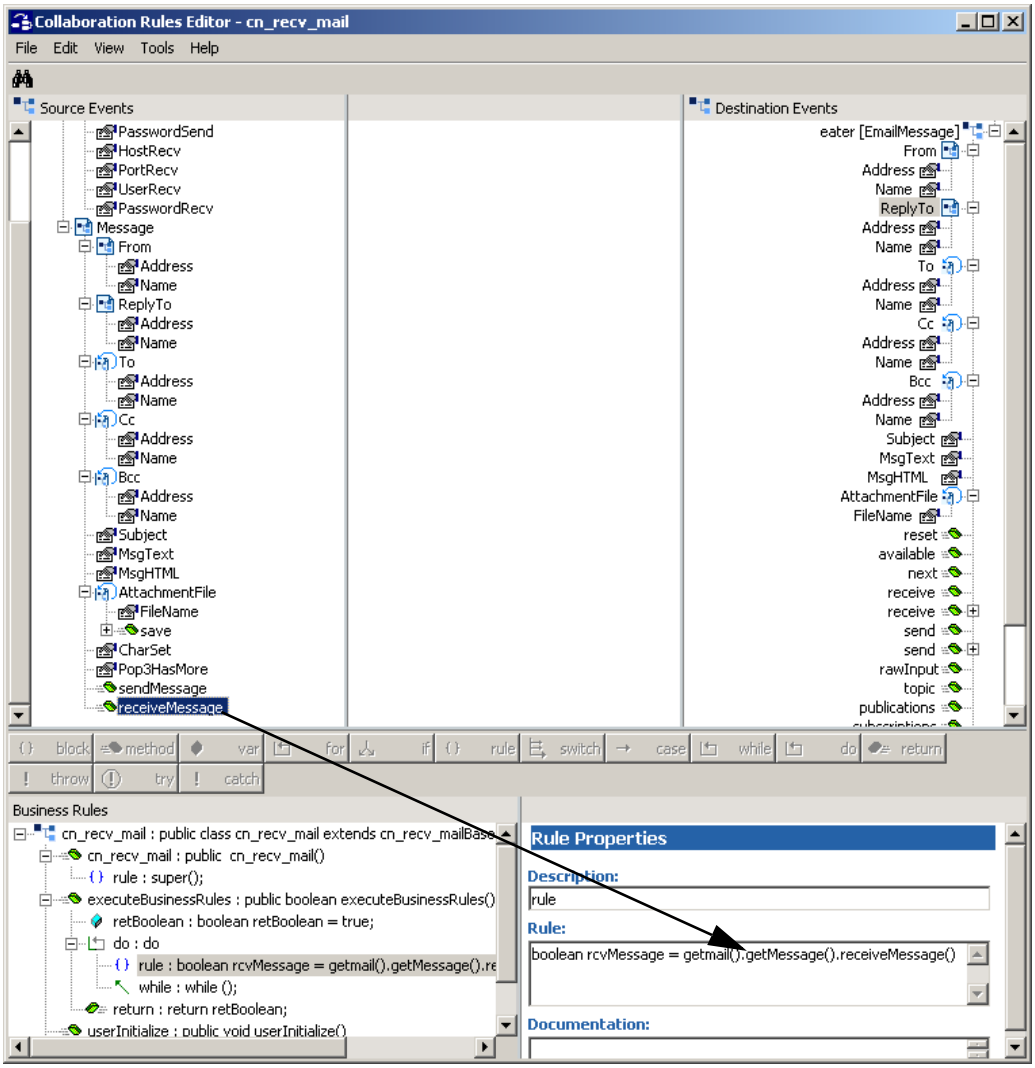
```
} else {
      exit the loop and collaboration
}
```

4 Select the new rule. In the Rule Properties, Rules field, type the following:

```
boolean rcvMessage =
```

Drag the **receiveMessage** method from the Source Events pane to the end of the entry in the **Rule Properties**, **Rules** field (see Figure 16).

Figure 16 Collaboration Rules Editor



5 To create the **if** expression, click **if** on the toolbar. An **if** expression is added under the **rule**.

6 Select **then** under the **if** expression. Drag-and-drop **Address** under **Message**, **From** from the Source Events pane to **Address** under **From** in the Destination Events pane. A new **rule** appears under the **then** expression with the following code:

```
geteater().getFrom().setAddress(getmail().getMessage().getFrom().getAddress())
```

Type **GetFromAddress** in the Rules Properties, Description field. This places a description (label) of the rule in the Business Rules window.

7 The next **rule** is created by typing the following in the Rule Properties, Rules window:

```
EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****Send Address "
+ (getmail().getMessage().getFrom().getAddress()));
```

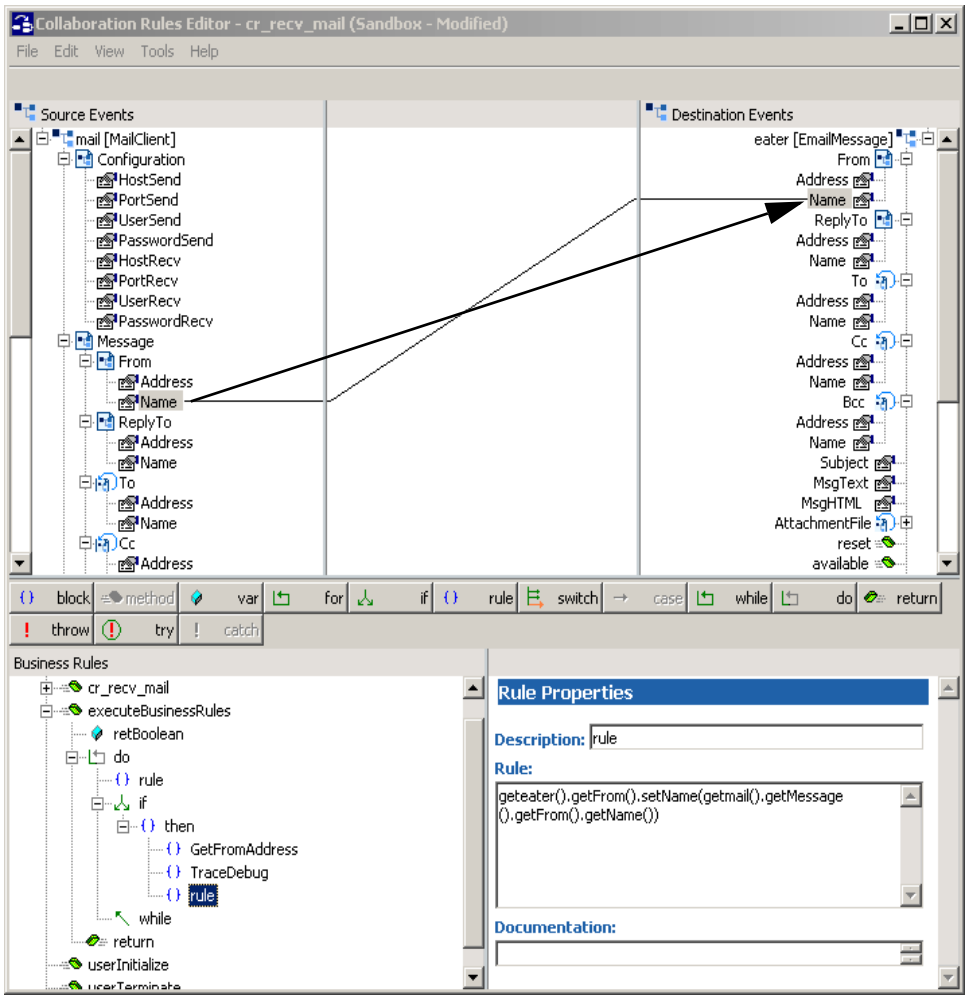
This is an available trace statement used for debugging. Type **TraceDebug** in the Description field.

8 To create the next rule, drag-and-drop **Name** under **Message**, **From** in the Source Events pane to **Name** under **eater**, **From** in the Destination Events pane (see Figure 17). A new rule appears in the Business Rules pane with the following code:

```
geteater().getFrom().setName(getmail().getMessage().getFrom().getName())
```

Type **GetNameFromSender** in the Description field to label the rule.

Figure 17 Collaboration Rules Editor



9 To create the next **rule**, drag-and-drop **Subject** under **Message** in the Source Events pane to **Subject** in the Destination Events pane. A new rule appears with the following code:

```
geteater().setSubject(getmail().getMessage().getSubject())
```

Type **SetTheSubject** in the Description field.

10 To create the next **rule**, drag-and-drop **MsgText** under **Message** in the Source Events pane to **MsgText** in the Destination Events pane. A new rule appears with the following code:

```
geteater().setMsgText(getmail().getMessage().getMsgText())
```

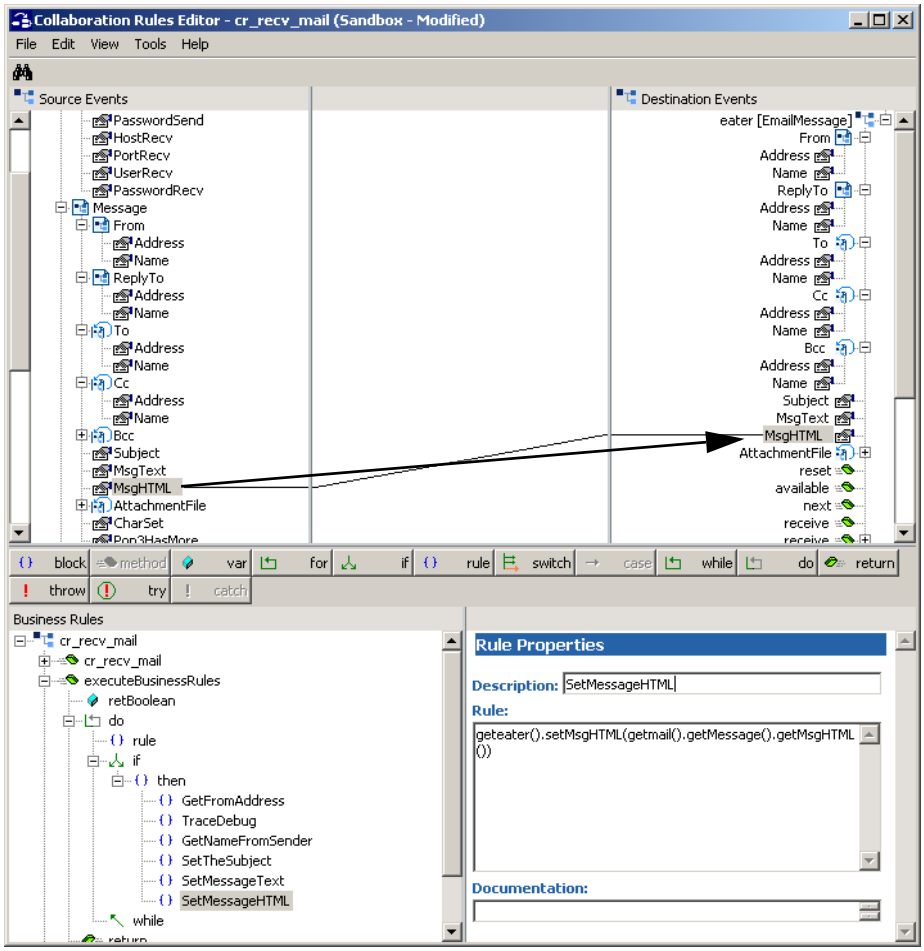
Type **SetMessageText** in the Description field.

11 To create the next **rule**, drag-and-drop **MsgHTML** under **Message** in the Source Events pane to **MsgHTML** in the Destination Events pane. A new rule appears with the following code:

```
geteater().setMsgHTML(getmail().getMessage().getMsgHTML())
```

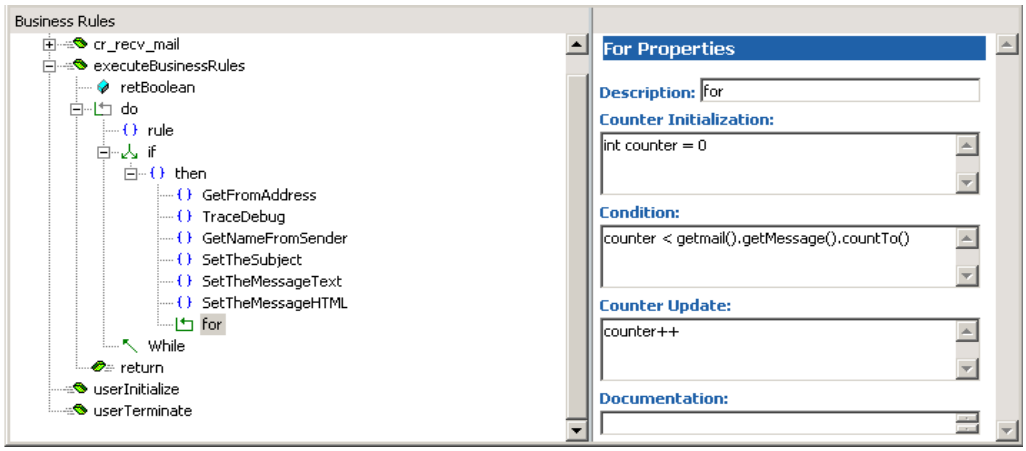
Type **SetMessageHTML** in the Description field (see Figure 18).

Figure 18 Collaboration Rules Editor



12 To create the first **for** loop, click on the **for** button on the toolbar. Select the new **for**. The For Properties pane now has fields for **Counter Initialization**, **Condition**, **Counter Update** and **Documentation**. Enter "**int counter = 0**" in the **Counter Initialization** field, "**counter < getinput().countTo()**" in the **Condition** field, and "**counter++**" in the **Counter Update** filed (see Figure 19).

Figure 19 Collaboration Rules Editor, Business Rules



13 To create the first **rule** under the first **for** loop, drag-and-drop **Address** under **Message, To** in the Source Events pane to **Address, To** in the Destination Events pane. A dialog box appears prompting for Source Node Repetition and Destination Node Repetition. Type **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **child**. A new rule is added under the **for** loop.

14 To create the next **rule**, drag-and-drop **Name** under **Message**, **To** in the Source Events pane to **Name** under **To** in the Destination Events pane. When prompted for Source Node Repetition and Destination Node Repetition. Type **counter** as the value for both and click **OK**.

15 Click on the **for** button again to create a second **for** loop. In the **for** Properties fields enter **int counter = 0** in the **Counter Initialization** field, **counter < getmail().getMessage().countCc()** in the **Condition** field, and **counter++** in the **Counter Update** field.

16 Select the second **for** loop. Drag-and-drop **Address** under **Message**, **Cc** in the Source Events pane to **Address** under **Cc** in the Destination Events pane. A dialog box appears, prompting for Source Node Repetition and Destination Node Repetition. Type **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **Child**. A new rule is added under the **for** loop.

17 To create the next **rule** drag-and-drop **Name** under **Message**, **Cc** in the Source Events pane to **Name** under **Cc** in the Destination Events pane. A dialog box appears, prompting for Source Node Repetition and Destination Node Repetition. Type **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **Child**.

18 Click on the **for** button again to create a third **for** loop. In the **for** Properties fields enter **int counter = 0** in the **Counter Initialization** field, **counter < getmail().getMessage().countBcc()** in the **Condition** field, and **counter++** in the **Counter Update** field.

19 To create the next **rule**, drag-and-drop **Address** under **Message**, **Bcc** in the Source Events pane to **Address** under **Bcc** in the Destination Events pane. When prompted

for Source Node Repetition and Destination Node Repetition enter **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **child**.

20 Create the next **rule** by dragging-and-dropping **Name** under **Message**, **Bcc** in the Source Events pane to **Name** under **Bcc** in the Destination Events pane. When prompted for Source Node Repetition and Destination Node Repetition enter **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **child**.

21 Create the last **for** loop. In the Properties fields enter **int counter = 0** in the **Counter Initialization** field, **counter < getmail().getMessage().countAttachmentFile()** in the **Condition** field, and **counter++** in the **Counter Update** field.

22 Create the next **rule** by dragging-and-dropping **FileName** under **AttachmentFile** in the Source Events pane to **FileName** under **AttachmentFile** in the Destination Events pane. When prompted for Source Node Repetition and Destination Node Repetition enter **counter** as the value for both and click **OK**. When prompted to add the **rule** as a **sibling** or a **child**, select **child**.

23 To create the next **rule**, Type **String value =** in the Rule Properties, Rule field. Drag **FileName** under **AttachmentFile** in the Source Events pane into the Rule field and type in **counter** as the value when prompted to **Select Repetition Instance**, to create the following code:

```
String value = getmail().getMessage().getAttachmentFile(counter).getFileName()
```

24 For the next **rule**, enter the following in the Rule Properties, Rule field:

```
value = value + counter
```

25 To create the next **rule,** drag-and-drop the **save** method under **AttachmentFile** in the Source Events pane to the Rule Properties, Rule field and type in **counter** as the value when prompted to **Select Repetition Instance**. When prompted for parameters for the method, enter **"C:/eGate/data/email/attach"** for **path**, and **value** for **name**.

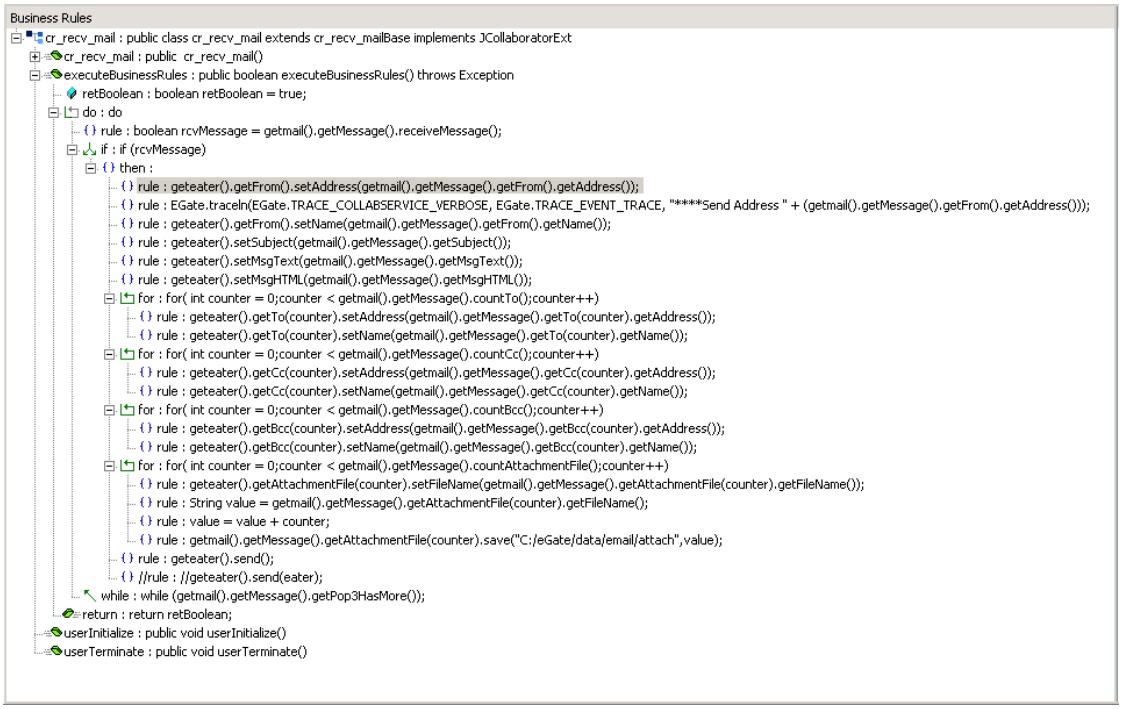
26 To create the next **rule**, select the **then** expression and click the **rule** button. A new **rule** expression appears as a sibling below the **for** loops. Drag the first **send** method under **eater** in the Destination Events pane to the Rule Properties, Rule field.

27 The last **rule** has been commented out for this sample. It can be used by removing the two slashes located just before "geteater().send(eater)". It's created by dragging second **send** method under **eater** in the Destination Events pane to the Rule Properties, Rule field. When prompted for a parameter, enter **eater**.

28 Select the **while** expression and drag-and-drop **Pop3HasMore** into the While Properties, Condition field.

29 When satisfied with the Collaboration Rules (see Figure 20), from the editor's **File** menu, click **Compile.** The Compile pane appears at the bottom of the Collaboration Rules Editor and displays **Compile Complete** or lists compile errors.

Figure 20 Business Rules - Receiving Collaboration



30 When the compile completes successfully, from the editor's **File** menu, click **Save** and click **Promote** to promote the file to the run-time environment.

Creating the cr_send_mail Collaboration Rules Class

To open the Collaboration Rules Editor for **cr_send_mail**, from cr_send_mail Properties click **New** (for a new Collaboration) or **Edit** (for an existing Collaboration) under the Collaboration Rules field.

Each new rule is created by clicking the **rule** button on the Business Rules toolbar. For additional information on using the Java Collaboration Rules Editor, see the *e*Gate Integrator User's Guide*.

Note: *It is necessary, when sending messages, to create a new object ("EmailAddress"), and instantiate that object for each of the addresses - To, Cc, and Bcc. In addition, the object "EmailAttachmentFile", must be created and instantiated for email attachments. See the example in* **Figure 22** *. This is necessary only for sending messages.*

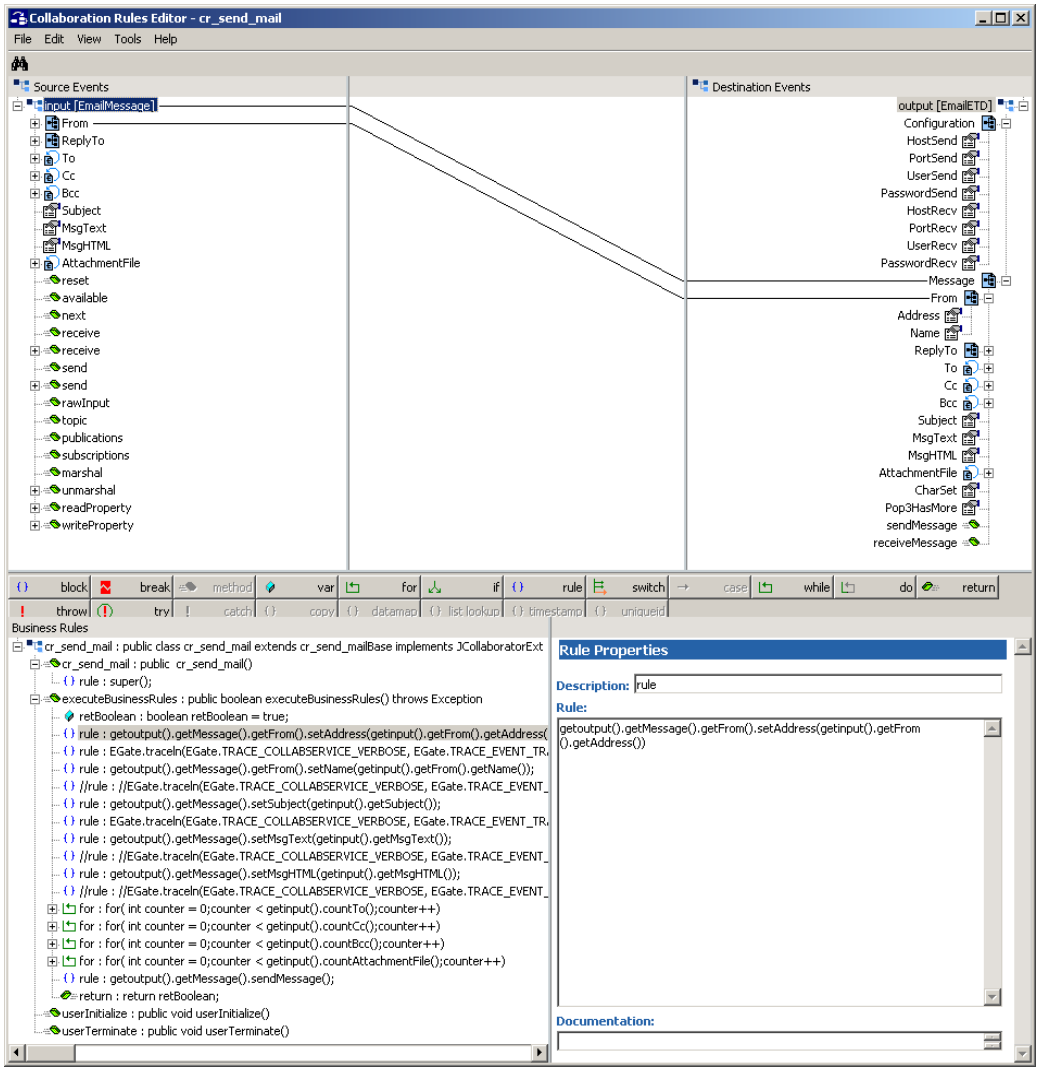
1 The first **rule** is created by dragging **Address** located under input\From (Source Events) and dropping it on **Address** located under output\Message\From (Destination Events).

2 The second **rule** is trace used for debugging. It's created by typing the following in the Rule Properties, Rule field:

```
EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****Send Address "
+ ());
```

Drag **Address** located under input\From (Source Events) into the set of parentheses.

Figure 21 Collaboration Rules Editor - cr_send_mail



3 The third **rule** is created by dragging **Name** located under input\From (Source Events) and dropping it on **Name** located under output\Message\From (Destination Events).

4 The forth **rule,** another trace statement that has been commented out, can be used by removing the two slashes located just before "EGate.traceIn". It's created by typing the following in the Rule Properties, Rule field:

```
EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****Send Address "
+ ());
```

Drag **Name** located under input\From (Source Events) into the set of parentheses.

5 The fifth **rule** is created by dragging **Subject** located under input (Source Events) and dropping it on **Subject** located under output\Message (Destination Events).

6 The sixth **rule** is another available trace statement created by typing the following in the Rule Properties, Rule field:

```
EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****Subject " +
(getinput().getSubject() + Calendar.getInstance().getTime()));
```

7 The seventh **rule** is created by dragging **MsgText** located under input (Source Events) and dropping it on **MsgText** located under output\Message (Destination Events).

8 The eighth **rule** is another available trace statement that has been commented out for this sample. It can be used by removing the two slashes located just before "EGate.traceIn". It's created by typing the following in the Rule Properties, Rule field:

```
//EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE,"****MsgText " +
(getinput().getMsgText()));
```

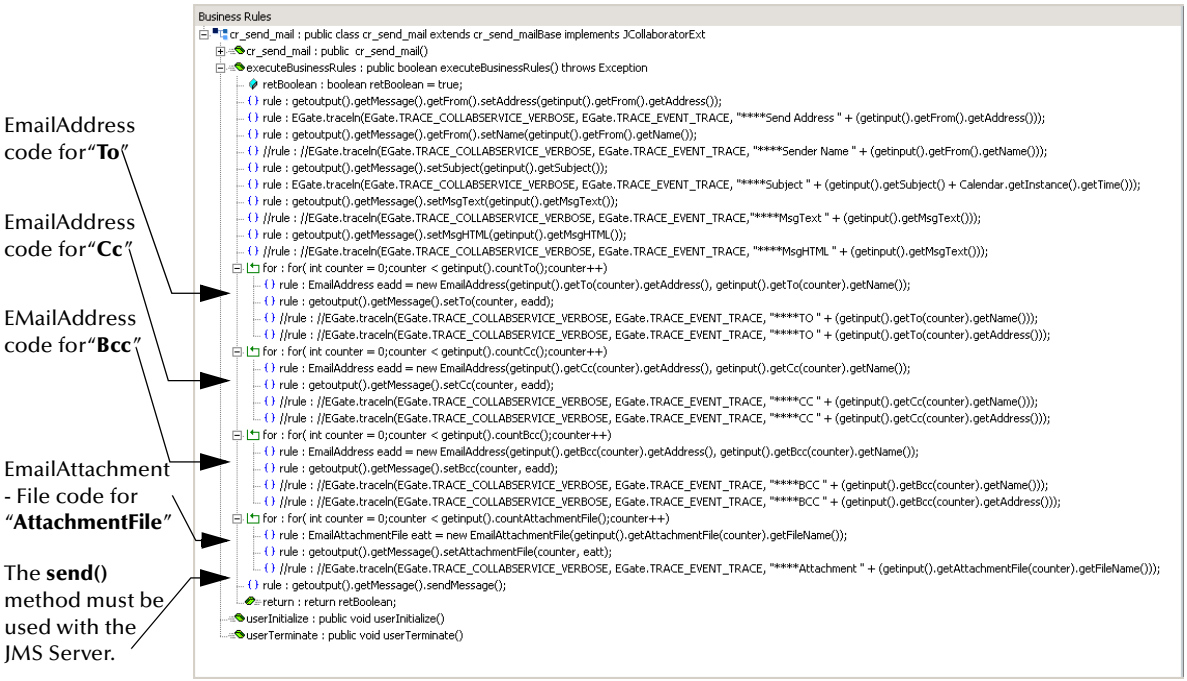
9 The ninth **rule** is created by dragging **MsgHTML** located under input (Source Events) and dropping it on **MsgHTML** located under output\Message (Destination Events).

10 The tenth **rule** is another available trace statement that has been commented out for this sample. It can be used by removing the two slashes located just before "EGate.traceIn". It's created by typing the following in the Rule Properties, Rule field:

```
//EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****MsgHTML " +
(getinput().getMsgText()));
```

Important: *When sending messages, the user must create a new object ("**EmailAddress**"), and instantiate that object for each of the addresses - **To**, **Cc**, and **Bcc**. In addition, the user must create the object ("**EmailAttachmentFile**"), and instantiate that object for email attachments (see Figure 22). This is necessary only for sending messages.*

Figure 22 Creating EmailAddress and EmailAttachment Objects



11 A **for** loop is created by clicking on the **for** button on the Business Rules toolbar. Click on the **for** button to create the first of **for** loops.

12 The first **rule** under the first **for** loop is created by typing the following in the Rule Properties, Rule field:

```
EmailAddress eadd = new EmailAddress()
```

Complete the **rule** by dragging **Address** under input\To (Source Events) and dropping it the parentheses. Do this also with **Name** under input\To (Source Events) to create the following code:

```
EmailAddress eadd = new EmailAddress(getinput().getTo(counter).getAddress(),
getinput().getTo(counter).getName())
```

13 Create the second **rule** under the first **for** loop by dragging **To** located under output\Message (Destination Events) and dropping it in the Rule Properties, Rule field. The **Select Repetition Instance** dialog box appears. Enter **counter** as the value and click **OK**. Place the cursor in the last parentheses and type in **eadd** to create the following code:

```
getoutput().getMessage().setTo(counter, eadd)
```

14 The third **rule** under the first **for** loop is another available trace statement that has been commented out for this sample. It can be used by removing the two slashes located just before "EGate.traceIn". It's created by typing the following in the Rule Properties, Rule field:

```
//EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****TO " +
(getinput().getTo(counter).getName()));
```

15 The forth **rule** under the first **for** loop is another available trace statement created by entering the following:

```
//EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****TO " +
(getinput().getTo(counter).getAddress()));
```

16 The second and third **for** loops and their underlying **rules** are created similar to the first using steps 11-15, replacing To with Cc or Bcc when applicable (see **Figure 22** for the correct code).

17 Create the forth **for** loop.

18 Create the first **rule** under the forth **for** loop by entering the following:

```
EmailAttachmentFile eatt = new EmailAttachmentFile()
```

Drag and drop **FileName** under input\AttachmentFile (Source Events) into the last parentheses. The **Select Repetition Instance** dialog box appears. Enter **counter** as the value and click **OK**. The resulting code should appear as follows:

```
EmailAttachmentFile eatt = new
EmailAttachmentFile(getinput().getAttachmentFile(counter).getFileName())
```

19 Create the second **rule** under the forth **for** loop by dragging **AttachmentFile** located under output\Message (Destination Events) and dropping it in the Rule Properties, Rule field. The **Select Repetition Instance** dialog box appears. Enter **counter** as the value and click **OK**. Place the cursor in the last parentheses and type in **eatt** to create the following code:

```
getoutput().getMessage().setAttachmentFile(counter, eatt)
```

20 The third **rule** under the forth **for** loop is another available trace statement created by entering the following:

```
//EGate.traceln(EGate.TRACE_COLLABSERVICE_VERBOSE, EGate.TRACE_EVENT_TRACE, "****Attachment "
+ (getinput().getAttachmentFile(counter).getFileName()));
```

21 The last **rule** is created by dragging **sendMessage()** under output\Message (Destination Events) and dropping it in the Rule Properties, Rule field.

22 When satisfied that the Collaboration Rules are complete, from the editor's **File** menu, click **Compile**. When the compile completes successfully, from the editor's **File** menu, click **Save** and **Promote** to promote the file to the run-time environment.

5.5 Enabling Japanese and Korean Character Support

The Java-enabled e-Mail e*Way supports Korean (Hangul) and Japanese character encoding in both the header and message content of the e-mail message (for both text/plain and text/HTML). File attachment names are converted to "B" encoding. The e-Mail e*Way conforms to RFC2047 standards for Multipurpose Internet Mail Extensions (MIME).

In order for email to be in encoded to support Korean character text (ISO-2022-KR) or Japanese character text (ISO-2022-JP), the value for **CharSet** must be set to one of the following:

- **ISO-2022-KR** for Korean character support.
- **ISO-2022-JP** for Japanese character support.

The **CharSet** property is only valid for sending messages and is only available in mailclient.xsc. To enable Korean or Japanese character support to **send** messages do the following:

1 Open the Collaboration Rules Editor for the *send_mail*.class file (in this case, **cr_send_mail.class**).

2 Create a new **rule** under **retBoolean** in the Business Rules pane. Select the new rule.

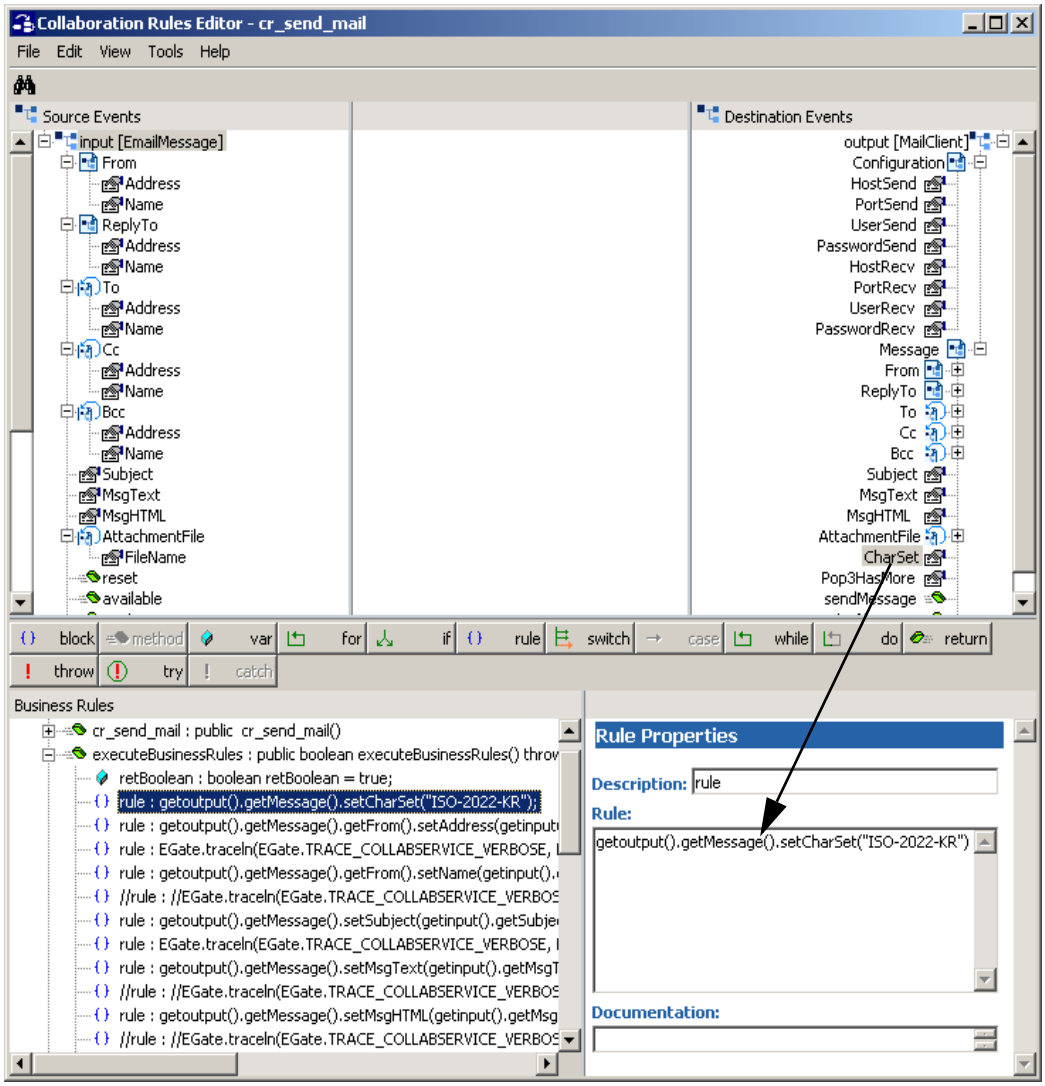
3 Drag-and-drop **CharSet** from the Destination Events pane into the Rule Properties, Rule field to create the following code:

```
getoutput().getMessage().setCharSet();
```

4 Type the value (**"ISO-2022-KR"** or **"ISO-2022-JP"**, including the quotation marks) into the last set of parentheses (see Figure 23).

5 Add a description for the rule (for example “Korean character support”) by typing one in the Rule Properties, Description field.

Figure 23 Collaboration Rules Editor - Korean Character Support



6 Compile, save, and promote the file to run-time.

5.6 Creating the JMS Queue Server (IQ Manager)

The next step is to create and associate the JMS Queue Server (IQ Manager) for the e-Mail e*Way. The IQ Manager the governs the exchange of information between components within the e*Gate system, providing non-volatile storage for data as it passes from one component to another.

To create and modify the JMS Queue Server for the e-Mail e*Way

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

2 Open the host on which the **JMS Queue Server** is to be created.

3 Select the **Control Broker** and click the **Create a New IQ Manager** button. Enter a name for the IQ Manager (for this example, **JMS_Queue_Server**).

4 Right-click the **IQ Manager** and select **Properties**. The IQ Manager Properties dialog box appears.

5 Select **SeeBeyond JMS** as the IQ Manager Type.

6 Click **New** under the Configuration File field. The **Edit Settings** dialog box appears.

7 From the **File** menu click **Save** (saving all the default settings). Save the file as **JMS_Queue_Server.cfg**.

8 From the **File** menu, click **Promote to Run Time**.

9 On the Start Up tab select **Start automatically**.

10 Click **OK** to close the IQ Manager dialog box.

5.7 Creating e*Way Connections

This schema uses two **Email** type e*Way Connections and four **JMS** e*Way Connections to define the External Systems used for communication.

Important: *Having multiple connection points accessing the same e-mail account will result in receiving the same e-mail message from more than one connection point. Therefore to avoid duplicates, only one Java E-mail e*Way connection point should be receiving from each e-mail account.*

To create and configure cp_rcvEmail e*Way Connection

1 In the Enterprise Manager's **Component** editor, select the **e*Way Connections** folder.

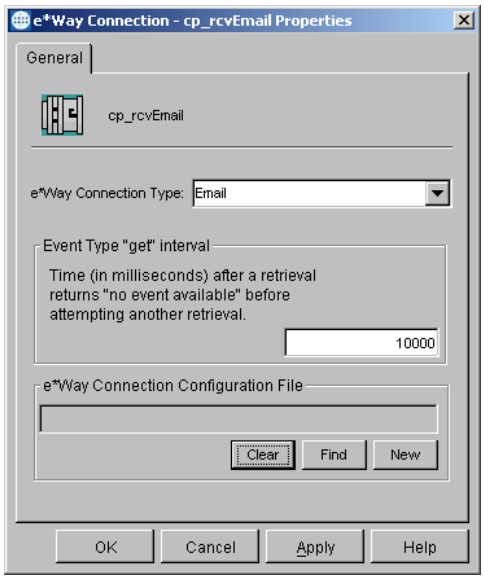
2 On the Palette, click the **Create a New e*Way Connection** button.

3 The **New e*Way Connection Component** dialog box appears. Enter a name for the e*Way Connection. (For this case, **cp_rcvEmail**.) Click **OK**.

4 Double-click the new **e*Way Connection**. The e*Way Connection Properties dialog box appears (see Figure 24). Select **Email** as the e*Way Connection Type.

5 Enter the **Event Type "get" interval** in the dialog box provided. The configured default for the "get interval is 10000 milliseconds.

Figure 24 e*Way Connection Properties Dialog Box



6 From the **e*Way Connection Configuration File**, click **New** to create a new Configuration File for this e*Way Connection.

7 The e*Way Connection editor appears, select the necessary parameters for a "**receiving**" Email e*Way connection. For more information on the **Email** e*Way Connection Type parameters, see **"e-Mail e*Way Connections" on page 17**.

8 From the **File** menu, click **Save** and **Promote to Run Time**.

To create and configure cp_sendEmail e*Way Connection

1 Repeat steps 1-6 above changing the name in step 3 to **cp_sendEmail**.

2 The e*Way Connection editor appears, select the necessary parameters for a "**sending**" Email e*Way connection. For more information on the **Email** e*Way Connection Type parameters, see "e-Mail e*Way Connections" on page 17.

3 From the **File** menu, click **Save** and **Promote to Run Time**.

To create and configure JMS e*Way Connections

1 In the Enterprise Manager's **Component** editor, select the **e*Way Connections** folder.

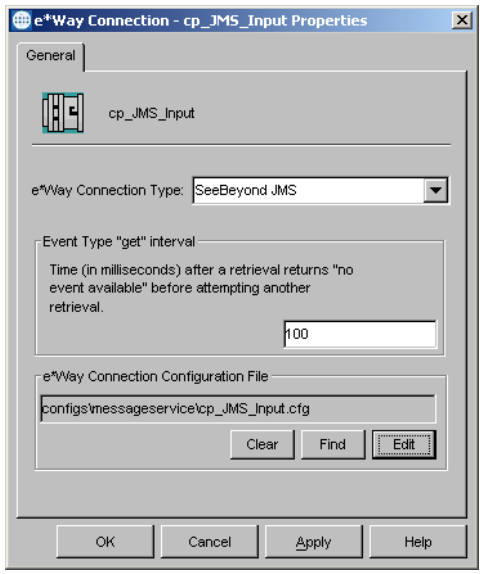
2 On the Palette, click the **Create a New e*Way Connection** button.

3 The **New e*Way Connection Component** dialog box appears. Enter a name for the e*Way Connection. (For this case, **JMS_Input**.) Click **OK**.

4 Double-click on the new **e*Way Connection**. The **e*Way Connection Properties** dialog box appears (see Figure 25). From the **e*Way Connection Type** drop-down box, select **SeeBeyond JMS**.

5 Enter the **Event Type "get" interval** in the dialog box provided. The configured default for the "get interval is 100 milliseconds.

Figure 25 JMS e*Way Connection



6 From the **e*Way Connection Configuration File**, click **New** to create a new Configuration File for this e*Way Connection.

7 The e*Way Connection editor appears, select the necessary parameters for a **SeeBeyond JMS** e*Way connection. For more information on the SeeBeyond JMS e*Way Connection Type parameters, see **"SeeBeyond JMS e*Way Connections" on page 22**.

8 From the **File** menu, click **Save** and **Promote to Run Time**.

9 Repeat steps 1-8 above three more times, replacing the name in step 3 with **JMS_Output**, **JMS_fmEmail**, and **JMS_toEmail**, to create the other 3 required JMS e*Way connections.

Note: *For more information about the e*Way Connection parameters, see* **"e*Way Connection Configuration" on page 17**.

5.8 Collaborations

Collaborations are the components that receive and process Event Types, then forward the output to other e*Gate components or an external. Collaborations consist of the Subscriber, which "listens" for Events of a known type (sometimes from a given source), and the Publisher, which distributes the transformed Event to a specified recipient. Four Collaborations (one for each e*Way) will be created for this schema.

Creating the Input Collaboration

1 In the e*Gate Enterprise Manager, select the Navigator's **Components** tab.

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

3 Select a **Control Broker.**

4 Select the **Input** e*Way.

5 On the palette, click the **Create a New Collaboration** button.

6 Enter the name of the new Collaboration, then click **OK** (for this example, **coll_input**).

7 Double-click the new Collaboration to edit its properties. The **Collaboration Properties** dialog box appears.

8 From the **Collaboration Rules** drop-down list box, select the Collaboration Rules file that you created earlier (for this example, **cr_inputJava**).

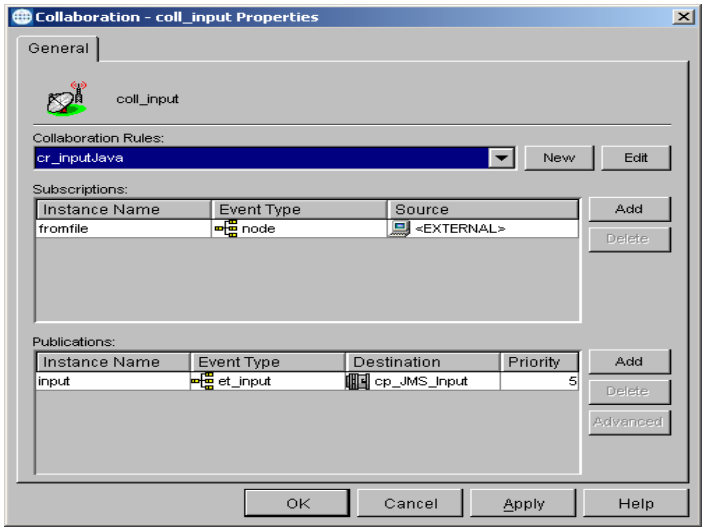
9 In the **Subscriptions** area, click **Add** to define the Instance, Event Type and source to which this Collaboration will subscribe.

- A Under the Instance Name in the Subscriptions field, select **fromfile**.
- B From the Event Type list, select the Event Type that you previously defined as **node**.
- C From Source list, select the subscription source (for this example, **<EXTERNAL>**).

10 In the **Publications** area, click **Add** to define the Instance, Event Type and destination.

- A Under the Instance Name in the Subscriptions field, select **input**.
- B From the Event Type list, select the Event Type that you previously defined as **et_input**.
- C From Destination list, select the subscription source (for this example, **cp_JMS_Input**).
- D The Priority field defaults to **5**.

Figure 26 Collaboration Properties



11 Click on **OK** to close the **Collaboration Properties** dialog box.

Creating the Output Collaboration

1. Select the **output** e*Way to which the Collaboration will be associated.
2. On the palette, click the **Create a New Collaboration** button.
3. Enter the name of the new Collaboration, then click **OK** (for this example, **collab_output**).
4. Double-click the new Collaboration to edit its properties. The **Collaboration Properties** dialog box appears.
5. From the **Collaboration Rules** drop-down list box, select the Collaboration Rules file that you created earlier (for this example, **cr_output_java**).
6. In the **Subscriptions** area, click **Add** to define the Instance, Event Type and source to which this Collaboration will subscribe.
   A. Under the Instance Name in the Subscriptions field, select **eater**.
   B. From the Event Type list, select the Event Type that you previously defined as **et_output**.
   C. From Source list, select the subscription source (for this example, **cp_JMS_Output**).
7. In the **Publications** area, click **Add** to define the Instance, Event Types and destination to which this Collaboration will publish.
   A. Under the Instance Name in the Publications field, select **eater**.
   B. From the **Event Type** list, select the Event Type that you previously defined as **et_output**.
   C. From **Destination** list, Select the destination (for this example, **<EXTERNAL>**).
   D. The setting for **Priority** will default to **5**.
8. Click on **OK** to close the **Collaboration Properties** dialog box.

Creating the Receive Collaboration

1. Select the **rcv_Mail** Multi-Mode e*Way.
2. On the palette, click the **Create a New Collaboration** button.
3. Enter the name of the new Collaboration, then click **OK** (for this example, **collab_rcv**).
4. Double-click the new Collaboration to edit its properties. The **Collaboration Properties** dialog box appears.
5. From the **Collaboration Rules** drop-down list box, select the Collaboration Rules file that you created earlier (for this example, **cr_recv_mail**).
6. In the **Subscriptions** area, click **Add** to define the Instances and input Event Types to which this Collaboration will subscribe.
   A. In the **Instance Name** box type **mail**.

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

C From **Source** list, select the subscription source (for this example, **cp_rcvEmail**).

7 In the **Publications** area, click **Add** to define the Instances and Event Types that this Collaboration will publish.

A In the **Instance Name** field, select the name of this instance (in this example, **eater**).

B From the **Event Type** list, select the outbound Event Type that you previously defined as **et_output**.

C Select the publication destination from the **Destination** list (for this example, **cp_JMS_fmEmail**).

D The setting for **Priority** will default to **5**.

8 Click on **OK** to close the **Collaboration Properties** dialog box.

Creating the Send Collaboration

1 Select the **send_Mail** Multi-Mode e*Way.

2 On the palette, click the **Create a New Collaboration** button.

3 Enter the name of the new Collaboration, then click **OK** (for this example, **coll_send**).

4 Double-click the new Collaboration to edit its properties. The **Collaboration Properties** dialog box appears.

5 From the **Collaboration Rules** drop-down list box, select the Collaboration Rules file that you created earlier (for this example, **cr_send_mail**).

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

A In the **Instance Name** box type **input**.

B From the **Event Type** list, select the Event Type that you previously defined as **et_input**.

C From **Source** list, select the subscription source (for this example, **cp_JMS_toEmail**).

7 In the **Publications** area, click **Add** to define the Instances and Event Types that this Collaboration will publish.

A In the **Instance Name** field, select the name of this instance (in this example, **output**).

B From the **Event Type** list, select the outbound Event Type that you previously defined as **et_Send**.

C Select the publication destination from the **Destination** list (for this example, **cp_sendEmail**).

D The setting for **Priority** will default to **5**.

8 Click **Advanced** to set additional properties for the Collaboration.

9 Click on **OK** to close the **Collaboration Properties** dialog box.

5.9 Executing the Schema

To execute the IMS_Sample schema

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

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

Substitute ***hostname***, ***username*** and ***user password*** as appropriate.

2 Start the e*Gate Monitor. Specify the server that contains the Control Broker you started in Step 1 above.

3 Select the sample schema.

4 Verify that the Control Broker is connected. To do this, select and right-click the Control Broker in the e*Gate Monitor, and select **Status**. (The message in the Control tab of the console will indicate command *succeeded* and status as *up*.)

5 Select the IQ Manager, ***hostname***_igmgr, then right-click and select **Start**. (This will already be started if **Start automatically** is selected in the IQ Manager properties.)

6 Select each of the e*Ways, right-click and select **Start**. (These will already be started if **Start automatically** is selected in the e*Way's properties.)

7 To view the output, copy the output file (specified in the Outbound e*Way configuration file). Save to a convenient location and open.

Opening the destination file while the schema is running will cause errors.

Chapter 6

Java Methods

The e-Mail e*Way contains Java methods that are used to extend the functionality of the e*Way. These methods are contained in the following classes:

- **The EmailAddress Class** on page 60
- **The EmailAttachmentFile Class** on page 61
- **The EmailMessage Class** on page 62

6.1 The EmailAddress Class

`com.stc.eways.email.EmailAddress`

EmailAddress constructors are:

EmailAddress on page 60

EmailAddress methods are:

`None.`

EmailAddress

Description

Constructor. Sets the fields for address and name to "".

Constructor. Sets the fields for address and name respectively.

Syntax

```
public EmailAddress()
```

```
Public EmailAddress (java.lang.String address, java.lang.String name)
```

Parameters

Name	Type	Description
address	java.lang.String	Sets the field for the address
name	java.lang.String	Sets the field for the name

Throws

None.

6.2 The EmailAttachmentFile Class

`com.stc.eways.email.EmailAttachmentFile`

The EmailAttachmentFile Constructors are:

EmailAttachmentFile on page 61

The EmailAttachmentFile methods are:

save on page 61

EmailAttachmentFile

Description

Constructor.

Constructor. Sets the field for name.

Syntax

```
public EmailAttachmentFile()
public EmailAttachmentFile(java.lang.String name)
```

Parameters

Name	Type	Description
name	java.lang.String	Sets the field for the name

Returns

None.

Throws

None.

save

Description

Saves a received attachment to the specified path under specified name.

Syntax

```
public void save(java.lang.String path, java.lang.String name)
```

Parameters

Name	Type	Description
path	java.lang.String	The path to which attachments are stored.
name	java.lang.String	This name is used to override the name specified in the received email message.

Returns

None.

Throws

CollabDataException on error.

6.3 The EmailMessage Class

```
com.stc.eways.email.EmailMessage
```

TheEmailMessage methods are:

sendMessage on page 62

receiveMessage on page 63

sendMessage

Description

Call to send a message. Sends the message specified by the *EmailMessage* Instance.

Syntax

```
public void sendMessage()
```

Parameters

None.

Returns

None.

Throws

CollabDataException on error.

receiveMessage

Description

Call to receive a message. Receives a message and stores it in the *EmailMessage* instance.

Note: *After a successful call to* ***EmailMessage.receiveMessage()*** *and before a successful call to* ***send()****, the received e-mail message is contained ONLY in the EmailMessage instance in the Collaboration Rule. Failure to process and/or store it properly will result in a message loss.*

Important: ***Pop3HasMore*** *does not contain a value until* ***receiveMessage()*** *is called. After the method "****receiveMessage()****" is called,* ***Pop3HasMore*** *contains either* ***True*** *or* ***False****, based on the response of the POP3 server.*

Syntax

```
public boolean receiveMessage()
```

Parameters

None.

Returns

boolean.

Call to receive a message. Returns true if a valid message is received or false if no message is available.

Throws

CollabDataException on error.

Index

C

CharSet
  enabling Japanese or Korean characters **51**
CICS e*Way
  UNIX installation **10**
Classpath Override **13**
Collaboration Rules **34**
Collaborations **55**
components **7**
configuration parameters
  Allow Remote Debugging of JVM **16**
  Asynchronous Garbage Collection **15**
  Class Garbage Collection **15**
  CLASSPATH Append From Environment Variable **14**
  CLASSPATH Override **13**
  CLASSPATH Prepend **13**
  Disable JIT **16**
  Garbage Collection Activity Reporting **15**
  Initial Heap Size **14**
  JNI DLL Absolute Pathname **12**
  JVM settings **12**
  Maximum Heap Size **14**
  Maximum Stack Size for JVM Threads **15**
  Maximum Stack Size for Native Threads **15**
  Password **20**, **21**, **22**
  Port Number **20**, **21**
  Report JVM Info and all Class Loads **16**
  Server Name **20**, **21**
  Suspend option for debugging **16**
  User Name **20**, **21**
Connection Type **24**

D

Default Outgoing Message Type **25**
Delivery Mode **24**

E

e*Way connections
  configuration parameters **18**
  create and configure **17**
e*Ways
  creating **32**
  Multi-Mode **34**
Email e*Way connection configuration **17**
EmailAddress Class **60**
EmailAddress() **60**
EmailAttachmentFile Class **61**
EmailAttachmentFile() **61**
EmailMessage Class **62**
event type
  from XSC **32**
event types
  from DTD **29**, **30**

F

Factory Class Name **25**

G

General Settings **23**

H

Hangul character support **51**
Host Name **25**

I

implementation **27**
  Collaborations **55**
  Collaborations Rules **34**
  e*Way connections **53**
  outbound e*Way **33**
  overview **27**
installation **9**
  files/directories created **11**
  Windows NT or Windows 2000 **9**
  Windows NT/Windows 2000 **9**
intended reader **6**
ISO-2022-JP
  Japanese character support **51**
ISO-2022-KR
  Korean character support **51**

J

Japanese character support **7**, **51**
JMS e*Way connection configuration **22**

K

Korean character support **7**, **51**

M

Maximum Message Cache Size **26**
Maximum Number of Bytes to Read **24**
Message Service **25**
Multi-Mode e*Way **12**
    configuration parameters **12**

O

overview **6**

P

Pop3HasMore **63**
Port Number **26**
pre-installation
    UNIX **10**

R

receiveMessage() **63**

S

sample schema
    executing the schema **59**
save() **61**
sendMessage() **62**
Server Name **25**

T

Transaction Type **24**

U

UNIX
    CICS e*Way installation **10**
    pre-installation **10**