

***SeeBeyond ICAN Suite***

# **e\*Way Intelligent Adapter for Siebel EAI User's Guide**

*Release 5.0.5 for Schema Run-time Environment (SRE)*

*Java Version*



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

SeeBeyond, e\*Gate, e\*Way, and e\*Xchange are the registered trademarks of SeeBeyond Technology Corporation in the United States and/or select foreign countries. The SeeBeyond logo, SeeBeyond Integrated Composite Application Network Suite, eGate, eWay, eInsight, eVision, eXchange, eView, eIndex, eTL, ePortal, eBAM, and e\*Insight are trademarks of SeeBeyond Technology Corporation. The absence of a trademark from this list does not constitute a waiver of SeeBeyond Technology Corporation's intellectual property rights concerning that trademark. This document may contain references to other company, brand, and product names. These company, brand, and product names are used herein for identification purposes only and may be the trademarks of their respective owners.

© 2005 SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

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

Version 20050406040111.

# Contents

---

<b>Preface</b>	<b>9</b>
Intended Reader	9
Organization	9
Nomenclature	10
Online Use	10
Writing Conventions	10
Additional Documentation	11

---

## Chapter 1

<b>Introduction</b>	<b>12</b>
Overview	12
Communicating with Siebel EAI	13
e*Way Operation	14
e*Way as Publisher	14
e*Way as Subscriber	15
CGI Mode	16
MUX Mode	16
SeeBeyond Workflow Templates	17
e*Way Components	17
Supported Operating Systems	18

---

## Chapter 2

<b>Installation</b>	<b>19</b>
System Requirements	19
Environment Configuration	19
External System Requirements	20
Client	20
Server	20
Installing the e*Way	21

Windows Systems	21
Installation Procedure	21
Subdirectories and Files	23
UNIX Systems	24
Installation Procedure	24
Subdirectories and Files	24
<b>Optional Example Files</b>	<b>26</b>
Installation Procedure	27
Subdirectories and Files	28
Siebel 2000 Example	28
Siebel 7 Examples	29

## Chapter 3

<b>Web Server Setup</b>	<b>31</b>
<b>Overview</b>	<b>31</b>
Preliminary Installations	31
<b>Installing the Siebel Web Server Extension</b>	<b>32</b>
Siebel 2000 Only	32
Siebel 7 Only	35
<b>Installing the SeeBeyond Plug-ins</b>	<b>37</b>
Transport Mechanisms	37
CGI	37
MUX	37
<b>Installing CGI Components</b>	<b>38</b>
Microsoft IIS	38
iPlanet Web Server	40
IBM HTTP Server	41
<b>Verifying CGI Operation</b>	<b>42</b>
<b>Installing the MUX ASP</b>	<b>43</b>
ActiveX Client	43
Active Server Page File	43

## Chapter 4

<b>Siebel 2000 Implementation</b>	<b>44</b>
<b>Overview</b>	<b>44</b>
Pre-Implementation Tasks	44
<b>Implementation Overview</b>	<b>45</b>
General Sequence	45
e*Gate to Siebel	46
Siebel to e*Gate	47
Viewing e*Gate Components	47
<b>SeeBeyond Workflow Templates</b>	<b>48</b>
Overview	48
Siebel XML Messages	53
Format	53

Examples	53
Importing SeeBeyond Workflow Templates	55
Modifying SeeBeyond Workflow Templates	57
Setting Up SeeBeyond Workflow Processes	58
<b>Creating a Schema</b>	<b>62</b>
<b>Generating the Integration Object DTD</b>	<b>63</b>
<b>Verifying the Integration Object DTD</b>	<b>63</b>
<b>Creating Event Type Definitions</b>	<b>66</b>
Using the DTD Builder	66
<b>Assigning ETDs to Event Types</b>	<b>72</b>
<b>Defining Collaborations</b>	<b>73</b>
The Java Collaboration Rules Editor	73
<b>Creating Intelligent Queues</b>	<b>74</b>
<b>Using the Siebel EAI ETD in a Collaboration</b>	<b>75</b>
Overview	75
Helper Methods	75
Call Sequence	76
<b>Using the e*Way</b>	<b>77</b>
Connecting to Siebel	77
Specifying the Business Service	78
The Siebel Workflow Process	78
e*Gate-to-Siebel Example Procedure	79
Siebel-to-e*Gate Example Procedure	80
<b>Sample Schema</b>	<b>81</b>
Components	82
Event Types	82
Collaborations	84
feeder_collab	84
bob_collab	84
eater_collab	85

---

## Chapter 5

<b>Siebel 7 Implementation</b>	<b>86</b>
<b>Overview</b>	<b>86</b>
Pre-Implementation Tasks	86
<b>Implementation Overview</b>	<b>87</b>
General Sequence	87
e*Gate to Siebel	88
Siebel to e*Gate	89
Troubleshooting Your Implementation	89
<b>SeeBeyond Workflow Templates</b>	<b>90</b>
Overview	90
Siebel XML Messages	95
Format	95

Examples	95
Importing SeeBeyond Workflow Templates	97
Modifying SeeBeyond Workflow Templates	99
Setting Up SeeBeyond Workflow Processes	100
<b>Creating a Schema</b>	<b>105</b>
<b>Generating the Integration Object DTD</b>	<b>106</b>
<b>Verifying the Integration Object DTD</b>	<b>106</b>
<b>Creating Event Type Definitions</b>	<b>108</b>
Using the DTD Builder	108
<b>Assigning ETDs to Event Types</b>	<b>113</b>
<b>Defining Collaborations</b>	<b>115</b>
The Java Collaboration Rules Editor	115
<b>Creating Intelligent Queues</b>	<b>116</b>
<b>Using the Siebel EAI ETD in a Collaboration</b>	<b>116</b>
Overview	117
Helper Methods	117
Post/Retrieve Call Sequence	118
<b>Using the e*Way</b>	<b>119</b>
Connecting to Siebel	119
Specifying the Business Service	120
The Siebel Workflow Process	120
e*Gate-to-Siebel Example Procedure	121
Siebel-to-e*Gate Example Procedure	122
<b>Sample Schema</b>	<b>123</b>
JavaSiebelOutbound	123
Components	124
Event Types	124
Collaborations	126
JavaSiebelInbound	127
Components	127
Event Types	128
Collaborations	129

---

## Chapter 6

<b>e*Way Setup</b>	<b>130</b>
<b>Overview</b>	<b>130</b>
<b>Setting Up the e*Way</b>	<b>131</b>
Creating the e*Way	131
Modifying e*Way Properties	132
Configuring the e*Way	133
Changing the User Name	134
Setting Startup Options or Schedules	134
Activating or Modifying Logging Options	136
Activating or Modifying Monitoring Thresholds	137

Creating e*Way Connections	138
Using the e*Way Configuration Editor	141
Section and Parameter Controls	142
Parameter Configuration Controls	142
Command-line Configuration	143
Getting Help	143
Troubleshooting the e*Way	144
Configuration Problems	144
System-related Problems	145

## Chapter 7

<b>Operational Overview</b>	<b>146</b>
Overview	146
Multi-Mode e*Way Architecture	147
Collaborations and Event Type Definitions	149
Java Collaboration Service	152
e*Way Connections	153
Establishing Connections	153

## Chapter 8

<b>Configuration Parameters</b>	<b>155</b>
Overview	155
Multi-Mode e*Way	155
e*Way Connections	155
CGI Components	156
Multi-Mode e*Way	157
JVM Settings	157
General Settings	162
e*Way Connections	163
connector	163
HTTP	164
Proxies	165
HTTP Authentication	167
SSL	168
Siebel Configuration	173
CGI Web Server	175
JMS Connection Section	175
Host	175
Port	175
RequestReply	175
Timeout	175
TopicRequest	176

## Contents

Topic	176
Queue	176
ClientID	176
CGI Data Section	177
EnvInBody	177
EnvEnd	177
EnvsAsProps	177
ReadChunksize	177
WriteChunksize	178
Log Section	179
LogFile	179
Trace	179

---

## Chapter 9

<b>Java Methods</b>	<b>180</b>
Overview	180
Object Classes	180
Siebel2000 Class	181
Methods	183

<b>Index</b>	<b>200</b>
--------------	------------



# Preface

This Preface contains information regarding the User's Guide itself.

---

## P.1 Intended Reader

The reader of this guide is presumed to be a developer or system administrator with responsibility for maintaining the SeeBeyond™ e\*Gate™ Integrator system, and have a working knowledge of:

- Operation and administration of the appropriate operating systems (see [Supported Operating Systems](#) on page 18)
- Windows-style GUI operations
- Siebel EAI concepts and operations
- Integrating Siebel EAI with external systems

---

## P.2 Organization

This User's Guide is organized into two parts. The first part, consisting of Chapters 1-6, introduces the e\*Way and describes the procedures for installing and setting up the e\*Way, configuring Siebel, and implementing a working system incorporating the e\*Way. This part should be of particular interest to a System Administrator or other user charged with the task of getting the system up and running.

The second part, consisting of Chapters 7-9, describes the e\*Way operation, configuration parameters, and exposed Java methods. This part should be of particular interest to a Developer involved in customizing the e\*Way for a specific purpose. Information contained in this part that is necessary for the initial setup of the e\*Way is cross-referenced in the first part of the guide, at the appropriate points in the procedures.

---

## P.3 Nomenclature

For the sake of brevity, the e\*Way Intelligent Adapter for Siebel EAI is frequently referred to as the Siebel EAI e\*Way, or simply the e\*Way.

---

## P.4 Online Use

This User's Guide is provided in Adobe Acrobat's Portable Document Format (PDF). As such, it can be printed out on any printer or viewed online. When viewing online, you can take advantage of the extensive hyperlinking imbedded in the document to navigate quickly throughout the Guide.

Hyperlinking is available in:

- The Table of Contents
- The Index
- Within the chapter text, indicated by **blue print**

Existence of a hyperlink *hotspot* is indicated when the hand cursor points to the text. Note that the hotspots in the Index are the *page numbers*, not the topics themselves. Returning to the spot you hyperlinked from is accomplished by right-clicking the mouse and selecting **Go To Previous View** on the resulting menu.

---

## P.5 Writing Conventions

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

### Monospaced (Courier) Font

Computer code and text to be typed at the command line are set in Courier as shown below.

```
Configuration for BOB_Promotion
java -jar ValidationBuilder.jar
```

Variables within a command line, or attributes within a method signature, are set in italics as shown below:

```
stcregutl -rh host-name -un user-name -up password -sf
```

### Bold Sans-serif Font

- User Input: Click **Apply** to save, or **OK** to save and close.
- File Names and Paths: In the **Open** field, type **D:\setup\setup.exe**.
- Parameter, Function, and Command Names: The default parameter **localhost** is usually used only for testing.

---

## P.6 Additional Documentation

- Many of the procedures included in this User's Guide are described in greater detail in the *e\*Gate Integrator User's Guide*
- For more information on the Java Collaboration Service, see the *e\*Gate Integrator Collaboration Services Reference*
- For additional information on the Multi-Mode e\*Way, see the *Standard e\*Way Intelligent Adapter User's Guide*
- For additional information on the Multiplexer e\*Way, see the *e\*Gate API Kit User's Guide*
- For additional information on the CGI e\*Way, see the *CGI Web Server e\*Way Intelligent Adapter User's Guide*
- For information on requirements for the Siebel environment, see the *Siebel System Requirements and Supported Platforms* document for the version of Siebel you are using

# Introduction

This chapter provides a brief introduction to the SeeBeyond Java e\*Way Intelligent Adapter for Siebel EAI.

---

## 1.1 Overview

The Java e\*Way Intelligent Adapter for Siebel EAI is a software interface that enables the e\*Gate system to exchange Events (messages) with Siebel EAI via a Web server. The e\*Way communicates with Siebel via open standards such as HTTP and XML, and incorporates elements of three other SeeBeyond products:

- HTTPS e\*Way Intelligent Adapter
- CGI Web Server e\*Way Intelligent Adapter
- e\*Gate API Kit

The e\*Gate-to-Siebel implementation of the Siebel EAI e\*Way uses components of the HTTPS e\*Way to forward Siebel XML message to Siebel, while the Siebel-to-e\*Gate implementation uses components of the CGI Web Server e\*Way (or the MUX ASP from the API Kit) to relay XML messages from Siebel. Common elements of both e\*Ways are installed automatically as part of the Siebel EAI e\*Way installation.

The e\*Way uses Java methods to exchange data with the external system, package data as e\*Gate *Events*, send those Events to Collaborations, and manage the connection between the e\*Way and the external system. The operation of the e\*Way is described in [Operational Overview](#) on page 146.

---

## 1.2 Communicating with Siebel EAI

A traditional Siebel infrastructure is composed of four basic components:

- A **Siebel Database** to hold the data
- A **Siebel Gateway Server** to store enterprise configuration
- At least one **Siebel Application Server** to manage components of Siebel applications
- A **Siebel Client** to provide a user interface

To make use of the Web, Siebel adds another component: the Siebel Web Server Extension (SWSE). This is a shared library that runs inside a Web server to direct user requests to the appropriate Application Object Manager service via the Siebel Web Engine (SWE). The Application Object Manager is a component in the Siebel Server, which passes Siebel object definitions and data between the database and the SWSE. These object definitions provide the application logic and enable the user to interact with the database.

Communication with the Siebel 7 EAI application also involves one of the following Web servers:

- Microsoft Internet Information Service (IIS) for Windows
- iPlanet Web Server for Solaris
- IBM HTTP Server (Limited Release) for AIX

Communication with Siebel 2000 EAI involves only IIS.

Internally, Siebel EAI executes the Transport, Business Service and Workflow in the Business Integration Manager (BIM). BIM provides the development and run-time tools to configure and deploy integration between the Siebel EAI system and other applications. It includes the following components, which are used by the Siebel EAI e\*Way in the manner indicated:

- **Siebel Integration Objects**, to generate the ETD
- **Transport Adapters**, to send and receive messages
- **Business Service**, to start the workflow
- **Workflow Process Designer**, to convert XML messages and update Siebel
- **EAI Siebel Adapter**, to populate the Siebel database

The workflow process uses two Siebel EAI Toolkit components: **EAI XML Converter** and **EAI Siebel Adapter**. The EAI XML Converter uses the **XML to Property Set** method to convert the Siebel XML message to a property set format that can be used by EAI Siebel Adapter to query, insert, update, or delete the Business Object. In case of a query, the EAI XML Converter converts the property set back to an XML message.

## 1.3 e\*Way Operation

### 1.3.1 e\*Way as Publisher

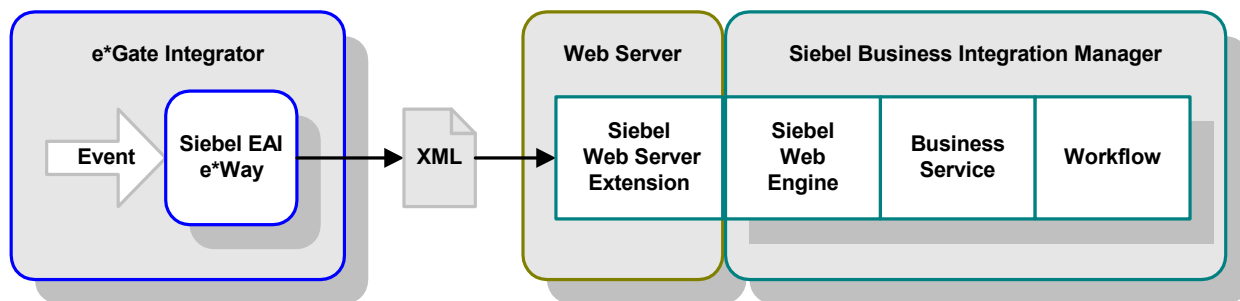
The Siebel EAI e\*Way acts as a Web browser, and uses HTTP to forward a Siebel XML-formatted Event to Siebel. It also specifies one of the following actions to be performed on the XML message:

- Delete
- Insert/Update
- Query

The result is that a corresponding Workflow is executed to process the message. A Siebel Workflow is a customized business application for managing and enforcing business processes.

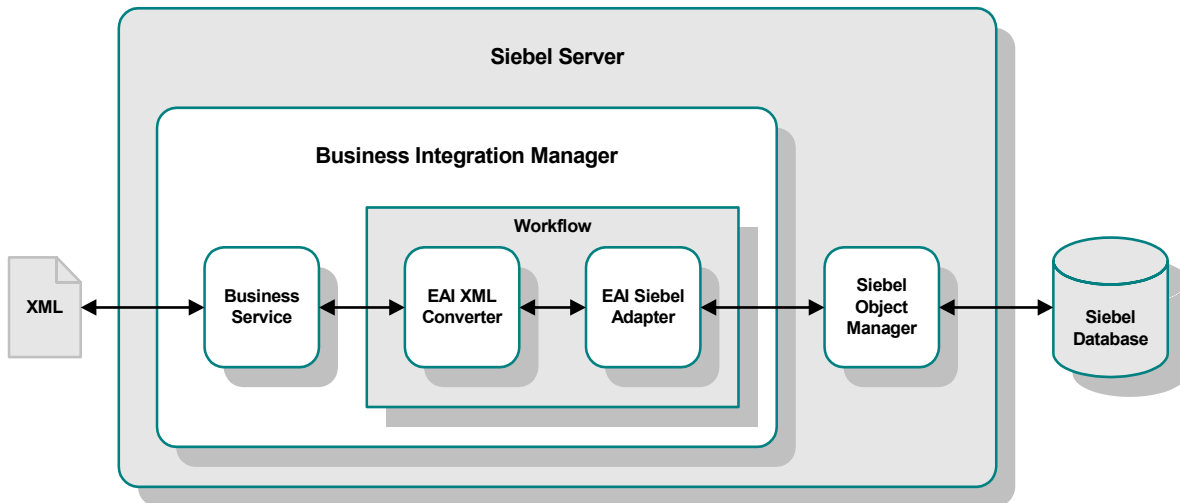
The Siebel EAI e\*Way receives an Event, which originated in some external application, from the e\*Gate system. The e\*Way passes the Event via HTTP to the Web server as a Siebel XML Message. The Siebel Web Server Extension invokes the specified Business Service which, in turn, starts an internal Workflow. Figure 1 illustrates the process.

**Figure 1** e\*Gate-to-Siebel Data Flow



The Workflow invokes the Siebel EAI XML Converter, which converts the information from XML into the Siebel internal format and presents it to the Siebel EAI Adapter. The information is then sent to the Siebel Server via the Siebel Object Manager (see Figure 2).

**Figure 2** Siebel Internal Processing

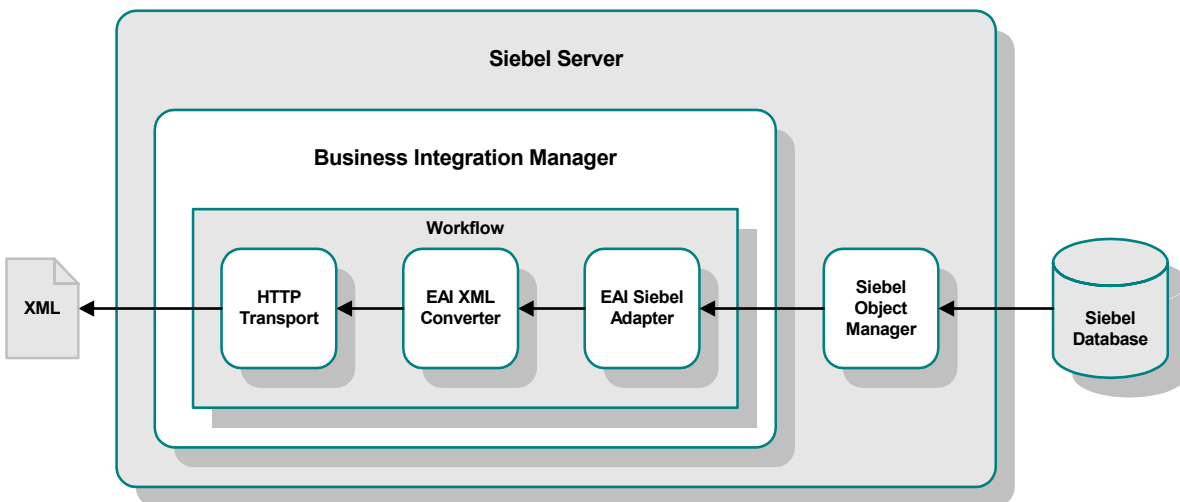


If there are data to be returned, the EAI Siebel Adapter can pass the result to the EAI XML Converter and send the data back to the e\*Way as a Siebel XML message.

### 1.3.2 e\*Way as Subscriber

The Siebel EAI e\*Way also allows the Siebel server to send a Siebel XML message to e\*Gate via HTTP. The data flow within Siebel is shown in Figure 3. This process is event-driven, and can be initiated, for example, by a feature added to the user interface of the Siebel application.

**Figure 3** Siebel Internal Processing

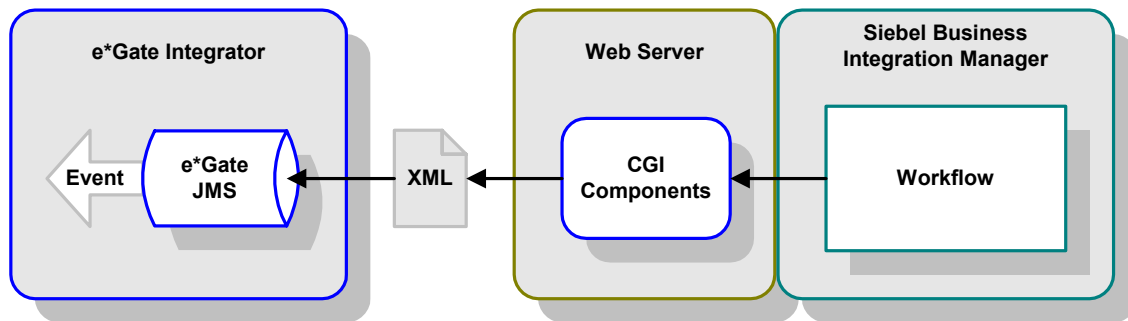


When a Siebel client initiates a data transfer, the Siebel Object Manager retrieves an Event from the Siebel database and starts a Workflow that resembles the Siebel-inbound workflow in reverse. The EAI Siebel Adapter relays the Event to the EAI XML Converter, which hands it off to a HTTP Transport module. The resulting Siebel XML Message is then sent to the Web server, which sends it to e\*Gate via CGI or MUX.

## CGI Mode

In this preferred method, e\*Gate receives the message forwarded by the CGI e\*Way's Web server components, which are installed on the Web server (Microsoft IIS, iPlanet, or IBM HTTP Server). The CGI components publish the message to a JMS IQ, and the Event then becomes available to other e\*Gate components. The concept is diagrammed in Figure 4.

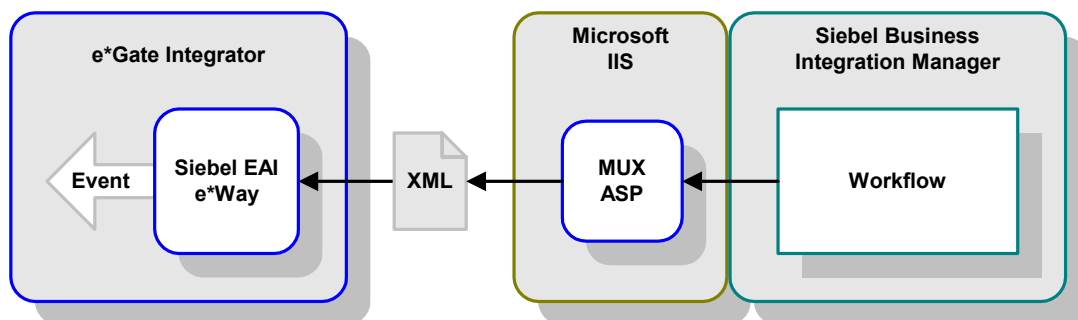
**Figure 4** Siebel-to-e\*Gate Data Flow - CGI



## MUX Mode

In this alternative method for Windows platforms, the e\*Way receives the message forwarded by the MUX Active Server Page (ASP) located in the Microsoft IIS. The Event then becomes available to other e\*Gate components. The concept is diagrammed in Figure 5.

**Figure 5** Siebel-to-e\*Gate Data Flow - MUX



**Note:** This method is included primarily for backward compatibility with existing (Siebel 2000) implementations. New implementations should employ the CGI method, publishing to the e\*Gate JMS, which is more robust.



### 1.3.3 SeeBeyond Workflow Templates

A set of SeeBeyond Workflow Templates is included with the Siebel EAI e\*Way. These Workflow Templates invoke the necessary Workflow Processes to map the data directly to or from the Siebel database.

*Note:* Different sets of Workflow Templates are used for Siebel 2000 and Siebel 7.

Additional information can be found in [SeeBeyond Workflow Templates](#) on page 48 or page 90. The referenced section also includes instructions on setting up the Business Service to execute the Workflows (see [Setting Up SeeBeyond Workflow Processes](#) on page 58 or page 100).

If you are using Siebel 2000 (Japanese), also see the information included in [Using the e\\*Way](#) on page 77.

---

## 1.4 e\*Way Components

The Java e\*Way Intelligent Adapter for Siebel EAI incorporates the following components:

- Executable files:
  - ♦ `stceway.exe` (SeeBeyond Multi-Mode e\*Way, installed with e\*Gate Integrator)
  - ♦ `stcewimp.exe` (e\*Gate API Kit)
  - ♦ `stccgi.exe` (SeeBeyond CGI e\*Way)
- A Java archive file, which customizes the Multi-Mode e\*Way for Siebel EAI:
  - ♦ `stcsiebel2000.jar`
- Dynamic-load libraries, used by the CGI e\*Way to communicate with the JMS Connection/IQ:
  - ♦ `stc_msapi.dll`
  - ♦ `stc_mscommon.dll`
  - ♦ `stc_msclient.dll`
- Configuration definition files, which you need to customize for your system (see [Chapter 8](#)):
  - ♦ `siebel2000.def` (for e\*Gate-to-Siebel operation)
  - ♦ `httpclient.def` (for Siebel-to-e\*Gate operation)
- Example schemas, discussed in [Chapter 4](#)

For a list of installed files, see [Installing the e\\*Way](#) on page 21.

*Note:* The e\*Gate Schema Designer GUI runs only on the Windows operating system.

## 1.5 Supported Operating Systems

The Java e\*Way Intelligent Adapter for Siebel EAI currently supports the following combinations of operating systems and Siebel versions.

**Table 1** English-language Version

Operating System	Siebel 2000	Siebel 7.0.3	Siebel 7.0.4	Siebel 7.5.3
Windows 2000, Windows XP, and Windows Server 2003	X	X	X	-
IBM AIX 5.1L and 5.2	-	-	X	-
Sun Solaris 8 and 9	-	X	X	-
HP-UX 11.0	-	-	-	X

**Table 2** Japanese-language Version

Operating System	Siebel 2000	Siebel 7.0.3	Siebel 7.0.4
Windows 2000, Windows XP, and Windows Server 2003	X	-	X

**Table 3** Korean-language Version

Operating System	Siebel 2000	Siebel 7.0.3	Siebel 7.0.4
Windows 2000, Windows XP, and Windows Server 2003	-	-	X

# Installation

This chapter describes the requirements and procedures for installing the e\*Way software. Procedures for implementing a working system, incorporating instances of the e\*Way, are described in [Chapter 4](#) (for Siebel 2000) and [Chapter 5](#) (for Siebel 7).

**Note:** Please read the *readme.txt* file located in the *addons\ewsiebelhttp* directory on the installation CD-ROM for important information regarding this installation.

---

## 2.1 System Requirements

To use the e\*Way Intelligent Adapter for Siebel EAI, you need the following:

- 1 An e\*Gate Participating Host.
- 2 A TCP/IP network connection.
- 3 Sufficient free disk space on both the Participating Host and the Registry Host to accommodate e\*Way files (not including sample schemas):
  - ♦ Approximately 2.6 MB on Windows systems
  - ♦ Approximately 10.9 MB on Solaris systems
  - ♦ Approximately 10.4 MB s on AIX systems

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.

**Note:** *It is not necessary to install the e\*Gate components on the Siebel Application server; however, the e\*Way must have access to the Siebel File system.*

### 2.1.1 Environment Configuration

No changes are required to the Participating Host's operating environment to support this e\*Way.

## 2.1.2 External System Requirements

In the following, please use the appropriate version of Siebel eBusiness and operating system(s) for your installation (see [Supported Operating Systems](#) on page 18). For full information on requirements for the Siebel environment, see the *Siebel System Requirements and Supported Platforms* document for the version of Siebel you are using.

### Client

The following software must be installed on all clients prior to installation of the e\*Way:

- Siebel 2000 or Siebel 7
  - ♦ Siebel Client
  - ♦ Siebel Tools

### Server

The following software must be installed on the server prior to installation of the e\*Way:

- Siebel 2000 or Siebel 7
  - ♦ Siebel Database Server
  - ♦ Siebel Gateway Server
  - ♦ Siebel Server
  - ♦ Siebel Tools
  - ♦ Siebel Web Server Extension

### Windows 2000 Platforms

- Microsoft Internet Information Server 5.0 (see [Microsoft IIS](#) on page 38 and [Installing the MUX ASP](#) on page 43)
- Libraries `stdole2.tlb` and `stdole32.tlb`

### Solaris Platforms

- iPlanet Web Server 4.1 with SP8 or above (see [iPlanet Web Server](#) on page 40)

### AIX Platforms

- IBM HTTP Server 2.0 Limited Release (see [IBM HTTP Server](#) on page 41)

## 2.2 Installing the e\*Way

### 2.2.1 Windows Systems

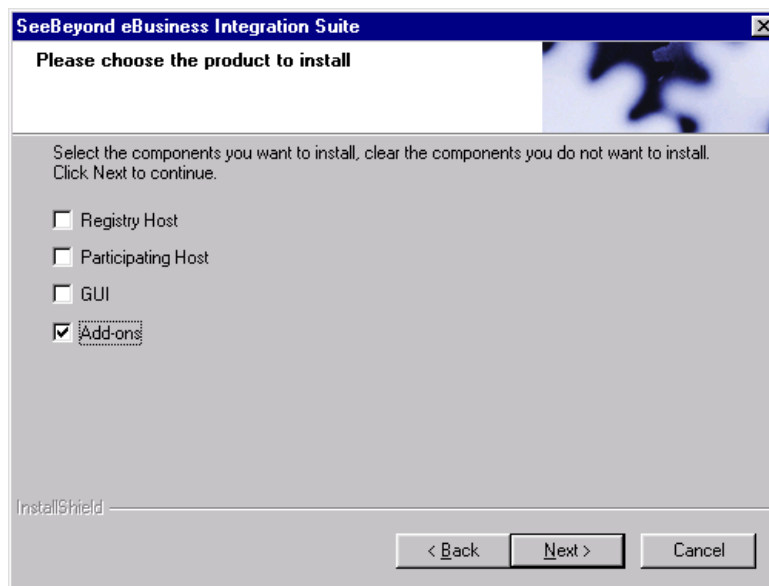
#### Installation Procedure

**Note:** *The installation utility detects and suggests the appropriate installation directory. Use this directory unless advised otherwise by SeeBeyond.*

#### To Install the e\*Way on a Microsoft Windows System

- 1 Log in as an Administrator on the workstation on which you want to install the e\*Way (you must have Administrator privileges to install this e\*Way).
- 2 Exit all Windows programs and disable any anti-virus applications before running the setup program.
- 3 Insert the e\*Way installation CD-ROM into the CD-ROM drive.
- 4 Launch the setup program.
  - A If the CD-ROM drive's Autorun feature is enabled, the setup program should launch automatically. Follow the on-screen instructions until the **Choose Product** dialog box appears (see Figure 6). Check **Add-ons**, then click **Next**.

**Figure 6** Choose Product Dialog

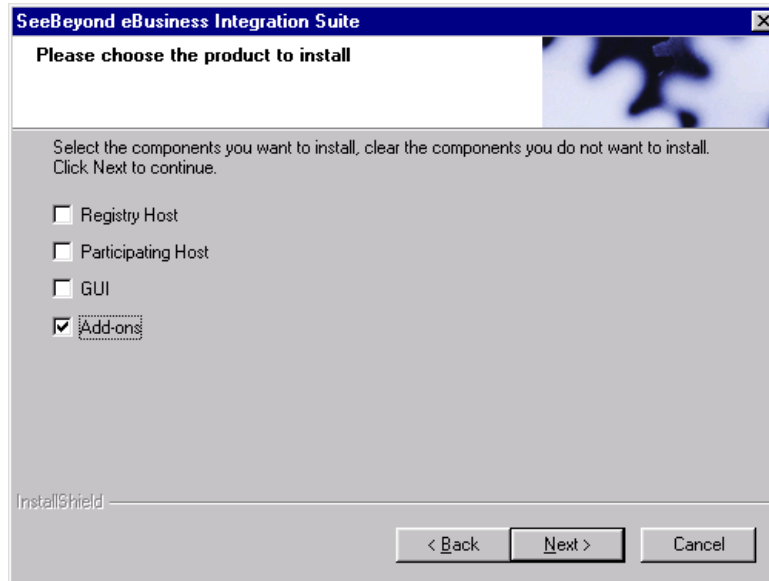


- B If the setup program does not launch automatically, use the Windows Explorer or the Control Panel's **Add/Remove Applications** feature to launch the following file on the CD-ROM drive (bypassing the **Choose Product** dialog):

setup\addons\setup.exe

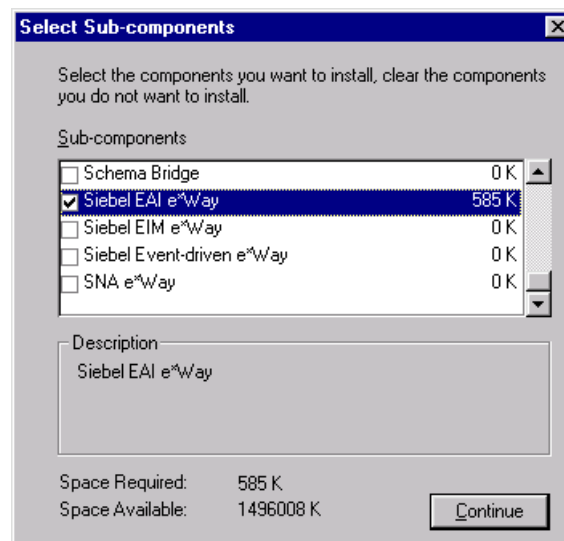
- 5 Follow the on-screen instructions until the **Select Components** dialog box appears (see Figure 7). Highlight—but do not check—**eWays** and then click **Change**.

**Figure 7** Select Components Dialog



- 6 When the **Select Sub-components** dialog box appears (see Figure 8), check the **Siebel EAI e\*Way**.

**Figure 8** Select e\*Way Dialog



- 7 Click **Continue**, and the **Select Components** dialog box reappears.
- 8 Click **Next** and continue with the installation.

## Subdirectories and Files

**Note:** *Installing the e\*Way Intelligent Adapter for Siebel EAI installs both Java and Monk versions. Only the files used by the Java version are listed in this section.*

By default, the InstallShield installer creates the following subdirectories and installs the following files within the \eGate\client tree on the Participating Host, and the \eGate\Server\registry\repository\default tree on the Registry Host.

**Table 4** Participating Host & Registry Host

Subdirectories	Files
\classes\	stcsiebel2000.jar
\configs\siebel2000\	siebel2000.def
\configs\httpclient\	httpclient.def
\etd\	siebeleai.ctl
\etd\siebel2000\	sample.jar sample.xsc siebel2000.xsc
\monk_scripts\common\	siebel-http-outgoing-delete.dsc siebel-http-outgoing-delete-sjis.dsc siebel-http-outgoing-execute.dsc siebel-http-outgoing-execute-sjis.dsc siebel-http-outgoing-insert.dsc siebel-http-outgoing-insert-sjis.dsc

By default, the InstallShield installer also installs the following file within the \eGate\Server\registry\repository\default tree on the Registry Host.

**Table 5** Registry Host Only

Subdirectories	Files
\	ewsiebelhttp.ctl

## 2.2.2 UNIX Systems

### Installation Procedure

**Note:** You are not required to have root privileges to install this e\*Way. Log on under the user name that you wish to own the e\*Way files. Be sure that this user has sufficient privilege to create files in the e\*Gate directory tree.

#### To Install the e\*Way on a UNIX System

- 1 Log onto the workstation containing the CD-ROM drive and, if necessary, mount the drive.
- 2 Insert the e\*Way installation CD-ROM into 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 appears. Select the **Install e\*Way** option and follow any additional on-screen instructions.

**Note:** The installation utility detects and suggests the appropriate installation directory. Use this directory unless advised otherwise by SeeBeyond. Note also that **no spaces** should appear in the installation path name.

### Subdirectories and Files

**Note:** Installing the e\*Way Intelligent Adapter for Siebel EAI installs both Java and Monk versions. Only the files used by the Javak version are listed in this section.

The preceding installation procedure creates the following subdirectories and installs the following files within the /eGate/client tree on the Participating Host, and the /eGate/Server/registry/repository/default tree on the Registry Host.

**Table 6** Participating Host & Registry Host

Subdirectories	Files
/classes/	stcsiebel2000.jar
/configs/siebel2000/	siebel2000.def
/configs/httpclient/	httpclient.def
/etd/	siebeleai.ctl
/etd\siebel2000/	sample.jar sample.xsc siebel2000.xsc



**Table 6** Participating Host & Registry Host

Subdirectories	Files
/monk_scripts/common/	siebel-http-outgoing-delete.dsc siebel-http-outgoing-delete-sjis.dsc siebel-http-outgoing-execute.dsc siebel-http-outgoing-execute-sjis.dsc siebel-http-outgoing-insert.dsc siebel-http-outgoing-insert-sjis.dsc

The preceding installation procedure also installs the following files only within the /eGate/Server/registry/repository/default tree on the Registry Host.

**Table 7** Registry Host Only

Subdirectories	Files
/	ewsiebelhttp.ctl

## 2.3 Optional Example Files

The installation CD contains three sample schema, located in the `samples\ewsiebelhttp` directory. Two versions of the Outbound schema are supplied, one for use with Siebel 2000 and another for use with Siebel 7.

- `\Siebel2000\JavaSiebelOutbound.zip`
- `\Siebel7\JavaSiebelOutbound.zip`
- `\Siebel7\JavaSiebelInbound.zip`

Note that sample schema for the Monk version of this e\*Way are located in the same directory. These are described in the User’s Guide for the Monk version.

**Table 8** e\*Gate CD-ROM Directory Structure

Subdirectory	Files	Description
\Siebel2000\	ewsiebelhttpsample.zip	Monk example—ignore.
	ewsiebelhttpsample2.zip	Monk example—ignore.
	JavaSiebelOutbound.zip	Java example for Siebel 2000.
	SiebelAccount.xml	A sample account.
\Siebel2000\inputdata\ewsiebelhttpsample\	input.fin	Input data for Monk example—ignore.
\Siebel2000\inputdata\JavaSiebelOutbound\	sample.xml	Input data for Java example—Siebel 2000.
\Siebel7\	JavaSiebelInbound.zip	Java Inbound example for Siebel 7.
	JavaSiebelOutbound.zip	Java Outbound example for Siebel 7.
\Siebel7\inputdata\JavaSiebelOutbound\	sample.xml	Input data for Java Outbound example—Siebel 7.

To use a schema, you must load it onto your system using the following procedure. See [Sample Schema](#) on page 81 for descriptions of the sample schema and instructions regarding its use.

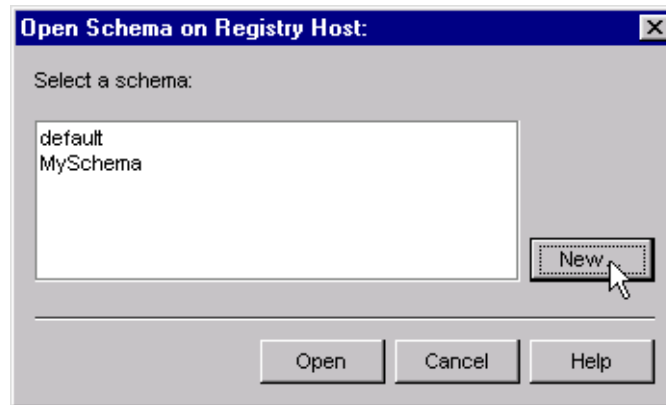
**Note:** *The Siebel EAI e\*Way must be properly installed on your system before you can run the sample schema.*

## 2.3.1 Installation Procedure

To load a sample schema

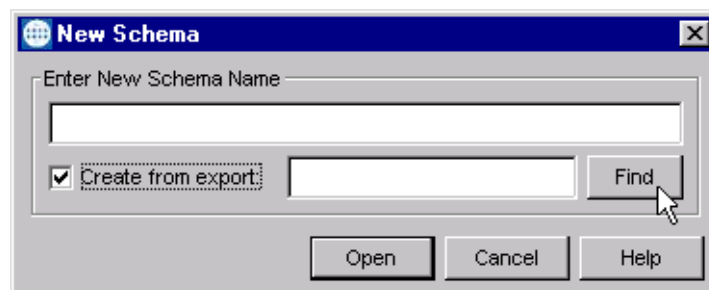
- 1 Invoke the **Open Schema** dialog box and select **New** (see Figure 9).

**Figure 9** Open Schema Dialog



- 2 Type the name you want to give to the schema (for example, **SiebelOut.Sample**)
- 3 Select **Create from export** and navigate to the directory containing the sample schema by clicking the **Find** button (see Figure 10).

**Figure 10** New Schema Dialog



- 4 Navigate to the desired archive file (\*.zip) and click **Open**.

**Note:** *The schema installs with the host name **localhost** and control broker name **localhost\_cb**. If you want to assign your own names, copy the file \*.zip to a local directory and extract the files. Using a text editor, edit the file \*.exp, replacing all instances of the name **localhost** with your desired name. Add the edited .exp file back into the .zip file.*

## 2.3.2 Subdirectories and Files

The preceding procedure creates the following subdirectories and installs the following files within the `\eGate\Server\registry\repository\<SchemaName>` tree on the Registry Host, where `<SchemaName>` is the name you have assigned to the schema in step 2.

### Siebel 2000 Example

**Table 9** Subdirectories and Files - JavaSiebelOutbound (Siebel 2000)

Subdirectories	Files
\	JavaSiebelOutbound.ctl
\runtime\collaboration_rules\siebel2000\	bob_rule.class bob_rule.ctl bob_rule.java bob_rule.xpr bob_rule.xts bob_ruleBase.class
\runtime\configs\siebel2000\	siebel_conn.cfg siebel_conn.sc
\runtime\configs\stcewfile\	eater.cfg eater.sc feeder.cfg feeder.sc
\runtime\etd\siebel2000\	postresponse.jar postresponse.ssc postresponse.xsc sampleAcount.jar sampleAcount.xsc
\sandbox\Administrator\etd\	common.ctl rtjar.ctl siebeleai.ctl TestSP.jar TestSP.xsc

## Siebel 7 Examples

**Table 10** Subdirectories and Files - JavaSiebelOutbound (Siebel 7)

Subdirectories	Files
\	JavaSiebel7Outbound.ctl
\runtime\collaboration_rules\	bob_rule.class bob_rule.ctl bob_rule.java bob_rule.xpr bob_rule.xts bob_ruleBase.class
\runtime\configs\siebel2000\	siebel_conn.cfg siebel_conn.sc
\runtime\configs\stcewfile\	eater.cfg eater.sc
\runtime\etd\	SampleAccount.jar SampleAccount.xsc
\sandbox\Administrator\collaboration_rules\	bob_rule.class bob_rule.ctl bob_rule.java bob_rule.xpr bob_rule.xts bob_ruleBase.class
\sandbox\Administrator\configs\stcewfile\	feeder.cfg feeder.sc
\sandbox\Administrator\etd\	common.ctl postresponse.jar postresponse.ssc postresponse.xsc sampleAccount.jar sampleAccount.xsc siebeleai.ctl
\userlocks\collaboration_rules\	bob_rule.class bob_rule.ctl bob_rule.java bob_rule.xpr bob_rule.xts bob_ruleBase.class
\userlocks\etd\	common.ctl siebeleai.ctl

**Table 11** Subdirectories and Files - JavaSiebellnbound (Siebel 7)

Subdirectories	Files
\	JavaSiebellnbound.ctl
\runtime\collaboration_rules\	crGetRequestJMSPProperty.class crGetRequestJMSPProperty.ctl crGetRequestJMSPProperty.java crGetRequestJMSPProperty.xpr crGetRequestJMSPProperty.xts crGetRequestJMSPPropertyBase.class crRequestReply_webRequestETDReplyETD.class crRequestReply_webRequestETDReplyETD.ctl crRequestReply_webRequestETDReplyETD.java crRequestReply_webRequestETDReplyETD.xpr crRequestReply_webRequestETDReplyETD.xts crRequestReply_webRequestETDReplyETDbase.class
\runtime\configs\messageservice\	cpBackstayJMS.cfg cpBackstayJMS.sc webRequestETDJMS.cfg webRequestETDJMS.sc
\runtime\configs\stceway\	ewRequestReply.cfg ewRequestReply.sc
\runtime\configs\stcewfile\	eater.cfg eater.sc
\runtime\configs\stcmsagent\	backstay_jmsmgr.cfg backstay_jmsmgr.sc
\runtime\etd\	common.ctl rtjar.ctl webReplyETD.ctl webRequestETD.ctl

# Web Server Setup

This chapter describes procedures for setting up the appropriate Web server to provide access to the Siebel EAI system.

---

## 3.1 Overview

Communication with the Siebel 7 EAI application involves one of the following Web servers:

- Microsoft Internet Information Service (IIS) for Windows
- iPlanet Web Server for Solaris
- IBM HTTP Server (Limited Release) for AIX

**Note:** *This is a special release of IBM HTTP Server 2.0. See the Siebel Systems Requirements and Supported Platforms documentation for information.*

Communication with Siebel 2000 EAI involves only IIS.

Certain components from both SeeBeyond and Siebel must be installed into the Web server to complete the communications link. Two stages are involved in setting up the Web server:

[Installing the Siebel Web Server Extension](#) on page 32

[Installing the SeeBeyond Plug-ins](#) on page 37 (for Siebel-to-e\*Gate operation only)

### 3.1.1 Preliminary Installations

The following software must be in place and operating correctly:

- Siebel 2000 or Siebel 7
- Siebel Tools
- Siebel Server
- The appropriate Web Server

## 3.2 Installing the Siebel Web Server Extension

The Siebel Web Server Extension (SWSE) is a shared library that runs inside the Web server to direct user requests to the appropriate Application Object Manager service via the Siebel Web Engine (SWE).

### 3.2.1 Siebel 2000 Only

#### To Install the Siebel Web Server Extension (SWSE) for Siebel 2000

- 1 From the Siebel installation media, run `\eappweb\setup.exe`, which invokes the installation wizard.
- 2 Follow the instructions presented by the wizard. Use the naming conventions for your Siebel EAI Application Server.
- 3 For **Connection Protocol**, specify the default port for an HTTP server, which is **80**.
- 4 Do *not* use any encryption or compression methods.
- 5 For **Anonymous Employee** and **Anonymous Contact** login and password, use **SADMIN**.
- 6 For **Error Level for Logging**, enter **All Errors and Warnings**. You can change this once correct system operation has been verified.
- 7 In the `\bin` directory where you have installed the Siebel Web Server Extension, open the `eapps.cfg` file and note the following (typical values are shown):

```
[defaults]
AnonUserName = SADMIN
AnonPassword = SADMIN
AnonUserPool = 10
StatsPage = _stats.swe

[/eai]
ConnetString = siebel.TCPIP.none.none://MyGatewayServer:3230/
MyEnterpriseServer/eaiObjMgr/MyAppServer
EnableExtServiceOnly = TRUE
```

- 8 In the `\bin` directory where you have installed the Siebel Server, open the corresponding application configuration file (for example, `eai.cfg`).
- 9 In the `[Server]` section, comment out the following line:

```
;SecurityAdapter = LDAP
and set
```

```
ContactLogin = FALSE
```

- 10 In the `[SWE]` section, comment out the following lines:

```
;UserSWFName =
;ContactLogin = TRUE
```

- 11 If LDAP is not used, comment out all of the following lines:

```
:[SecurityAdapters]
;LDAP = LDAP
```



```
;[LDAP]
;DllName = sscfldap.dll
;ServerName =
;Port = 389
;BaseDN =
;UsernameAttributeType = uid
;PasswordAttributeType = userPassword
;CredentialsAttributeType = credentials
;RolesAttributeType = roles
;SslDatabase =
```

12 After modifying these files, stop and then restart the following services:

- ◆ Siebel Server
- ◆ World Wide Web Publishing Service

13 Log in to Siebel Sales 6.0 and follow the Screens menu path:

Server Administration > Enterprise Config > Enterprise Component Groups

**Figure 11** Enterprise Component Groups

Component Group	Component Group Alias	Number of Components	Enable state	Description
Field Service	FieldSvc	6	Disabled	Field Service Components
Workflow Management	Workflow	5	Enabled	Workflow Management Components
Data Quality	DataQual	1	Disabled	Data Quality Components
Assignment Management	AsgnMgmt	2	Disabled	Assignment Management Components
SAP Connector	SAP	2	Disabled	SAP Connector Components
Incentive Compensation	IComp	4	Disabled	Incentive Compensation Components
Marketing	Mktng	8	Disabled	Marketing Components
Dun and Bradstreet	DandB	3	Disabled	Dun and Bradstreet Components
Web Collaboration	WebColab	1	Disabled	Web Collaboration Components
Siebel Thin Client	ThinClient	15	Disabled	Siebel Thin Client Components
Enterprise Application Integration	EAI	5	Enabled	Enterprise Application Integration Components
Siebel Remote	Remote	7	Disabled	Siebel Remote Components
System Management	System	5	Enabled	System Management Components
Communications Management	CommMgmt	5	Enabled	Communications Management Components

14 Select the following items, and click **Enable**:

- ◆ Enterprise Application Integration
- ◆ Workflow Management
- ◆ Communication Management

15 Open the browser and type:

<yourservename>/<module> (for example, 10.1.3.135/eai)

and then click **Enter**.

Make sure that your Siebel Server ODBC data source is configured. You can verify which one you are using by examining the Siebel Server log directory—it contains a file listing all the parameters.

### To Verify SWSE Operation for Siebel 2000

- 1 Verify that the **Server Request Processor** is running correctly. You may need to synchronize the Server Request Components with the Gateway Server—follow the menu/command path:  
**Enterprise Configuration > Batch Components Admin > Synchronize**
- 2 Verify that the **EAI Object Manager** is running correctly.
  - A Follow the path:  
**Server Admin > Servers > Server Components**
  - B Under **Assignment Components**, select **EAI Object Manager**.
- 3 For Windows installations only, verify that the .swe file is associated with **sweiis.dll** in the web site. Use the following procedure to set the association:
  - A Run the **IIS 4.0 Management Console** application.
  - B Right-click on your Web site entry in the tree display, and select **Properties**.
  - C Select the **Home Directory** tab.
  - D In the **Application Settings** box, select **Configuration**.
  - E On the **App Mappings** tab, select **Add**.
  - F Type the following line:  

```
.make association swe - sweiis.dll
```
  - G Select **All Siebel apps**.
- 4 Verify that the configuration files are set up properly. If LDAP is not used, comment out all LDAP-related parameters in the configuration files of the corresponding application (see step 11 in the previous section).

### 3.2.2 Siebel 7 Only

#### To Install Siebel Web Server Extension (SWSE) for Siebel 7

- 1 From the Siebel installation media, run `\eappweb\setup.exe`, which invokes the installation wizard.
- 2 Follow the instructions presented by the wizard. Use the naming conventions for your Siebel EAI Application Server.
- 3 For **Connection Protocol**, specify the default port for an HTTP server, which is **80**.
- 4 Do *not* use any encryption or compression methods.
- 5 For **Anonymous Employee** and **Anonymous Contact** login and password, use **SADMIN**.
- 6 For **Error Level for Logging**, enter **All Errors and Warnings**. You can change this once correct system operation has been verified.
- 7 In the `\bin` directory where you have installed the Siebel Web Server Extension, open the `eapps.cfg` file and note the following (typical values are shown):

```
[defaults]
AnonUserName = SADMIN
AnonPassword = SADMIN
AnonUserPool = 10
StatsPage = _stats.swe

[/eai]
ConnetString = siebel.TCPIP.none.none://MyGatewayServer:2320/
MyEnterpriseServer/eaiObjMgr/MyAppServer
EnableExtServiceOnly = TRUE
```

- 8 After modifying these files, stop and then restart the following services:
  - ◆ Siebel Server
  - ◆ World Wide Web Publishing Service
- 9 Log in to Siebel Sales 7.0 and follow the Screens menu path:

**Ctrl+Shift+A > Server Administration > Enterprise Configuration**  
**> Enterprise Component Groups**

**Figure 12** Enterprise Component Groups

Component Group	Component Grou	Number of Comp	Enable state	Description
Field Service	FieldSvc	10	Enabled	Field Service Components
Workflow Managem	Workflow	5	Enabled	Workflow Management Components
Assignment Manage	AsgnMgmt	2	Disabled	Assignment Management Components
Data Quality	DataQual	1	Disabled	Data Quality Components
Siebel Sales	Sales	2	Enabled	Siebel Sales Components
Siebel eDocuments	eDocuments	1	Disabled	Siebel eDocuments Components
Siebel Call Center	CallCenter	2	Enabled	Siebel Center Components

- 10 Select the following items, and click **Enable**:
  - ♦ Enterprise Application Integration
  - ♦ Workflow Management
  - ♦ Communication Management
- 11 Open the browser and type:  
`<yourservename>/<module>` (for example, 10.1.3.135/eai)  
and then click **Enter**.

Make sure that your Siebel Server ODBC data source is configured. You can verify which one you are using by examining the Siebel Server log directory—it contains a file listing all the parameters.

#### To Verify SWSE Operation for Siebel 7

- 1 Verify that the **Server Request Processor** is running correctly. You may need to synchronize the Server Request Components with the Gateway Server—follow the menu/command path:  
**Enterprise Configuration > Batch Components Admin > Synchronize**
- 2 Verify that the **EAI Object Manager** is running correctly.
  - A Follow the path:  
**Server Admin > Servers > Server Components**
  - B Under **Assignment Components**, select **EAI Object Manager**.
- 3 *For Windows installations only*, verify that the .swe file is associated with sweiis.dll in the web site. Use the following procedure to set the association:
  - A Run the IIS 4.0 **Management Console** application.
  - B Right-click on your Web site entry in the tree display, and select **Properties**.
  - C Select the **Home Directory** tab.
  - D In the **Application Settings** box, select **Configuration**.
  - E On the **App Mappings** tab, select **Add**.
  - F Type the following line:  

```
.make association swe - sweiis.dll
```
  - G Select **All Siebel apps**.
- 4 Verify that the configuration files are set up properly. If LDAP is not used, comment out all LDAP-related parameters in the configuration files of the corresponding application (see step 11 in the previous section).

---

## 3.3 Installing the SeeBeyond Plug-ins

*Note:* The following sections pertain only to Siebel-to-e\*Gate operation.

### 3.3.1 Transport Mechanisms

#### CGI

The **mscgi.properties** file must be edited before running the CGI e\*Way. The file contains the information pertaining to the JMS Connection, CGI Data, and Logging values.

The properties file is loaded by the SeeBeyond JMS CGI. Each property is a name/value pairing. The name uniquely identifies the property. The value is the content associated with that name. The name is separated from the value with the ':' character.

*Important:* Do not change the names.

Setup procedures differ according to the specific Web server you are using. See the section appropriate to your system:

[Microsoft IIS](#) on page 38

[iPlanet Web Server](#) on page 40

[IBM HTTP Server](#) on page 41

#### MUX

*Note:* This method is included primarily for backward compatibility with existing (Siebel 2000) implementations. New implementations should employ the CGI method, publishing to the e\*Gate JMS, which is more robust.

For **Siebel-to-e\*Gate** operation using MUX instead of CGI, setup is required to enable operation of the MUX ASP. Note that the existing Active Server Page file, **Mux.asp**, serves as a template that you must modify to suit your system. See the following section:

[Installing the MUX ASP](#) on page 43

## 3.3.2 Installing CGI Components

### Microsoft IIS

For **Siebel-to-e\*Gate** operation, the web server should execute the client executable, **stccgi.exe**, when a request arrives. It also needs to set the dynamic-load library path in order for **stc\_msapi.dll**, **stc\_mscommon.dll** and **stc\_msclient.dll** to be loaded by **stccgi.exe**.

To configure the IIS Web server to use the CGI e\*Way Web server components

- 1 It is recommended (but not mandatory) that you create a **cgi-bin** directory in the IIS root directory to store all CGI applications. If the default IIS server installation was used, the root directory is:

`\inetpub`

and the new directory should be:

`\inetpub\cgi-bin`

- 2 Using the Internet Information Services Manager, go to:

**Start > Settings > Control Panel > Administrative Tools > Internet Services Manager**

(or use Internet Information Services snap-in contained in Windows 2000 Advanced Server) and create a virtual directory:

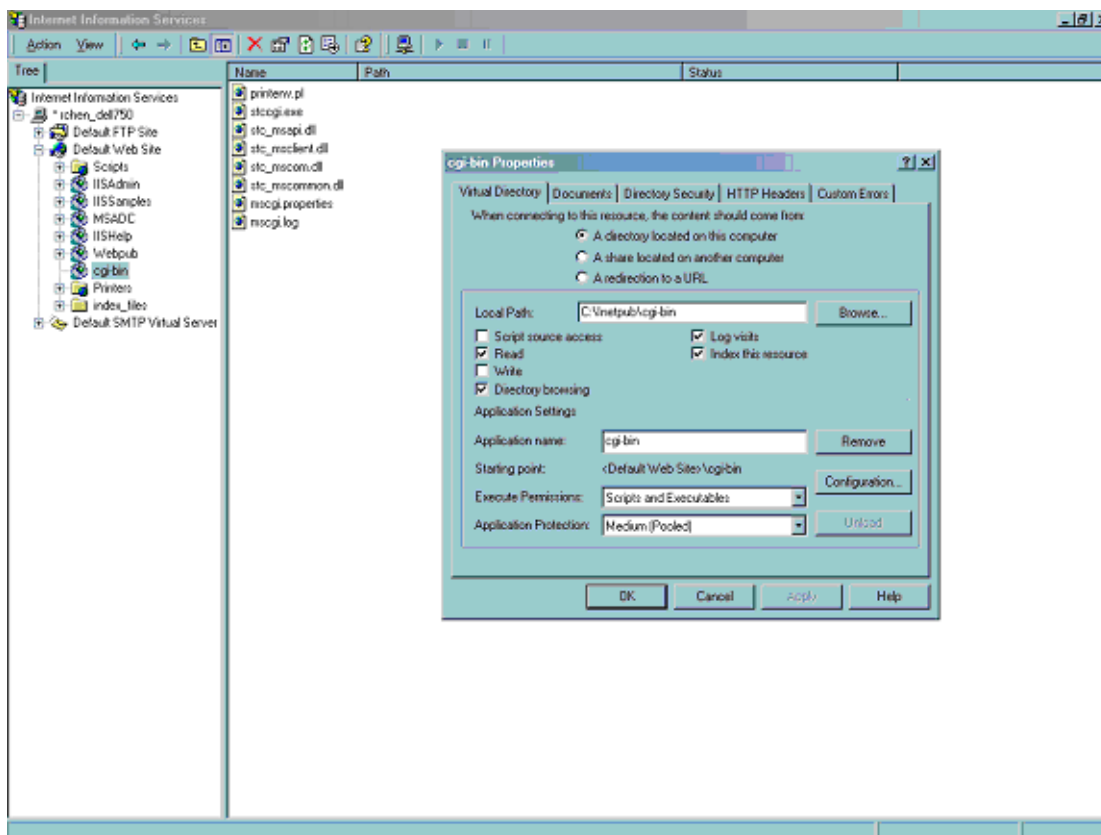
- A Select **Default Web Site** in IIS manager.
- B Right-click and select the action **New > Virtual Directory**.
  - ♦ Alias: **cgi-bin**
  - ♦ Directory: **C:\inetpub\cgi-bin** (use the same directory as created in step 1)
  - ♦ Access permissions: **Read, Run Scripts, and Execute**.
- 3 Copy the **stccgi.exe**, **stc\_msapi.dll**, **stc\_msclient.dll** and **stc\_mscommon.dll** to the **cgi-bin** directory.
- 4 Create or copy a **test\*.html** file to the document root directory that was configured for IIS server, for example:

`C:\inetpub\wwwroot`
- 5 You must modify **mscgi.properties** to configure the CGI executable. Change the permission on **stccgi.exe**, **stc\_msapi.dll**, **stc\_msclient.dll** and **stc\_mscommon.dll**, to enable the Web server to read and execute them.

For IIS, ensure that for the directory created above, **cgi-bin**, the **Execute Permissions** setting is set to **Scripts and Executables**. To modify this setting:

- A Go to Internet Service Manager (see Figure 13).
- B Click on your Web site (for example, Default Web Site).
- C Right-click on **Scripts** and select **Properties**.
- D In the **Scripts Properties** window, click on the **Virtual Directory** tab.
- E Select **Scripts and Executables** on the Execute Permissions scroll menu.
- F Select **OK**, then restart the Web server.

**Figure 13** IIS Internet Services Manager



- 6 Verify proper operation as described in [Verifying CGI Operation](#) on page 42.

**Note:** Consult the Web server documentation for more information.

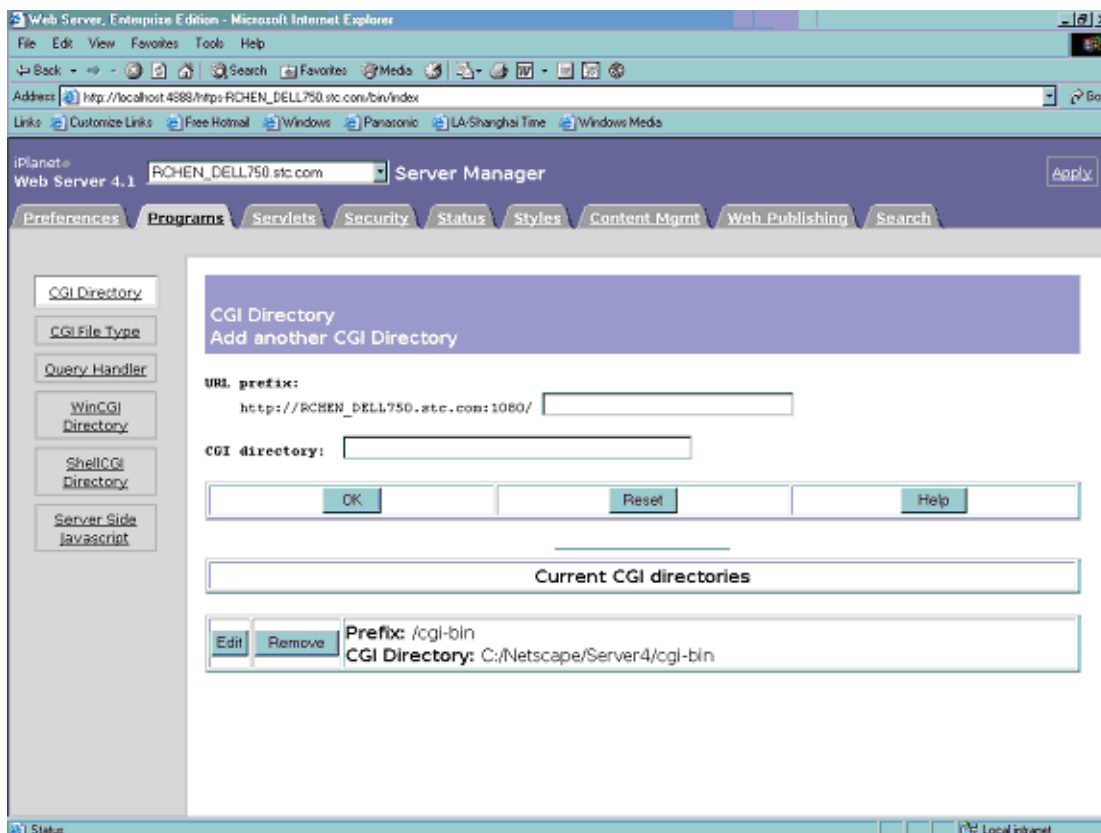
## iPlanet Web Server

For **Siebel-to-e\*Gate** operation, the web server should execute the client executable, **stccgi.exe**, when a request arrives. It also needs to set the dynamic-load library path in order for **stc\_msapi.dll**, **stc\_mscommon.dll** and **stc\_msclient.dll** to be loaded by **stccgi.exe**.

To configure the iPlanet Web server to use the CGI e\*Way Web server components

- 1 Using your Web browser, access the iPlanet Web Server and select **Server Manager**.
- 2 Select the **Programs** tab, and then **CGI Directory**, as shown below:

**Figure 14** iPlanet Server Manger



- 3 Copy the **stccgi.exe**, **stc\_msapi.dll**, **stc\_msclient.dll** and **stc\_mscommon.dll** to the indicated directory, for example:  
`/NetScape/Srever4/cgi-bin`
- 4 You must modify **mscgi.properties** to configure the CGI executable. Change the permissions on **stccgi.exe**, **stc\_msapi.dll**, **stc\_msclient.dll** and **stc\_mscommon.dll** to enable the Web server to read and execute them.
- 5 Verify proper operation as described in **Verifying CGI Operation** on page 42.

**Note:** Consult the Web server documentation for more information.



## IBM HTTP Server

For **Siebel-to-e\*Gate** operation, the web server should execute the client executable, **stccgi.exe**, when a request arrives. It also needs to set the dynamic-load library path in order for **stc\_msapi.dll**, **stc\_mscommon.dll** and **stc\_msclient.dll** to be loaded by **stccgi.exe**.

To configure the IBM HTTP Server to use the CGI e\*Way Web server components

- 1 Locate the scripts **restartapa**, **startapa**, and **stopapa** in the following directory:  
`/usr/IBMIHS/bin`
- 2 Each of these scripts contains a **LIBPATH** statement. Add the path to the directory containing the following files to each **LIBPATH** statement:
  - ♦ **stccgi.exe**
  - ♦ **stc\_msapi.dll**
  - ♦ **stc\_msclient.dll**
  - ♦ **stc\_mscommon.dll**
- 3 You must modify **mscgi.properties** to configure the CGI executable. Change the permissions on **stccgi.exe**, **stc\_msapi.dll**, **stc\_msclient.dll** and **stc\_mscommon.dll** to enable the Web server to read and execute them.
- 4 Start the Web server using the **startapa** or **restartapa** command.
- 5 Verify proper operation as described in [Verifying CGI Operation](#) on page 42.

**Note:** Consult the Web server documentation for more information.

### 3.3.3 Verifying CGI Operation

Access the `test*.html` file from a Web browser, send a file to the CGI Web Server e\*Way server. If successful, you will see the file you send to the server displayed. The URL to access the `stccgi.exe` is:

```
http://hostname/cgi-bin/stccgi.exe
```

Following is a sample HTML form used to access `stccgi.exe`:

```
<HTML>

<FORM ACTION="/cgi-bin/stccgi.exe" METHOD="POST"
ENCTYPE="multipart/form-data">
Multipart test
<P>
<TABLE>
  <TR>
    <TD><LABEL for="fname">First name: </LABEL>
    <TD> <INPUT type="text" name="firstname" id="fname">
  <TR>
    <TD><LABEL for="lname">Last name: </LABEL>
    <TD><INPUT type="text" name="lastname" id="lname">
  </TABLE>
  <LABEL for="email">email: </LABEL>
    <INPUT type="text" name="email"><BR>
  <INPUT type="radio" name="sex" value="Male"> Male<BR>
  <INPUT type="radio" name="sex" value="Female"> Female<BR>
  <LABEL for="filename">What files are you sending? </LABEL>
    <INPUT type="file" name="filename"><BR>
  <INPUT type="submit" value="Send"> <INPUT type="reset">
</P>
</FORM>

</HTML>
```

### 3.3.4 Installing the MUX ASP

**Note:** *The following is applicable only to Windows platforms using the IIS Web Server. It is included primarily for backward compatibility with existing (Siebel 2000) implementations. CGI is the preferred method.*

For **Siebel-to-e\*Gate** operation using MUX instead of CGI, additional setup is required to enable operation of the MUX ASP. Note that the existing Active Server Page file, **Mux.asp**, serves as a template that you must modify to suit your system.

## ActiveX Client

### To Register the ActiveX Client

- If your e\*Gate installation is co-located with your Siebel application, launch a command shell and, at the command prompt, type:

```
regsvr32 <drive>:\egate\client\bin\stc_xipmpclnt.dll <enter>
regsvr32 <drive>:\egate\client\bin\stc_common.dll <enter>
regsvr32 <drive>:\egate\client\bin\stc_ewipmpclnt.dll <enter>
```

A message box appears after each entry, confirming that the command was performed successfully; no messages are displayed in the command window itself.

- If your e\*Gate installation is located on a different platform from your Siebel 2000 application:

**A** Copy the \*.dll files from the \eGate\Client\bin directory to a directory on your Siebel host.

**B** Launch a command shell and, at the command prompt, type:

```
regsvr32 <drive>:\<path>\stc_xipmpclnt.dll <enter>
regsvr32 <drive>:\<path>\stc_common.dll <enter>
regsvr32 <drive>:\<path>\stc_ewipmpclnt.dll <enter>
```

where <path> is the path to your chosen directory location.

A message box appears after each entry, confirming that the command was performed successfully; no messages are displayed in the command window itself.

## Active Server Page File

### To Install the Active Server Page File

- 1 Locate the file **Mux.asp**, which is contained in the following sample schema file on the installation CD-ROM:

```
<cdrom>\setup\samples\ewsiebelhttp\ewsiebelhttpsample.zip
```

- 2 Copy the file to the directory of the Web Server that you want to access.
- 3 Modify the following line:

```
const strHost = "localhost"
```

using the IP address or host name corresponding to your MUX server. If your MUX server uses a port number other than the default, also change the value of **dwPort**.

# Siebel 2000 Implementation

This chapter describes the procedures for creating a functional Siebel-e\*Gate system incorporating the Siebel EAI e\*Way. Please refer to the *e\*Gate Integrator User's Guide* for additional information.

---

## 4.1 Overview

This e\*Way provides a specialized transport component for incorporation in an operational schema. The schema also contains Collaborations, linking different data or Event types, and Intelligent Queues. Typically, other e\*Way types also are used as components of the schema.

One or more sample schemas, included in the software package, are described at the end of this chapter. These can be used to test your system following installation and, if appropriate, as a template that you can modify to produce your own schema.

### 4.1.1 Pre-Implementation Tasks

#### Install the SeeBeyond Software

The first task is to install the SeeBeyond software as described in [Installing the e\\*Way](#) on page 21.

#### Import the Sample Schema

If you want to use the sample schema supplied with the e\*Way, the schema files must be imported from the installation CD-ROM (see [Optional Example Files](#) on page 26).

**Note:** *It is highly recommended that you make use of the sample schemas to familiarize yourself with e\*Way operation, test your system, and use as templates for your working schemas.*

#### Configure the Siebel EAI System

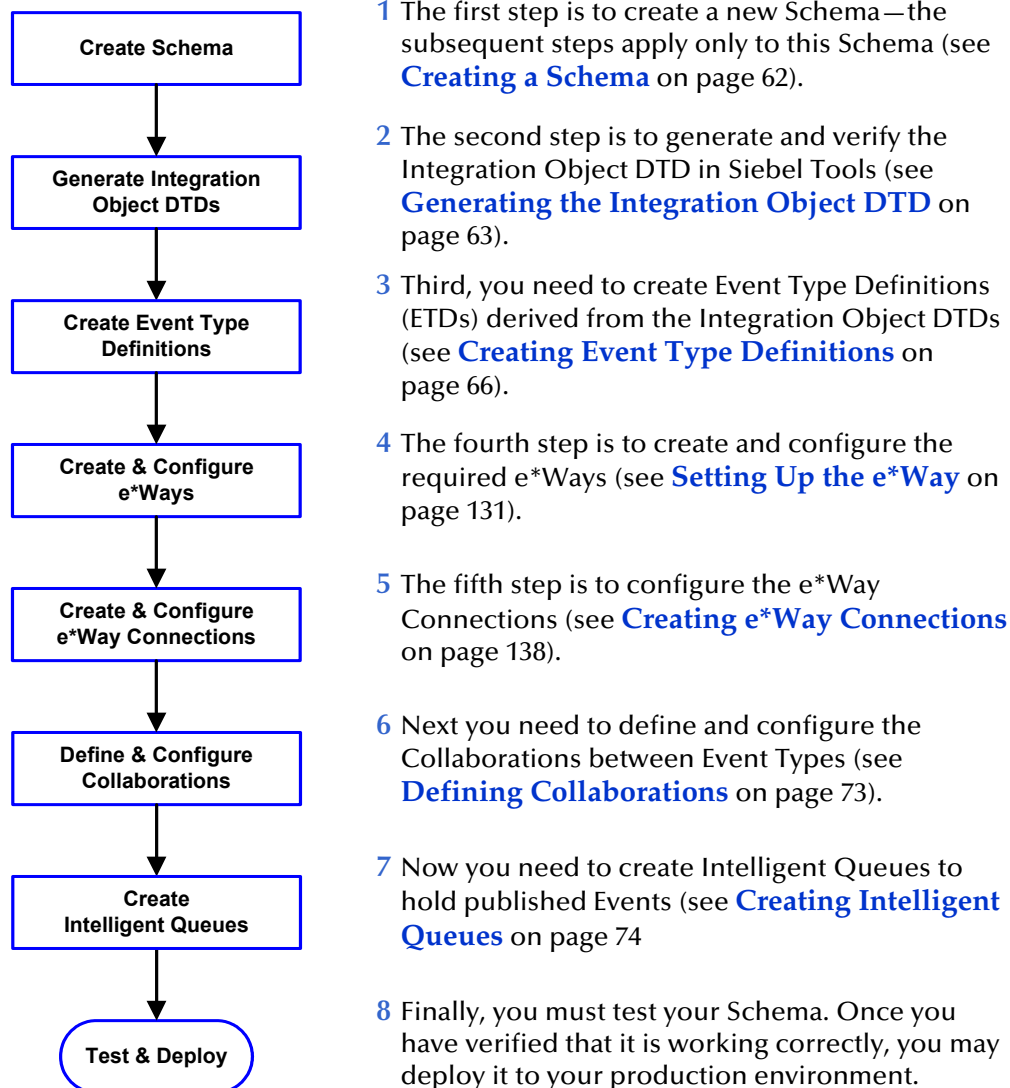
Follow the procedure described in [Web Server Setup](#) on page 31.

## 4.2 Implementation Overview

### 4.2.1 General Sequence

The high-level implementation sequence for a system incorporating the Siebel EAI e\*Way is depicted below.

#### General Implementation Sequence

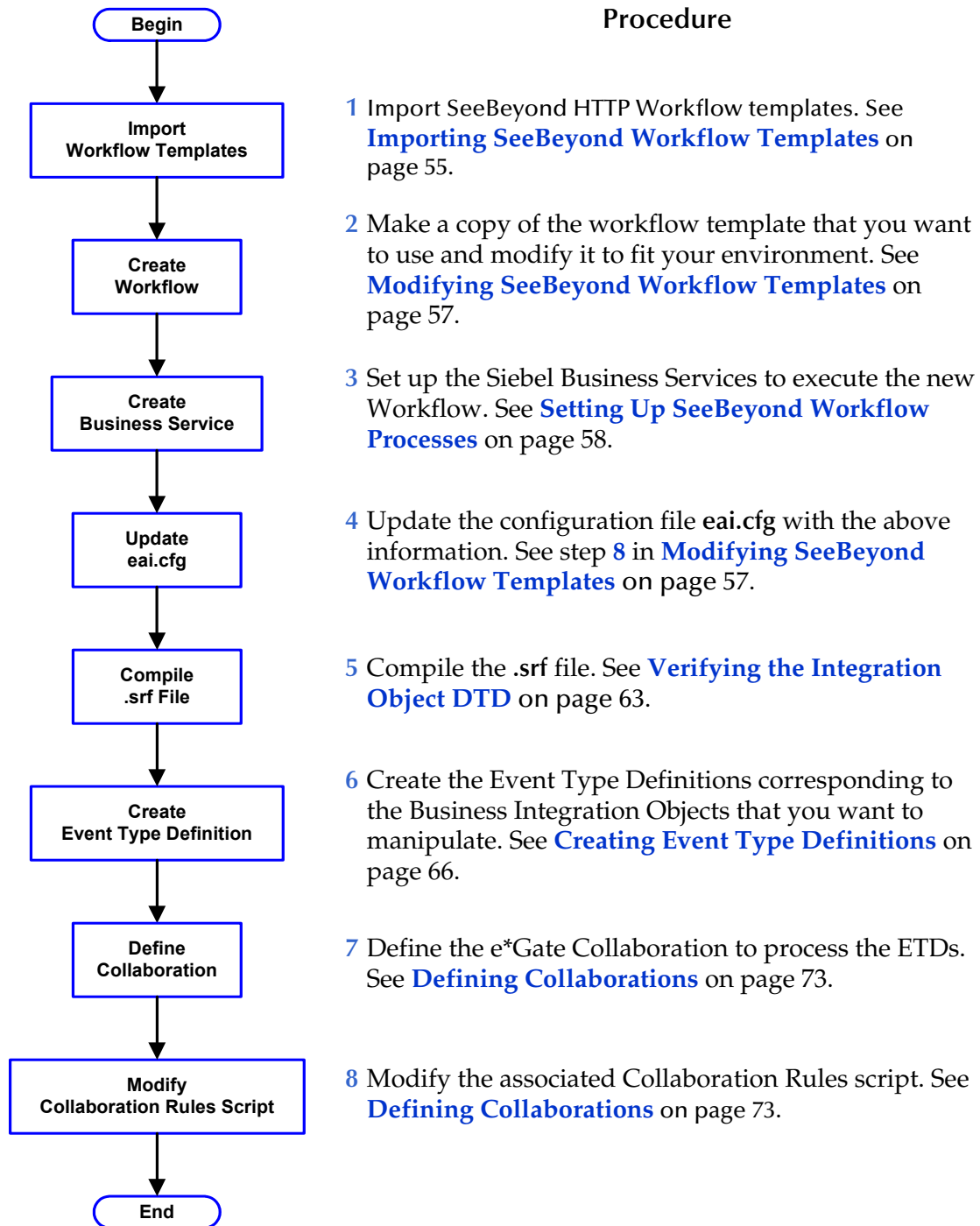


Included with the Siebel EAI e\*Way are several [SeeBeyond Workflow Templates](#), which furnish pre-defined workflows within the Siebel application. More detailed implementation sequences, making use of these templates, appear on the following pages. See [e\\*Gate to Siebel](#) on page 46 and [Siebel to e\\*Gate](#) on page 47.

Also included with the e\*Way are sample schema, which provide pre-defined templates that can be modified to suit your specific requirements.

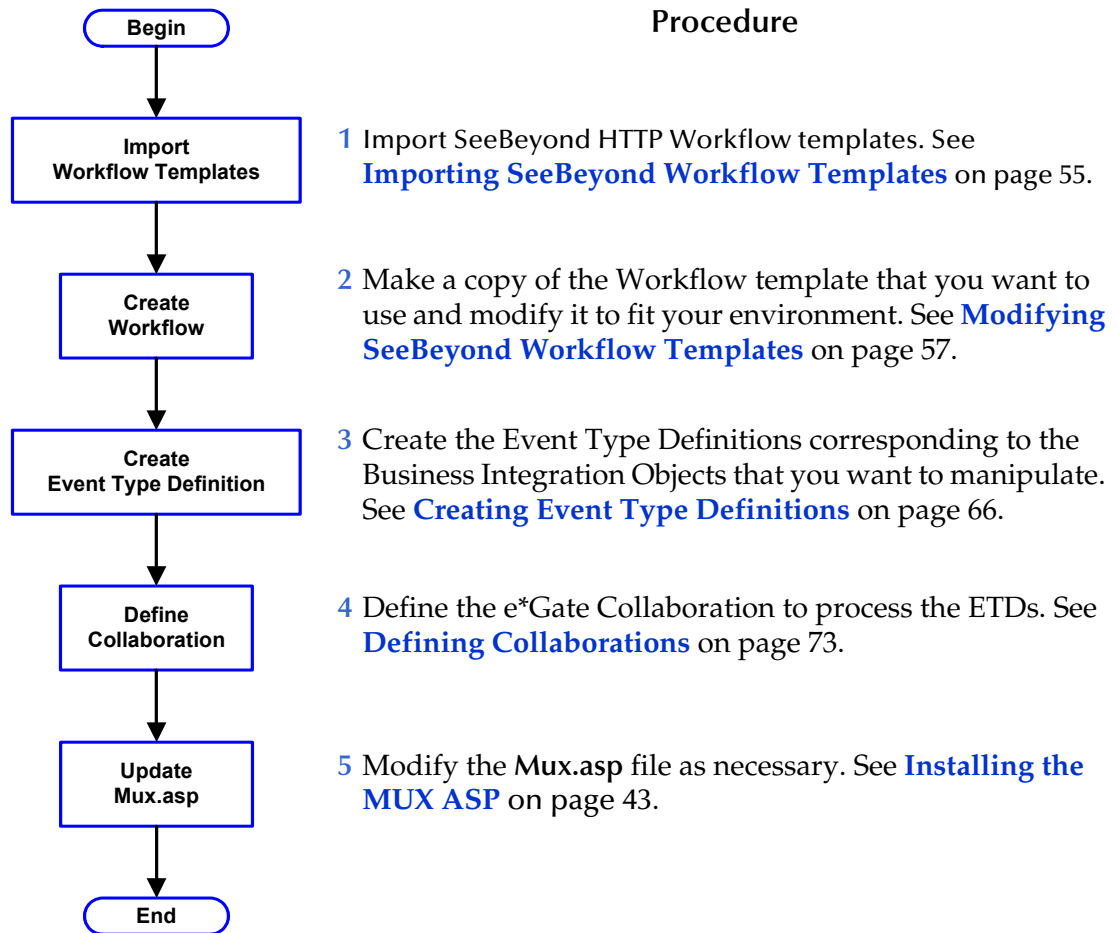
## 4.2.2 e\*Gate to Siebel

### e\*Gate-to-Siebel Implementation



### 4.2.3 Siebel to e\*Gate

#### Siebel-to-e\*Gate Implementation



### 4.2.4 Viewing e\*Gate Components

Use the Navigator and Editor panes of the e\*Gate Schema Designer to view the various e\*Gate components. Note that you may only view components of a single schema at one time, and that all operations apply only to the current schema. All procedures in this chapter should be performed while displaying the **Components** Navigator pane. See the *e\*Gate Integrator User's Guide* for a detailed description of the features and use of the Schema Designer.

---

## 4.3 SeeBeyond Workflow Templates

### 4.3.1 Overview

A set of SeeBeyond Workflow templates is included with the Siebel EAI e\*Way. These workflow templates invoke the following workflow processes to map the data directly to or from the Siebel database.

- SeeBeyond HTTP Delete (see [Figure 16 on page 49](#))
- SeeBeyond HTTP Query (see [Figure 17 on page 50](#))
- SeeBeyond HTTP Update (see [Figure 18 on page 50](#))

Inserts or Updates according to the provided input values.

- SeeBeyond HTTP Execute (see [Figure 19 on page 51](#))

The preferred Workflow for receiving Siebel XML messages from e\*Gate; combines **Delete**, **Query** and **Update** functionality into a single Workflow.

- SeeBeyond HTTP Send (see [Figure 20 on page 51](#))
- SeeBeyond HTTP Send Receive (see [Figure 21 on page 52](#))
- SeeBeyond HTTP Post (see [Figure 22 on page 52](#))

The preferred Workflow for sending Siebel XML messages to e\*Gate; combines **Send** and **Send/Receive** functionality into a single Workflow.

The names entered above are used to set up the Business Service for the sample program supplied with the e\*Way. You should use them as templates to create new processes corresponding to the Workflows you create for your own system.

Examples of XML messages used with different Workflow templates are given in [Siebel XML Messages](#) on page 53.



Following the Screens menu path shown below displays the installed Workflow templates, as shown in Figure 15.

Siebel Workflow Administration > Workflow Processes > All Processes

Figure 15 SeeBeyond Workflow Processes

Name	Business Object	Status	Group	Activation Date/Time	Expiration Date/Time
Export Account (File)	Account	In Progress	Sample		
SeeBeyond HTTP Delete	Account	Active	Sample		
SeeBeyond HTTP Execute	Account	Active	Sample		
SeeBeyond HTTP Post	Account	Active	Sample		
SeeBeyond HTTP Query	Account	Active	Sample		
SeeBeyond HTTP Send	Account	Active	Sample		
SeeBeyond HTTP Send Receive	Account	Active	Sample		
SeeBeyond HTTP Update	Account	Active	Sample		

Clicking the process name to invoke a Workflow Process Designer display for that process, such as shown in Figures 7-13.

Figure 16 DELETE Workflow Template

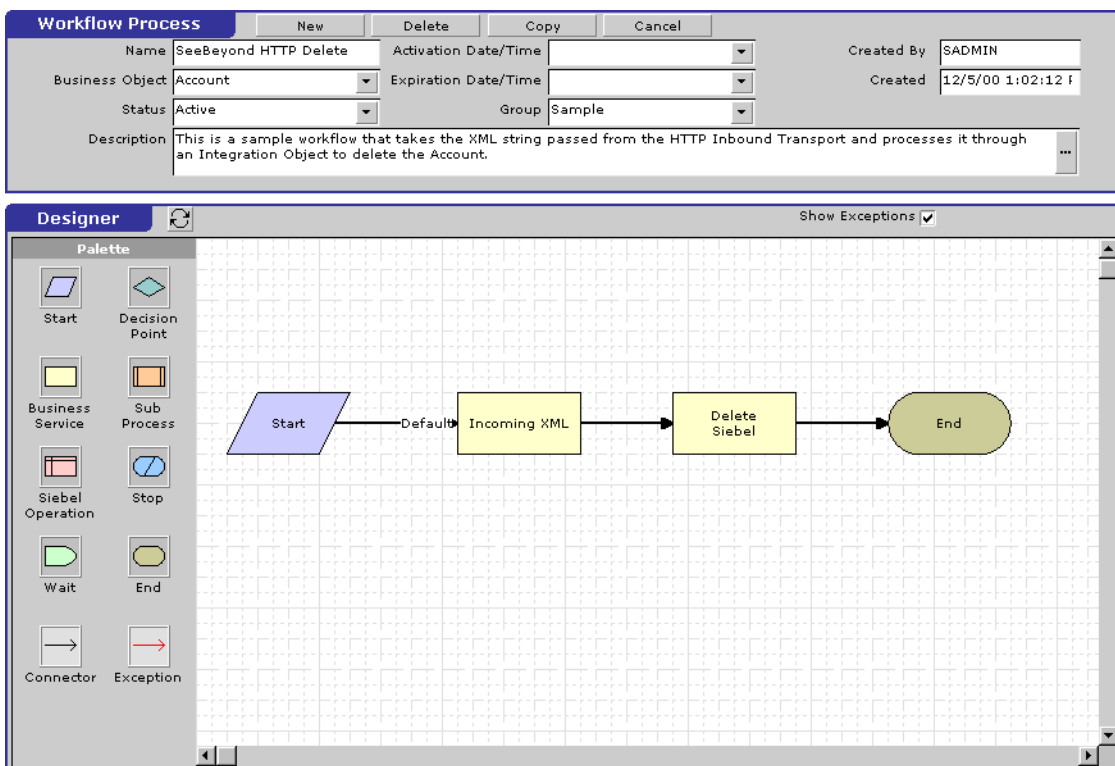


Figure 17 QUERY Workflow Template

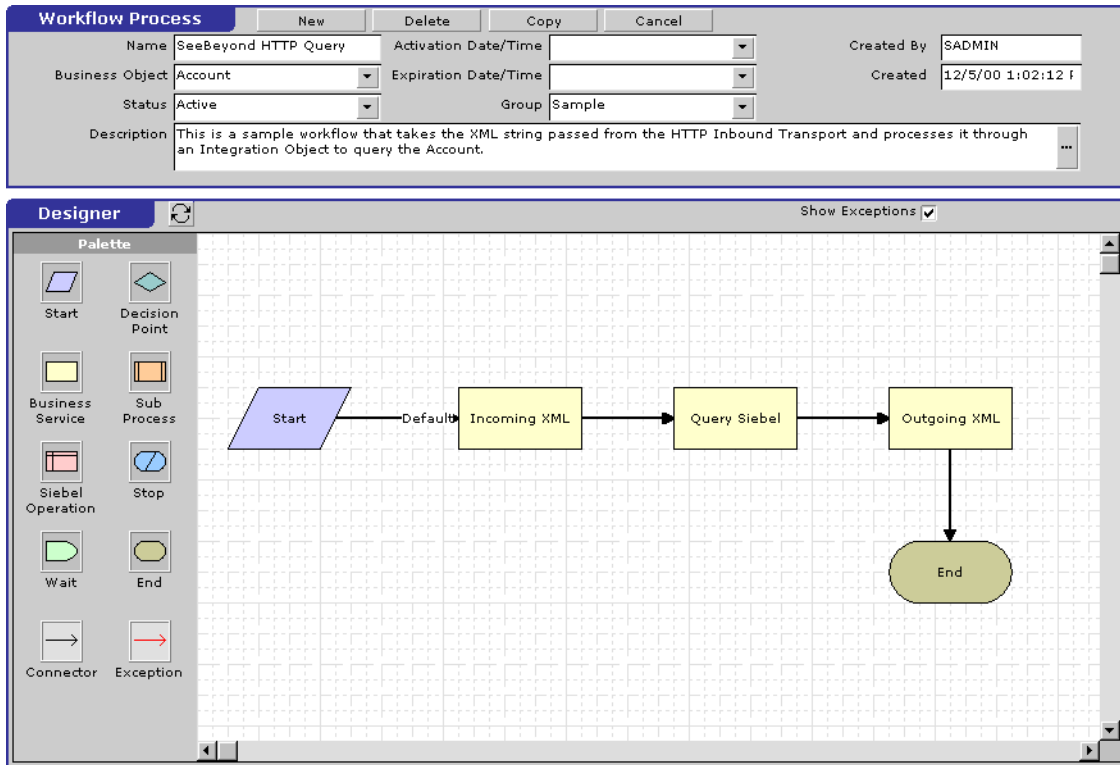


Figure 18 INSERT/UPDATE Workflow Template

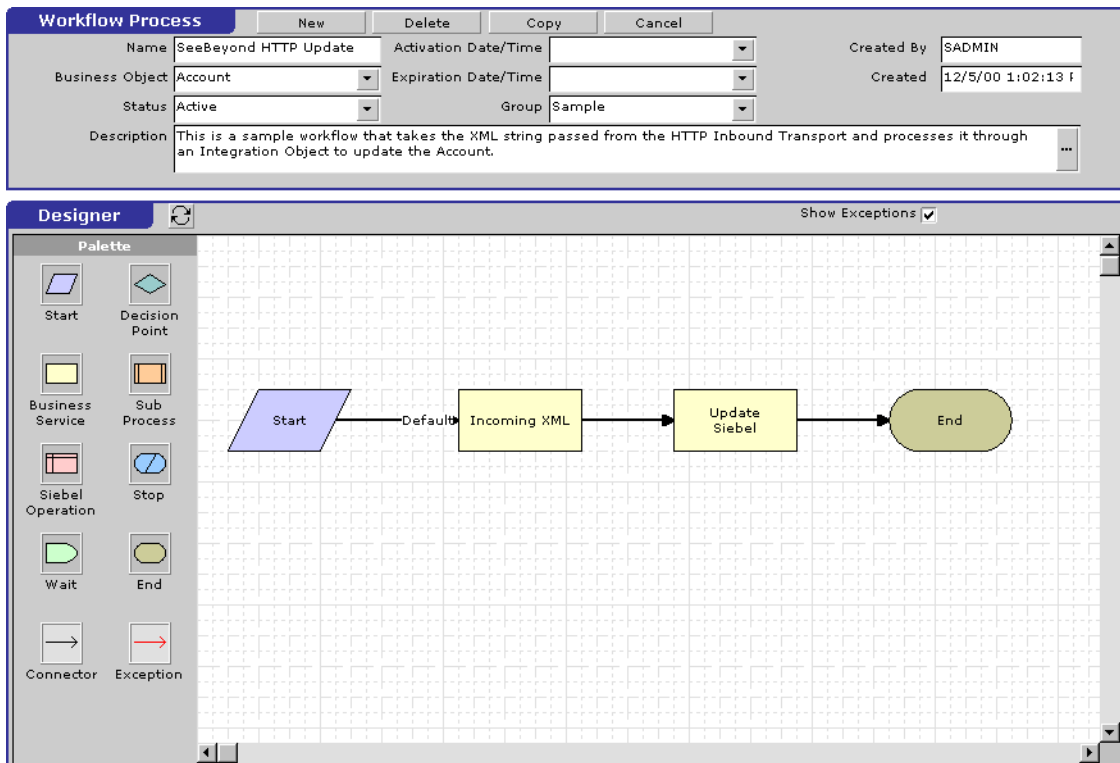


Figure 19 EXECUTE Workflow Template

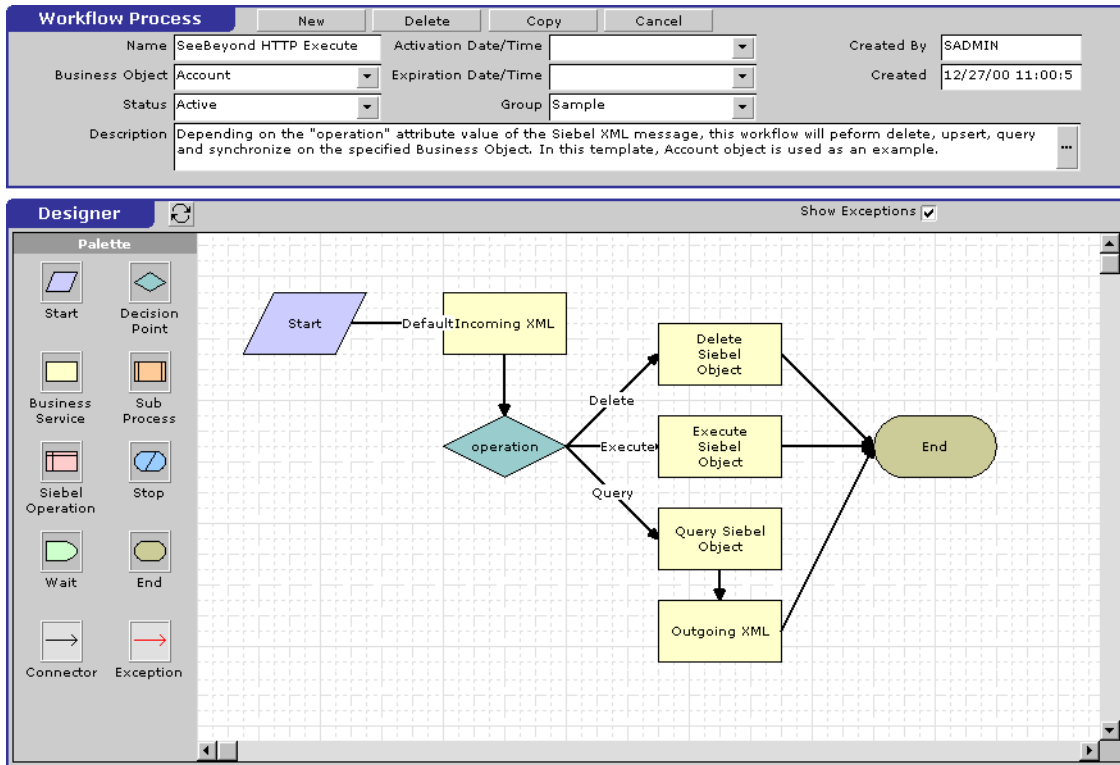


Figure 20 SEND Workflow Template

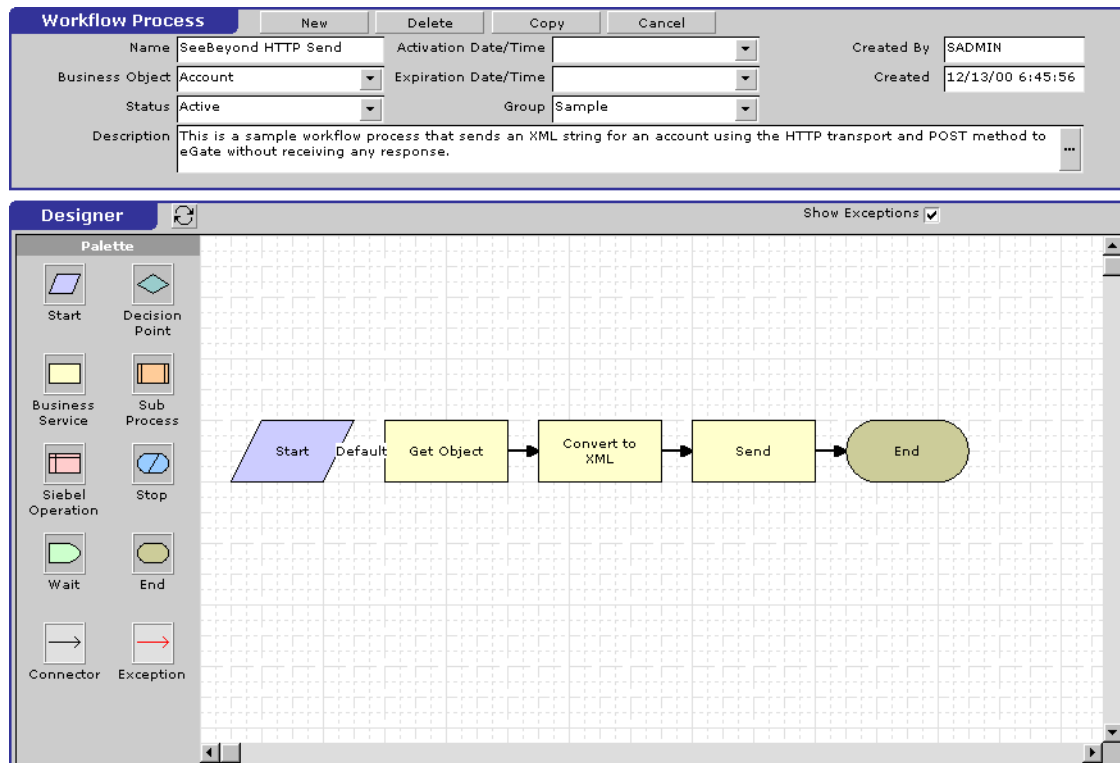


Figure 21 SEND/RECEIVE Workflow Template

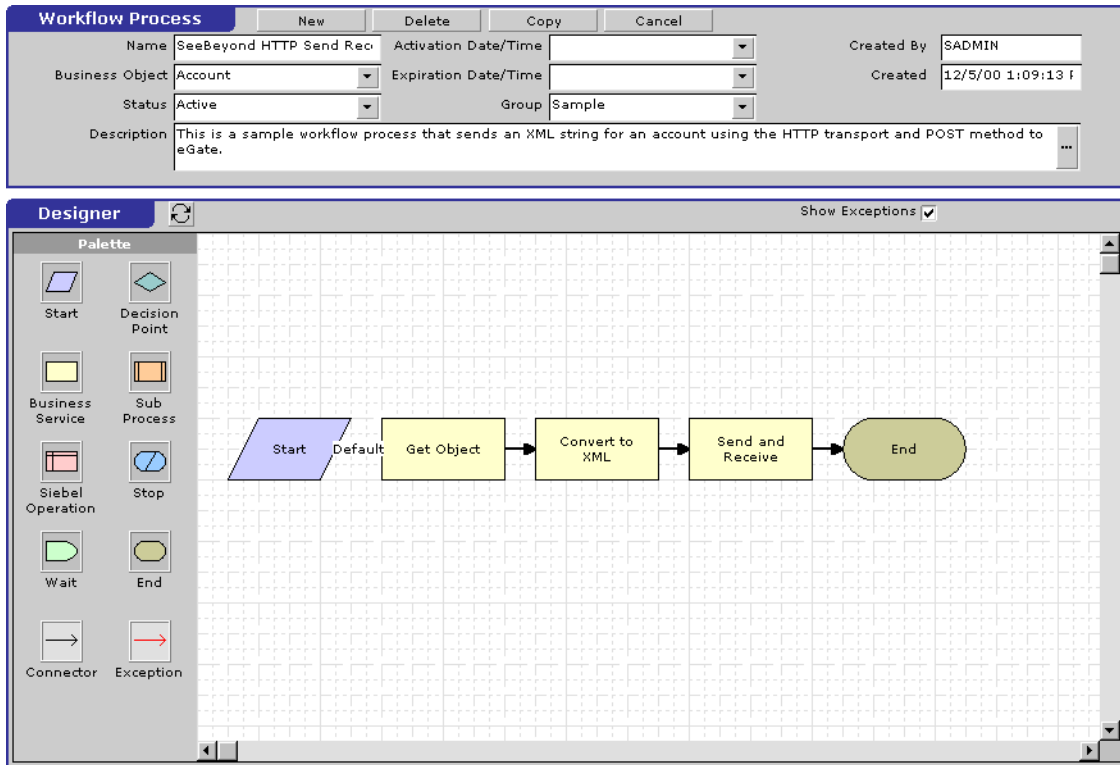
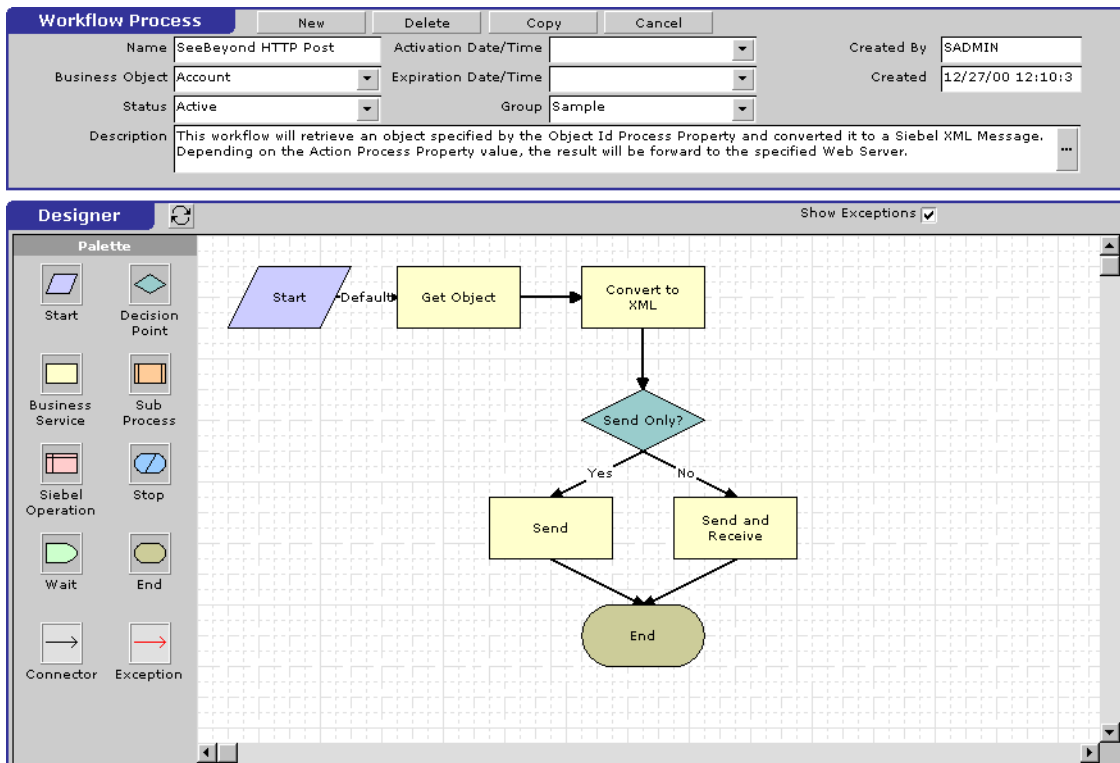


Figure 22 POST Workflow Template



## 4.3.2 Siebel XML Messages

### Format

A Siebel XML Message used by Siebel EAI e\*Way has the following format:

---

Header/Prefix  
Integration Object (in XML format)  
Footer/Suffix

---

where:

**Header =**

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="(Name of Integration Object)" operation=(action)>
```

**Footer =**

```
</SiebelMessage>
```

and **(action)** can be any of the following values:

- ♦ delete
- ♦ query
- ♦ upsert

**Note:** `operation=(action)` is used only with the EXECUTE workflow template.

### Examples

#### Example 1

The following Siebel XML message specifies that the Integration Object that we are dealing with is **Sample Account**. If we send this message to Siebel EAI using the INSERT/UPDATE workflow template, either a new record is generated or an existing record is updated.

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account">
  <ListofSampleAccount>
  <Account>
  <Name>A. K. Parker Distribution</Name>
  <Location>HQ-Distribution</Location>
  <Organization>North American Organization</Organization>
  <Division></Division>
  <CurrencyCode>USD</CurrencyCode>
  <Description>This is THE key account in the AK Parker Family</
  Description>
  <HomePage>www.parker.com</HomePage>
  <LineofBusiness>Manufacturing</LineofBusiness>
  </Account>
  </ListofSampleAccount>
```

```
</SiebelMessage>
```

---

### Example 2

The following Siebel XML message specifies that the Integration Object that we are dealing with is **Sample Account**. If we send this message to Siebel EAI using the **QUERY** workflow template, it returns the object that matches the Name **A. K\***

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account">
  <ListofSampleAccount>
  <Account>
  <Name>A. K*</Name>
  </Account>
  </ListofSampleAccount>
</SiebelMessage>
```

---

### Example 3

The following Siebel XML message provides an example of how to use the **operation** attribute with the **Execute** workflow. Here we send the message to Siebel EAI using the **EXECUTE** workflow template to perform a **query** operation. The result is the same as in Example 2.

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account" operation=query>
  <ListofSampleAccount>
  <Account>
  <Name>A. K*</Name>
  </Account>
  </ListofSampleAccount>
</SiebelMessage>
```

---

### 4.3.3 Importing SeeBeyond Workflow Templates

**Note:** *If you are using Siebel 2000 Japanese, the file SeeBeyondHTTPWorkflowJPN.xml replaces the file SeeBeyondHTTPWorkflow.xml in the following procedure.*

#### To Import the SeeBeyond Workflow Templates

- 1 On the e\*Gate installation CD-ROM, go to:  
`\setup\addons\ewsiebelhttp\common.taz`
- 2 Decompress the .taz file and open the .tar file contained within.
- 3 Extract the file SeeBeyondHTTPWorkflow.xml to an appropriate directory.
- 4 Start Siebel EAI Client and select Siebel Sales.
- 5 Follow the Screens menu path:  
**Siebel Workflow Administration > Workflow Processes > All Processes**
- 6 Click **Import** and browse to the directory that contains SeeBeyondHTTPWorkflow.xml.
- 7 Click **Open** to begin importing the Workflow template.
- 8 Check or set up the following configuration files:

- ◆ In the file SWEApp\eapps.cfg, verify that the following section is specified correctly:

```
[/eai]
ConnectString = siebel.TCPIP.none.none://<Your Gateway
Server>:3230/ <Your Enterprise Server>/eaiObjMgr/<Your App
Server>
EnableExtServiceOnly = TRUE
```

- ◆ For the e\*Gate-to-Siebel sample, add the following sections in the file siebsrvr\eai.cfg:

```
[HTTP Services]
...
SEEBEYOND_HTTP_DELETE = SB_HTTP_DELETE
SEEBEYOND_HTTP_QUERY = SB_HTTP_QUERY
SEEBEYOND_HTTP_UPDATE = SB_HTTP_UPDATE
SEEBEYOND_HTTP_EXECUTE = SB_HTTP_EXECUTE

[SB_HTTP_DELETE]
Mode = Document
Service = SeeBeyond HTTP Delete
Method = RunProcess

[SB_HTTP_QUERY]
Mode = Document
Service = SeeBeyond HTTP Query
Method = RunProcess

[SB_HTTP_UPDATE]
Mode = Document
Service = SeeBeyond HTTP Update
Method = RunProcess
```

```
[SB_HTTP_EXECUTE]  
Mode = Document  
Service = SeeBeyond HTTP Execute  
Method = RunProcess
```



## 4.3.4 Modifying SeeBeyond Workflow Templates

**Note:** *The SeeBeyond Workflow templates provided with the e\*Way use **Account** as the Business Object—you must modify them for use with a different Business Object.*

### To Modify a SeeBeyond Workflow Template

- 1 Log in to **Siebel Client 6.0**, designating the appropriate Siebel server.
- 2 Follow the **Screens** menu path:  
**Siebel Workflow Administration > Workflow Processes > All Processes**
- 3 Highlight the SeeBeyond Workflow Process template you want to modify.
- 4 Right-click and select **Copy Record**.
- 5 Rename the copied Process.
- 6 Specify the Business Object to which you want to apply the template, and any other fields that may be necessary (for example, Description).
- 7 After modifying a Workflow template you must create the Business Service to execute it, using the supplied Workflow processes as templates. This procedure is described in the following section.
- 8 A new Services section should be added to your `siebsrvr\ei.cfg` file, as shown in the preceding section.

For example, if you have a Business Service named Employee Execute, you should add the following lines to the `ei.cfg` file:

```
[HTTP Services]
...
EMPLOYEE_EXECUTE = EE

[EE]
Mode = Document
Service = Employee Execute
Method = RunProcess
```

### 4.3.5 Setting Up SeeBeyond Workflow Processes

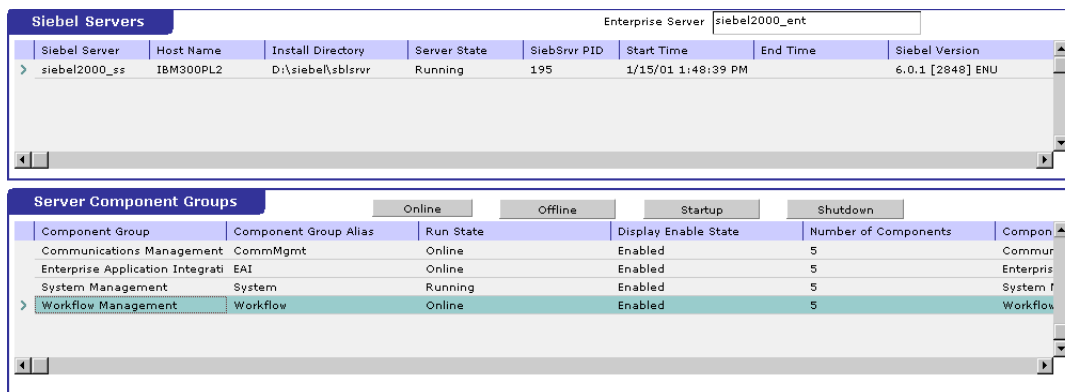
The Workflow processes invoked by the SeeBeyond Workflow Templates must be set up in Siebel Business Services.

**Note:** The names entered in step 8 are used to set up the Business Service for the sample program supplied with the e\*Way. You should use them as templates to create new processes corresponding to the Workflows you create for your own system.

To set up the Business Service to execute the Workflow

- 1 Make sure the following services are running:
  - ♦ Siebel Gateway Server
  - ♦ Siebel Server
  - ♦ World Wide Web Publishing Service
- 2 Log in to Siebel Sales 6.0.
- 3 Follow the Screens menu path:  
**Server Administration > Servers > Server Component Groups**

**Figure 23** Server Component Groups



- 4 Make sure that **Workflow Management** is **Online** and **Enabled**.
- 5 Log in to Siebel Tools 6.0 and designate the server as the database by entering **sadmin, sadmin, server**.
- 6 In Object Explorer, go to **Siebel Objects > Project** and lock the project (see Figure 24).

**Figure 24** Lock Project

Projects							
	Name	Changed	Inactive	Locked	Locked By Name	Locked Date	Language
	EAI						
	EAI Account						
	EAI Business Services						ENU
	EAI Converter Services						ENU
	EAI CreditCard						ENU
	EAI DTE						ENU
	EAI Demo						
	EAI Design						
	EAI Dispatch Service						ENU
	EAI Envelope Services						
	EAI Product						
	EAI Queue						
	EAI Sample Perf Test						ENU
	EAI Tax						ENU
>	EAI Test			✓	SADMIN	2002/03/21 19:33:	ENU
	EIM						ENU
	EIM Accounts and Quotes						ENU
	EIM Activity						ENU
	EIM Administrative						ENU
	EIM Agreement						ENU
	EIM Asset Management						ENU
	EIM Auction Item						ENU
	EIM Audit Trail						ENU
	EIM Bussiness Unit						ENU
	EIM CHAMP						ENU
	EIM CTI						ENU
	EIM Call Script						ENU

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z \*

- In Object Explorer, go to **Business Service**, make a copy of **Workflow Process Manager** (menu path **Edit > Copy Record**).

**Figure 25** Business Services View - Workflow Process Manager

Name	Changed	Project	Cache	Class	Display Name
UI Import/Export		Front Office Workflow		CSSUIImpExpService	Import/Export Integration Object
Web Engine Interface		SWE		CSSServiceSWEInterface	
Webphone Push		Webphone Push		CSSWAPPushService	Webphone Push
Workflow FlowChart UI		Front Office Workflow	✓	CSSSvcWfFlowCht	
Workflow Process Manager		Front Office Workflow	✓	CSSWfEngine	Workflow Process Manager
Workflow Process Manager (Server R		Front Office Workflow	✓	CSSSrvrReqSyncService	
Workflow Siebel Operation		Front Office Workflow		CSSBcOperService	Workflow Siebel Operation
Workflow UI Utilities		Front Office Workflow		CSSWfUIUtilService	
Workflow Utilities		Front Office Workflow		CSSWfUtilService	
XML Converter		Business Service	✓	CSSXMLCnvService	
XML Converter (Data in Child PropSet		Business Service	✓	CSSXMLCnvService	
XML Gateway		XML Gateway		CSSServiceXMLGateway	

8 Type the Process Name into the Name and Display Names fields, as shown in Figure 26 (this name is specified in the eai.cfg file).

For e\*Gate-to-Siebel operation, perform this step for:

- ◆ SeeBeyond HTTP Delete
- ◆ SeeBeyond HTTP Execute
- ◆ SeeBeyond HTTP Query
- ◆ SeeBeyond HTTP Update

**Figure 26** Business Services View - Renamed Fields

Name	Changed	Project	Cache	Class	Display Name
Proposal Integrator		Proposal		CSSProposalUIService	
Query Exporter		System		CSSQueryExporter	
Report Execution Engine		Report Server		CSSRunReportService	
Report Viewing Engine		Report Server		CSSViewReportService	
SeeBeyond HTTP Execute	✓	EAI Test	✓	CSSWfEngine	SeeBeyond HTTP Execute
SeeBeyond HTTP Query	✓	EAI Test	✓	CSSWfEngine	SeeBeyond HTTP Query
SeeBeyond HTTP Update	✓	EAI Test	✓	CSSWfEngine	SeeBeyond HTTP Update
Siebel Anywhere Upgrade		Software Upgrade		CSSSvcAnyUpgr	
Siebel Tools Actuate Report Generator		Siebel Tools		CSSActuateReportGenerator	
Siebel Tools Applet Designer		Siebel Tools		CSSAppletDesigner	
Siebel Tools Application Upgrader		Siebel Tools		CSSMerge	

9 In Object Explorer, go to **Business Service > Business Service Userprops:**

A Type **ProcessName** into the **Name** field.

B Type the actual Process Name into the **Value** field (see Figure 27).

For e\*Gate-to-Siebel operation, perform this step for:

- ♦ SeeBeyond HTTP Delete
- ♦ SeeBeyond HTTP Execute
- ♦ SeeBeyond HTTP Query
- ♦ SeeBeyond HTTP Update

**Figure 27** Business Services User Properties

The screenshot shows two tables in a software interface. The top table, titled "Business Services", has columns: Name, Changed, Project, Cache, Class, and Disp. It contains two rows: "SeeBeyond HTTP Update" (with checkmarks in Changed and Cache, and "CSSwfEngine" in Class) and "Siebel Anywhere Upgrade" (with "Software Upgrade" in Project and "CSSSvcAnyUpg" in Class). The bottom table, titled "Business Service User Props", has columns: Name, Changed, Value, Inactive, and Comments. It contains one row: "ProcessName" (with a checkmark in Changed and "SeeBeyond HTTP Update" in Value).

Business Services					
Name	Changed	Project	Cache	Class	Disp
SeeBeyond HTTP Update	✓	EAI Test	✓	CSSwfEngine	See
Siebel Anywhere Upgrade		Software Upgrade		CSSSvcAnyUpg	

Business Service User Props				
Name	Changed	Value	Inactive	Comments
ProcessName	✓	SeeBeyond HTTP Update		

The Workflow Processes you create in the Business Services are similar to those shown in Figure 26.

## 4.4 Creating a Schema

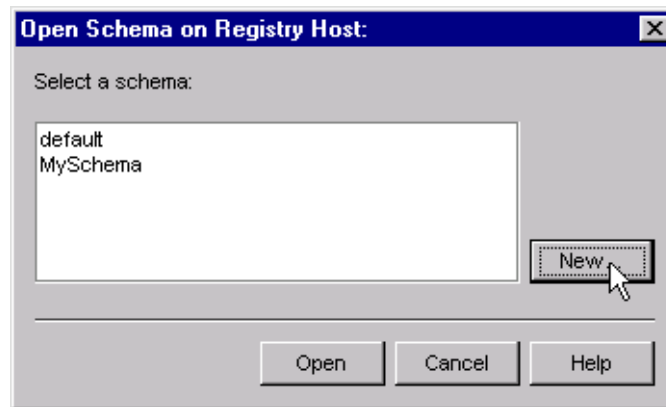
A schema is the structure that defines e\*Gate system parameters and the relationships between components within the e\*Gate system. Schemas can span multiple hosts.

Because all setup and configuration operations take place within an e\*Gate schema, a new schema must be created, or an existing one must be started before using the system. Schemas store all their configuration parameters in the e\*Gate Registry.

To select or create a schema

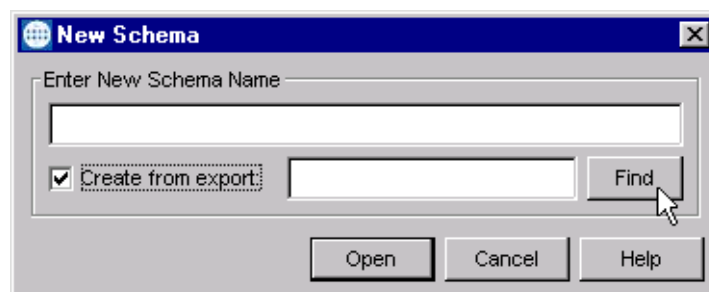
- 1 Invoke the **Open Schema** dialog box and **Open** an existing schema or click **New** to create a new schema.

**Figure 28** Open Schema Dialog



- 2 Clicking **New** invokes the **New Schema** dialog box (Figure 29).

**Figure 29** New Schema Dialog



- 3 Enter a new schema name and click **Open**.
- 4 The e\*Gate Schema Designer then opens under your new schema name.
- 5 From the **Options** menu, click on **Default Editor** and select **Monk**.
- 6 Select the **Components** tab, found at the bottom of the Navigator pane of the e\*Gate Schema Designer window.
- 7 You are now ready to begin creating the necessary components for this new schema.

---

## 4.5 Generating the Integration Object DTD

### To generate the DTD

- 1 In Siebel Tools, click on an Integration Object to activate it.
- 2 Click **Generate Schema**, which displays the initial page of the Generate Schema Wizard.
- 3 Select the **EAI XML DTD Generator** business service.
- 4 Select a location to store the resulting file.
- 5 Click **Finish**.

The Wizard generates an XML DTD of the Integration Object you selected. You can use this DTD to create an ETD using the SeeBeyond XML Converter/ETD Builder, as described in [Using the DTD Builder](#) on page 66.

**Note:** *There is a defect in Siebel's Integration Object DTD. Element names are **not** unique. It has been reported as a product defect #12-1TQJN7. Following is the workaround recommended by Siebel:*

- Do **not** specify an XML Parent Element name in the Integration Component
- Add the prefix **ListOf** to the XML Tag

---

## 4.6 Verifying the Integration Object DTD

The next step is to confirm that the Integration Object DTD is generated correctly. You should export the DTD and run the SeeBeyond XML Converter/ETD Builder to verify that it can generate the Event Type Definition correctly. An incorrect ETD build usually indicates that the Siebel DTD has a repeated element name, in which case you need to modify the Integration Object. See the *Note* in [Generating the Integration Object DTD](#) on page 63.

The supplied sample program uses the **Sample Account** Integration Object, the integrity of which is verified as follows:

- 1 Navigate to the **Integration Object** view and select **Sample Account**.

**Figure 30** Integration Object View - Sample Account

Name	Changed	Project	External Name	Base Object Type	XML Tag
SAP IDOC Wizard - Get RFC Table En		EAI Design	TABLE_ENTRIES_GET_VIA_RFC	SAP BAPI Output	ListofTable_Entries_C
SAP Wizards - Get Field Info (BAPI In		EAI Design	DDIF_FIELDINFO_GET	SAP BAPI Input	Listofddif_Fieldinfo
SAP Wizards - Get Field Info (BAPI O		EAI Design	DDIF_FIELDINFO_GET	SAP BAPI Output	Listofddif_Fieldinfo
Sample Account	✓	EAI Test	Account	Siebel Business Object	ListofSampleAccount
Sample Contact		EAI Test	Contact	Siebel Business Object	ListOfSampleContac
Sample Employee		EAI Test	Employee	Siebel Business Object	ListOfSampleEmplo
Sample Order		EAI Test	Order Entry	Siebel Business Object	ListOfOrderEntrySe
Sample Service Request		EAI Test	Service Request	Siebel Business Object	ListOfSampleServic
Siebel OLEDB: AccContact		OLEDB	AccContact	OLE DB	Listofaccounts

- 2 Navigate to the **Integration Object > Integration Component** view.
  - A In the **Business Address/XML Parent Element** field, type **ListOf**.
  - B In the **Contact/XML Parent Element** field, type **ListOf**.

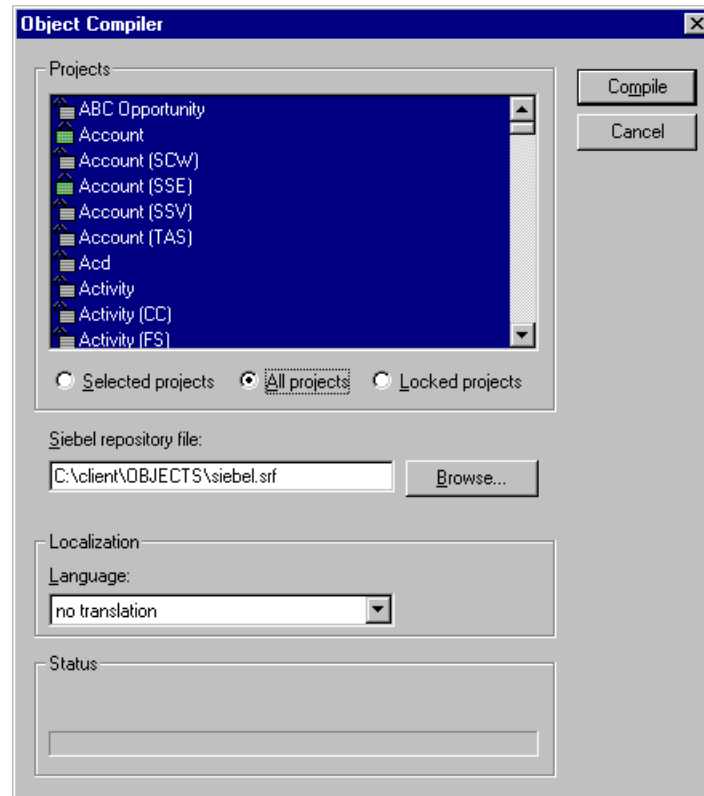
**Figure 31** Integration Object > Integration Component View

External Name Context	Name	Changed	XML Parent Element	XML Tag
Account	Account			Account
Business Address	Business Address	✓	ListOf	BusinessAddress
Contact	Contact	✓	ListOf	Contact
Contact_Business Address	Contact_Business Address			Contact_BusinessAddress
Sub Account	Sub Account			SubAccount

- 3 Stop the following services, in this order:
  - A Siebel Server.
  - B Siebel Gateway Name Server.
- 4 Follow the menu path **Repository > Compile**, which invokes the **Object Compiler** dialog box.



**Figure 32** Objects Compiler Dialog Box



- 5 If you have completed all projects, select the **All Projects** option button; otherwise, select **Locked Projects** to shorten the compilation time.
- 6 Select the Siebel repository file `\client\OBJECTS\siebel.srf`.
- 7 Click **Compile** and copy the Siebel `.srf` file to the `siebel\sblsrvr\OBJECTS` directory.
- 8 Start the following services, in this order:
  - A Siebel Gateway Name Server.
  - B Siebel Server.
- 9 Verify that the EAI Object Manager is running.

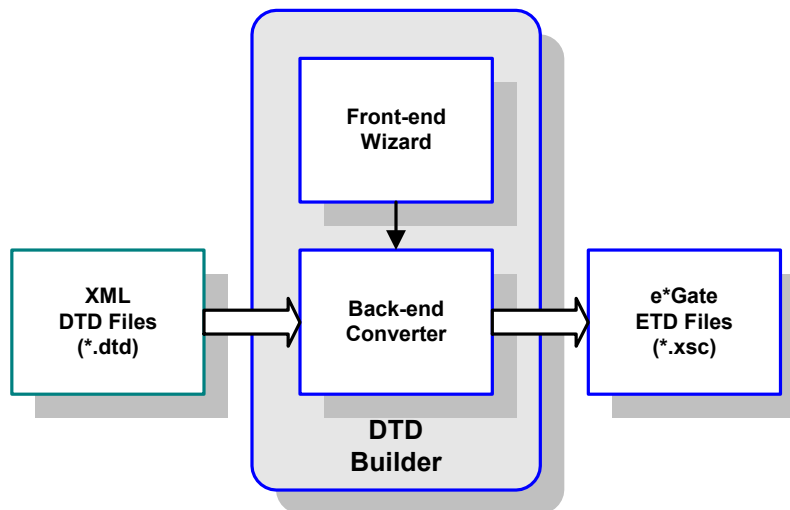
## 4.7 Creating Event Type Definitions

Before e\*Gate can process any data to or from a Siebel EAI system, you must create an Event Type Definition to package and route that data within the e\*Gate system. The ETD is derived from a Siebel Integration Object Data Type Definition (DTD). See the *e\*Gate Integrator User's Guide* for additional information about Event Type Definitions and the e\*Gate ETD Editor.

### 4.7.1 Using the DTD Builder

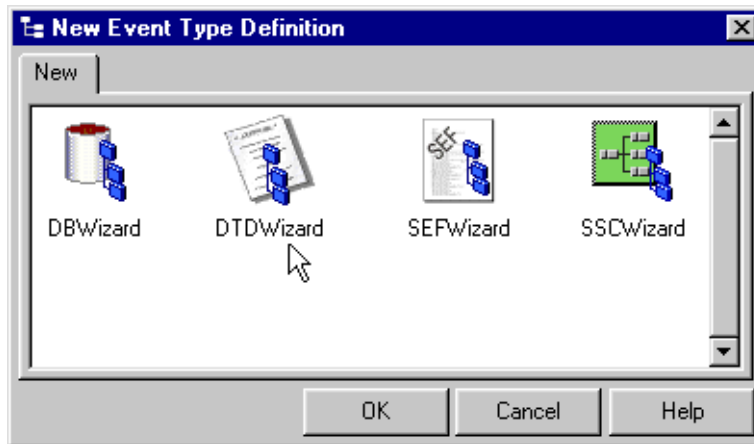
The ETD Editor contains a DTD Builder, which takes a Siebel XML DTD and converts it to a .xsc file. See the *XML Toolkit User's Guide* for detailed information on the DTD Builder.

Figure 33 DTD Builder



To access the Wizard, select the **New** option in the ETD Editor's **File** menu. The New Event Type Definitions window appears, displaying all installed ETD Builder Wizards. Select the **DTD Wizard**, and follow the instructions presented in the dialog.

**Figure 34** New Event Type Definitions Window



To run the DTD Builder

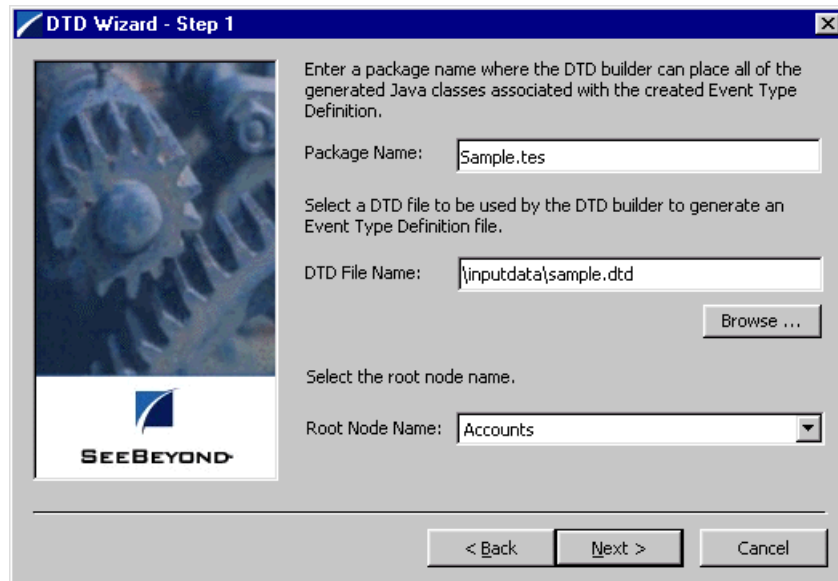
- 1 Invoke the DTD Wizard by clicking its icon.

**Figure 35** DTD Wizard – Introduction



- 2 Read the instructions carefully, and click Next. Step 1 of the DTD Wizard dialog appears (see Figure 36).

Figure 36 DTD Wizard — Step 1



3 Enter the following information:

♦ **Java Package Name**

Type in the name you want to give the Java package, for example, **Sample.tes**. This name must conform to Java package name requirements. See the appropriate Java documentation for details.

♦ **DTD File Name**

Type in the name of the DTD file you want to convert. Click **Browse** to access an Open (file selection) dialog box, allowing you to choose the desired file.

♦ **Root Node Name**

This text box is a pull-down menu. Select the desired root node name from the menu. For more information on root nodes and ETDs, see the *e\*Gate Integrator User's Guide*.

4 When you are finished, click **Next**. Step 2 of the DTD Wizard dialog appears (see Figure 37).

Figure 37 DTD Wizard — Step 2



- 5 Specify the options you want used by the DTD Builder.
  - ◆ Allow whitespace in EMPTY elements
  - ◆ Ignore #FIXED attributes
  - ◆ Ignore all attributes
  - ◆ Include XML declarations
  - ◆ Include DOC Type Reference (if selected, specify DTR name)
- 6 When you are finished, click Next. Step 3 of the DTD Wizard dialog appears (see Figure 38).

Figure 38 DTD Wizard — Step 3



- 7 Review the information you have entered in the Wizard. If it is correct, click **Finish** to generate a Java ETD (.xsc file) from the original DTD file.

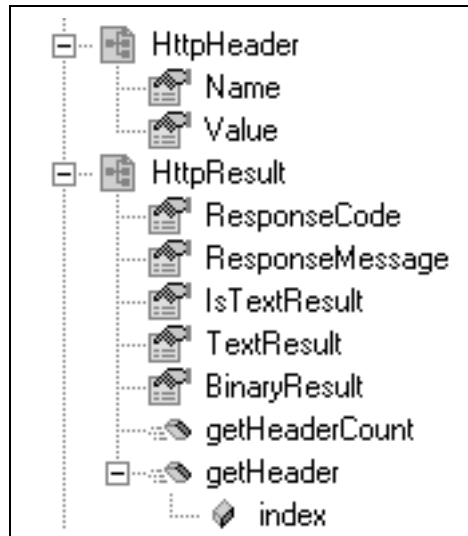
The Wizard closes, and the new ETD appears in the ETD Editor Main window. See the *e\*Gate Integrator User's Guide* for details on how to use this editor, including an explanation of the information it shows.

- 8 To save the new ETD, click the **Save** button on the Toolbar or select the **Save** command from the **File** menu. A Save dialog box appears.
- 9 Select the desired directory location, give the new ETD your desired name, and click **Save**. The ETD Editor saves the new Java ETD.
- 10 You can continue to use the ETD Editor or select the **Close** command from the **File** menu to exit the GUI.

**Note:** *The ETD nodes created using the DTD Builder appear shaded in the ETD Editor, indicating that you cannot edit an ETD created by the Builder.*

After converting the DTD to an ETD, return to the e\*Gate Schema Designer to verify the process (see Figure 39).


Figure 39 ETD Structure Example



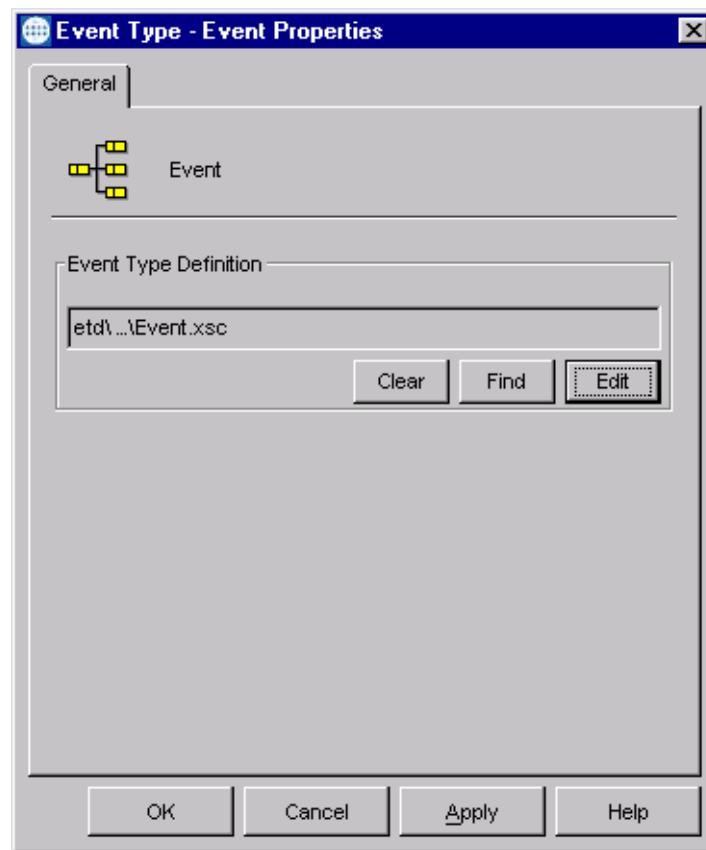
## 4.8 Assigning ETDs to Event Types

After you have created the e\*Gate system's ETD files, you can assign them to Event Types you have already created.

### To assign ETDs to Event Types

- 1 In the Schema Designer window, select the **Event Types** folder in the Navigator/Components pane.
- 2 In the Editor pane, select one of the Event Types you created.
- 3 Right-click on the Event Type and select **Properties** (or click  in the toolbar). The Event Type Properties dialog box appears. See Figure 40.

**Figure 40** Event Type Properties Dialog Box



- 4 Under Event Type Definition, click **Find**.

The Event Type Definition Selection dialog box appears; it is similar to the Windows Open dialog box.

**Note:** Clicking **New** in the Event Type Properties dialog box opens the ETD Editor window, allowing you to create a new ETD.



- 5 Open the **etd** folder, then select the desired file name (.xsc).
- 6 Click **Select**. The file populates the Event Type Definition field.
- 7 To save any work in the properties dialog box, click **Apply** to enter it into the system.
- 8 When finished assigning ETDs to Event Types, click **OK** to close the properties dialog box and apply all the properties.

Each Event Type is associated with the specified Event Type Definition.

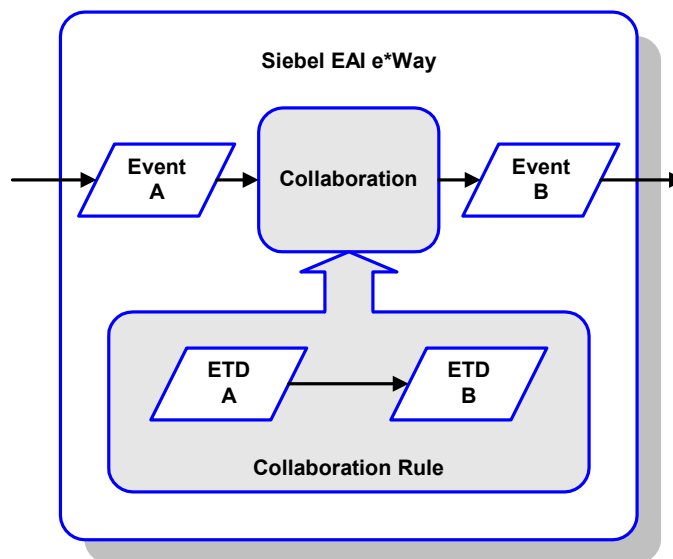
---

## 4.9 Defining Collaborations

After you have created the required Event Type Definitions, you must define a Collaboration to transform the incoming Event into the desired outgoing Event.

Collaborations are e\*Way components that receive and process Event Types, then forward the output to other e\*Gate components. Collaborations consist of the Subscriber, which “listens” for Events of a known type or from a given source, and the Publisher, which distributes the transformed Event to a specified recipient. The same Collaboration cannot be assigned to more than one e\*Gate component.

**Figure 41** Collaborations



### 4.9.1 The Java Collaboration Rules Editor

Java Collaborations are defined using the e\*Gate Java Collaboration Rules Editor. Note that the Java Collaboration environment supports multiple source and destination ETDs. The file extension for Java Collaboration Rules is **.xpr**. See the *e\*Gate Integrator User's Guide* for descriptions of the Java Collaboration Rules Editor and its use.

---

## 4.10 Creating Intelligent Queues

IQs are components that provide nonvolatile storage for Events within the e\*Gate system as they pass from one component to another. IQs are *intelligent* in that they are more than just a “holding tank” for Events. They actively record information about the current state of Events.

Each schema must have an IQ Manager before you can add any IQs to it. You must create at least one IQ per schema for published Events within the e\*Gate system. Note that e\*Ways that publish Events externally do not need IQs.

For more information on how to add and configure IQs and IQ Managers, see the *e\*Gate Integrator System Administration and Operations Guide*. See the *e\*Gate Integrator Intelligent Queue Services Reference Guide* and the *SeeBeyond JMS Intelligent Queue User's Guide* for complete information on working with IQs.

## 4.11 Using the Siebel EAI ETD in a Collaboration

The Siebel EAI ETD contains the following attributes which users can **set** and/or **get**:

SWExtCmd	deleteSource
SWExtData	executeSource
SWExtSource	querySource
URL	updateSource
xmlData	

The ETD also contains the following methods:

delete	insert
getResultData	postSiebelForm
getResponseHeaderString	query

### 4.11.1 Overview

See [Call Sequence](#) on page 76 for a detailed call sequence.

#### To Post data to Siebel

You can set most of the relevant parameters for posting such as **URL**, **SWExtSource**, **SWExtCmd** and **SWExtData**, and then call the **postSiebelForm** method to perform the HTTP post using the **set** parameters. Note that the username and password are always obtained from the configuration file. When setting **SWExtData**, you insert the correct XML string to pass, based on the operation to be performed. See [Chapter 9](#) for details of these Java methods and their attributes.

#### To get the HTTP response

After the call to **postSiebelForm()**, the HTTP response can be obtained by calling **getResultData()**.

#### To get the response header

After the call to **postSiebelForm()**, the HTTP response header can be obtained by calling **getResponseHeaderString()**.

### 4.11.2 Helper Methods

You have the option of specifying your designated sources for **execute**, **update**, **delete** and **query**. These source names are used in the helper methods **insert**, **delete**, and **query**.

**Note:** *The helper methods may be used **only** if you want to use the following hard-coded XML tags for **SWExtData** (along with the value they set for the **xmlData** attribute):*

**For insert:**

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account" operation="upsert"> + xmlData +
</SiebelMessage>
```

**For delete:**

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account"
  operation="delete"><ListofSampleAccount><Account><Name> +
xmlData +</Name></Account></ListofSampleAccount></SiebelMessage>
```

**For query:**

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account"
  operation="query"><ListofSampleAccount><Account><Name> +
xmlData +</Name></Account></ListofSampleAccount></>
```

### 4.11.3 Call Sequence

#### To Post data to Siebel and retrieve a response

- 1 Specify **SWEEExtSource**, which includes **update**, **insert**, and **delete**; for example:

```
setSWEEExtSource(Siebel2000API.SBYN_UPDATE_SOURCE);
```

where Siebel2000API.SBYN\_UPDATE\_SOURCE refers to the service "SEEBEYOND\_HTTP\_UPDATE" that you specified in the HTTP Service section of the eai.cfg file.

- 2 Specify **SWEEExtCmd**, which currently only uses **execute**; for example:

```
setSWEEExtCmd("Execute");
```

- 3 Specify the Siebel Integration Object, for example:

```
setIntegrationObjectName("Sample Account");
```

- 4 Set the XML message, for example:

```
setXmlData(getinAccount().toString());
```

- 5 Format the message with prefix and suffix to create a Siebel Message, for example:

```
setSWEEExtData(getoutSiebel().getTAG_SIEBEL_EXECUTE_UPSERT_PREFIX()
  + getoutSiebel().getXmlData() +
  getoutSiebel().getTAG_SIEBEL_MSG_SUFFIX());
```

- 6 Post the message to Siebel, for example:

```
postSiebelForm();
```

- 7 Retrieve any return data, for example:

```
String httpResponseStr = null;
httpResponseStr = getoutSiebel().getResultData();
```

---

## 4.12 Using the e\*Way

In the following example procedures, we assume that you have already imported the SeeBeyond HTTP Workflow templates (see [Importing SeeBeyond Workflow Templates](#) on page 55).

Three sample direct-database-access script files are included in the installation:

- `siebel-http-outgoing-delete.dsc`
- `siebel-http-outgoing-execute.dsc`
- `siebel-http-outgoing-insert.dsc`

If you are using Siebel 2000 (Japanese), you should use the following alternate files instead:

- `siebel-http-outgoing-delete-sjis.dsc`
- `siebel-http-outgoing-execute-sjis.dsc`
- `siebel-http-outgoing-insert-sjis.dsc`

### 4.12.1 Connecting to Siebel

When an HTML form is submitted to the Web server and the specified action is `http://webserver/eai/start.swe`, the Web server loads the Siebel Web Server Extension (SWSE) plug-in. The SWSE then obtains the connection string from the `[/eai]` section of the configuration file `eapps.cfg`. This connection string contains the following information:

- Transport
- Siebel Gateway Server
- Siebel Enterprise Server
- Siebel Object Manager (`eaiObjectManager`)
- Siebel Application Server

Below is an example of a connection string:

```
ConnectionString = siebel.TCPIP.none.none://MyGatewayServer:3230/  
MyEnterpriseServer/eaiObjMgr/MyAppServer
```

With this information, the Web server can connect to the Siebel Server utilizing the user name and password given in the form.

## 4.12.2 Specifying the Business Service

Additional information must be provided to specify the specific method of the business service to be executed. Typically, this information is placed in the configuration file associated with the application. Since the e\*Way uses the EAI Object Manager, the appropriate file is `eai.cfg`. This file has two sections that are used by the HTTP adapter, **HTTP Services** and a user-defined method information section. **HTTP Services** is the section in which you define the **SWExtSource** and the name of the method. The method section allows you to define the adapter mode and the name and method of the Business Service.

Below is an example of how an HTTP Service is specified:

```
[HTTP Services]
ACCOUNT_UPSERT_SERVICE = ACCOUNT_UPSERT_METHOD

[ACCOUNT_UPSERT_METHOD]
Mode = Document
Service = ACCOUNT_UPSERT
Method = RunProcess
```

In this example, the method **RunProcess** of the Business Service **ACCOUNT\_UPSERT** is executed if the form has an “input” **SWExtSource** with the value “**ACCOUNT\_UPSERT\_SERVICE**”.

An adapter in **Document** mode sends data across a specific data transport without converting the data to a property set. A Business Service of class **CSSWfEngine** is provided, which has a **RunProcess** method to execute a workflow process. The name of the process (i.e., **ProcessName**) needs to be specified in the **BIM BS User Property**.

## 4.12.3 The Siebel Workflow Process

The Workflow process has the following properties:

- **<Value>** with a type **String**

This property refers to the **Value** attribute of the property set that is currently active. In the workflow, it can be either the **Inputs** or **Outputs** property set that executes it. In the **Inputs** property set, **Value** contains the incoming XML message; in the **Outputs** property set, **Value** consists of a result string that can be sent back to the Web page.

- **IncomingXML** with a type **String** and a default value **<Value>**

Anything you pass along to the URL as data is placed in this variable.

- **Message** with a type **Hierarchy**

The message is used to hold the intermediate property set that is generated by the EAI XML Converter.

## 4.12.4 e\*Gate-to-Siebel Example Procedure

### To insert or update an Employee Record

- 1 Make a copy of the template `SeeBeyond HTTP Update`.
  - A Change the name of the Workflow to `Employee Update Workflow`.
  - B Specify the **Business Object** to be `Employee`.
- 2 The Update Siebel Business Service is hard-coded with the return value `<h1>Update completed. </h1>`. You may leave it as it is.
- 3 In **Siebel Tools**, make a copy of the **Workflow Process Manager Business Service**.
  - A Change the value of **Name** to `Employee Update Business Service`.
  - B Change the value of **Project** to `EAI`.
- 4 Add a new **Business Service User Property** named `ProcessName` with the value of `Employee Update Workflow`.
- 5 Next, add the following sections in the `eai.cfg` file. It should be located in `siebsrvr\bin` directory.

```
[HTTP Services]
...
EUHS = EMPLOYEE_UPDATE_HTTP_SERVICE

[EMPLOYEE_UPDATE_HTTP_SERVICE]
Mode = Document
Service = Employee Update Business Service
Method = RunProcess
```
- 6 Compile the `.srf` file.
- 7 In **Siebel Tools**, export the **Employee Integration Object**.
- 8 Run the **SeeBeyond XML Converter** to generate the **Employee Integration Object ETD**.
- 9 Assuming that you have defined a **Collaboration** that satisfies your requirements, you are now ready to modify the **Collaboration Rules script**.
  - A Using the **e\*Gate Editor**, open the direct-database-access script `siebel-http-outgoing-insert.dsc`.
  - B Change the **Integration Object** from `Sample Account` to `Employee`.
  - C Change the **HTTP Service** name from `SEEBYOND_HTTP_UPDATE` to `EUHS`.
  - D Since you only want to perform an **Insert/Update**, delete the `siebel-http-process` call that performs the query operation.
  - E You also need to modify the script to match the **Collaboration** that you defined.
  - F Save the modified **Collaboration Rules script** under a different name.

**Note:** See [Siebel XML Messages](#) on page 53 regarding the message format.

## 4.12.5 Siebel-to-e\*Gate Example Procedure

### To retrieve an Employee Record and forward it to the e\*Gate system

- 1 Make a copy of the template **SeeBeyond HTTP Send**.
  - A Change the name of the Workflow to **Employee Send Workflow**.
  - B Specify the **Business Object** to be **Employee**.
- 2 The **Send Business Service** is hard-coded with the **Request URL Template** value **http://<web server>/mux.asp**. You need to specify the **MS IIS** as the web server.
- 3 Since you are testing the implementation in **Siebel Workflow Designer**, you need to change the value of **Object Id** of the **Process Properties** to the value used in your system (in this example, assume that **1-D9T** is the correct ID).
- 4 In the **MS IIS**:
  - A Modify the **Mux.asp** to have the IP address and port number of the Siebel EAI (MUX) e\*Way.
  - B Since you are not gathering data from a form, set **blnUseBinary = true**.
- 5 In **Siebel Tools**, export the **Employee Integration Object**.
- 6 Run the **SeeBeyond XML Converter** to generate the **Employee Integration Object ETD**.
- 7 Create the e\*Gate Collaboration to process the ETD.

*Note:* See [Siebel XML Messages](#) on page 53 regarding the message format.



## 4.13 Sample Schema

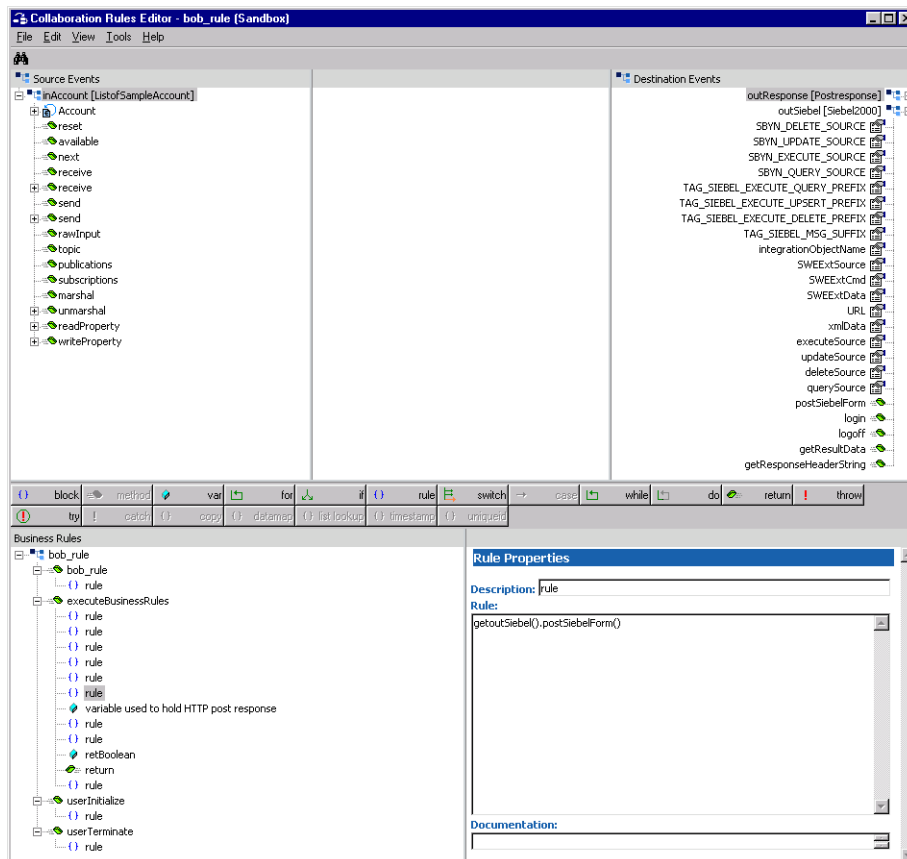
A sample implementation is located in the `\samples\ewsiebelhttp\Siebel2000` directory of the e\*Gate CD-ROM (see [Optional Example Files](#) on page 26 for installation instructions):

- **JavaSiebelOutbound.zip**

This e\*Gate-to-Siebel sample schema can be used to test your system following installation. Note that it is hard-coded to work only with the *sample account* integration object. It also substitutes a Business Object Broker (BOB) for the Multi-Mode e\*Way, with no change in functionality. For your own schema, you should use the Multi-Mode e\*Way executable.

The sample schema makes use of the SeeBeyond Workflow Templates included with the e\*Way. You must set up your environment by following the instructions on setting up the templates to execute the Workflow in [SeeBeyond Workflow Templates](#) on page 48. Figure 42 shows a sample Collaboration Rule within the sample schema.

**Figure 42** Collaboration Rules Editor Window - JavaSiebelOutbound



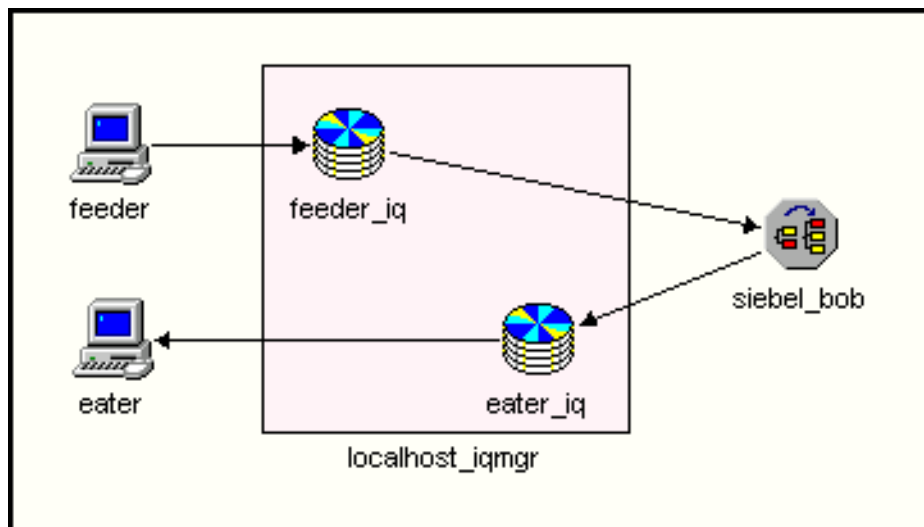
### 4.13.1 Components

The e\*Gate-to-Siebel example, **JavaSiebelOutbound**, sets up a single instance of the Siebel EAI e\*Way and two of the File e\*Way, having the logical names shown in the following table.

e*Way Type	Logical Name
Siebel EIM e*Way	siebel_bob
File e*Way	feeder
	eater

It also sets up two Intelligent Queues, with the logical names **feeder\_iq** and **eater\_iq**.

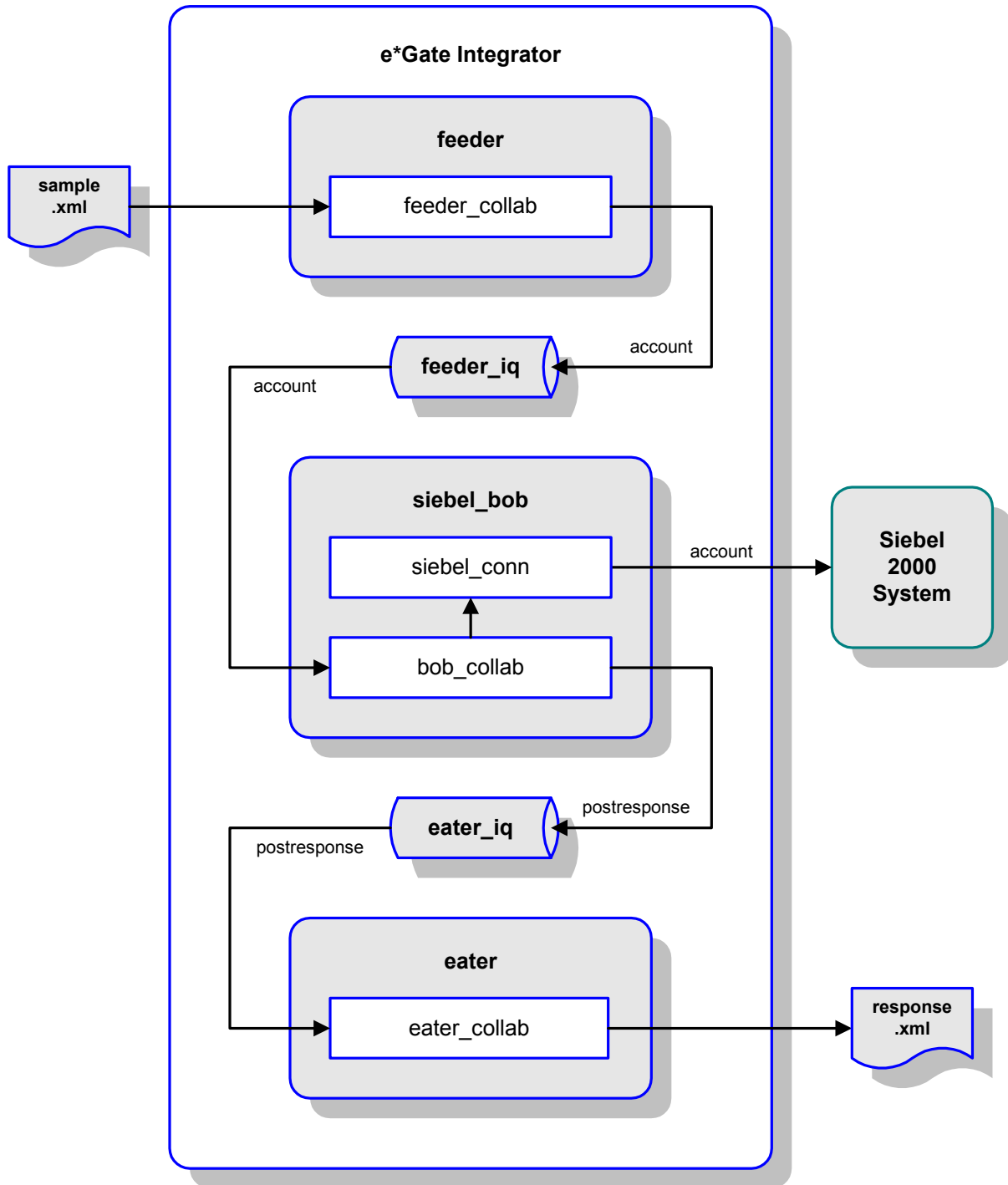
**Figure 43** JavaSiebelOutbound Components



### 4.13.2 Event Types

There are two Event types, **account** and **postresponse**, representing account data from another source to be posted to Siebel, and a response message or acknowledgement. These Event types are passed from one component to another following three Collaborations, as outlined next and diagrammed in Figure 44.

Figure 44 JavaSiebelOutbound Schema (Siebel 2000)

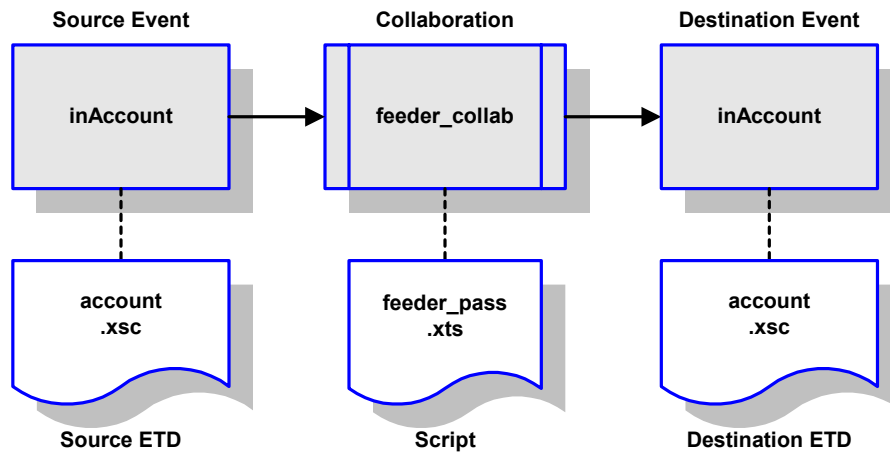


### 4.13.3 Collaborations

#### feeder\_collab

This pass-through Collaboration, performed by the e\*Way **feeder**, subscribes to an Event **InAccount** of Event Type **account** from an external source and publishes it to the **feeder\_iq** without transformation.

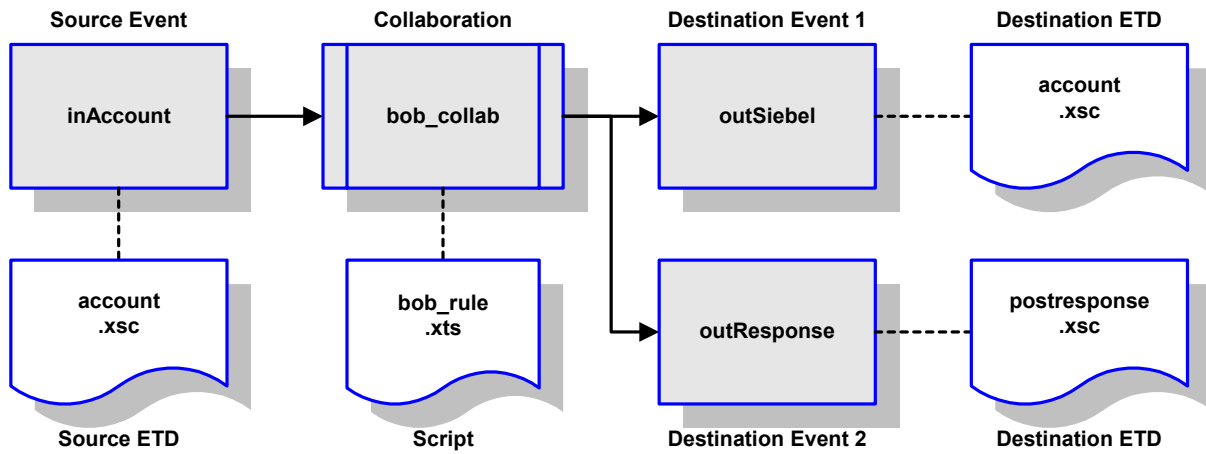
**Figure 45** feeder\_collab Collaboration



#### bob\_collab

This Collaboration, performed by the e\*Way **siebel\_bob**, subscribes to the Collaboration **feeder\_collab** and receives the Event **InAccount**. It then publishes it as Event **outSiebel**, still of Event Type **account**, through the e\*Way Connection **siebel\_conn** to the Siebel system. It also publishes the message **outResponse** of Event Type **postresponse** to the **eater\_iq**.

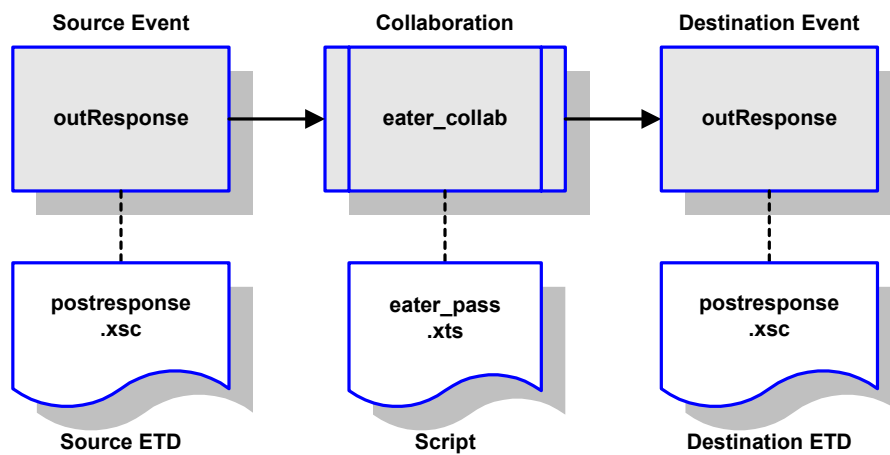
**Figure 46** bob\_collab Collaboration



### eater\_collab

This pass-through Collaboration, performed by the e\*Way **eater**, subscribes to the Collaboration **bob\_collab** through the **eater\_iq**. It receives the Event **outResponse** of Event Type **postresponse** and publishes it to an external destination without transformation.

**Figure 47** eater\_collab Collaboration



# Siebel 7 Implementation

This chapter describes the procedures for creating a functional Siebel 7-e\*Gate system incorporating the Siebel EAI e\*Way. Please refer to the *e\*Gate Integrator User's Guide* for additional information.

---

## 5.1 Overview

This e\*Way provides a specialized transport component for incorporation in an operational schema. The schema also contains Collaborations, linking different data or Event types, and Intelligent Queues. Typically, other e\*Way types also are used as components of the schema.

One or more sample schemas, included in the software package, are described at the end of this chapter. These can be used to test your system following installation and, if appropriate, as a template that you can modify to produce your own schema.

### 5.1.1 Pre-Implementation Tasks

#### Install the SeeBeyond Software

The first task is to install the SeeBeyond software as described in [Installing the e\\*Way](#) on page 21.

#### Import the Sample Schema

If you want to use the sample schema supplied with the e\*Way, the schema files must be imported from the installation CD-ROM (see [Optional Example Files](#) on page 26).

**Note:** *It is highly recommended that you make use of the sample schemas to familiarize yourself with e\*Way operation, test your system, and use as templates for your working schemas.*

#### Configure the Siebel EAI System

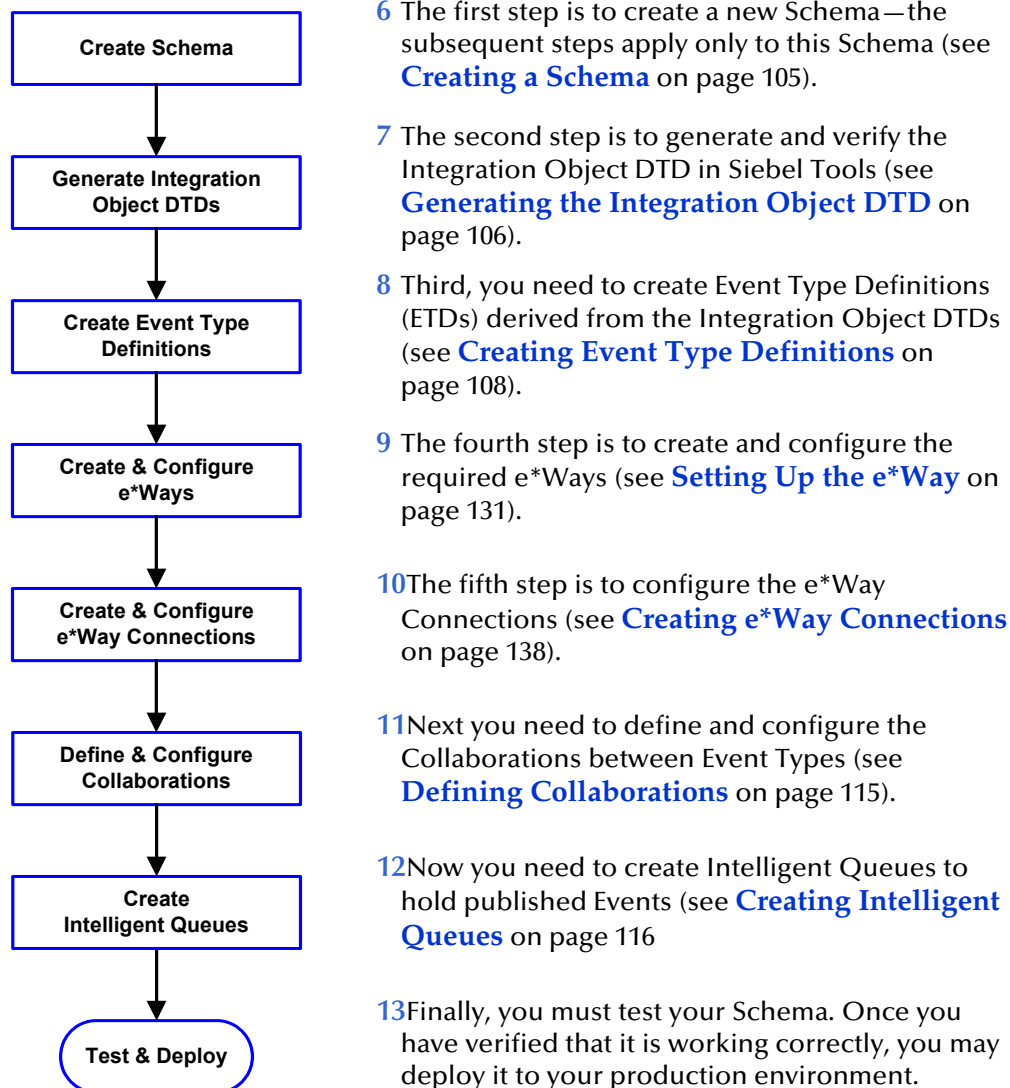
Follow the procedure described in [Web Server Setup](#) on page 31.

## 5.2 Implementation Overview

### 5.2.1 General Sequence

The high-level implementation sequence for a system incorporating the Siebel EAI e\*Way is depicted below.

#### General Implementation Sequence

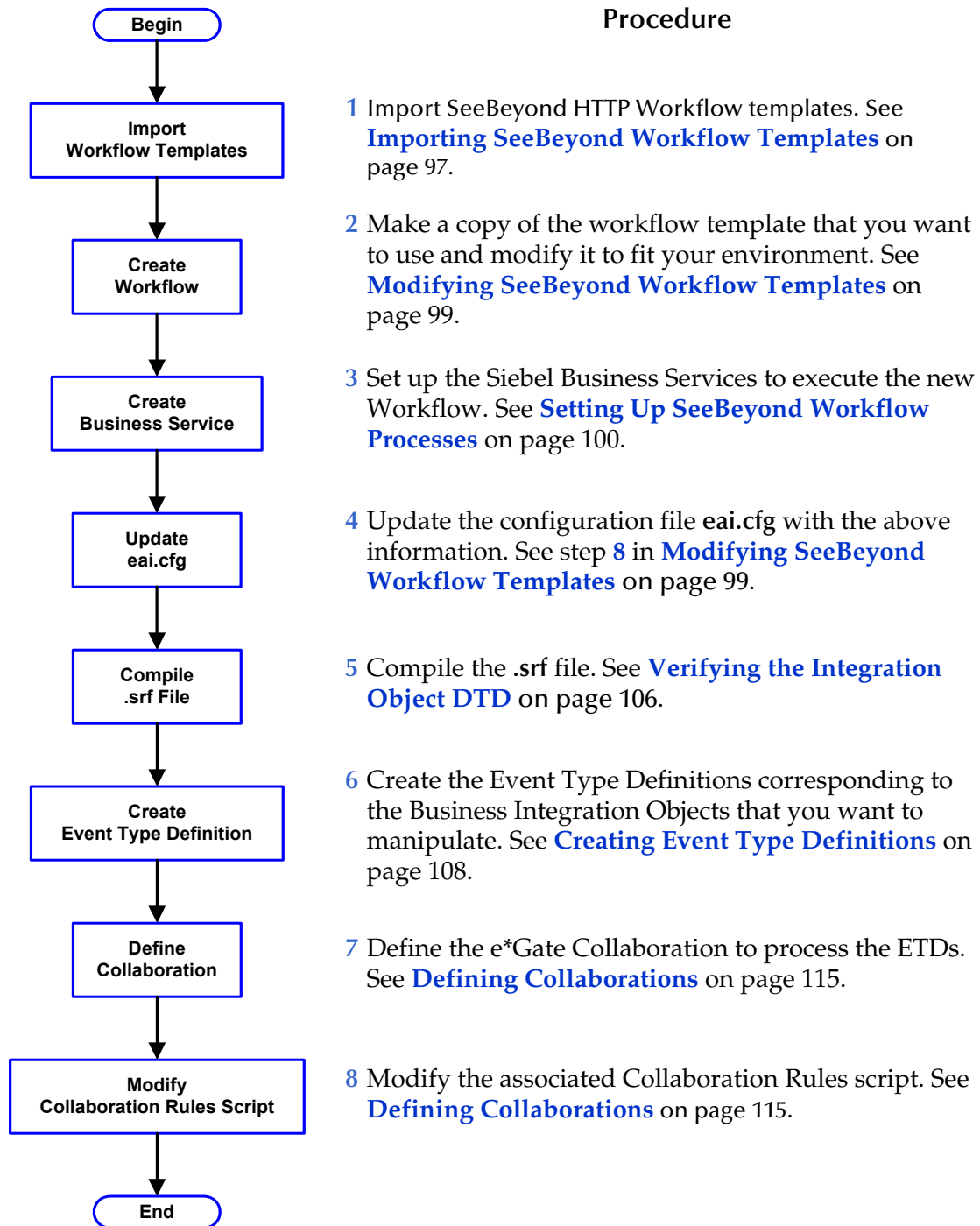


Included with the Siebel EAI e\*Way are several [SeeBeyond Workflow Templates](#), which furnish pre-defined workflows within the Siebel application. More detailed implementation sequences, making use of these templates, appear on the following pages. See [e\\*Gate to Siebel](#) on page 88 and [Siebel to e\\*Gate](#) on page 89.

Also included with the e\*Way are sample schema, which provide pre-defined templates that can be modified to suit your specific requirements.

## 5.2.2 e\*Gate to Siebel

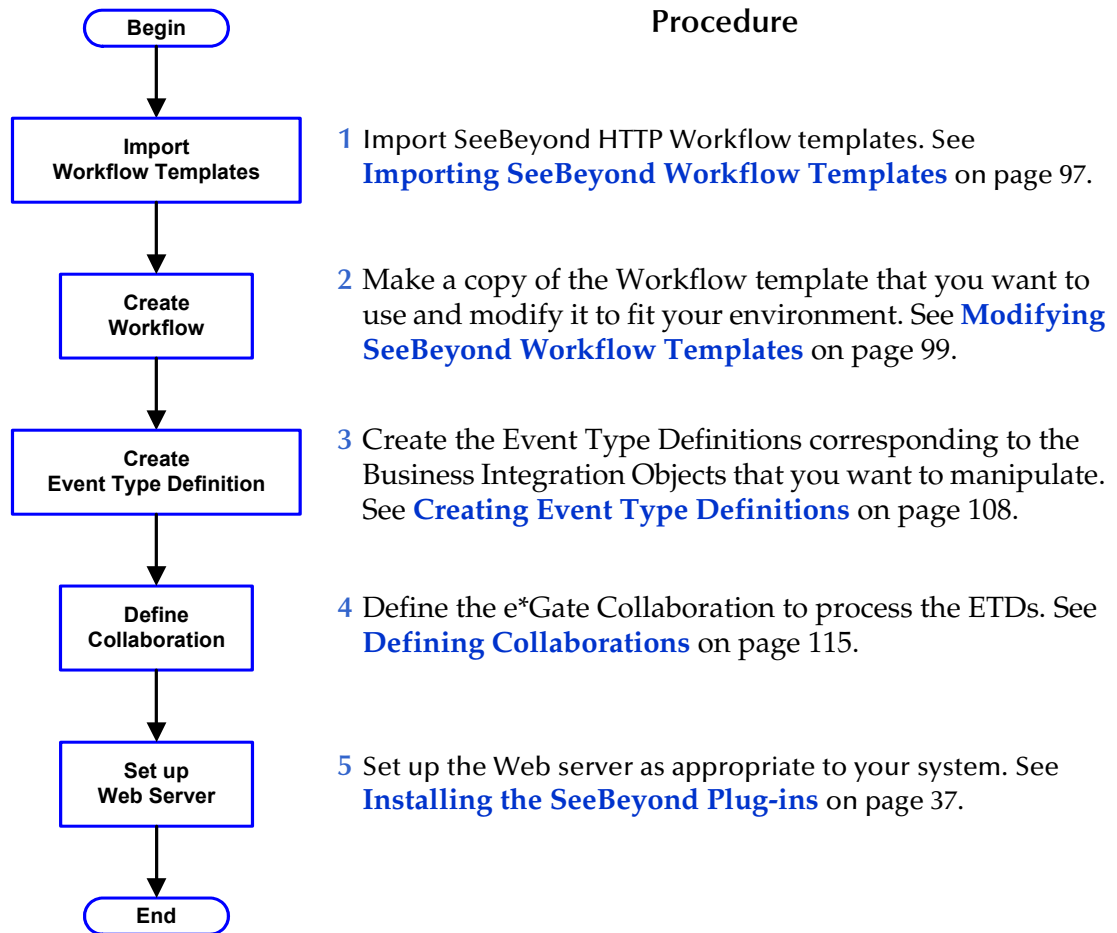
### e\*Gate-to-Siebel Implementation





### 5.2.3 Siebel to e\*Gate

#### Siebel-to-e\*Gate Implementation



### 5.2.4 Troubleshooting Your Implementation

To assist in debugging, you can view the log files generated by Siebel.

- On UNIX, the log files for Siebel 7 can be found in:  
`<Siebel7 root>/enterprises/siebel/siebel/log`
- On Windows, the log files can be found in:  
`siebsrvr\log,`  
`sweapp\log.`

---

## 5.3 SeeBeyond Workflow Templates

### 5.3.1 Overview

A set of SeeBeyond Workflow templates is included with the Siebel EAI e\*Way. These workflow templates invoke the following workflow processes to map the data directly to or from the Siebel database.

- SeeBeyond HTTP Delete (see [Figure 49 on page 91](#))
- SeeBeyond HTTP Query (see [Figure 50 on page 92](#))
- SeeBeyond HTTP Update (see [Figure 51 on page 92](#))

Inserts or Updates according to the provided input values.

- SeeBeyond HTTP Execute (see [Figure 52 on page 93](#))

The preferred Workflow for receiving Siebel XML messages from e\*Gate; combines **Delete**, **Query** and **Update** functionality into a single Workflow.

- SeeBeyond HTTP Send (see [Figure 53 on page 93](#))
- SeeBeyond HTTP Send Receive (see [Figure 54 on page 94](#))
- SeeBeyond HTTP Post (see [Figure 55 on page 94](#))

The preferred Workflow for sending Siebel XML messages to e\*Gate; combines **Send** and **Send/Receive** functionality into a single Workflow.

The names entered above are used to set up the Business Service for the sample program supplied with the e\*Way. You should use them as templates to create new processes corresponding to the Workflows you create for your own system.

Examples of XML messages used with different Workflow templates are given in [Siebel XML Messages](#) on page 95.

Following the Screens menu path shown below displays the installed Workflow templates, as shown in Figure 48.

Siebel Workflow Administration > Workflow Processes > All Processes

Figure 48 SeeBeyond Workflow Processes

Name	Business Objec	Status	Group	Activation Date	Expiration Date	Version
SeeBeyond HTTP Delete	Account	In Progress	Sample			0
SeeBeyond HTTP Execute	Account	In Progress	Sample			0
SeeBeyond HTTP Post	Account	In Progress	Sample			0
SeeBeyond HTTP Query	Account	In Progress	Sample			0
SeeBeyond HTTP Send	Account	In Progress	Sample			0
SeeBeyond HTTP Send Receive	Account	In Progress	Sample			0
SeeBeyond HTTP Update	Account	In Progress	Sample			0

Clicking the process name to invoke a Workflow Process Designer display for that process, such as shown in Figures 7-13.

Figure 49 DELETE Workflow Template

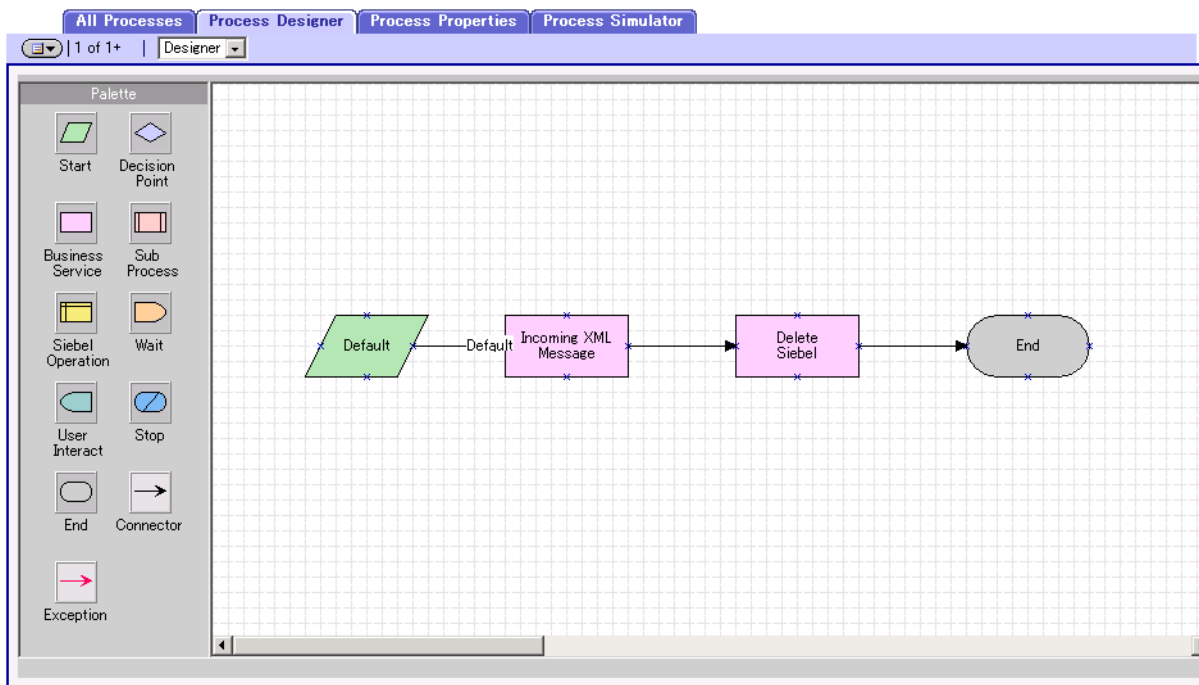


Figure 50 QUERY Workflow Template

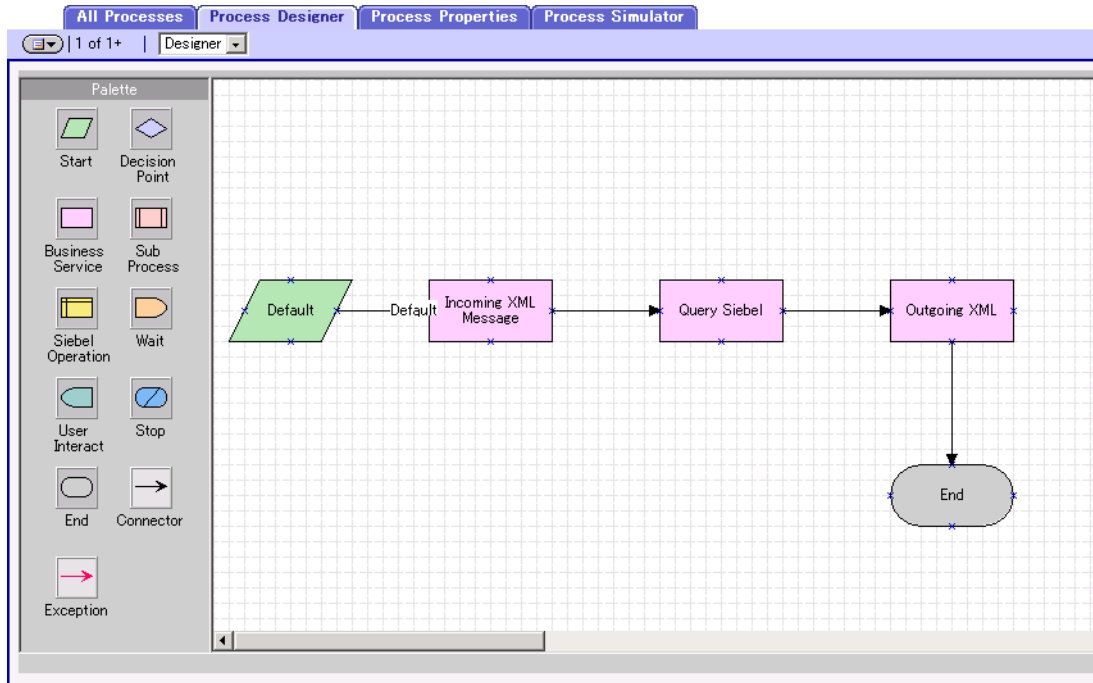


Figure 51 INSERT/UPDATE Workflow Template

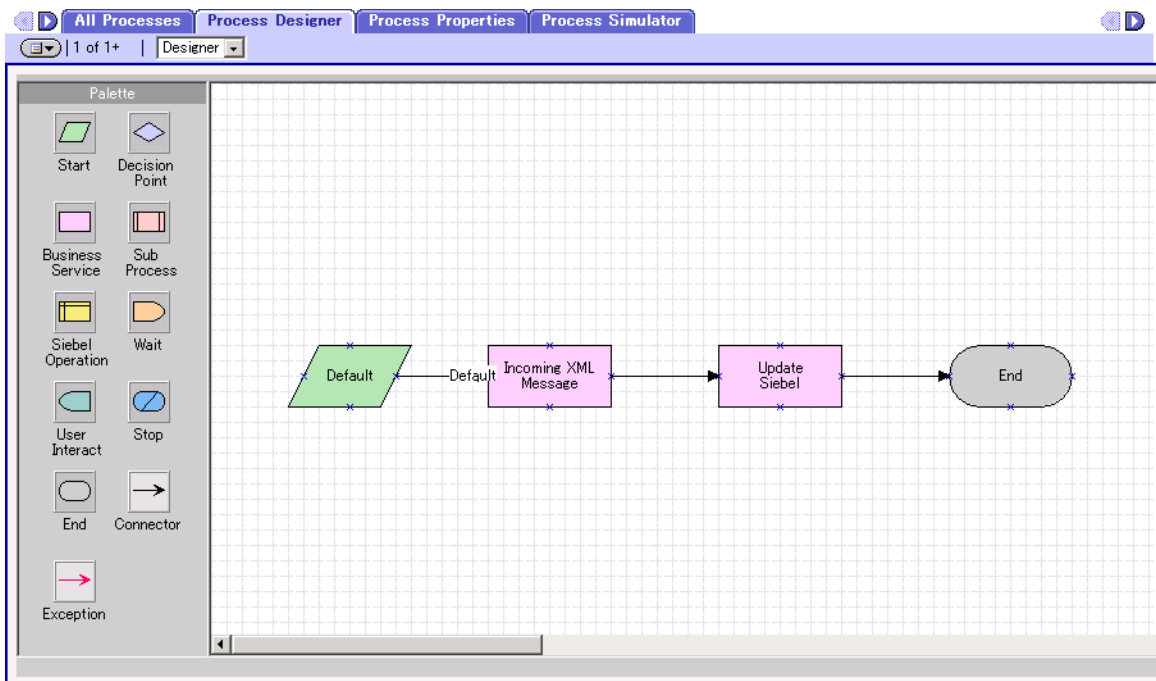


Figure 52 EXECUTE Workflow Template

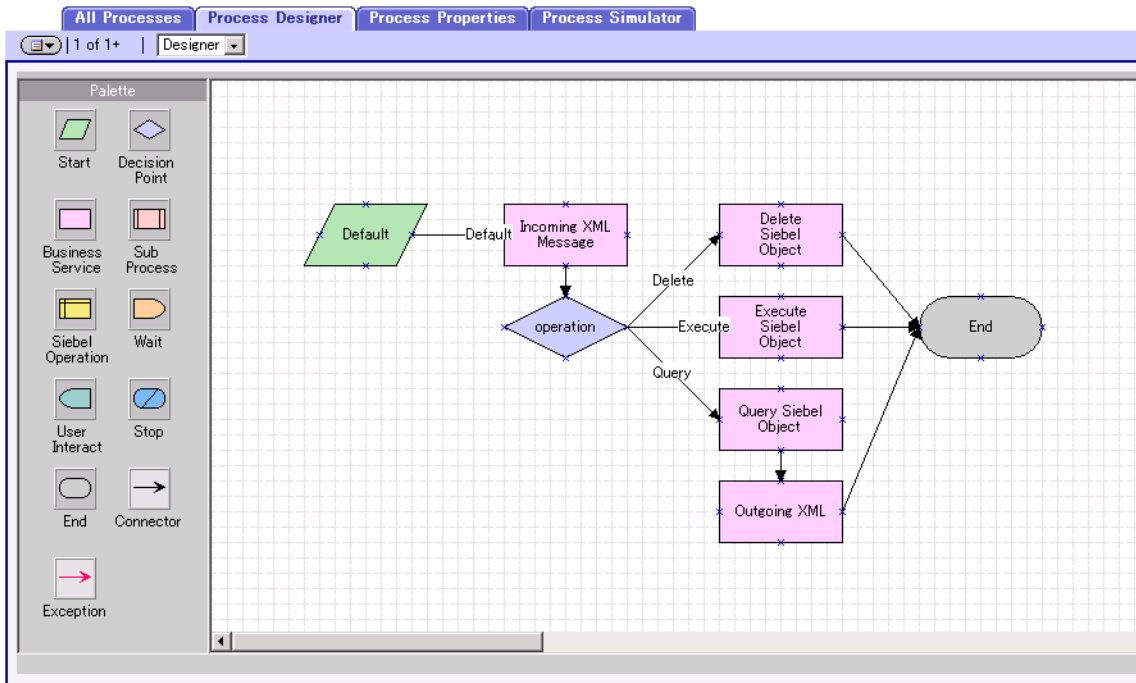


Figure 53 SEND Workflow Template

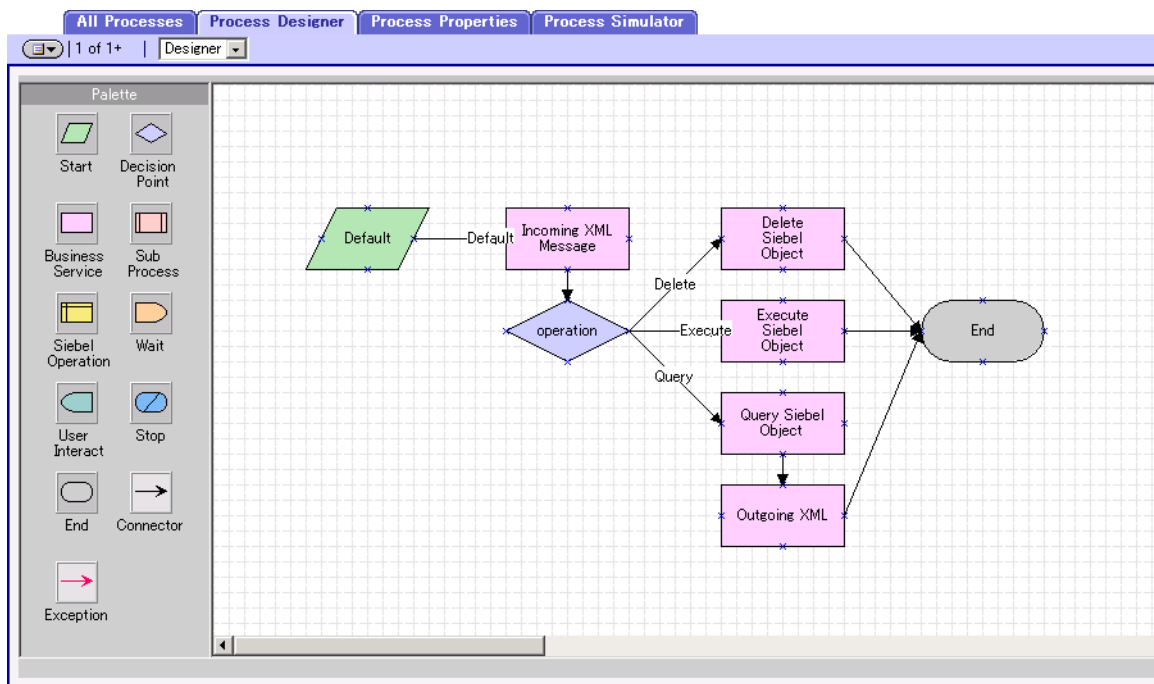


Figure 54 SEND/RECEIVE Workflow Template

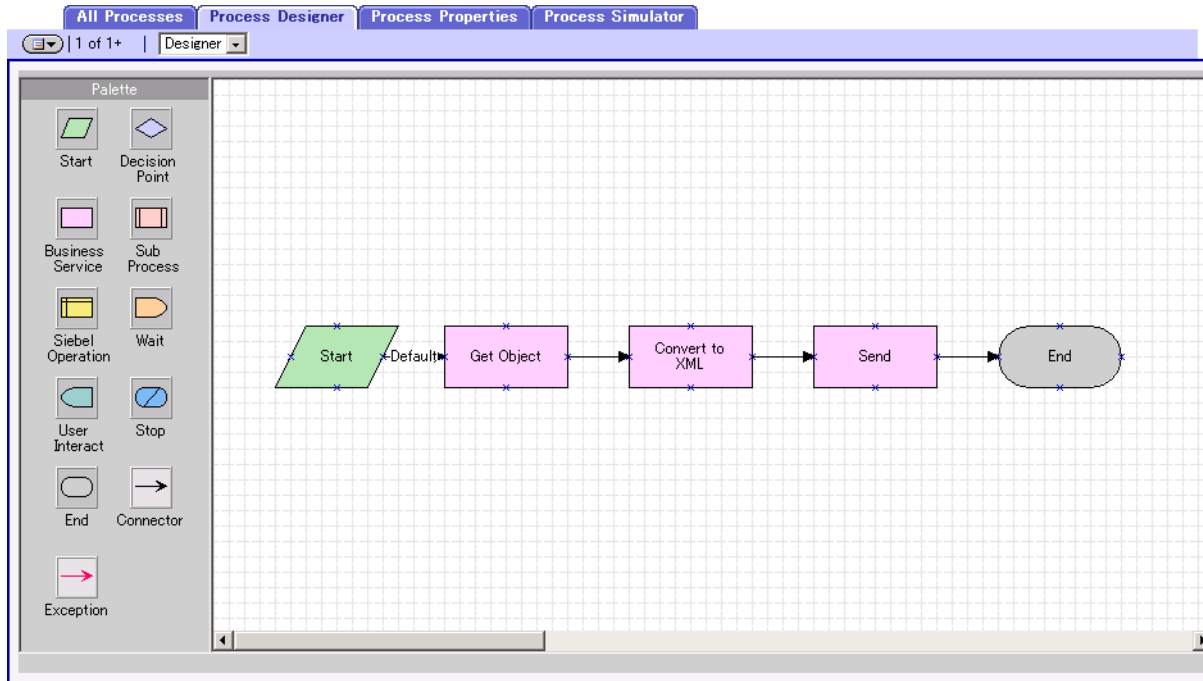
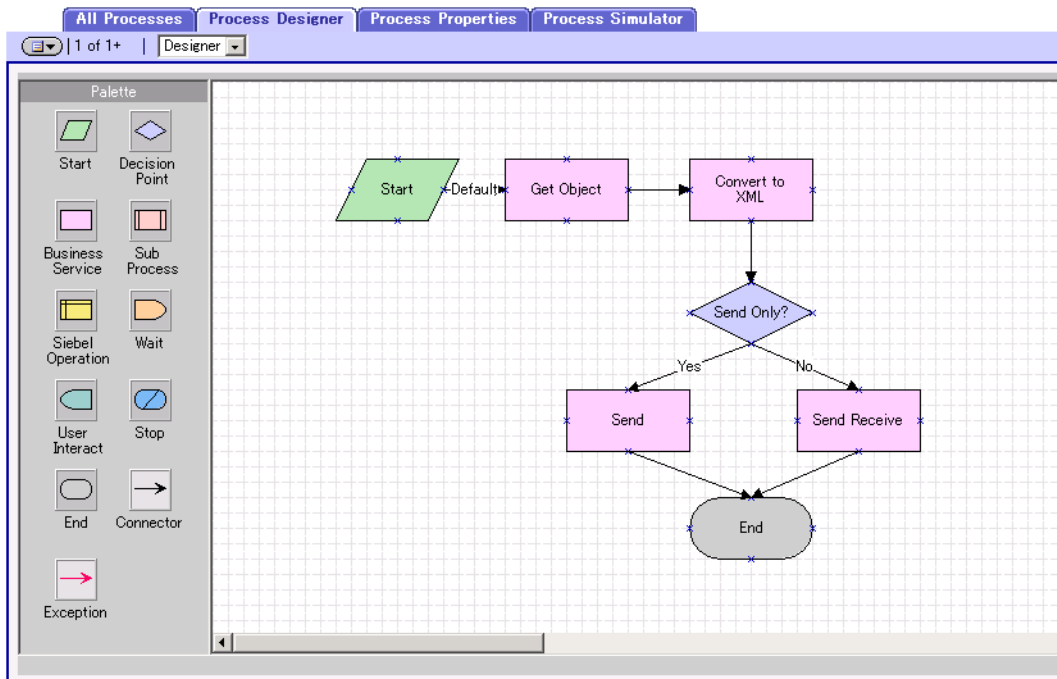


Figure 55 POST Workflow Template



## 5.3.2 Siebel XML Messages

### Format

A Siebel XML Message used by Siebel EAI e\*Way has the following format:

---

**Header/Prefix**  
**Integration Object (in XML format)**  
**Footer/Suffix**

---

where:

**Header =**

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="(Name of Integration Object)"
  operation=(action)>
```

**Footer =**

```
</SiebelMessage>
```

and **(action)** can be any of the following values:

- ♦ delete
- ♦ query
- ♦ upsert

**Note:** operation=(action) is used only with the EXECUTE workflow template.

### Examples

#### Example 1

The following Siebel XML message specifies that the Integration Object that we are dealing with is **Sample Account**. If we send this message to Siebel EAI using the INSERT/UPDATE workflow template, either a new record is generated or an existing record is updated.

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account">
<ListofSampleAccount>
<Account>
<Name>A. K. Parker Distribution</Name>
<Location>HQ-Distribution</Location>
<Organization>North American Organization</Organization>
<Division></Division>
<CurrencyCode>USD</CurrencyCode>
<Description>This is THE key account in the AK Parker Family</
  Description>
<HomePage>www.parker.com</HomePage>
<LineofBusiness>Manufacturing</LineofBusiness>
</Account>
```

```
</ListofSampleAccount>  
</SiebelMessage>
```

---

### Example 2

The following Siebel XML message specifies that the Integration Object that we are dealing with is **Sample Account**. If we send this message to Siebel EAI using the **QUERY** workflow template, it returns the object that matches the Name **A. K\***

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"  
  IntObjectName="Sample Account">  
<ListofSampleAccount>  
<Account>  
<Name>A. K*</Name>  
</Account>  
</ListofSampleAccount>  
</SiebelMessage>
```

---

### Example 3

The following Siebel XML message provides an example of how to use the **operation** attribute with the **Execute** workflow. Here we send the message to Siebel EAI using the **EXECUTE** workflow template to perform a **query** operation. The result is the same as in Example 2.

---

```
<SiebelMessage MessageId="" MessageType="Integration Object"  
  IntObjectName="Sample Account" operation=query>  
<ListofSampleAccount>  
<Account>  
<Name>A. K*</Name>  
</Account>  
</ListofSampleAccount>  
</SiebelMessage>
```

---



## 5.3.3 Importing SeeBeyond Workflow Templates

### To Import the SeeBeyond Workflow Templates

- 1 On the e\*Gate installation CD-ROM, go to:  
`\setup\addons\ewsiebelhttp\common.taz`
- 2 Decompress the .taz file and open the .tar file contained within.
- 3 Extract the file **SeeBeyondHTTPWorkflow.xml** to an appropriate directory.
- 4 Start **Siebel EAI Client** and select **Siebel Sales**.
- 5 Follow the menu path:  
**Ctrl+Shift+A > Siebel Workflow Administration > Workflow Processes**
- 6 Click **Import** and browse to the directory that contains **SeeBeyondHTTPWorkflow.xml**.
- 7 Click **Open** to begin importing the Workflow template.
- 8 Check or set up the following configuration files:
  - ♦ In the file `SWEApp\eapps.cfg`, verify that the following section is specified correctly:

```
[/eai]
ConnectString = siebel.TCPIP.none.none://<Your Gateway
Server>:3230/ <Your Enterprise Server>/eaiObjMgr/<Your App
Server>
EnableExtServiceOnly = TRUE
```

- ♦ For the **e\*Gate-to-Siebel** sample, add the following sections in the file `siebsrvr\bin\ENU\ei.cfg`:

```
[HTTP Services]
...
SEEBEYOND_HTTP_DELETE = SB_HTTP_DELETE
SEEBEYOND_HTTP_QUERY = SB_HTTP_QUERY
SEEBEYOND_HTTP_UPDATE = SB_HTTP_UPDATE
SEEBEYOND_HTTP_EXECUTE = SB_HTTP_EXECUTE

[SB_HTTP_DELETE]
Mode = Document
Service = SeeBeyond HTTP Delete
Method = RunProcess

[SB_HTTP_QUERY]
Mode = Document
Service = SeeBeyond HTTP Query
Method = RunProcess

[SB_HTTP_UPDATE]
Mode = Document
Service = SeeBeyond HTTP Update
Method = RunProcess
```

```
[SB_HTTP_EXECUTE]  
Mode = Document  
Service = SeeBeyond HTTP Execute  
Method = RunProcess
```

- 9 The imported Workflow must be in *active* mode before it can be used; this is accomplished by clicking the **active** button.

## 5.3.4 Modifying SeeBeyond Workflow Templates

**Note:** *The SeeBeyond Workflow templates provided with the e\*Way use **Account** as the Business Object—you must modify them for use with a different Business Object.*

### To Modify a SeeBeyond Workflow Template

- 1 Log in to **Siebel Client 7.0**, designating the appropriate Siebel server.
- 2 Follow the menu path:  
**Ctrl+Shift+A > Siebel Workflow Administration > Workflow Processes**
- 3 Highlight the SeeBeyond Workflow Process template you want to modify.
- 4 Right-click and select **Copy Record**.
- 5 Rename the copied Process.
- 6 Specify the Business Object to which you want to apply the template, and any other fields that may be necessary (for example, Description).
- 7 After modifying a Workflow template you must create the Business Service to execute it, using the supplied Workflow processes as templates. This procedure is described in the following section.
- 8 A new Services section should be added to your `siebsrvr\bin\ENU\esai.cfg` file, as shown in the preceding section.

For example, if you have a Business Service named Employee Execute, you should add the following lines to the `esai.cfg` file:

```
[HTTP Services]
...
EMPLOYEE_EXECUTE = EE

[EE]
Mode = Document
Service = Employee Execute
Method = RunProcess
```

### 5.3.5 Setting Up SeeBeyond Workflow Processes

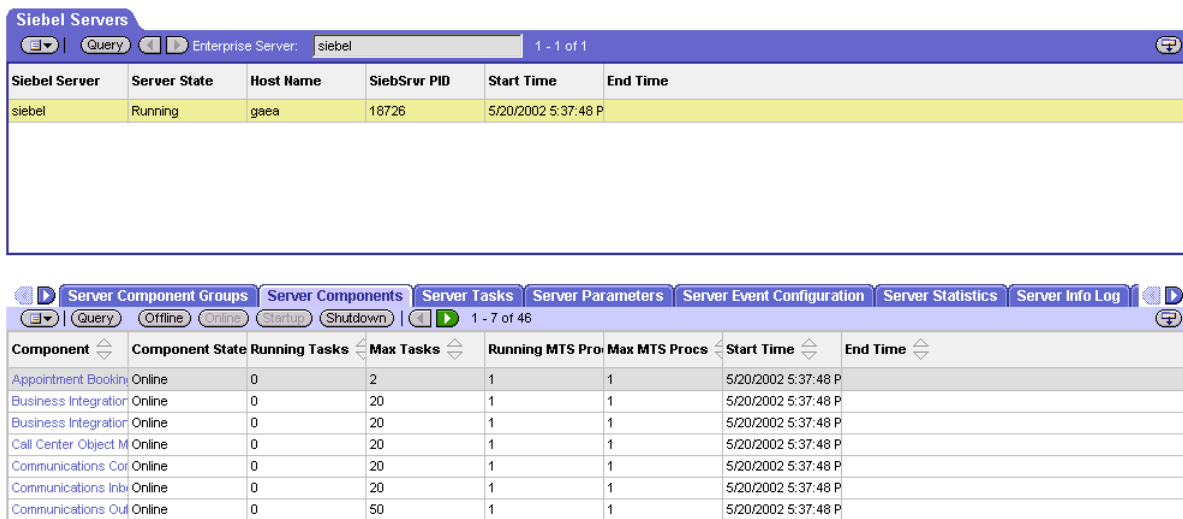
The Workflow processes invoked by the SeeBeyond Workflow Templates must be set up in Siebel Business Services.

**Note:** The names entered in step 8 are used to set up the Business Service for the sample program supplied with the e\*Way. You should use them as templates to create new processes corresponding to the Workflows you create for your own system.

To set up the Business Service to execute the Workflow

- 1 Make sure the following services are running:
  - ♦ Siebel Gateway Server
  - ♦ Siebel Server
  - ♦ World Wide Web Publishing Service
- 2 Log in to Siebel Sales 7.0.
- 3 Follow the Screens menu path:  
Server Administration > Servers

**Figure 56** Server Component Groups



- 4 Make sure that Workflow Management is Online and Enabled.
- 5 Log in to Siebel Tools 7.0 and designate the server as the database by entering `sadmin`, `sadmin`, `server`.
- 6 In Object Explorer, go to Siebel Objects > Project and lock the project (see Figure 57).

**Figure 57** Lock Project

Name	Changed	Inactive	Locked	Locked By Name	Locked Date	Language Locked
EAI						
EAI Account						
EAI Business Services						ENU
EAI Converter Services						ENU
EAI CreditCard						ENU
EAI DTE						ENU
EAI Demo						
EAI Design						
EAI Dispatch Service						ENU
EAI Envelope Services						
EAI Product						
EAI Queue						
EAI Sample Perf Test						ENU
EAI Tax						ENU
▶ EAI Test						ENU
EIM						ENU
EIM Accounts and Quotes						ENU
EIM Activity						ENU
EIM Administrative						ENU
EIM Agreement						ENU
EIM Asset Management						ENU
EIM Auction Item						ENU
EIM Audit Trail						ENU
EIM Bussiness Unit						ENU
EIM CHAMP						ENU
EIM CTI						ENU
EIM Call Script						ENU
EIM Class Systems						ENU
EIM Contact						
EIM Correspondence and Fulfillment						ENU
EIM DNB						ENU
EIM ERM						ENU

- In Object Explorer, go to **Business Service**, make a copy of **Workflow Process Manager** (menu path **Edit > Copy Record**).

**Figure 58** Business Services View - Workflow Process Manager

Business Services						
<span>Export</span> <span>Import</span> <span>Generate Code</span>						
W	Name	Changed	Project	Cache	Class	Display Name
	WI Web Proxy Service		WI - Web Integration		CSSWIService	Web Proxy
	Web Collab Service		Web Collaboration	✓	CSSWebCollabService	Web Collab
	Web Engine HTTP TXN		SWE		CSSServiceSweHttpTxn	Web Engin
	Web Engine Interface		SWE	✓	CSSServiceSWEIface	Web Engin
	Web Engine Mobile Device		SWE		CSSMobileDeviceService	Mobile Dev
	Web Engine State Properties		SWE		CSSWEStateService	Web Engin
	Web Engine UI Preferences		SWE		CSSServiceSWEUIPrefs	Web Engin
	Web Engine User Agent		SWE		CSSServiceSweUserAgent	Web Engin
	WebPhone Push Test		eAuction Test		CSSWAPPushService	Webphone
	Webphone Push		SWLS Push		CSSWAPPushService	Webphone
	Workflow FlowChart UI		Workflow Process	✓	CSSSvcWfFlowCht	Workflow F
	Workflow Process Manager		Workflow Process	✓	CSSWfEngine	Workflow P
	Workflow Process Manager (Server R		Workflow Process	✓	CSSSrmService	Workflow P
	Workflow Process Manager (Server R		Workflow Process	✓	CSSSrvrReqSyncService	Workflow P
	Workflow Siebel Operation		Workflow Process		CSSBCOperService	Workflow P
	Workflow UI Utilities		Workflow Process		CSSWfUtilService	Workflow U
	Workflow Utilities		Workflow Process		CSSWfUtilService	Workflow U

8 Type the Process Name into the **Name** and **Display Names** fields, as shown in Figure 59 (this name is specified in the `eai.cfg` file).

For e\*Gate-to-Siebel operation, perform this step for:

- ◆ SeeBeyond HTTP Delete
- ◆ SeeBeyond HTTP Execute
- ◆ SeeBeyond HTTP Query
- ◆ SeeBeyond HTTP Update

**Figure 59** Business Services View - Renamed Fields

Business Services						
<span>Export</span> <span>Import</span> <span>Generate Code</span>						
W	Name	Changed	Project	Cache	Class	Display Name
	S2S Inbound Dispatch Service		EAI Business Services		CSSEAIUtilService	S2S Inbound
	S2S Inbound Receive Processor		EAI Business Services		CSSWFEngine	S2S Inbound
	SAP 4x Account - SAP Maps (MO)		SAP Account 4x	✓	CSSEAITEScriptService	SAP 4x Map
	SME Filter Expression		Filters (DBM)	✓	CSSFilterExpressionService	SME Java E
	SME Java Expression		Segment (DBM)	✓	CSSExpressionService	SME Java E
	SSE Command Service		Siebel Sales Enterprise		CSSSrcvSSECmd	SSE Comm
	SWE Command Manager		System		CSSCommandMgrService	SWE Comm
	SWE Locale Service		System		CSSLocaleService	SWE Local
	SWE Query Exporter		SWE Import Export		CSSSWEQueryExporter	SWE Query
	Search Admin Service		Search Administration	✓	CSSSearchAdminService	Search Adm
	Search Client Service		Search Execution	✓	CSSSearchClientService	Search Clie
	Search Execution Service		Search Execution	✓	CSSSearchExecService	Search Exe
	Search Routing Service		Search Execution	✓	CSSSearchRouteService	Search Rou
	SeeBeyond HTTP Delete		Account	✓	CSSWFEngine	SeeBeyond
	SeeBeyond HTTP Execute		Account	✓	CSSWFEngine	SeeBeyond
	SeeBeyond HTTP Query		Account	✓	CSSWFEngine	SeeBeyond
	SeeBeyond HTTP Update		Account	✓	CSSWFEngine	SeeBeyond
	Server Requests		Business Service	✓	CSSSRmService	Server Req
	Service Provider Search Engine		Service Locator	✓	CSSSearchServiceService	Service Pro
	Shipping Cost Service (eScript)		EAI CreditCard	✓	CSSEAITEScriptService	Shipping C
	Shopping Service		eSales	✓	CSSShoppingService	Shopping S
	Siebel Anywhere Upgrade		Software Upgrade		CSSSvcAnyUpg	Siebel Any
	Siebel Code Generator		Repdtd		CSSEAICodeGenService	Siebel Code
	Siebel Tools Actuate Report Generator		Siebel Tools		CSSActuateReportGenerator	Siebel Tool
	Siebel Tools Applet Designer		Siebel Tools		CSSAppletDesigner	Siebel Tool
	Siebel Tools Application Converter		Siebel Tools		CSSAppConvSvc	Siebel Tool
	Siebel Tools Application Upgrader		Siebel Tools		CSSMerge	Siebel Tool
	Siebel Tools Assemble		Siebel Tools		CSSAssembleService	Siebel Tool
	Siebel Tools Batch Archive Export		Siebel Tools		CSSRepositoryBatchExport	Siebel Tool
	Siebel Tools Batch Archive Import		Siebel Tools		CSSRepositoryBatchImport	Siebel Tool

9 In Object Explorer, go to **Business Service > Business Service Userprops**:

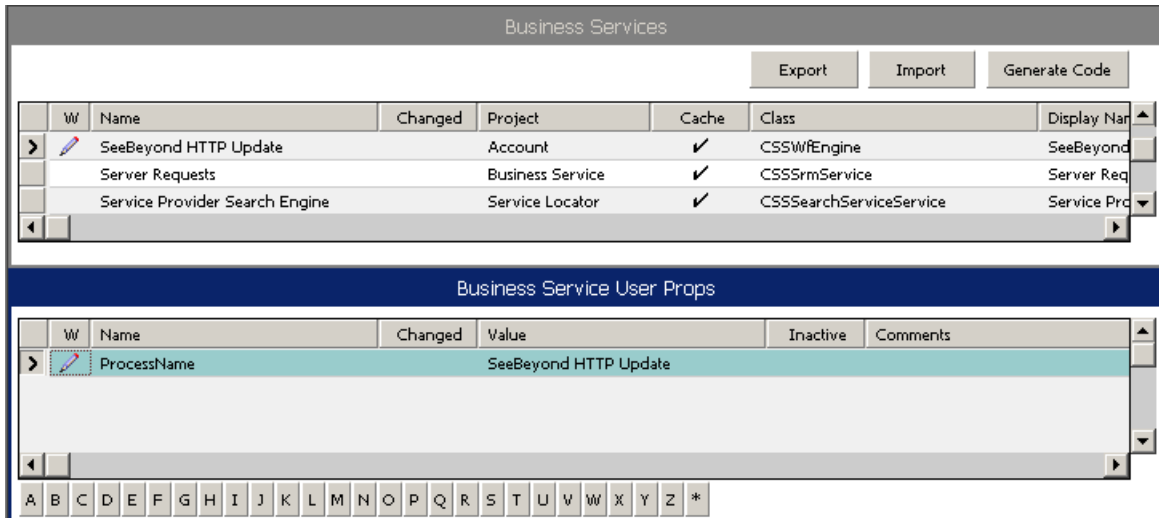
A Type **ProcessName** into the **Name** field.

B Type the actual Process Name into the **Value** field (see Figure 60).

For e\*Gate-to-Siebel operation, perform this step for:

- ◆ SeeBeyond HTTP Delete
- ◆ SeeBeyond HTTP Execute
- ◆ SeeBeyond HTTP Query
- ◆ SeeBeyond HTTP Update

**Figure 60** Business Services User Properties



The Workflow Processes you create in the Business Services are similar to those shown in Figure 59.



## 5.4 Creating a Schema

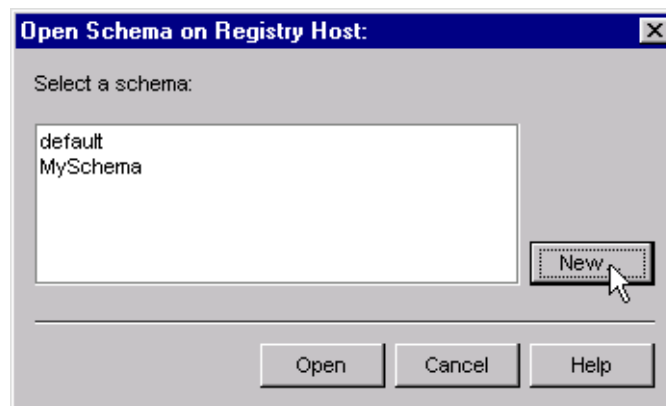
A schema is the structure that defines e\*Gate system parameters and the relationships between components within the e\*Gate system. Schemas can span multiple hosts.

Because all setup and configuration operations take place within an e\*Gate schema, a new schema must be created, or an existing one must be started before using the system. Schemas store all their configuration parameters in the e\*Gate Registry.

To select or create a schema

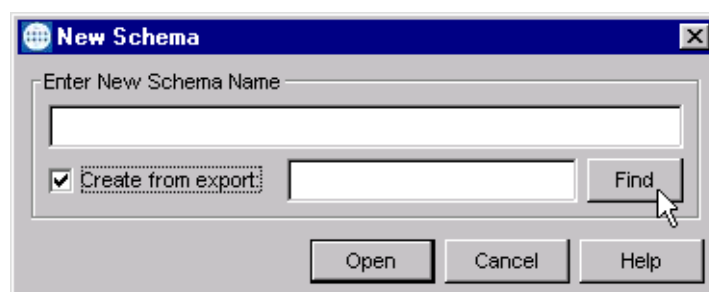
- 1 Invoke the **Open Schema** dialog box and **Open** an existing schema or click **New** to create a new schema.

**Figure 61** Open Schema Dialog



- 2 Clicking **New** invokes the **New Schema** dialog box (Figure 62).

**Figure 62** New Schema Dialog



- 3 Enter a new schema name and click **Open**.
- 4 The e\*Gate Schema Designer then opens under your new schema name.
- 5 From the **Options** menu, click on **Default Editor** and select **Monk**.
- 6 Select the **Components** tab, found at the bottom of the Navigator pane of the e\*Gate Schema Designer window.
- 7 You are now ready to begin creating the necessary components for this new schema.

---

## 5.5 Generating the Integration Object DTD

### To Generate the DTD

- 1 In Siebel Tools, click on an Integration Object to activate it.
- 2 Click **Generate Schema**, which displays the initial page of the Generate XML Schema Wizard.
- 3 Select the **EAI XML DTD Generator** business service.
- 4 Select a location to store the resulting file.
- 5 Click **Finish**.

The Wizard generates an XML DTD of the Integration Object you selected. You can use this DTD to create an ETD using the SeeBeyond XML Converter/ETD Builder, as described in [Using the DTD Builder](#) on page 108.

---

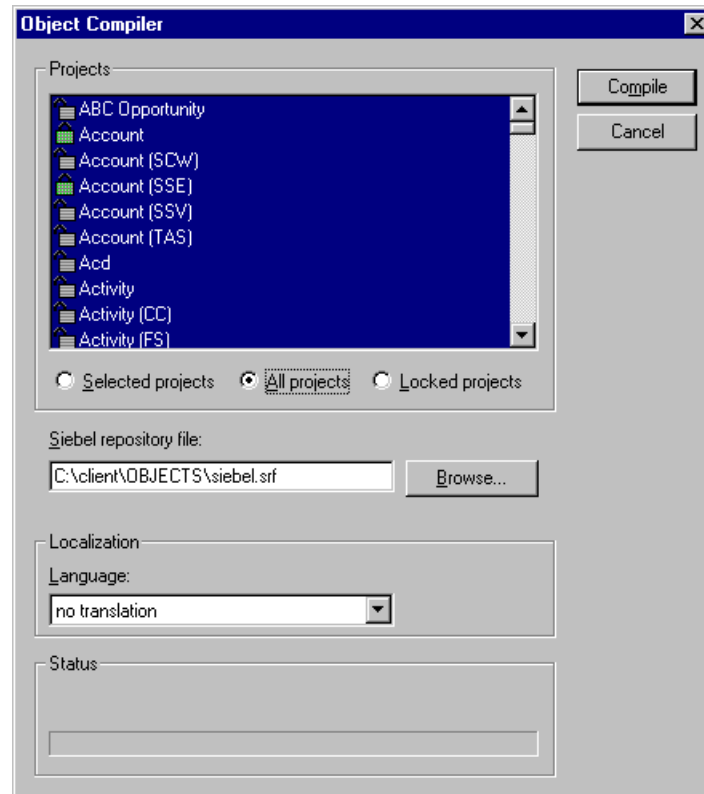
## 5.6 Verifying the Integration Object DTD

The next step is to confirm that the Integration Object DTD is generated correctly. You should export the DTD and run the SeeBeyond XML Converter/ETD Builder to verify that it can generate the Event Type Definition correctly. An incorrect ETD build usually indicates that the Siebel DTD has a repeated element name, in which case you need to modify the Integration Object.

### To Verify the DTD

- 1 Stop the following services, in this order:
  - A Siebel Server.
  - B Siebel Gateway Name Server.
- 2 Follow the menu path **Tools > Compile Project**, which invokes the **Object Compiler** dialog box.

**Figure 63** Objects Compiler Dialog Box



- 3 If you have completed all projects, select the **All Projects** option button; otherwise, select **Locked Projects** to shorten the compilation time.
- 4 Select the Siebel repository file `\client\OBJECTS\ENU\siebel.srf`.
- 5 Click **Compile** and copy the Siebel `.srf` file to the `siebel\sblsrvr\ENU\OBJECTS` directory.
- 6 Start the following services, in this order:
  - A Siebel Gateway Name Server.
  - B Siebel Server.
- 7 Verify that the EAI Object Manager is running.

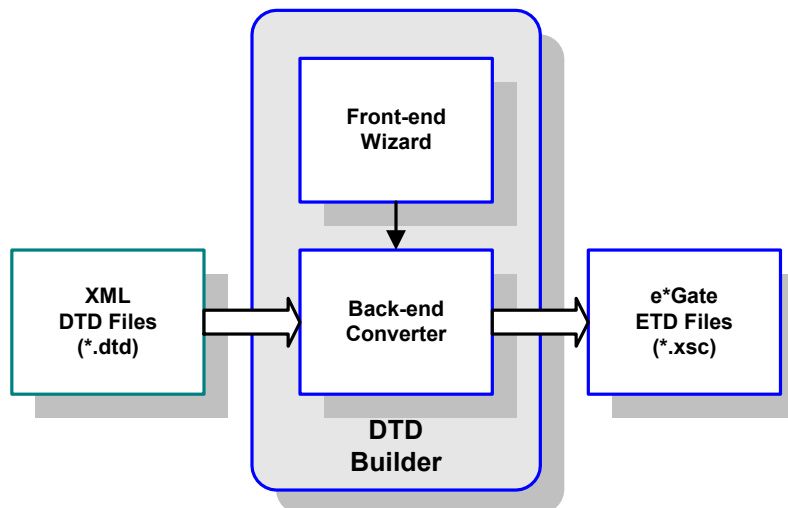
## 5.7 Creating Event Type Definitions

Before e\*Gate can process any data to or from a Siebel EAI system, you must create an Event Type Definition to package and route that data within the e\*Gate system. The ETD is derived from a Siebel Integration Object Data Type Definition (DTD). See the *e\*Gate Integrator User's Guide* for additional information about Event Type Definitions and the e\*Gate ETD Editor.

### 5.7.1 Using the DTD Builder

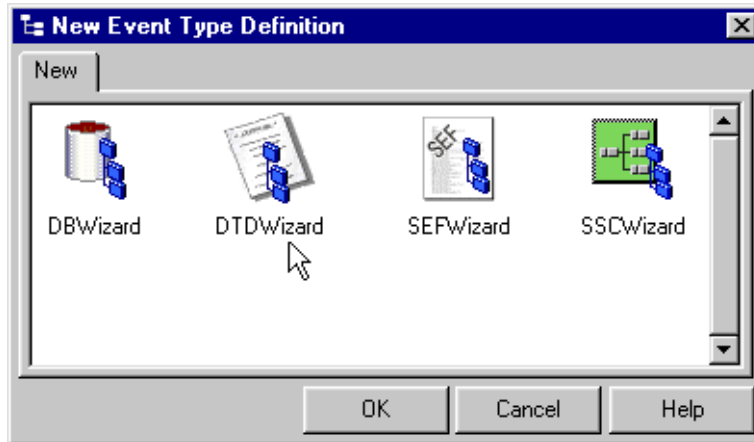
The ETD Editor contains a DTD Builder, which takes a Siebel XML DTD and converts it to a .xsc file. See the *XML Toolkit User's Guide* for detailed information on the DTD Builder.

**Figure 64** DTD Builder



To access the Wizard, select the **New** option in the ETD Editor's **File** menu. The New Event Type Definitions window appears, displaying all installed ETD Builder Wizards. Select the **DTD Wizard**, and follow the instructions presented in the dialog.

**Figure 65** New Event Type Definitions Window



**To run the DTD Builder**

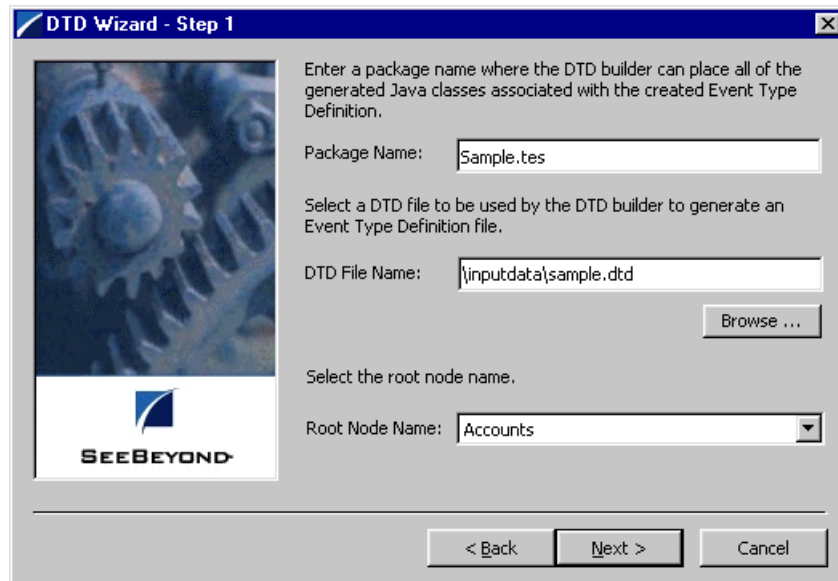
- 1 Invoke the DTD Wizard by clicking its icon.

**Figure 66** DTD Wizard – Introduction



- 2 Read the instructions carefully, and click Next. Step 1 of the DTD Wizard dialog appears (see Figure 67).

Figure 67 DTD Wizard — Step 1



3 Enter the following information:

♦ **Java Package Name**

Type in the name you want to give the Java package, for example, **Sample.tes**. This name must conform to Java package name requirements. See the appropriate Java documentation for details.

♦ **DTD File Name**

Type in the name of the DTD file you want to convert. Click **Browse** to access an Open (file selection) dialog box, allowing you to choose the desired file.

♦ **Root Node Name**

This text box is a pull-down menu. Select the desired root node name from the menu. For more information on root nodes and ETDs, see the *e\*Gate Integrator User's Guide*.

4 When you are finished, click **Next**. Step 2 of the DTD Wizard dialog appears (see Figure 68).

Figure 68 DTD Wizard — Step 2



- 5 Specify the options you want used by the DTD Builder.
  - ◆ Allow whitespace in EMPTY elements
  - ◆ Ignore #FIXED attributes
  - ◆ Ignore all attributes
  - ◆ Include XML declarations
  - ◆ Include DOC Type Reference (if selected, specify DTR name)
- 6 When you are finished, click Next. Step 3 of the DTD Wizard dialog appears (see Figure 69).

Figure 69 DTD Wizard — Step 3



- 7 Review the information you have entered in the Wizard. If it is correct, click **Finish** to generate a Java ETD (.xsc file) from the original DTD file.

The Wizard closes, and the new ETD appears in the ETD Editor Main window. See the *e\*Gate Integrator User's Guide* for details on how to use this editor, including an explanation of the information it shows.

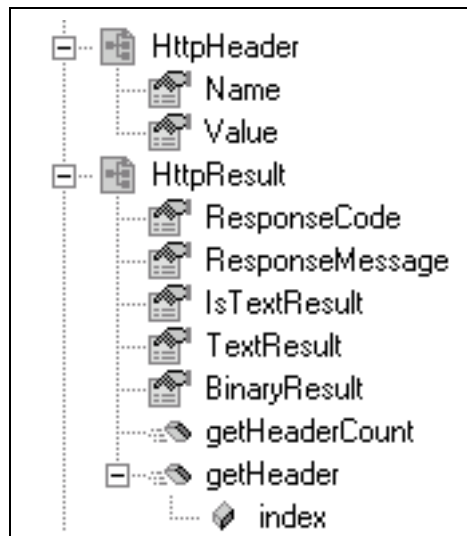
- 8 To save the new ETD, click the **Save** button on the Toolbar or select the **Save** command from the **File** menu. A Save dialog box appears.
- 9 Select the desired directory location, give the new ETD your desired name, and click **Save**. The ETD Editor saves the new Java ETD.
- 10 You can continue to use the ETD Editor or select the **Close** command from the **File** menu to exit the GUI.

**Note:** *The ETD nodes created using the DTD Builder appear shaded in the ETD Editor, indicating that you cannot edit an ETD created by the Builder.*

After converting the DTD to an ETD, return to the e\*Gate Schema Designer to verify the process (see Figure 70).



**Figure 70** ETD Structure Example




---

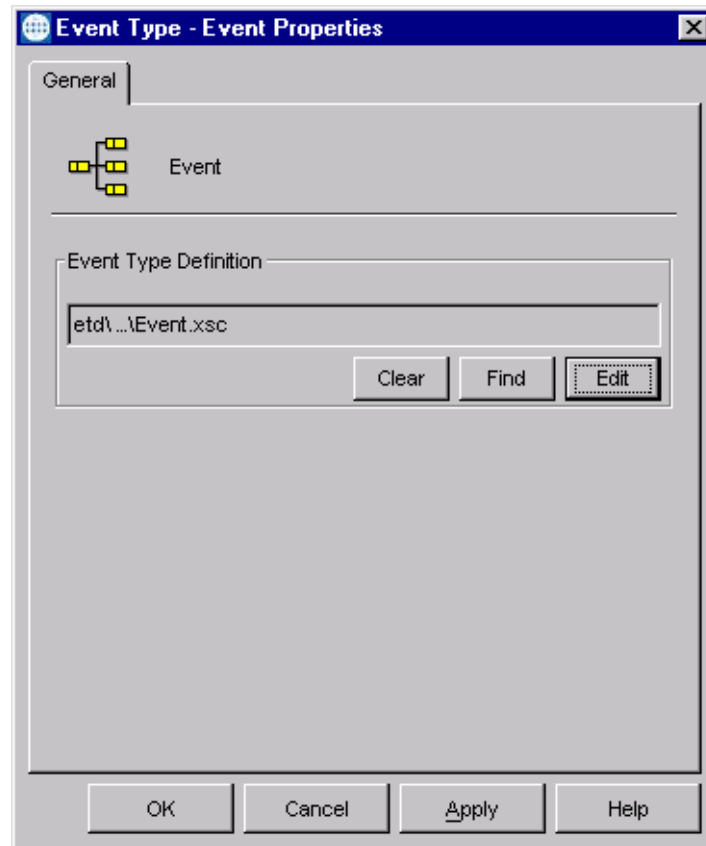
## 5.8 Assigning ETDs to Event Types

After you have created the e\*Gate system's ETD files, you can assign them to Event Types you have already created.

### To assign ETDs to Event Types

- 1 In the Schema Designer window, select the **Event Types** folder in the Navigator/Components pane.
- 2 In the Editor pane, select one of the Event Types you created.
- 3 Right-click on the Event Type and select **Properties** (or click  in the toolbar). The Event Type Properties dialog box appears. See Figure 71.

**Figure 71** Event Type Properties Dialog Box



- 4 Under Event Type Definition, click **Find**.

The Event Type Definition Selection dialog box appears; it is similar to the Windows Open dialog box.

**Note:** Clicking **New** in the Event Type Properties dialog box opens the ETD Editor window, allowing you to create a new ETD.

- 5 Open the **etd** folder, then select the desired file name (.xsc).
- 6 Click **Select**. The file populates the Event Type Definition field.
- 7 To save any work in the properties dialog box, click **Apply** to enter it into the system.
- 8 When finished assigning ETDs to Event Types, click **OK** to close the properties dialog box and apply all the properties.

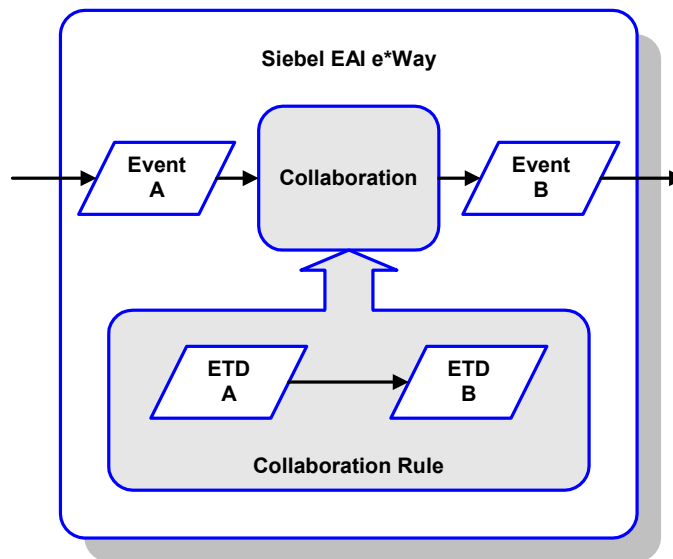
Each Event Type is associated with the specified Event Type Definition.

## 5.9 Defining Collaborations

After you have created the required Event Type Definitions, you must define a Collaboration to transform the incoming Event into the desired outgoing Event.

Collaborations are e\*Way components that receive and process Event Types, then forward the output to other e\*Gate components. Collaborations consist of the Subscriber, which “listens” for Events of a known type or from a given source, and the Publisher, which distributes the transformed Event to a specified recipient. The same Collaboration cannot be assigned to more than one e\*Gate component.

**Figure 72** Collaborations



### 5.9.1 The Java Collaboration Rules Editor

Java Collaborations are defined using the e\*Gate Java Collaboration Rules Editor. Note that the Java Collaboration environment supports multiple source and destination ETDs. The file extension for Java Collaboration Rules is **.xpr**. See the *e\*Gate Integrator User's Guide* for descriptions of the Java Collaboration Rules Editor and its use.

---

## 5.10 Creating Intelligent Queues

IQs are components that provide nonvolatile storage for Events within the e\*Gate system as they pass from one component to another. IQs are *intelligent* in that they are more than just a “holding tank” for Events. They actively record information about the current state of Events.

Each schema must have an IQ Manager before you can add any IQs to it. You must create at least one IQ per schema for published Events within the e\*Gate system. Note that e\*Ways that publish Events externally do not need IQs.

For more information on how to add and configure IQs and IQ Managers, see the *e\*Gate Integrator System Administration and Operations Guide*. See the *e\*Gate Integrator Intelligent Queue Services Reference Guide* and the *SeeBeyond JMS Intelligent Queue User’s Guide* for complete information on working with IQs.

---

## 5.11 Using the Siebel EAI ETD in a Collaboration

The Siebel EAI ETD contains the following attributes which users can **set** and/or **get**:

SWExtCmd	deleteSource
SWExtData	executeSource
SWExtSource	querySource
URL	updateSource
xmlData	

The ETD also contains the following methods:

delete	login
getResultData	logout
getResponseHeaderString	postSiebelForm
insert	query

**Note:** To run in Session mode, you must call **login** during initialization and **logout** during termination.

## 5.11.1 Overview

See [Post/Retrieve Call Sequence](#) on page 118 for a detailed call sequence.

### To Post Data to Siebel

You can set most of the relevant parameters for posting such as **URL**, **SWExtSource**, **SWExtCmd** and **SWExtData**, and then call the **postSiebelForm** method to perform the HTTP post using the **set** parameters. Note that the username and password are always obtained from the configuration file. When setting **SWExtData**, you insert the correct XML string to pass, based on the operation to be performed. See [Chapter 9](#) for details of these Java methods and their attributes.

### To get the HTTP response

After the call to **postSiebelForm()**, the HTTP response can be obtained by calling **getResultData()**.

### To get the response header

After the call to **postSiebelForm()**, the HTTP response header can be obtained by calling **getResponseHeaderString()**.

## 5.11.2 Helper Methods

You have the option of specifying your designated sources for **execute**, **update**, **delete** and **query**. These source names are used in the helper methods **insert**, **delete**, and **query**.

**Note:** *The helper methods may be used **only** if you want to use the following hard-coded XML tags for **SWExtData** (along with the value they set for the **xmlData** attribute):*

### For insert:

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account" operation="upsert"> + xmlData +
</SiebelMessage>
```

### For delete:

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account"
  operation="delete"><ListofSampleAccount><Account><Name> +
  xmlData +</Name></Account></ListofSampleAccount></SiebelMessage>
```

### For query:

```
<SiebelMessage MessageId="" MessageType="Integration Object"
  IntObjectName="Sample Account"
  operation="query"><ListofSampleAccount><Account><Name> +
  xmlData +</Name></Account></ListofSampleAccount></>
```

## 5.11.3 Post/Retrieve Call Sequence

### To Post Data to Siebel and Retrieve a Response

- 1 Specify **SWEEExtSource**, which includes **update**, **insert**, and **delete**; for example:

```
setSWEEExtSource("MY_UPDATE");
```

where MY\_UPDATE is the service user specifies in the HTTP Service section of the eai.cfg file.

- 2 Specify **SWEEExtCmd**, which currently only uses **execute**; for example:

```
setSWEEExtCmd("Execute");
```

- 3 Specify the Siebel Integration Object, for example:

```
setIntegrationObjectName("Sample Account");
```

- 4 Set the XML message, for example:

```
setXmlData(getinAccount().toString());
```

- 5 Format the message with prefix and suffix to create a Siebel Message, for example:

```
setSWEEExtData(getoutSiebel().getTAG_SIEBEL_EXECUTE_UPSERT_PREFIX()  
+ getoutSiebel().getXmlData() +  
getoutSiebel().getTAG_SIEBEL_MSG_SUFFIX());
```

- 6 Post the message to Siebel, for example:

```
postSiebelForm();
```

- 7 Retrieve any return data, for example:

```
String httpResponseStr = null;  
httpResponseStr = getoutSiebel().getResultData();
```

---

## 5.12 Using the e\*Way

In the following example procedures, we assume that you have already imported the SeeBeyond HTTP Workflow templates (see [Importing SeeBeyond Workflow Templates](#) on page 97).

### 5.12.1 Connecting to Siebel

When an HTML form is submitted to the Web server and the specified action is `http://webserver/eai/start.swe`, the Web server loads the Siebel Web Server Extension (SWSE) plug-in. The SWSE then obtains the connection string from the `[/eai]` section of the configuration file `eapps.cfg`. This connection string contains the following information:

- Transport
- Siebel Gateway Server
- Siebel Enterprise Server
- Siebel Object Manager (`eaiObjectManager`)
- Siebel Application Server

Below is an example of a connection string:

```
ConnectionString = siebel.TCPIP.none.none://MyGatewayServer:3230/  
MyEnterpriseServer/eaiObjMgr/MyAppServer
```

With this information, the Web server can connect to the Siebel Server utilizing the user name and password given in the form.

## 5.12.2 Specifying the Business Service

Additional information must be provided to specify the specific method of the business service to be executed. Typically, this information is placed in the configuration file associated with the application. Since the e\*Way uses the EAI Object Manager, the appropriate file is `eai.cfg`. This file has two sections that are used by the HTTP adapter, **HTTP Services** and a user-defined method information section. **HTTP Services** is the section in which you define the **SWExtSource** and the name of the method. The method section allows you to define the adapter mode and the name and method of the Business Service.

Below is an example of how an HTTP Service is specified:

```
[HTTP Services]
ACCOUNT_UPSERT_SERVICE = ACCOUNT_UPSERT_METHOD

[ACCOUNT_UPSERT_METHOD]
Mode = Document
Service = ACCOUNT_UPSERT
Method = RunProcess
```

In this example, the method **RunProcess** of the Business Service **ACCOUNT\_UPSERT** is executed if the form has an “input” **SWExtSource** with the value “**ACCOUNT\_UPSERT\_SERVICE**”.

An adapter in **Document** mode sends data across a specific data transport without converting the data to a property set. A Business Service of class **CSSWfEngine** is provided, which has a **RunProcess** method to execute a workflow process. The name of the process (i.e., **ProcessName**) needs to be specified in the **BIM BS User Property**.

## 5.12.3 The Siebel Workflow Process

The Workflow process has the following properties:

- **<Value>** with a type **String**  
This property refers to the **Value** attribute of the property set that is currently active. In the workflow, it can be either the **Inputs** or **Outputs** property set that executes it. In the **Inputs** property set, **Value** contains the incoming XML message; in the **Outputs** property set, **Value** consists of a result string that can be sent back to the Web page.
- **IncomingXML** with a type **String** and a default value **<Value>**  
Anything you pass along to the URL as data is placed in this variable.
- **Message** with a type **Hierarchy**  
The message is used to hold the intermediate property set that is generated by the EAI XML Converter.



## 5.12.4 e\*Gate-to-Siebel Example Procedure

### To insert or update an Employee Record

- 1 Make a copy of the template `SeeBeyond HTTP Update`.
  - A Change the name of the Workflow to `Employee Update Workflow`.
  - B Specify the **Business Object** to be `Employee`.
- 2 The Update Siebel Business Service is hard-coded with the return value `<h1>Update completed. </h1>`. You may leave it as it is.
- 3 In **Siebel Tools**, make a copy of the **Workflow Process Manager Business Service**.
  - A Change the value of **Name** to `Employee Update Business Service`.
  - B Change the value of **Project** to `EAI`.
- 4 Add a new **Business Service User Property** named `ProcessName` with the value of `Employee Update Workflow`.
- 5 Next, add the following sections in the `eai.cfg` file. It should be located in `siebsrvr\bin` directory.

```
[HTTP Services]
...
EUHS = EMPLOYEE_UPDATE_HTTP_SERVICE

[EMPLOYEE_UPDATE_HTTP_SERVICE]
Mode = Document
Service = Employee Update Business Service
Method = RunProcess
```
- 6 Compile the `.srf` file.
- 7 In **Siebel Tools**, export the **Employee Integration Object**.
- 8 Run the **SeeBeyond XML Converter** to generate the **Employee Integration Object ETD**.
- 9 Assuming that you have defined a **Collaboration** that satisfies your requirements, you are now ready to modify the **Collaboration Rules** script.
  - A Using the **e\*Gate Editor**, open the direct-database-access script `siebel-http-outgoing-insert.dsc`.
  - B Change the **Integration Object** from `Sample Account` to `Employee`.
  - C Change the **HTTP Service** name from `SEEBYOND_HTTP_UPDATE` to `EUHS`.
  - D Since you only want to perform an **Insert/Update**, delete the `siebel-http-process` call that performs the query operation.
  - E Modify the script to match the **Collaboration** that you defined.
  - F Save the modified **Collaboration Rules** script under a different name.

**Note:** See [Siebel XML Messages](#) on page 95 regarding the message format.

## 5.12.5 Siebel-to-e\*Gate Example Procedure

**Note:** This example is for Windows only. The sample schema [JavaSiebelInbound](#) on page 127 provides an example using the CGI e\*Way, which is the preferred method.

### To retrieve an Employee Record and forward it to the e\*Gate system

- 1 Make a copy of the template **SeeBeyond HTTP Send**.
  - A Change the name of the Workflow to **Employee Send Workflow**.
  - B Specify the **Business Object** to be **Employee**.
- 2 The Send Business Service is hard-coded with the Request URL Template value **http://<web server>/mux.asp**. You need to specify the **MS IIS** as the web server.
- 3 Since you are testing the implementation in Siebel Workflow Designer, you need to change the value of **Object Id** of the Process Properties to the value used in your system (in this example, assume that **1-D9T** is the correct ID).
- 4 In the **MS IIS**:
  - A Modify the **Mux.asp** to have the IP address and port number of the Siebel EAI (MUX) e\*Way.
  - B Since you are not gathering data from a form, set **blnUseBinary = true**.
- 5 In **Siebel Tools**, export the **Employee** Integration Object.
- 6 Run the **SeeBeyond XML Converter** to generate the **Employee** Integration Object ETD.
- 7 Create the e\*Gate Collaboration to process the ETD.

**Note:** See [Siebel XML Messages](#) on page 95 regarding the message format.

## 5.13 Sample Schema

Sample implementations are located in the `\samples\ewsiebelhttp\Siebel7` directory of the e\*Gate CD-ROM (see [Optional Example Files](#) on page 26 for installation instructions):

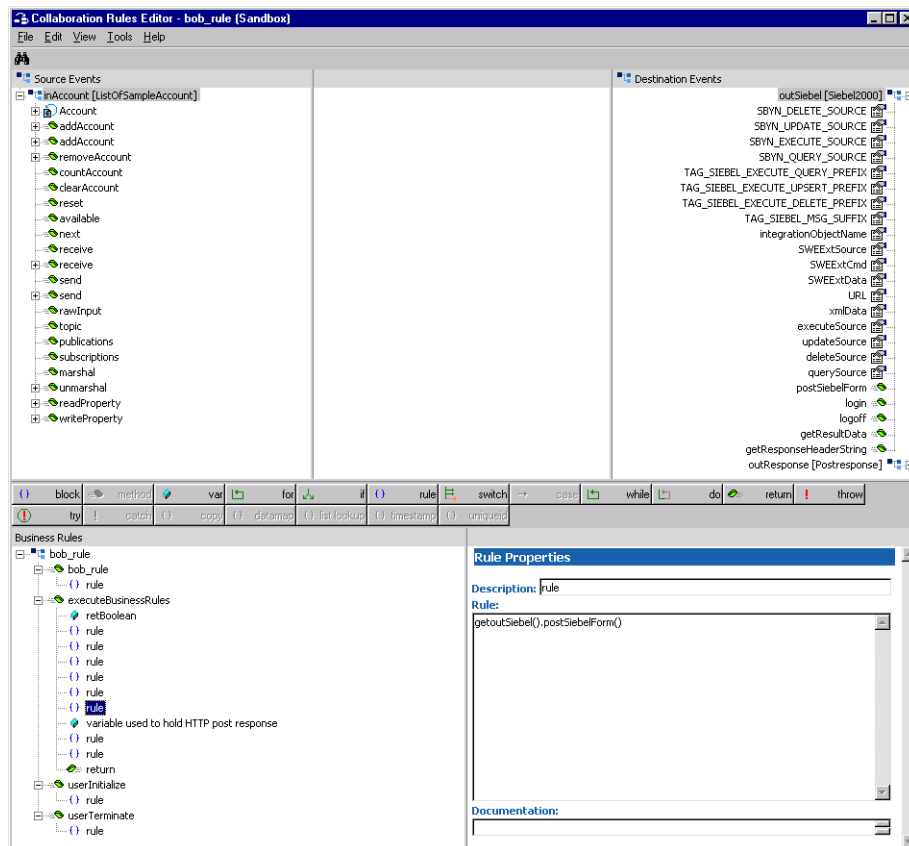
- [JavaSiebelOutbound.zip](#)
- [JavaSiebelInbound.zip](#)

These examples make use of the SeeBeyond Workflow Templates included with the e\*Way. You must set up your environment by following the instructions on setting up the templates to execute the Workflow in [SeeBeyond Workflow Templates](#) on page 90.

### 5.13.1 JavaSiebelOutbound

This e\*Gate-to-Siebel sample schema can be used to test your system following installation. Note that it uses a Business Object Broker (BOB) rather than the Multi-Mode e\*Way. For your own schemas, however, it is recommended that you use the Multi-Mode e\*Way executable. Figure 73 shows a sample Collaboration Rule within the schema.

**Figure 73** Collaboration Rules Editor Window - JavaSiebelOutbound



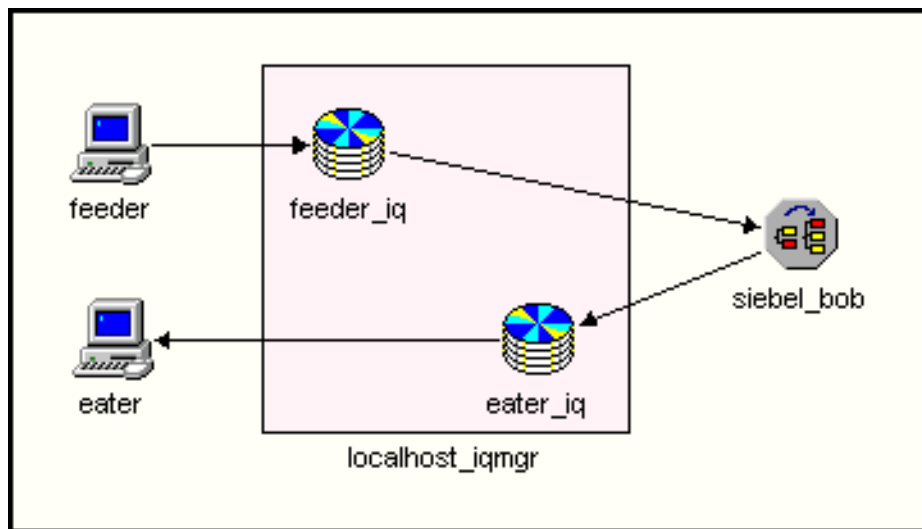
## Components

The e\*Gate-to-Siebel example, **JavaSiebelOutbound**, sets up a single instance of the Siebel EAI e\*Way and two of the File e\*Way, having the logical names shown in the following table.

e*Way Type	Logical Name
Siebel EIM e*Way	siebel_bob
File e*Way	feeder
	eater

It also sets up two Intelligent Queues, with the logical names **feeder\_iq** and **eater\_iq**.

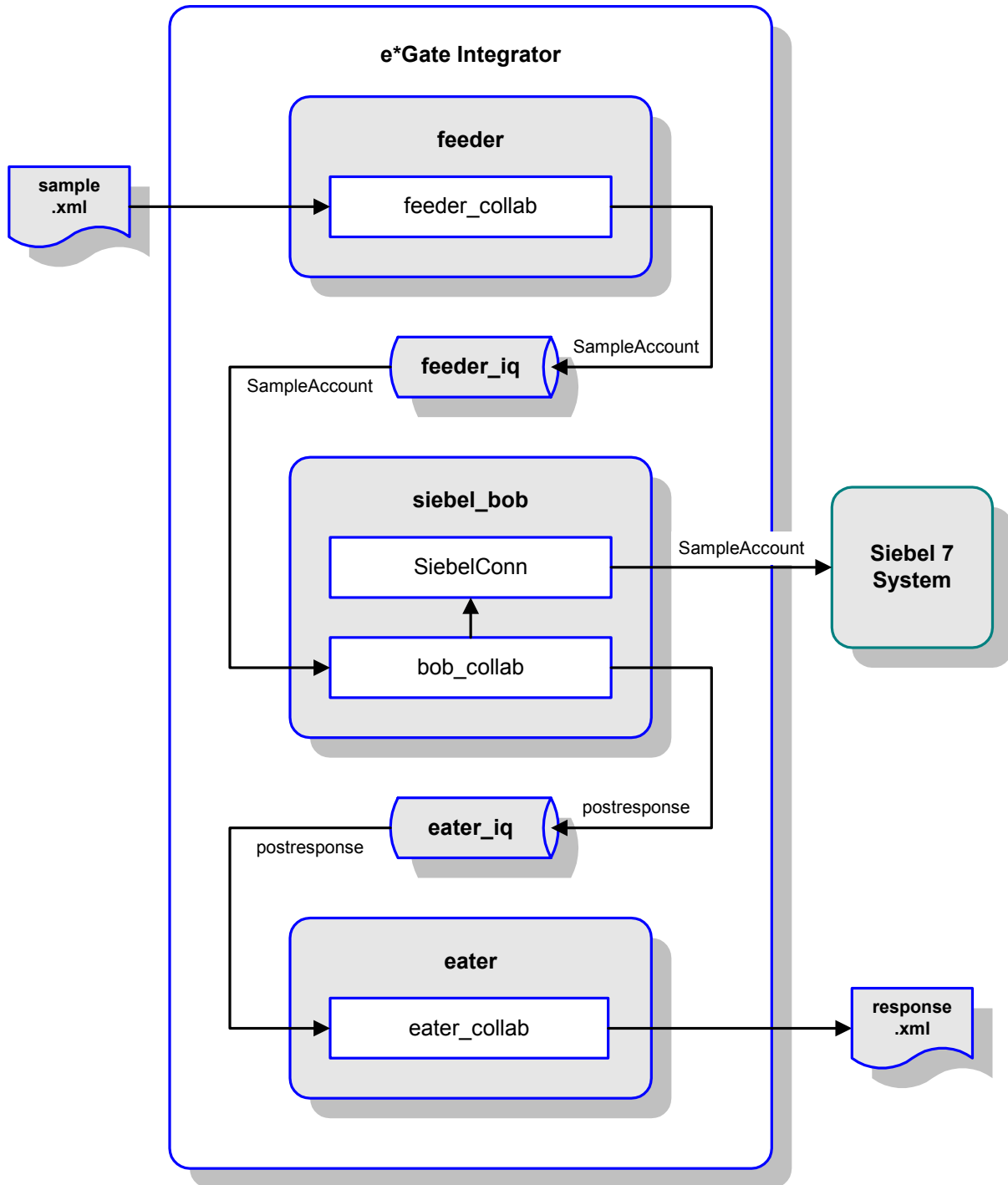
**Figure 74** JavaSiebelOutbound Components



## Event Types

There are two Event types, **SampleAccount** and **postresponse**, representing account data from another source to be posted to Siebel, and a response message or acknowledgement. These Event types are passed from one component to another following three Collaborations, as outlined next and diagrammed in Figure 75.

Figure 75 JavaSiebelOutbound Schema

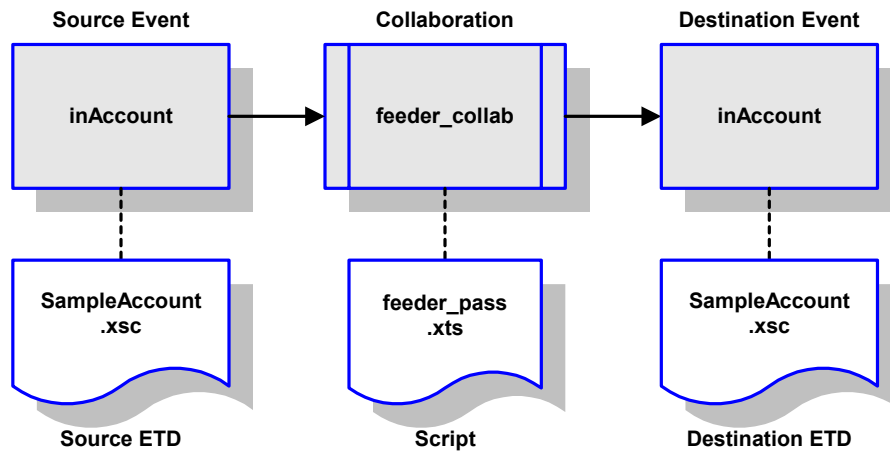


## Collaborations

### feeder\_collab

This pass-through Collaboration, performed by the e\*Way **feeder**, subscribes to an Event **InAccount** of Event Type **SampleAccount** from an external source and publishes it to the **feeder\_iq** without transformation.

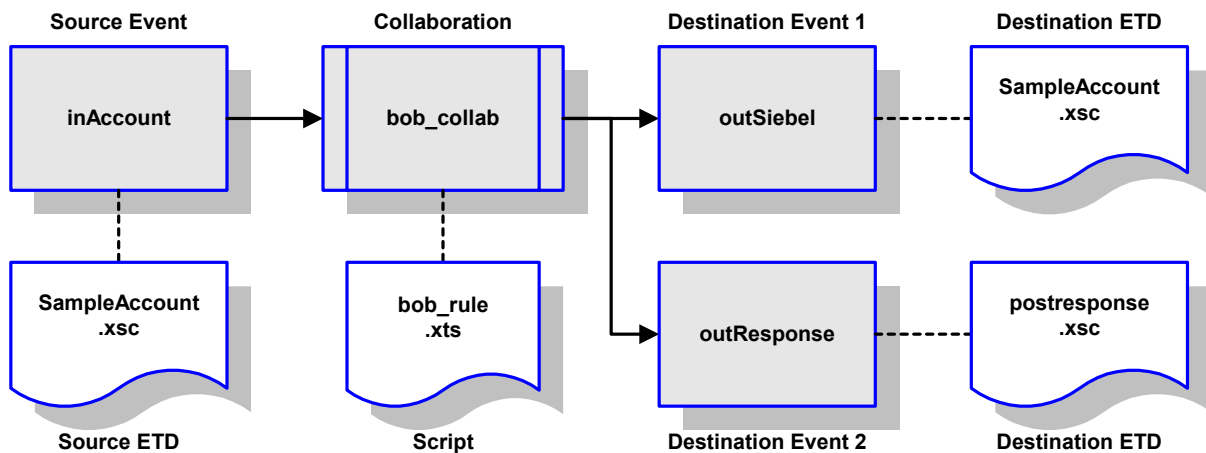
**Figure 76** feeder\_collab Collaboration



### bob\_collab

This Collaboration, performed by the e\*Way **siebel\_bob**, subscribes to the Collaboration **feeder\_collab** and receives the Event **InAccount**. It then publishes it as Event **outSiebel**, still of Event Type **SampleAccount**, through the e\*Way Connection **SiebelConn** to the Siebel system. It also publishes the message **outResponse** of Event Type **postresponse** to the **eater\_iq**.

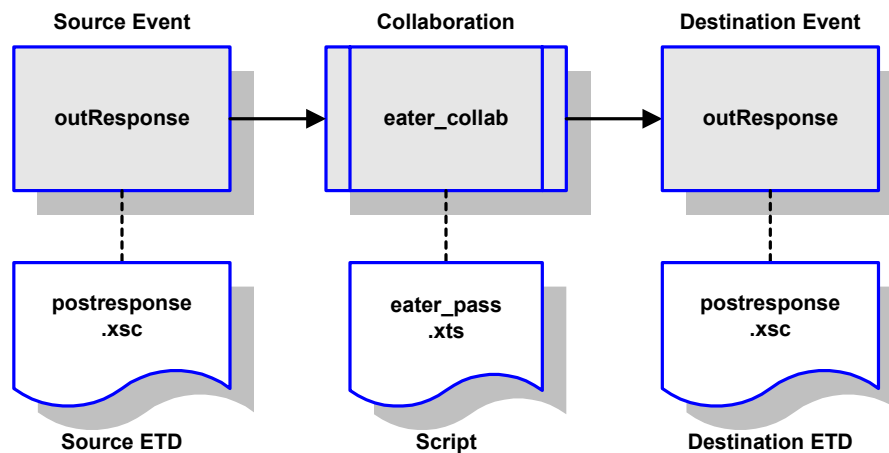
**Figure 77** bob\_collab Collaboration



## eater\_collab

This pass-through Collaboration, performed by the e\*Way **eater**, subscribes to the Collaboration **bob\_collab** through the **eater\_iq**. It receives the Event **outResponse** of Event Type **postresponse** and publishes it to an external destination without transformation.

**Figure 78** eater\_collab Collaboration



## 5.13.2 JavaSiebellnbound

This Siebel-to-e\*Gate sample schema can be used to test your system following installation. It makes use of the CGI e\*Way to perform a Collaboration that subscribes to a JMS e\*Way Connection. The e\*Way Connection receives data from, and returns an acknowledgment to, the Siebel 7 system through the IIS Web Server.

### Components

The Siebel-to-e\*Gate example, **JavaSiebellnbound**, sets up a single instance of the CGI e\*Way and one of the File e\*Way, having the logical names shown in the following table.

e*Way Type	Logical Name
CGI e*Way	ewwebRequestETDReplyETD
File e*Way	ewfileOut

It also sets up a JMS e\*Way Connection, with the logical name **cpBackstayJMS**.

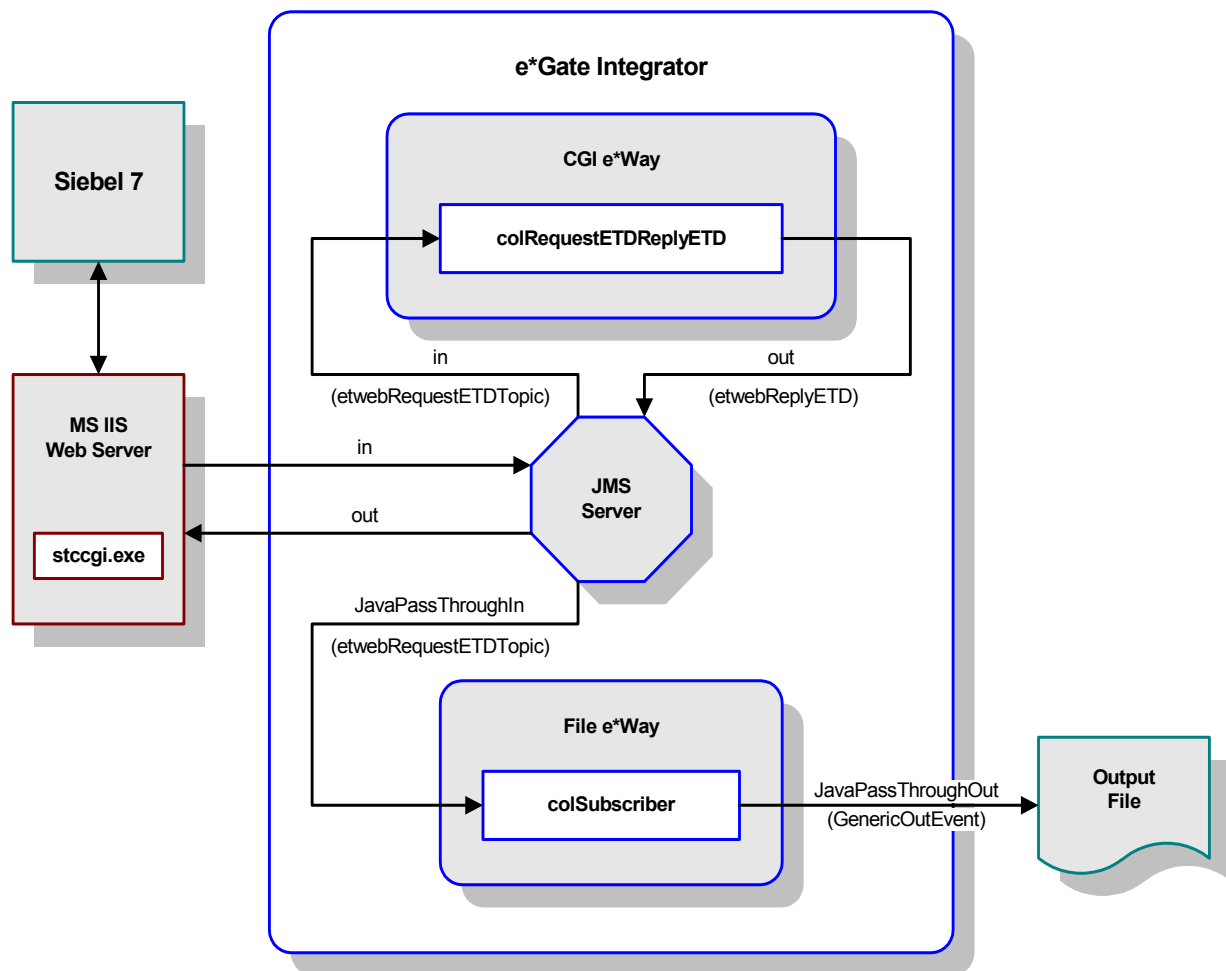
## Event Types

There are three Event types:

- etwebRequestETDTopic
- etwebReplyETD
- GenericOutEvent

which represent account data from Siebel, a response message or acknowledgement to Siebel, and an output data file, respectively. These Event types are passed from one component to another following two Collaborations, as outlined next and diagrammed in Figure 79.

Figure 79 JavaSiebellnbound Schema



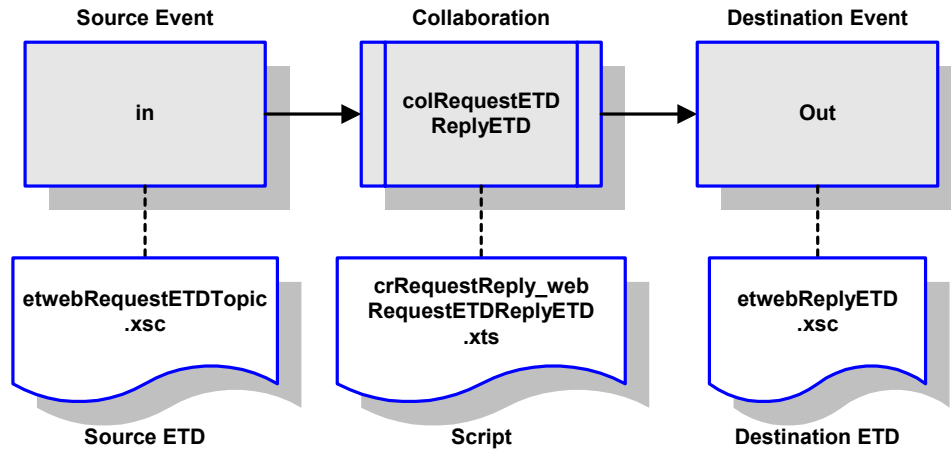


## Collaborations

### colRequestETDReplyETD

This Collaboration, performed by the CGI e\*Way **ewwebRequestETDReplyETD**, subscribes to the JMS e\*Way Connection. It receives the Event **in** of Event Type **etwebRequestETDTopic** and sends an acknowledgment **out**, of Event Type **etwebReplyETD**, to the JMS CP to publish to the Siebel 7 system via the IIS Web Server.

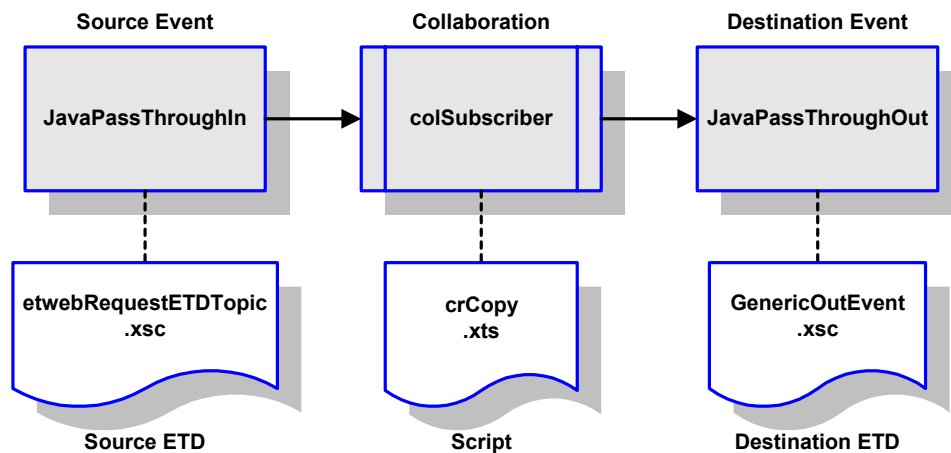
**Figure 80** colRequestETDReplyETD



### colSubscriber

This pass-through Collaboration, performed by the e\*Way **ewfileOut**, subscribes to the JMS e\*Way Connection. It receives the Event **JavaPassThroughIn** of Event Type **etwebRequestETDTopic** and publishes it to an external destination without transformation.

**Figure 81** colSubscriber



# e\*Way Setup

This chapter describes the procedures required to customize the SeeBeyond e\*Way Intelligent Adapter for Siebel EAI to operate within your production system.

---

## 6.1 Overview

After creating a schema, you must instantiate and configure the e\*Way Intelligent Adapter for Siebel EAI to operate within the schema. A wide range of setup options allow the e\*Way to conform to your system's operational characteristics and your facility's operating procedures.

The topics discussed in this chapter include the following:

[Setting Up the e\\*Way](#) on page 131

[Creating e\\*Way Connections](#) on page 138

[Using the e\\*Way Configuration Editor](#) on page 141

[Troubleshooting the e\\*Way](#) on page 144

## 6.2 Setting Up the e\*Way

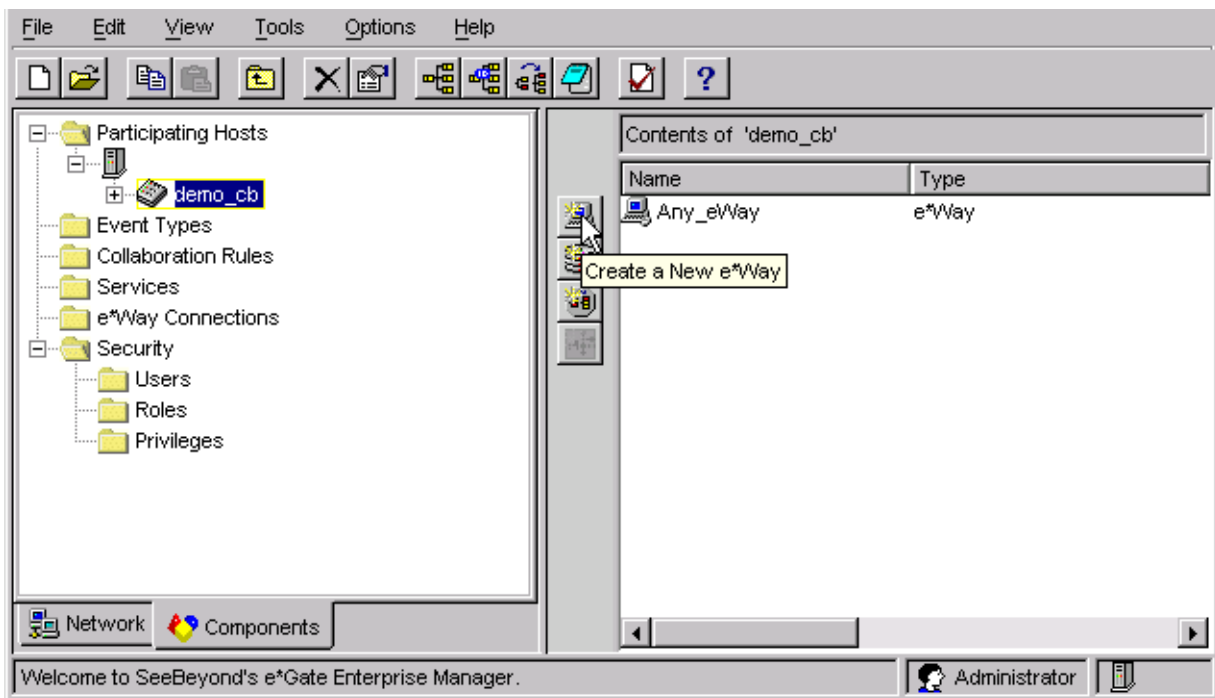
### 6.2.1 Creating the e\*Way

The first step in implementing an e\*Way is to define the e\*Way component using the e\*Gate Schema Designer.

To create an e\*Way

- 1 Open the schema in which the e\*Way is to operate.
- 2 Select the e\*Gate Schema Designer Navigator's **Components** tab.
- 3 Open the host on which you want to create the e\*Way.
- 4 Select the Control Broker you want to manage the new e\*Way.

**Figure 82** e\*Gate Schema Designer Window (Components View)



- 5 On the Palette, click **Create a New e\*Way**.
- 6 Enter the name of the new e\*Way, then click **OK**.
- 7 All further actions are performed in the e\*Gate Schema Designer Navigator's **Components** tab.

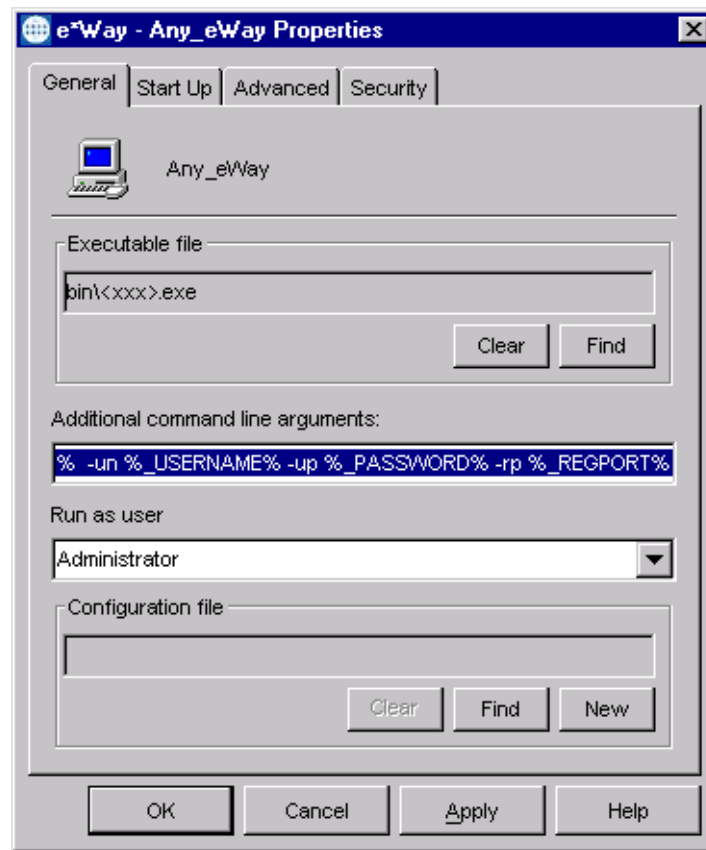
## 6.2.2 Modifying e\*Way Properties

To modify any e\*Way properties

- 1 Right-click on the desired e\*Way and select **Properties** to edit the e\*Way's properties. The properties dialog opens to the **General** tab (shown in Figure 83).

**Note:** The executable and default configuration files used by this e\*Way are listed in **e\*Way Components** on page 17.

**Figure 83** e\*Way Properties (General Tab)



- 2 Make the desired modifications, then click **OK**.

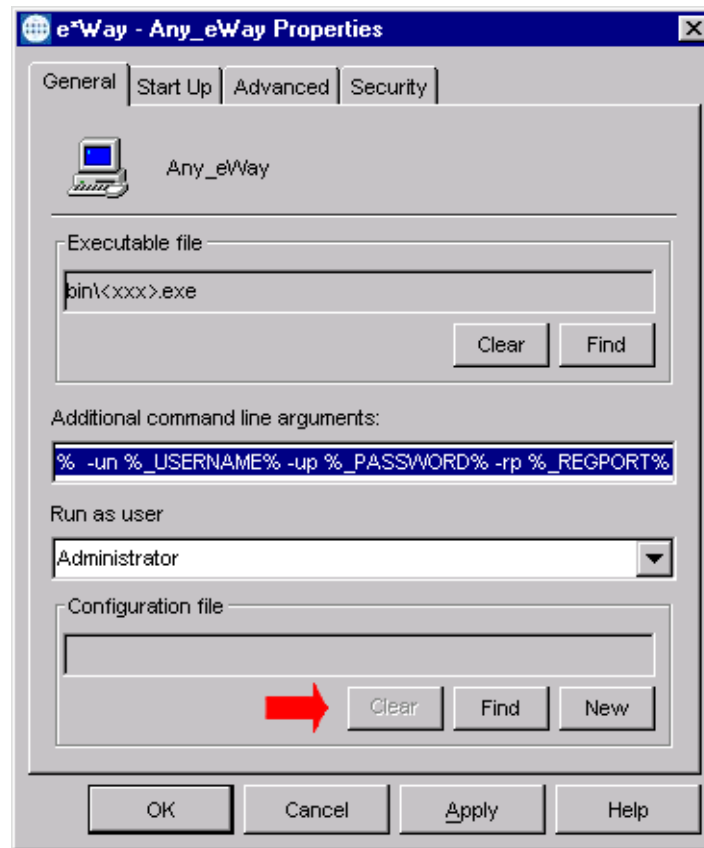
### 6.2.3 Configuring the e\*Way

The e\*Way's inherent configuration parameters are stored in an ASCII text file with a .def extension. The e\*Way Editor provides a simple graphical interface for viewing and changing those parameters to create a working configuration (.cfg) file.

To change e\*Way configuration parameters

- 1 In the e\*Gate Schema Designer's Component editor, select the e\*Way you want to configure and display its properties.

**Figure 84** e\*Way Properties - General Tab



- 2 Under **Executable File**, click **Find** to locate `stceway.exe`.
- 3 Under **Configuration File**, click **New** to create a new file or **Find** to select an existing configuration file. If you select an existing file, an **Edit** button appears; click the button to edit the currently selected file.
- 4 You now are in the e\*Way Configuration Editor (see [Using the e\\*Way Configuration Editor](#) on page 141). The e\*Way's inherent configuration parameters are described in [Multi-Mode e\\*Way](#) on page 157.

## 6.2.4 Changing the User Name

Like all e\*Gate executable components, e\*Ways run under an e\*Gate user name. By default, all e\*Ways run under the **Administrator** user name. You can change this if your site's security procedures so require.

### To change the user name

- 1 Display the e\*Way's properties dialog.
- 2 On the **General** tab, use the **Run as user** list to select the e\*Gate user under whose name this component runs.

See the *e\*Gate Integrator System Administration and Operations Guide* for more information on the e\*Gate security system.

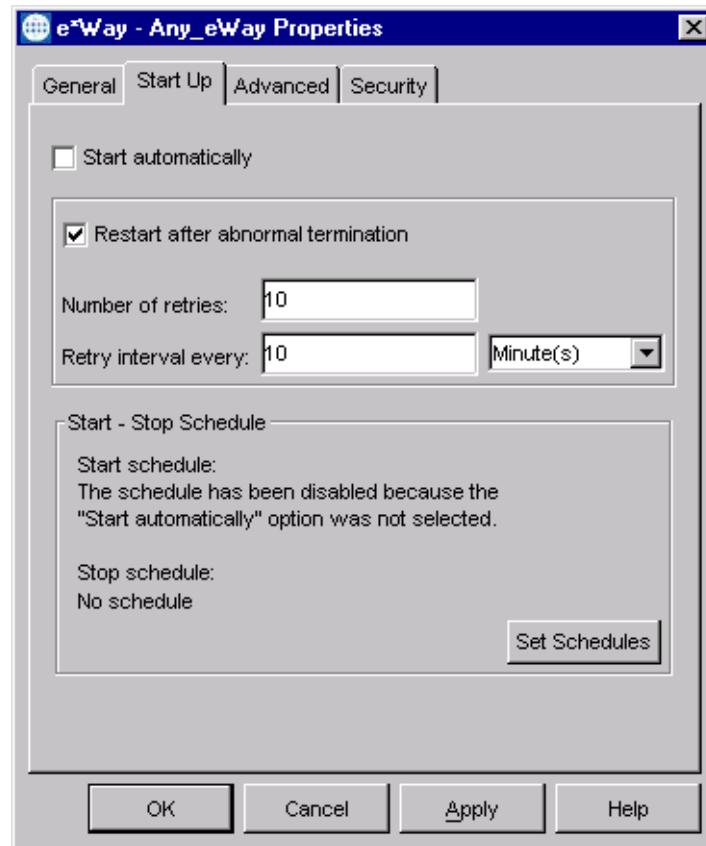
## 6.2.5 Setting Startup Options or Schedules

SeeBeyond e\*Ways can be started or stopped by any of the following methods:

- The Control Broker can start the e\*Way automatically whenever the Control Broker starts.
- The Control Broker can start the e\*Way automatically whenever it detects that the e\*Way terminated execution abnormally.
- The Control Broker can start or stop the e\*Way on a schedule that you specify.
- Users can start or stop the e\*Way manually using an interactive monitor.

You determine how the Control Broker starts or shuts down an e\*Way using options on the e\*Way properties **Start Up** tab (see Figure 85). See the *e\*Gate Integrator System Administration and Operations Guide* for more information about how interactive monitors can start or shut down components.

**Figure 85** e\*Way Properties (Start-Up Tab)



**To set the e\*Way's startup properties**

- 1 Display the e\*Way's properties dialog.
- 2 Select the **Start Up** tab.
- 3 To have the e\*Way start automatically when the Control Broker starts, select the **Start automatically** check box.
- 4 To have the e\*Way start manually, clear the **Start automatically** check box.
- 5 To have the e\*Way restart automatically after an abnormal termination:
  - A Select **Restart after abnormal termination**.
  - B Set the desired number of retries and retry interval.
- 6 To prevent the e\*Way from restarting automatically after an abnormal termination, clear the **Restart after abnormal termination** check box.
- 7 Click **OK**.

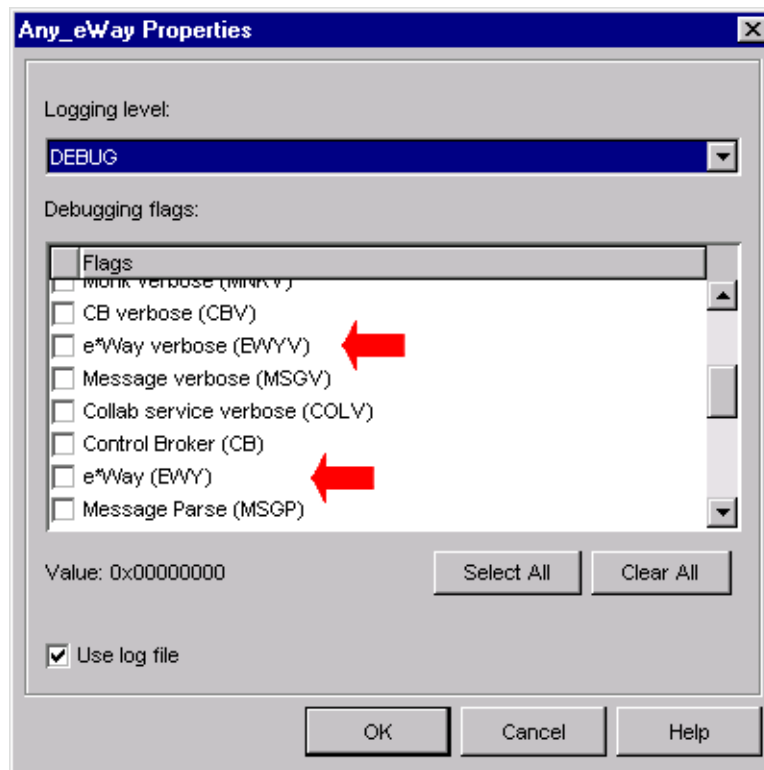
## 6.2.6 Activating or Modifying Logging Options

Logging options enable you to troubleshoot problems with the e\*Way and other e\*Gate components.

To set the e\*Way debug level and flag

- 1 Display the e\*Way's Properties dialog.
- 2 Select the **Advanced** tab.
- 3 Click **Log**. The dialog window appears as in Figure 86.

**Figure 86** e\*Way Properties (Advanced Tab - Log Option)



- 4 Select **DEBUG** for the **Logging level**.
- 5 Select either **e\*Way (EWY)** or **e\*Way Verbose (EWYV)** for the **Debugging flag**. Note that the latter has a significant impact on system performance.
- 6 Click **OK**.

The other options apply to other e\*Gate components and are activated in the same manner. See the *e\*Gate Integrator Alert and Log File Reference* for additional information concerning log files, logging options, logging levels, and debug flags.



## 6.2.7 Activating or Modifying Monitoring Thresholds

Monitoring thresholds enable you to monitor the throughput of the e\*Way. When the monitoring thresholds are exceeded, the e\*Way sends a Monitoring Event to the Control Broker, which is routed to the Schema Manager and any other configured destinations.

- 1 Display the e\*Way's properties dialog.
- 2 Select the **Advanced** tab.
- 3 Click **Thresholds**.
- 4 Select the desired threshold options and click **OK**.

See the *e\*Gate Integrator Alert and Log File Reference* for more information concerning threshold monitoring, routing specific notifications to specific recipients, or for general information about e\*Gate's monitoring and notification system.

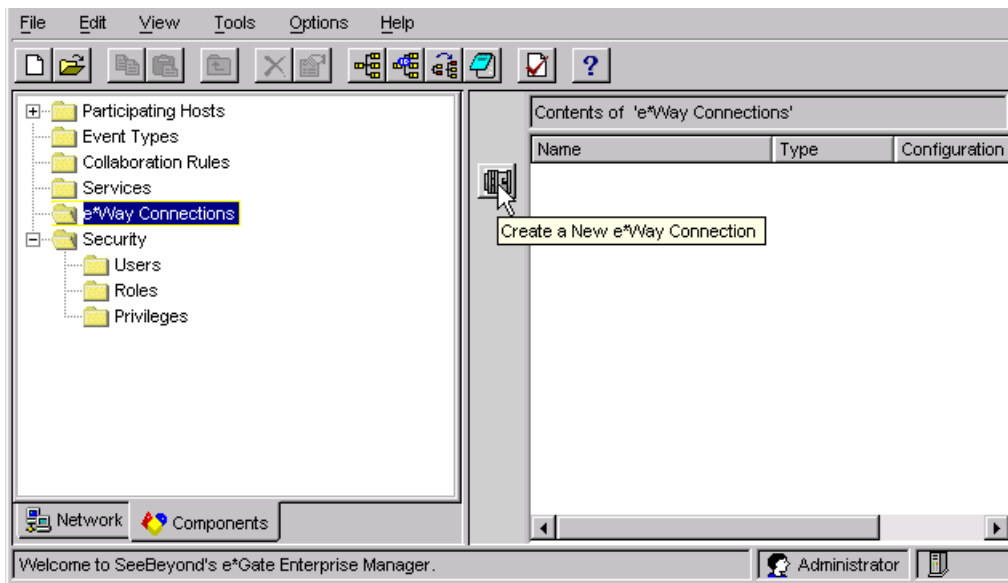
## 6.3 Creating e\*Way Connections


The e\*Way Connection's configuration parameters are stored in an ASCII text file with a .def extension. The e\*Way Editor provides a simple graphical interface for viewing and changing those parameters to create a working configuration (.cfg) file.

To create and configure the e\*Way Connections

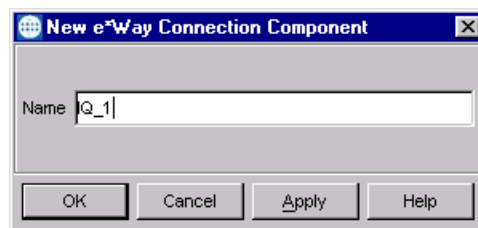
- 1 In the Schema Designer's Component editor, select the **e\*Way Connections** folder.

**Figure 87** Schema Designer - e\*Way Connections Folder (1)



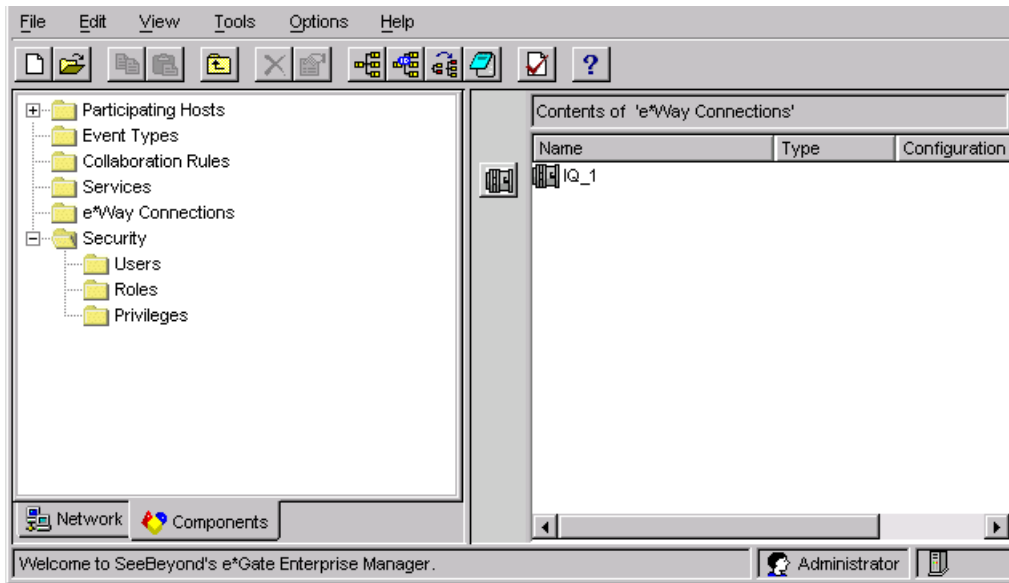
- 2 On the Palette, click the **Create a New e\*Way Connection** button , which opens the New e\*Way Connection Component dialog box.

**Figure 88** New e\*Way Connection Component Dialog Box



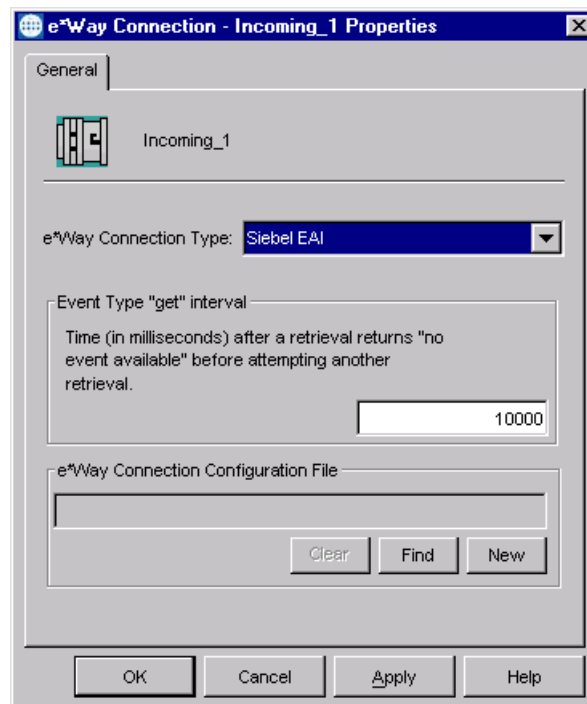
- 3 Enter a name for the e\*Way Connection and click **OK**. The new e\*Way Connection appears in the Schema Designer Contents pane.

Figure 89 Schema Designer - e\*Way Connections Folder (2)



- 4 Right-click the new e\*Way Connection icon and select **Properties** to open the e\*Way Connection Properties dialog box.

Figure 90 e\*Way Connection Properties Dialog Box



- 5 From the e\*Way Connection Type drop-down box, select **Siebel EAI**.
- 6 Enter the Event Type *get* interval in the dialog box provided (optional).

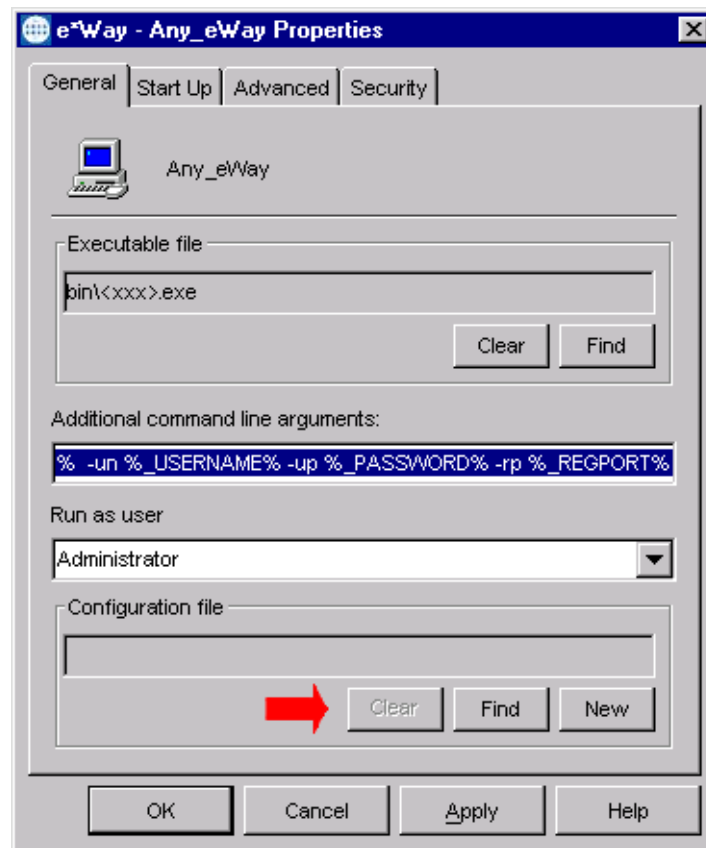
- Click **New** to invoke the e\*Way Connection Configuration File Editor, where you can create a new e\*Way Connection Configuration File.

**To change the e\*Way Connections**

- In the e\*Gate Schema Designer’s Component editor, select the e\*Way you want to configure and display its properties.

**Note:** The executable and default configuration files used by this e\*Way are listed in **e\*Way Components** on page 17.

**Figure 91** e\*Way Properties - General Tab



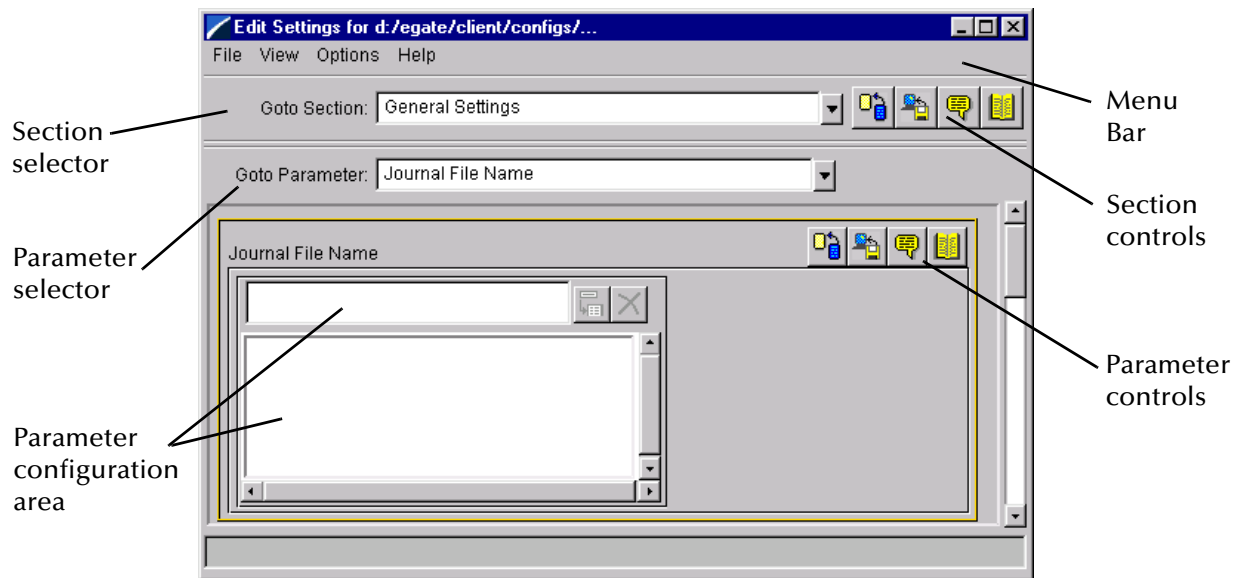
- Under **Configuration File**, click **New** to create a new file or **Find** to select an existing configuration file. If you select an existing file, an **Edit** button appears, which you can click to edit the currently selected file.
- You are now in the e\*Way Configuration Editor (see **Using the e\*Way Configuration Editor** on page 141). The e\*Way Connection’s configuration parameters are described in **e\*Way Connections** on page 163.

**Note:** You must restart the e\*Way after changing the e\*Way connection.

## 6.4 Using the e\*Way Configuration Editor

The e\*Way's default configuration parameters are stored in an ASCII text file with a **.def** extension. The e\*Way Editor provides a simple graphical interface for viewing and changing those parameters to create a working configuration (**.cfg**) file.

**Figure 92** The e\*Way Configuration Editor







The e\*Way Configuration Editor controls fall into one of six categories:

- The **Menu bar** allows access to basic operations (e.g., saving the configuration file, viewing a summary of all parameter settings, and launching the Help system)
- The **Section selector** at the top of the Editor window enables you to select the category of the parameters you wish to edit
- **Section controls** enable you to restore the default settings, restore the last saved settings, display tips, or enter comments for the currently selected section
- The **Parameter selector** allows you to jump to a specific parameter within the section, rather than scrolling
- **Parameter controls** enable you to restore the default settings, restore the last saved settings, display tips, or enter comments for the currently selected parameter
- **Parameter configuration controls** enable you to set the e\*Way's various operating parameters

## 6.4.1 Section and Parameter Controls

The section and parameter controls are shown in Table 12 below.

**Table 12** Parameter and Section Controls

Button	Name	Function
	<b>Restore Default</b>	Restores default values
	<b>Restore Value</b>	Restores saved values
	<b>Tips</b>	Displays tips
	<b>User Notes</b>	Enters user notes



*Note: The section controls affect all parameters in the selected section, whereas the parameter controls affect only the selected parameter.*

## 6.4.2 Parameter Configuration Controls

Parameter configuration controls fall into one of two categories:

- Option buttons
- Selection lists, which have controls as described in Table 13

**Table 13** Selection List Controls

Button	Name	Function
	<b>Add to List</b>	Adds the value in the text box to the list of available values.
	<b>Delete Items</b>	Displays a “delete items” dialog box, used to delete items from the list.

### 6.4.3 Command-line Configuration

In the **Additional Command Line Arguments** box, type any additional command line arguments that the e\*Way may require, taking care to insert them *at the end* of the existing command-line string. Be careful not to change any of the default arguments unless you have a specific need to do so.

### 6.4.4 Getting Help

To launch the e\*Way Editor's Help system

From the **Help** menu, select **Help topics**.

To display tips regarding the general operation of the e\*Way

From the **File** menu, select **Tips**.

To display tips regarding the selected Configuration Section

In the **Section Control** group, click .

To display tips regarding the selected Configuration Parameter

In the **Parameter Control** group, click .

**Note:** *"Tips" are displayed and managed separately from the online Help system. You cannot search for Tips within the Help system, or view Help system topics by requesting Tips.*

For detailed descriptions and procedures for using the e\*Way Configuration Editor, see the *e\*Gate Integrator User's Guide*.

---

## 6.5 Troubleshooting the e\*Way

In the initial stages of developing your e\*Gate Integrator system administration system, most problems with e\*Ways can be traced to configuration.

### 6.5.1 Configuration Problems

#### In the Schema Designer

- Does the e\*Way have the correct Collaborations assigned?
- Do those Collaborations use the correct Collaboration Services?
- Is the logic correct within any Collaboration Rules script employed by this e\*Way's Collaborations?
- Do those Collaborations subscribe to and publish Events appropriately?
- Are all the components that provide information to this e\*Way properly configured, and are they sending the appropriate Events correctly?
- Are all the components to which this e\*Way sends information properly configured, and are they subscribing to the appropriate Events correctly?

#### In the e\*Way Editor

- Check that all e\*Way connection options are set appropriately.
- Check that all settings you changed are set correctly.
- Check all required changes to ensure they have not been overlooked.
- Check the defaults to ensure they are acceptable for your installation.

#### On the e\*Way's Participating Host

- Check that the Participating Host is operating properly, and that it has sufficient disk space to hold the IQ data that this e\*Way's Collaborations publish.
- Check that the *path* environmental variable includes the location of the Siebel EAI dynamically-loaded libraries. The name of this variable on the different operating systems is:
  - ♦ PATH (Windows)
  - ♦ LD\_LIBRARY\_PATH (Solaris)
  - ♦ LIBPATH (AIX)

#### In the Siebel Application

- Check that the application is configured correctly, is operating properly, and is sending or receiving the correct data appropriately.



## 6.5.2 System-related Problems

- Check that the connection between the external application and the e\*Way is functioning appropriately.
- Once the e\*Way is up and running properly, operational problems can be due to:
  - ♦ External influences (network or other connectivity problems).
  - ♦ Problems in the operating environment (low disk space or system errors)
  - ♦ Problems or changes in the data the e\*Way is processing.
  - ♦ Corrections required to Collaboration Rules scripts that become evident in the course of normal operations.

One of the most important tools in the troubleshooter's arsenal is the e\*Way log file. See the *e\*Gate Integrator Alert and Log File Reference Guide* for an extensive explanation of log files, debugging options, and using the e\*Gate Schema Manager system to monitor operations and performance.

# Operational Overview

This chapter provides a brief overview of the way the SeeBeyond Java e\*Way Intelligent Adapter for Siebel EAI operates.

---

## 7.1 Overview

The e\*Way uses Java methods to exchange data with the external system, package data as e\*Gate *Events*, send those Events to Collaborations, and manage the connection between the e\*Way and the external system. This chapter gives an illustrated overview of the following topics:

[Multi-Mode e\\*Way Architecture](#) on page 147

[Collaborations and Event Type Definitions](#) on page 149

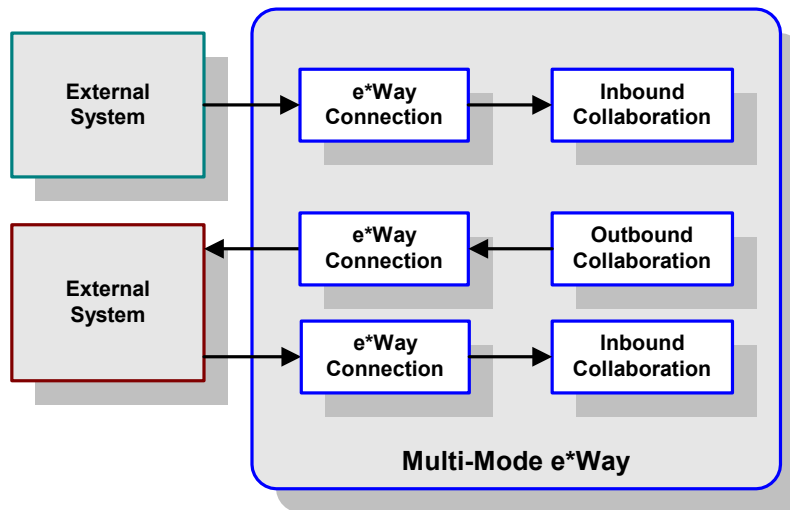
[e\\*Way Connections](#) on page 153

These topics also are covered in greater detail in the *e\*Gate Integrator User's Guide*.

## 7.2 Multi-Mode e\*Way Architecture

The Siebel EAI e\*Way is based on the SeeBeyond Multi-Mode e\*Way, which is a multi-threaded component forming an Intelligent Adapter for e\*Gate Integrator to exchange information with multiple external systems. The e\*Way connects to one or more external systems by means of *e\*Way Connections*, each of which must be configured for the specific external system to which it connects (see Figure 93).

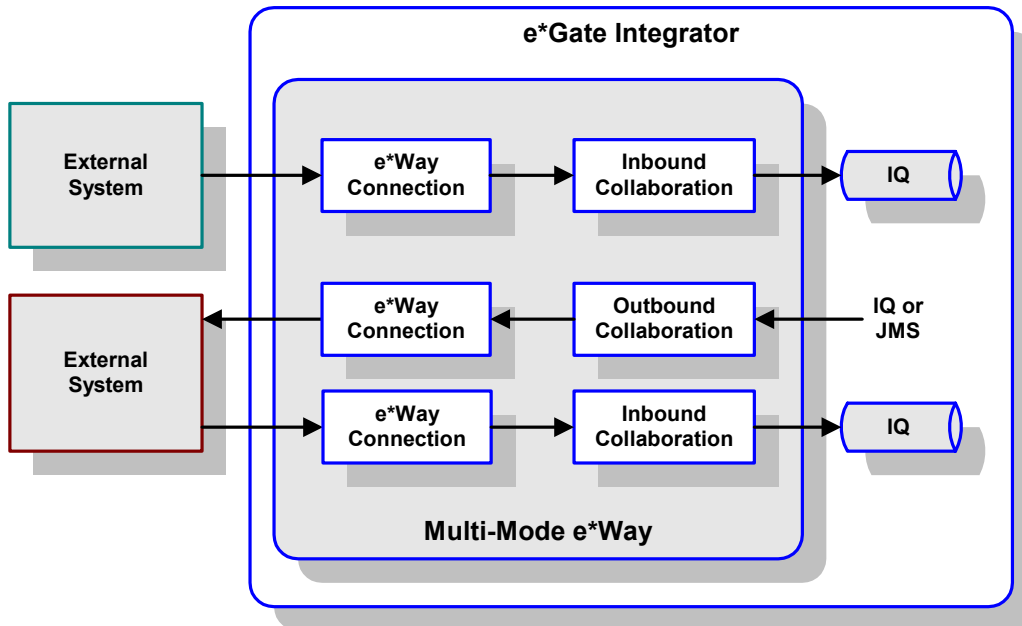
**Figure 93** Multi-Mode e\*Way



Each e\*Way performs one or more *Collaborations* (see [Collaborations and Event Type Definitions](#) on page 149). Bidirectional data flow requires at least two Collaborations, one *Inbound* and one *Outbound*, as shown in Figure 93. Each Collaboration processes a stream of messages, or *Events*, containing data or other information.

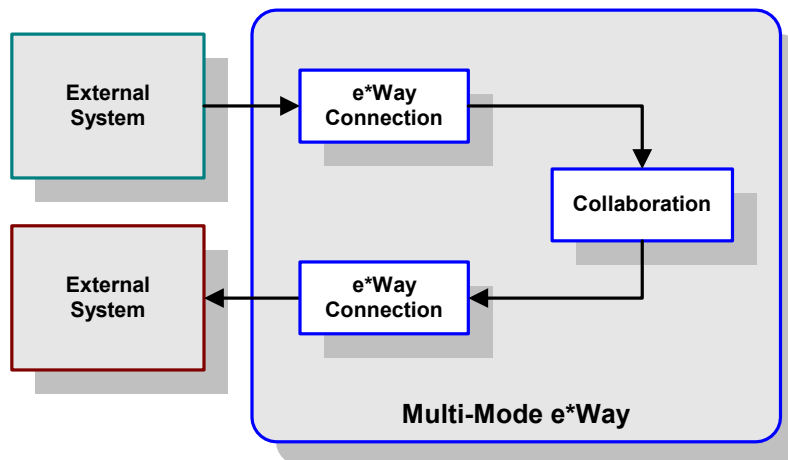
Each Collaboration that publishes its processed Events internally (within e\*Gate Integrator) requires one or more *Intelligent Queues* (IQs) to receive the Events (see Figure 94). Any Collaboration that publishes its processed Events only to an external system *does not* require an IQ to receive Events.

Figure 94 e\*Way within e\*Gate Integrator



Although usually implemented within e\*Gate Integrator as shown in Figure 94, this e\*Way also can be implemented as a stand-alone bridge between two or more external systems (see Figure 95).

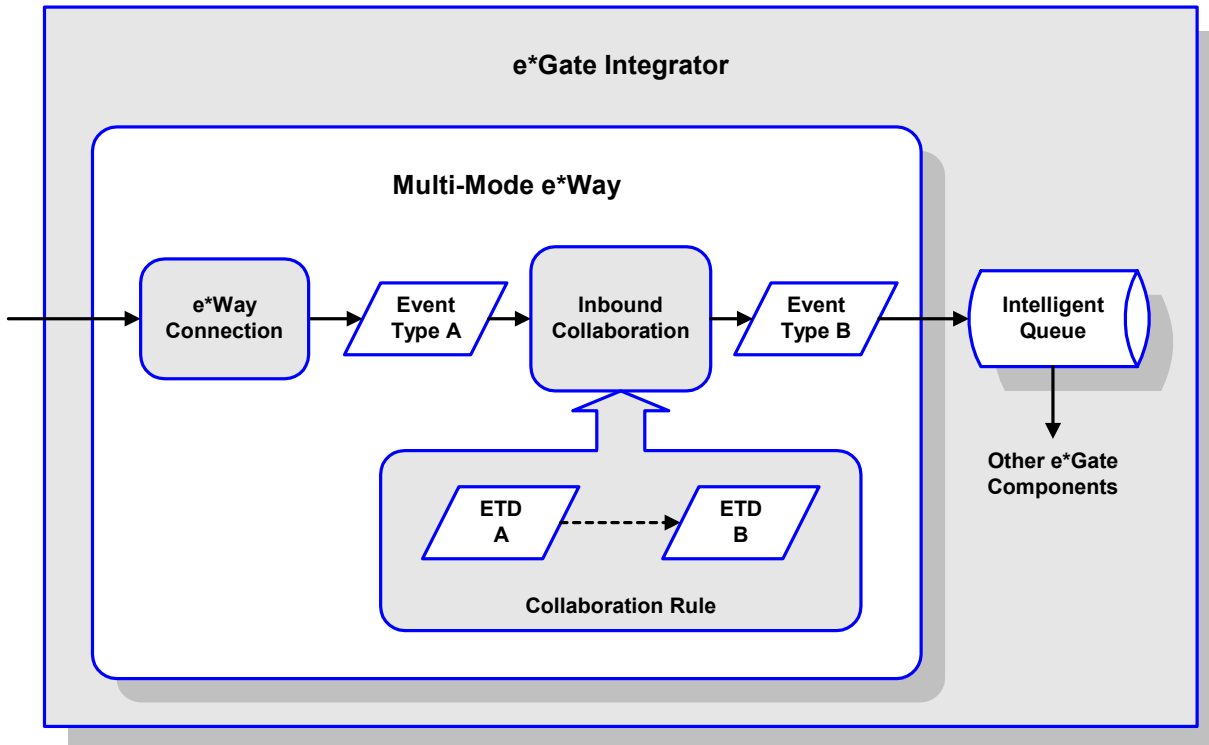
Figure 95 Stand-alone e\*Way



## 7.3 Collaborations and Event Type Definitions

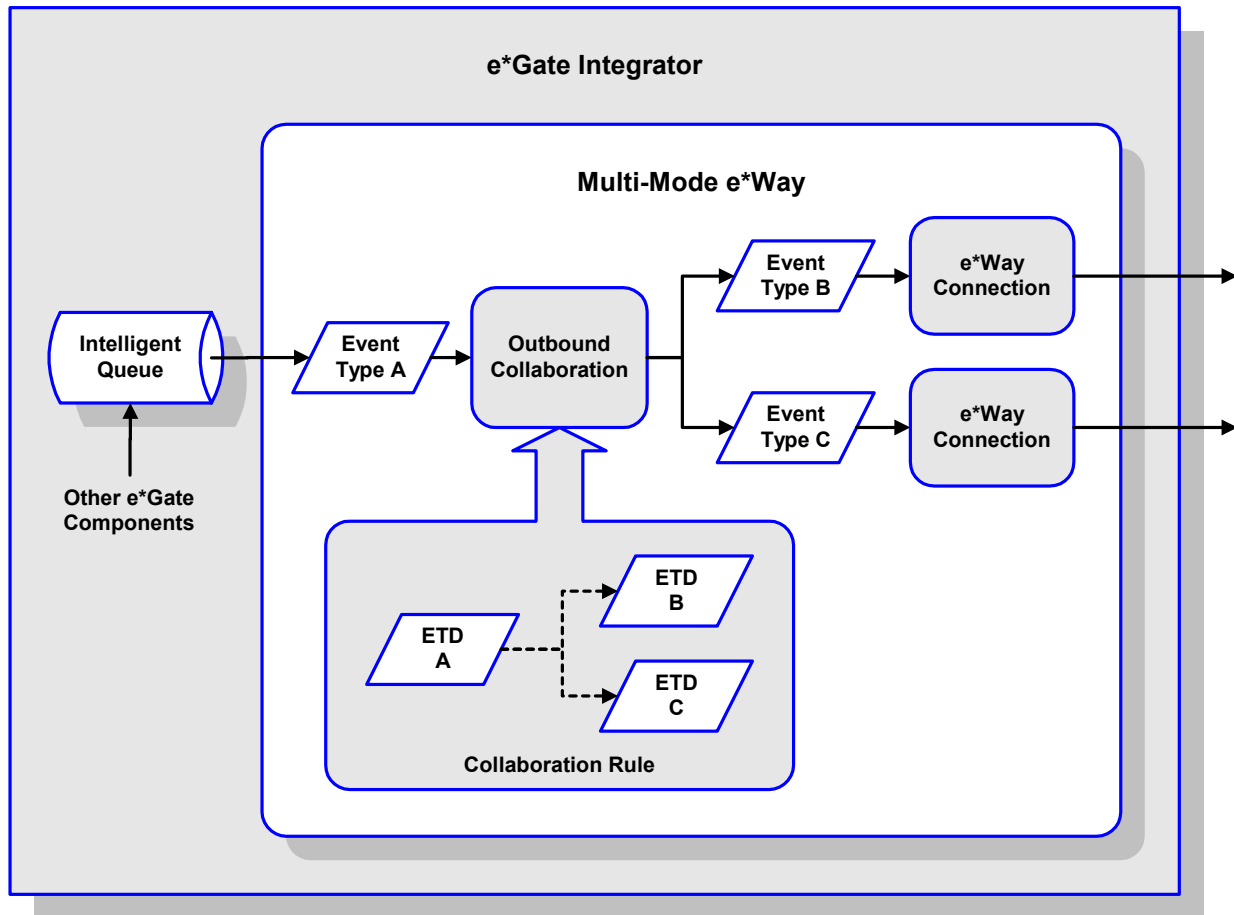
Collaborations execute the business logic that enable the e\*Way to perform its intended task. Each Collaboration executes a specified *Collaboration Rule*, which contains the actual instructions to execute the business logic and specifies the applicable *Event Type Definitions* (ETDs). Events Types represent *instances* of their corresponding ETDs. A look inside a typical inbound Collaboration is shown in Figure 96.

**Figure 96** Inbound Collaboration



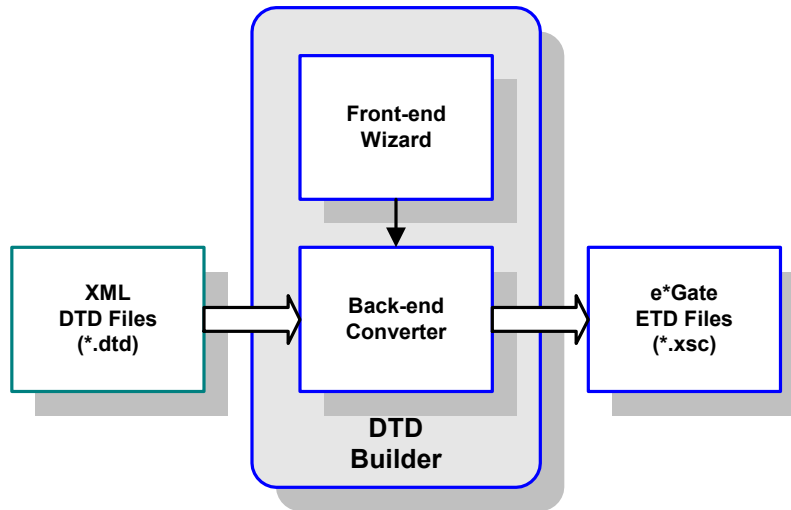
A corresponding look inside a typical outbound Collaboration is shown in Figure 97. In this diagram, two e\*Way Connections are shown, feeding two external systems. More than two e\*Way Connections can be accommodated in each e\*Way and, as stated previously, multiple Collaborations as well.

Figure 97 Outbound Collaboration



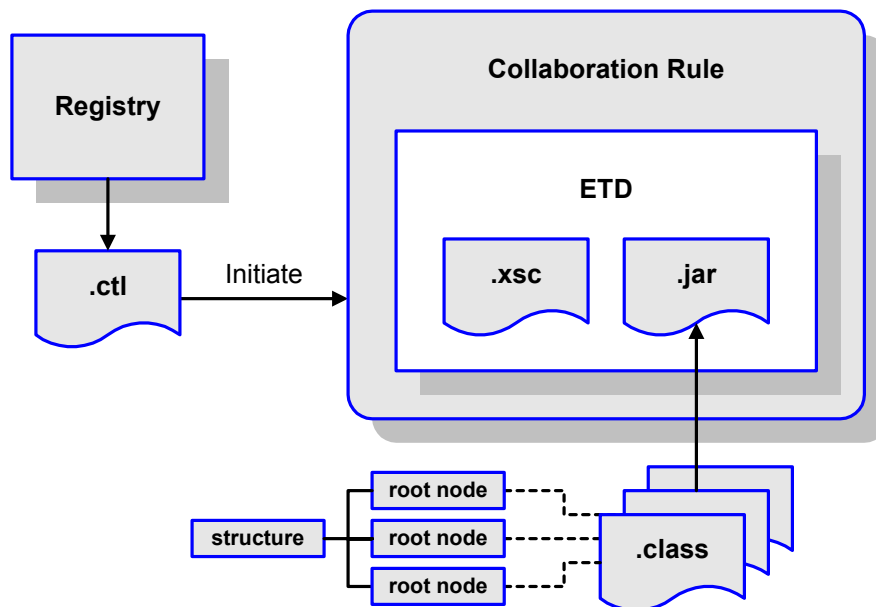
ETDs are representations of the data structure required by specific external systems, and transforming data from one format to another is a major part of the processing performed by the e\*Way. Building an ETD obviously requires knowledge of the internal data structure of the specific application. This information often can be obtained by extracting metadata from the external application, which can be automated by using an *ETD Builder*. In the case of the Siebel EAI e\*Way, these ETDs are built automatically by means of the Wizard-based XML DTD Builder (see Figure 98).

Figure 98 DTD Builder Operation



Once compiled, an ETD has two components, an `.xsc` file and a `.jar` file, both having the same file name. The `.jar` file contains `.class` files whose names correspond to the root node names in the ETD. Ultimately, the ETD is used within a Collaboration Rule to define the structure of the corresponding Event. At run time, the Collaboration Rule is initiated according to information contained in a `.ctl` file contained in the e\*Gate Registry (see Figure 99).

Figure 99 Event Type Definitions

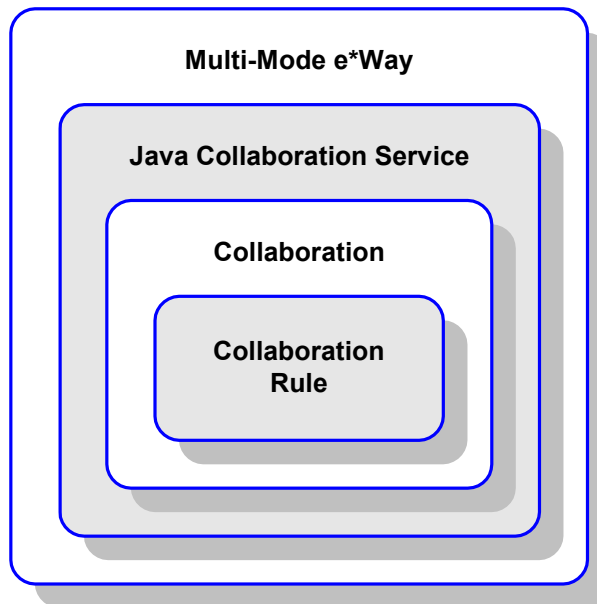


### 7.3.1 Java Collaboration Service

The Java Collaboration Service (JCS) provides an environment that allows you to use a Java class to implement the business logic that transforms Events as they move through e\*Gate. When data passes through e\*Gate using a Java Collaboration, a Java Virtual Machine (JVM) is instantiated and uses the associated Java Collaboration Rules class to accomplish the data transformation.

The relationships between the various Java e\*Way components can be depicted as a nested structure, as shown in Figure 100.

**Figure 100** Java Component Relationships



The Java Collaboration Service makes it possible to develop Collaboration Rules that execute e\*Gate business logic using Java code. Using the Java Collaboration Editor, you create Java classes that utilize the `executeBusinessRules()`, `userInitialize()`, and `userTerminate()` methods.

To use the Java Collaboration Service, you create a Collaboration Rule and select Java as the service. Using Event Type instances of previously defined Event Type Definitions (ETDs), you then use the Java Collaboration Rules Editor to add the rules and logic between the Event Type instances. Compiling the Collaboration Rule creates a Java Collaboration Rules class and all required supporting files. This Java class implements the data transformation logic.

For more information on the Java Collaboration Service, see the *e\*Gate Integrator Collaboration Services Reference Guide*.



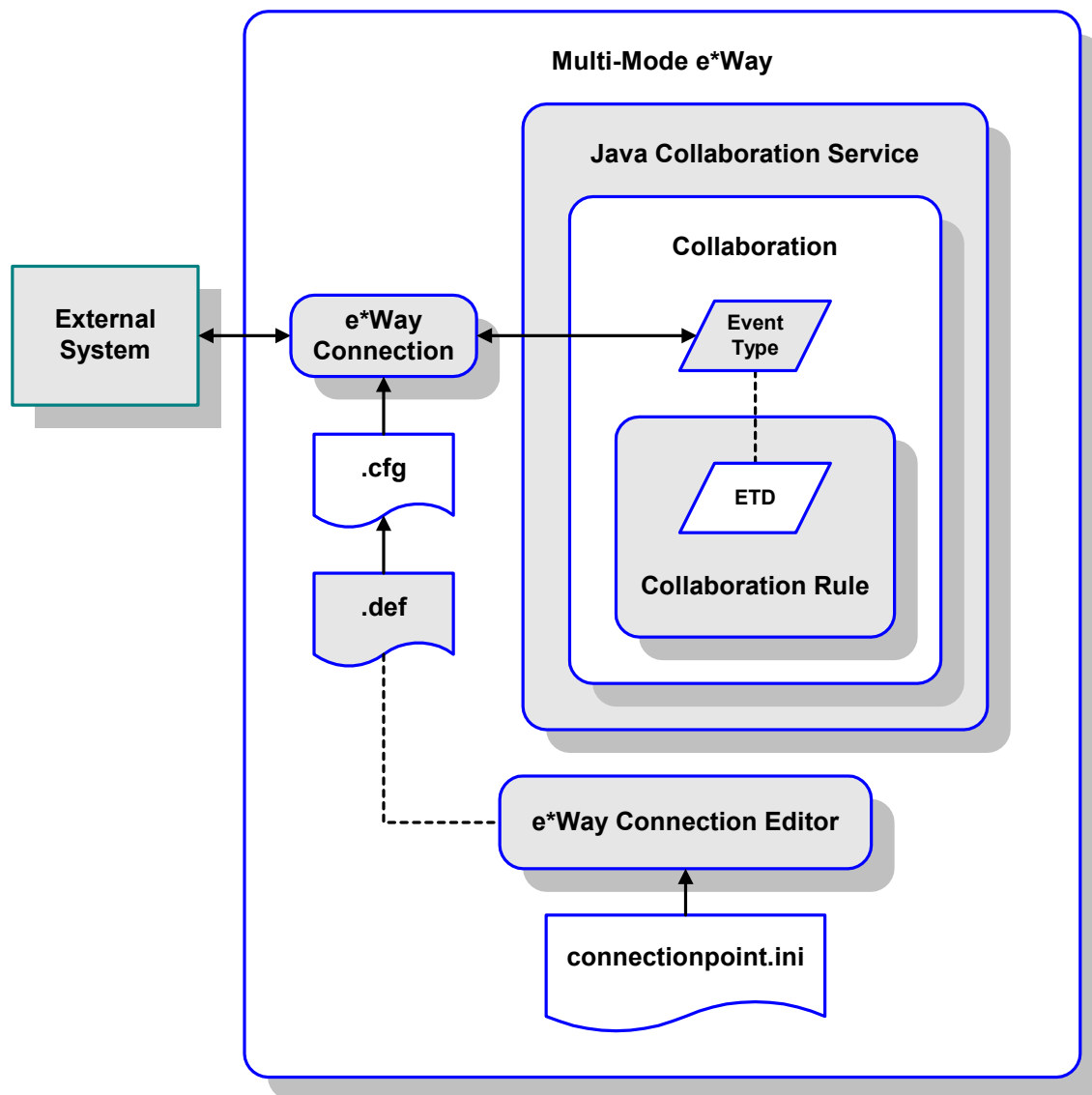
## 7.4 e\*Way Connections

The e\*Way Connections provide portals to external systems, allowing a single e\*Way to adopt several configuration profiles simultaneously. Individual e\*Way Connections can be configured using the e\*Way Connection Editor to establish a particular kind of interaction with the external system.

### 7.4.1 Establishing Connections

An e\*Way Connection to an external application is set up as depicted in Figure 101. The .def file supplied with the e\*Way is configured for the specific application using the e\*Way Connection Editor, and instantiated as a .cfg file for each e\*Way Connection.

Figure 101 e\*Way Connection Establishment



The e\*Way Connection Editor enables you to modify all parameters of a Multi-Mode e\*Way that control the way the e\*Way communicates with an external application. Because each e\*Way functions in a specific way to provide an interface to a specific external application or communications protocol, each e\*Way Connection has a unique set of configuration parameters.

For more information on the Java ETD Editor and the Java Collaboration Editor, see the *e\*Gate Integrator User's Guide*.

# Configuration Parameters

This chapter describes the configuration parameters for the Siebel EAI e\*Way Connections.

---

## 8.1 Overview

### 8.1.1 Multi-Mode e\*Way

The e\*Way's inherent configuration parameters are set using the e\*Way Configuration Editor; see [Configuring the e\\*Way](#) on page 133 for procedural information. The default configuration is provided in `sapeway.def`. The Siebel EAI e\*Way's configuration parameters are organized into the following sections:

[JVM Settings](#) on page 157

[General Settings](#) on page 162

### 8.1.2 e\*Way Connections

The Siebel EAI e\*Way's Connection parameters are set using the e\*Way Configuration Editor; see [Creating e\\*Way Connections](#) on page 138 for procedural information. The default configurations for e\*Gate-to-Siebel and Siebel-to-e\*Gate operation are provided in `siebel2000.def` and `httpclient.def`, respectively. The Siebel EAI e\*Way's configuration parameters are organized into the following sections:

[connector](#) on page 163

[HTTP](#) on page 164

[Proxies](#) on page 165

[HTTP Authentication](#) on page 167

[SSL](#) on page 168

[Siebel Configuration](#) on page 173

### 8.1.3 CGI Components

The Siebel EAI e\*Way makes use of selected components of the CGI Web Server e\*Way. Configuration of these components is covered in the following sections:

[JMS Connection Section](#) on page 175

[CGI Data Section](#) on page 177

[Log Section](#) on page 179

---

## 8.2 Multi-Mode e\*Way

### 8.2.1 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 SDK* is located on the Participating Host.

##### Required Values

A valid pathname.

*Note:* This parameter is **required**, and must **not** be left blank.

##### Additional Information

The JNI DLL name varies for different operating systems:

Operating System	Java 2 JNI DLL Name
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 file loads successfully, the Dynamic Load Library search path environment variable must be set appropriately to include all the directories under the Java SDK (or JDK) installation directory that contain shared libraries (UNIX) or .dll files (Windows).*

---

#### CLASSPATH Prepend

##### Description

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

### Required Values

An absolute path or an environmental variable.

*Note:* This parameter is optional and may be left blank.

### Additional Information

If left blank, no paths will be prefixed 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 blank, 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 environment variable.

*Note:* This parameter is optional and may be left blank.

### 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 whether to attach the environment variable to the end of CLASSPATH.

### Required Values

YES or NO. The default value is 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 Java VM will be used.

### Required Values

An integer between 0 and 2147483647.

*Note:* This parameter is optional and may be left blank.

---

## 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 Java VM will be used.

### Required Values

An integer between 0 and 2147483647.

*Note:* This parameter is optional and may be left blank.

---

## 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.

*Note:* This parameter is optional and may be left blank.

---

## 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 Java VM will be used.

### Required Values

An integer between 0 and 2147483647.

*Note:* This parameter is optional and may be left blank.

---

## Class Garbage Collection

### Description

Specifies whether the Class Garbage Collection will be done automatically by the Java VM. The selection affects performance issues.

### Required Values

YES or NO.

---

## Garbage Collection Activity Reporting

### Description

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

### 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.

### Required Values

YES or NO.

---

## Disable JIT

### Description

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

### Required Values

YES or NO.

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



---

## Remote debugging port number

### Description

Specifies whether to allow remote debugging of the JVM.

### Required Values

YES or NO.

---

## Suspend option for debugging

### Description

Specifies whether to suspend option for debugging on JVM startup.

### Required Values

YES or NO.

## 8.2.2 General Settings

---

### Rollback Wait Interval

#### Description

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

#### Required Values

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

---

## 8.3 e\*Way Connections

### 8.3.1 connector

The parameters in this section apply to the Siebel EAI connector.

---

#### type

##### Description

Specifies the connector type.

##### Required Values

`siebel2000`

---

#### class

##### Description

Specifies the implementing class for the specified type.

##### Required Values

`com.stc.jsiebel2000.Siebel2000Connector`

---

#### Property.Tag

##### Description

Specifies the data source.

##### Required Values

A valid data source package name.

## 8.3.2 HTTP

The parameters in this section furnish the required HTTP variables.

---

### Defaulter

#### Description

Specifies the destination URL. If HTTPS protocol is to be used, SSL also must be configured (see [SSL](#) on page 168).

#### Required Values

A full URL, for example:

```
http://www.seebeyond.com/servlet/sieb.sb7.gateway.GatewayServlet
```

---

### Allow Cookies

#### Description

Specifies whether or not cookies sent from servers is stored and sent on subsequent requests. If cookies are not allowed, then sessions are not supported.

#### Required Values

Yes or No; the default is Yes.

---

### Contentedly

#### Description

Specifies the parameters for the Content Type request header.

#### Required Values

A string of the form `*/*`. The default value is `application/x-www-form-urlencoded`. If you are sending other forms of data, enter the appropriate value; for example, `text/html` or `text/xml`.

---

### AcceptType

#### Description

Specifies the parameters for the Accept Type request header.

#### Required Values

A string of the form `text/*`. You must replace the asterisk with the appropriate value; for example, `text/html`, `text/xml`, `text/plain`, etc.

### 8.3.3 Proxies

The parameters in this section furnish the required HTTP Proxy variables.

---

#### UseProxy

##### Description

Specifies whether or not a proxy is used, and whether it is HTTP or HTTPS.

##### Required Values

One of the following values; the default is **NO**.

- HTTP
- HTTPS
- NO

If **HTTP** is selected, then an HTTP proxy for non-secured connection is used and the HTTP-related parameters in this section apply.

If **HTTPS** is selected, then an HTTPS proxy for secured connection is used and the HTTPS-related parameters in this section apply, as do the parameters in the SSL section (see [SSL](#) on page 168).

If **NO** is selected, then no proxy is used.

---

#### HttpProxyHost

##### Description

Specifies the proxy host for non-secured HTTP connections.

##### Required Values

A valid host name. There is no default value.

---

#### HttpProxyPort

##### Description

Specifies the proxy port for non-secured HTTP connections.

##### Required Values

A valid port name. There is no default value.

---

#### HttpsProxyHost

##### Description

Specifies the proxy host for secured HTTPS connections.

### Required Values

A valid host name. There is no default value.

---

## HttpsProxyPort

### Description

Specifies the proxy port for secured HTTPS connections.

### Required Values

A valid port name. There is no default value.

---

## UserName

### Description

Specifies the user name for Proxy Authentication.

### Required Values

A valid user name. There is no default value.

---

## Password

### Description

Specifies the user password for Proxy Authentication.

### Required Values

A valid user password. There is no default value.

## 8.3.4 HTTP Authentication

The parameters in this section furnish the required HTTP Authentication variables.

---

### UseHttpAuthentication

#### Description

Specifies whether or not standard HTTP Authentication is used (if required by the Web site).

#### Required Values

Yes or No; the default is No.

#### Additional Information

If this parameter is set to Yes, the parameters **UserName** and **PassWord** *must* be specified.

---

### UserName

#### Description

Specifies the user name for standard HTTP Authentication.

#### Required Values

A valid user name. There is no default value.

---

### PassWord

#### Description

Specifies the user password for standard HTTP Authentication.

#### Required Values

A valid user password. There is no default value.

### 8.3.5 SSL

The parameters in this section furnish the required Secure Sockets Layer (SSL) variables.

---

#### UseSSL

##### Description

Specifies whether or not to use SSL.

##### Required Values

Yes or No; the default is No.

If this parameter is set to **Yes**, the e\*Way uses the parameter values in this section to configure to use the **HTTPS** protocol.

If this parameter is set to **No**, the e\*Way ignores any certificate information.

##### Additional Information

If this parameter is set to **Yes**, the parameters **HttpsProtocolImpl** and **Provider** *must* be specified.

---

#### HttpsProtocolImpl

##### Description

Specifies the package that contains the **HTTPS** protocol implementation

##### Required Values

The default value is **com.sun.net.ssl.internal.www.protocol**.

*Note:* This parameter **must** be specified if **UseSSL** is set to **Yes**.

##### Additional Information

Specifying this parameter adds the **HTTPS URLStreamHandler** implementation by including the handler's implementation package name to the list of packages that are searched by the Java URL class. The default value specified is the package containing the Sun reference implementation of the **HTTPS URLStreamHandler**.

---

#### Provider

##### Description

Specifies the Cryptographic Service Provider.

##### Required Values

The default value is **com.sun.net.ssl.internal.ssl.Provider**.

*Note:* This parameter **must** be specified if **UseSSL** is set to **Yes**.



### Additional Information

Specifying this parameter adds a JSSE provider implementation to the list of provider implementations. The default value specified is the Sun reference implementation of the Cryptographic Service Provider, **SunJSSE**.

---

## X509CertificateImpl

### Description

Specifies the implementation class of **X509Certificate**.

### Required Values

This field should contain the concatenated values that represent the implementation class and package. For example, if the implementation class is called

```
MyX509CertificateImpl
```

and it appears in the package

```
com.radcrypto
```

then you should specify.

```
com.radcrypto.MyX509CertificateImpl.
```

*Note:* Specification of this parameter is optional.

---

## SSLSocketFactoryImpl

### Description

Specifies the implementation class of **SSL Socket Factory**.

### Required Values

This field should contain the concatenated values that represent the implementation class and package. For example, if the implementation class is called

```
MySSLSocketFactoryImpl
```

and it appears in the package

```
com.radcrypto
```

then you should specify.

```
com.radcrypto.MySSLSocketFactoryImpl.
```

*Note:* Specification of this parameter is optional.

---

## SSLServerSocketFactoryImpl

### Description

Specifies the implementation class of **SSL Server Socket Factory**.

## Required Values

This field should contain the concatenated values that represent the implementation class and package. For example, if the implementation class is called

```
MySSLServerSocketFactoryImpl
```

and it appears in the package

```
com.radcrypto
```

then you should specify.

```
com.radcrypto.MySSLServerSocketFactoryImpl.
```

*Note:* Specification of this parameter is optional.

---

## KeyStore

### Description

Specifies the default key store file for use by the Key Manager. If this parameter is not specified, then the key store managed by Key Manager is empty.

### Required Values

A valid key store file name. There is no default value.

*Note:* Specification of this parameter is optional.

---

## KeyStoreType

### Description

Specifies the default key store type. If this parameter is not specified, then the system sets the default key store type to **jks**.

### Required Values

A valid key store type. There is no default value.

*Note:* Specification of this parameter is optional.

---

## KeyStorePassword

### Description

Specifies the default key store password. If this parameter is not specified, then the default key store password is assumed to be a null string.

### Required Values

A valid key store password. There is no default value.

*Note:* Specification of this parameter is optional.

---

## TrustStore

### Description

Specifies the default trust store name.

### Required Values

A valid trust store name. There is no default value.

*Note:* Specification of this parameter is optional.

### Additional Information

If this parameter is not specified, then the system searches for a default trust store. If a trust store named `<java-home>/lib/security/jssecacerts` is found, it is used. If not, then the system searches for a trust store named `<java-home>/lib/security/cacerts`. If it is found, it is used. If neither is found, then the trust store managed by the Trust Manager is a new, empty trust store.

---

## TrustStoreType

### Description

Specifies the default trust store type.

### Required Values

A valid trust store type. There is no default value.

*Note:* Specification of this parameter is optional.

---

## TrustStorePassword

### Description

Specifies the default trust store password. If this parameter is not specified, then the default trust store password is assumed to be a null string.

### Required Values

A valid trust store name. There is no default value.

*Note:* Specification of this parameter is optional.

---

## KeyManagerAlgorithm

### Description

Specifies the default Key Manager Algorithm name.

### Required Values

The name of the key manager algorithm to use. For example, the default key manager algorithm used in the Sun reference implementation of JSSE is **SunX509**.

*Note:* Specification of this parameter is optional.

---

## TrustManagerAlgorithm

### Description

Specifies the default Trust Manager Algorithm name.

### Required Values

The name of the trust manager algorithm to use. For example, the default trust manager algorithm used in the Sun reference implementation of JSSE is **SunX509**.

*Note:* Specification of this parameter is optional.

## 8.3.6 Siebel Configuration

These parameters pertain only to e\*Gate-to-Siebel operation.

---

### SWExtSource

#### Description

Specifies the service the Siebel Web Engine calls. The value should match one of the services listed under the section [HTTP Services] in the file eai.cfg.

#### Required Values

One of the following:

- SEEBEYOND\_HTTP\_DELETE
- SEEBEYOND\_HTTP\_EXECUTE
- SEEBEYOND\_HTTP\_QUERY
- SEEBEYOND\_HTTP\_UPDATE

The default value is SEEBEYOND\_HTTP\_UPDATE.

*Note:* Specification of this parameter is required; it must **not** be left blank.

---

### SWExtCmd

#### Description

Specifies the command used by the Siebel Web Engine to execute the service specified in the previous parameter.

#### Required Values

The default value is **Execute**.

*Note:* Specification of this parameter is required; it must **not** be left blank.

---

### User Name

#### Description

Specifies the user name for authentication.

#### Required Values

A valid user name

## Encrypted password

### Description

Specifies the user password for authentication.

### Required Values

A valid user password.

---

## 8.4 CGI Web Server

### 8.4.1 JMS Connection Section

---

#### Host

##### Description

The name of the host on which the Message Service (MS) is running. The SeeBeyond JMS IQ Manager acts as the Message Service server.

##### Required Values

If Host is not specified, then **localhost** is the default value.

```
Host:localhost
```

---

#### Port

##### Description

The port at which the MS is listening for connections.

##### Required Values

If Port is not specified, then **7555** is the default value.

```
Port:24053
```

---

#### RequestReply

##### Description

Selects the JMS delivery mode as Request/Reply or Publish/Subscribe.

##### Required Values

Specify **True** for **Request/Reply** mode, **False** for **Publish** or **Send** mode.

```
RequestReply:True
```

##### Additional Information

If this parameter is set to **True**, go to [Timeout](#) to configure the reply timeout.

---

#### Timeout

##### Description

This parameter specifies the time period (in milliseconds) to wait for the reply when [RequestReply](#) is set to **True**.

```
Timeout:60000
```

---

## TopicRequest

### Description

Selects the JMS mode as Topic or Queue request.

### Required Values

Specify **True** for **Topic** requests (the default), **False** for **Queue** requests.

```
TopicRequest:True
```

### Additional Information

If this parameter is set to **True**, go to **Topic** to configure the JMS Topic.

If this parameter is set to **False**, go to **Queue** to configure the JMS Queue.

---

## Topic

### Description

The JMS Topic that the CGI will use to send a message to JMS when **TopicRequest** is set to **True**. Refer to the sample schema for more information.

### Required Values

Use the same value as the ETD type name, which the participating host receives,, for example, **etRequestReplyTopic121**. There is no default value for this parameter.

```
Topic:etwebRequestETDTopic
```

---

## Queue

### Description

The JMS Queue that CGI will use to send a message to JMS when **TopicRequest** is set to **False**. This must be specified for Queue requests.

### Required Values

The queue name, for example, **etRequestReplyQueue**. There is no default value for this parameter.

```
Queue:etRequestReplyQueue
```

---

## ClientID

### Description

The Client ID to use for the JMS connection.

### Required Values

A Client ID, for example, **SeeBeyondMSCGI**.

```
ClientID:SeeBeyondMSCGI1
```



## 8.4.2 CGI Data Section

---

### EnvInBody

#### Description

Include the CGI Environments in the message body. See [EnvEnd](#), below.

If set to **True**, then each CGI environment will be added to before the CGI message body. Each environment is a name/value pair with '=' separating the name from the value (**name=value**). Each environment is separated from the next by a **newline**.

If set to **False**, then the CGI environments will not be added to the message body.

#### Required Values

**True** or **False**; the default value is **True**.

```
EnvInBody:True
```

---

### EnvEnd

#### Description

The text denoting the End of the Environment values. If [EnvInBody](#) (above) is set to **True**, [EnvStart](#) will be used to separate the message body from the environments. **Do not** change this value.

#### Required Values

```
EnvEnd:<--End Environments-->
```

---

### EnvsAsProps

#### Description

Include the CGI Environments as JMS Properties.

If set to **True**, then each CGI environment will be added to the JMS message as a JMS string property.

If set to **False**, then the CGI environments will *not* be added as JMS properties.

#### Required Values

**True** or **False**; the default value is **True**.

```
EnvsAsProps:True
```

---

### ReadChunksize

#### Description

When CGI reads from a standard input, this parameter specifies the chunk size (in bytes) of data to be read; for example, if you specify 1024 then CGI will read 1024

bytes of data at a time. If the content length is less than the chunk size, CGI will read based on the content length.

### Required Values

An integer value; the maximum acceptable value is 2147483647 bytes. The default internal read chunk size is **409600** bytes.

```
ReadChunkSize:409600
```

---

## WriteChunksize

### Description

When CGI writes to a standard output, this parameter specifies the chunk size (in bytes) of the data to be written at one time; for example, if you specify 1024 then CGI will write 1024 bytes of data at a time.

### Required Values

An integer value; the maximum acceptable value is 2147483647 bytes. The default internal write chunk size is **409600** bytes.

```
WriteChunkSize:409600
```

## 8.4.3 Log Section

---

### LogFile

The log filename. Messages will be logged into this file. See [Trace](#) (below) to set the trace/log level.

```
LogFile:mscgi.log
```

---

### Trace

The trace level to use for trace/debug. The following are valid values:

- 0 - Information
- 1 - Warning
- 2 - Error
- 3 - Fatal

The default is 0.

# Java Methods

The Siebel EAI e\*Way contains Java methods that are used to extend the functionality of the basic e\*Way core.

---

## 9.1 Overview

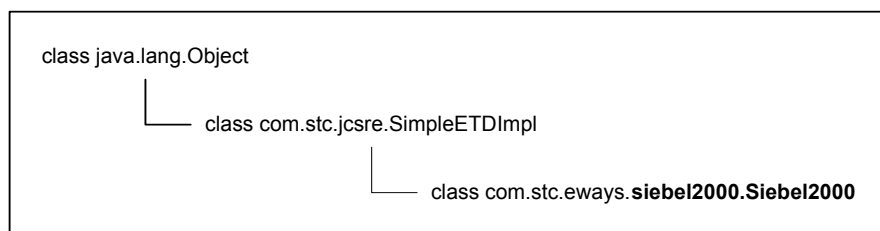
This chapter contains descriptions of methods that are exposed in the user interface. Additional methods contained in the e\*Way should only be accessed or modified by qualified SeeBeyond personnel. Unless otherwise noted, all classes and methods described in this chapter are **public**. Methods inherited from classes other than those described in this chapter are listed, but not described.

---

## 9.2 Object Classes

The Java Siebel EAI e\*Way object methods are contained in the following classes:

**Figure 102** Class Hierarchy



## 9.2.1 Siebel2000 Class

*Note:* This class also supports Siebel 7.

### Description

Extends `com.stc.jcsre.SimpleETDImpl` for Siebel EAI e\*Way.

### Definition

```
Siebel2000
```

### Constructors

```
Siebel2000()
```

### Methods

<a href="#">getDeleteSource</a>	<a href="#">getURL</a>
<a href="#">getExecuteSource</a>	<a href="#">getXmlData</a>
<a href="#">getHttpRequest</a>	<a href="#">initialize</a>
<a href="#">getQuerySource</a>	<a href="#">login</a>
<a href="#">getResponseHeaderString</a>	<a href="#">logout</a>
<a href="#">getResultData</a>	<a href="#">postSiebelForm</a>
<a href="#">getSBYN_DELETE_SOURCE</a>	<a href="#">reset</a>
<a href="#">getSBYN_UPDATE_SOURCE</a>	<a href="#">setDeleteSource</a>
<a href="#">getSBYN_EXECUTE_SOURCE</a>	<a href="#">setExecuteSource</a>
<a href="#">getSBYN_QUERY_SOURCE</a>	<a href="#">setIntegrationObjectName</a>
<a href="#">getSWEExtCmd</a>	<a href="#">setQuerySource</a>
<a href="#">getSWEExtData</a>	<a href="#">setSEWExtCmd</a>
<a href="#">getSWEExtSource</a>	<a href="#">setSWEExtData</a>
<a href="#">getTAG_SIEBEL_EXECUTE_QUERY_PREFIX</a>	<a href="#">setSWEExtSource</a>
<a href="#">getTAG_SIEBEL_EXECUTE_UPSERT_PREFIX</a>	<a href="#">setUpdateSource</a>
<a href="#">getTAG_SIEBEL_EXECUTE_DELETE_PREFIX</a>	<a href="#">setURL</a>
<a href="#">getTAG_SIEBEL_MSG_SUFFIX</a>	<a href="#">setXmlData</a>
<a href="#">getUpdateSource</a>	

### Methods Inherited from `com.stc.jcsre.SimpleETDImpl` Class

<code>available</code>	<code>receive</code>	<code>send</code>
<code>marshal</code>	<code>receive</code>	<code>subscriptions</code>
<code>next</code>	<code>retrieveKey</code>	<code>terminate</code>
<code>publications</code>	<code>retrieveMode</code>	<code>topic</code>
<code>rawInput</code>	<code>send</code>	<code>unmarshal</code>

**Methods Inherited from java.lang.Object Class**

clone	hashCode	wait
equals	notify	wait
finalize	notifyAll	wait
getClass	toString	

## 9.2.2 Methods

---

### getDeleteSource

#### Description

This method gets and returns the value of `deleteSource`. This method is not currently used.

#### Signature

```
getDeleteSource()
```

#### Parameters

None.

#### Return Type

`java.lang.String`

#### Overrides

None.

#### Throws

None.

---

### getExecuteSource

#### Description

This method gets and returns the value of `executeSource`. This method is not currently used.

#### Signature

```
getExecuteSource()
```

#### Parameters

None.

#### Return Type

`java.lang.String`

#### Overrides

None.

#### Throws

None.

---

## getHttpRequest

### Description

This method gets and returns the value of the **HttpRequest**. This is the **HttpRequest** object returned from the last post method execution.

### Signature

```
getHttpRequest()
```

### Parameters

None.

### Returns

HttpRequest Object

### Overrides

None.

### Throws

None.

---

## getQuerySource

### Description

This method gets and returns the value of **querySource**. This method is not currently used.

### Signature

```
getQuerySource()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getResponseHeaderString

### Description

This method returns the HTTP response header string from the last HTTP post to Siebel.



### Signature

```
getResponseHeaderString()
```

### Parameters

None.

### Return Type

java.lang.String

### Throws

None.

---

## getResultData

### Description

This method returns the result string from the last HTTP post method execution.

### Signature

```
getResultData()
```

### Parameters

None.

### Return Type

java.lang.String

### Throws

None.

---

## getSBYN\_DELETE\_SOURCE

### Description

This method gets and returns the constant value for the SeeBeyond DELETE source. This variable is read only.

### Signature

```
getSBYN_DELETE_SOURCE()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getSBYN\_UPDATE\_SOURCE

### Description

This method gets and returns the constant value for the SeeBeyond UPDATE source.  
This variable is read only.

### Signature

```
getSBYN_UPDATE_SOURCE()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getSBYN\_EXECUTE\_SOURCE

### Description

This method gets and returns the constant value for the SeeBeyond EXECUTE source.  
This variable is read only.

### Signature

```
getSBYN_EXECUTE_SOURCE()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getSBYN\_QUERY\_SOURCE

### Description

This method gets and returns the constant value for the SeeBeyond QUERY source. This variable is read only.

### Signature

```
getSBYN_QUERY_SOURCE()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getSWEEExtCmd

### Description

This method gets and returns the value of SWEEExtCmd.

### Signature

```
getSWEEExtCmd()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getSWEEExtData

### Description

This method gets and returns the value of SWEEExtData.

### Signature

```
getSWEEExtData()
```

### Parameters

None.

### Return Type

```
java.lang.String
```

### Overrides

None.

### Throws

None.

---

## getSWEEExtSource

### Description

This method gets and returns the value of SWEEExtSource.

### Signature

```
getSWEEExtSource()
```

### Parameters

None.

### Return Type

```
java.lang.String
```

### Overrides

None.

### Throws

None.

---

## getTAG\_SIEBEL\_EXECUTE\_QUERY\_PREFIX

### Description

This method gets and returns the value of the prefix tag for the XML QUERY Siebel message. This variable is read only.

### Signature

```
getTAG_SIEBEL_EXECUTE_QUERY_PREFIX()
```

### Parameters

None.

**Return Type**

java.lang.String

**Overrides**

None.

**Throws**

None.

---

**getTAG\_SIEBEL\_EXECUTE\_UPSERT\_PREFIX**

**Description**

This method gets and returns the value of the prefix tag for the XML UPSERT Siebel message. This variable is read only.

**Signature**

```
getTAG_SIEBEL_EXECUTE_UPSERT_PREFIX()
```

**Parameters**

None.

**Return Type**

java.lang.String

**Overrides**

None.

**Throws**

None.

---

**getTAG\_SIEBEL\_EXECUTE\_DELETE\_PREFIX**

**Description**

This method gets and returns the value of the prefix tag for the XML DELETE Siebel message. This variable is read only.

**Signature**

```
getTAG_SIEBEL_EXECUTE_DELETE_PREFIX()
```

**Parameters**

None.

**Return Type**

java.lang.String

**Overrides**

None.

### Throws

None.

---

## getTAG\_SIEBEL\_MSG\_SUFFIX

### Description

This method gets and returns the value of the closing tag for Siebel message. This variable is read only.

### Signature

```
getTAG_SIEBEL_MSG_SUFFIX()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getUpdateSource

### Description

This method gets and returns the value of `updateSource`. This method is not currently used.

### Signature

```
getUpdateSource()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getURL

### Description

This method gets and returns the URL.

### Signature

```
getURL()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## getXmlData

### Description

This method gets and returns the value of `xmlData`.

### Signature

```
getXmlData()
```

### Parameters

None.

### Return Type

java.lang.String

### Overrides

None.

### Throws

None.

---

## initialize

### Description

This method is called by the external application (via collaboration service) to initialize the Siebel2000 or Siebel 7 object. The e\*Way Connection configuration is then loaded.

### Signature

```
initialize(cntrCollab key mode)
```

### Parameters

Name	Type	Description
cntrCollab	com.stc.common.collabService.JCollabController	The Java Collaboration Controller object.
key	java.lang.String	Key to one of the JMsgObjects.
mode	integer	Mode for ETD (IN_MODE, OUT_MODE, or IN_OUT_MODE)

### Return Type

void

### Overrides

initialize in class `com.stc.jcsre.SimpleETDImpl`

### Throws

- `com.stc.common.collabService.CollabConnException`
- `com.stc.common.collabService.CollabDataException`

## login

### Description

For a session-mode connection, this method must be called to connect to Siebel. You need to call login only once.

**Note:** This method makes use of cookies; therefore, the e\*Way setting should have the cookies option enabled.

### Signature

```
login()
```

### Parameters

None.

### Return Type

boolean

### Overrides

None.

### Throws

`com.stc.common.collabService.CollabDataException`



---

## logout

### Description

For a session-mode connection, this method must be called to disconnect from Siebel. You need to call `logout` only once.

**Note:** *This method makes use of cookies; therefore, the `e*Way` setting should have the cookies option enabled.*

### Signature

```
logout()
```

### Parameters

None.

### Return Type

boolean

### Overrides

None.

### Throws

```
com.stc.common.collabService.CollabDataException
```

---

## postSiebelForm

### Description

This method performs an HTTP post to the Siebel Web Engine using the current values of `SWExtSource`, `SWExtCmd`, `SWExtData` and `xmlData`. The data is posted as a URL encoded string in form:

```
SWExtSource=...&SWExtCmd=...&username=...&Password=...&SWExtData=...
```

where ... is the value set in the corresponding attribute in this object and `SWExtData` is in the Siebel message format. The Siebel message format is composed of an operation prefix, the XML data, and the message suffix. See [Siebel XML Messages](#) on page 53.

### Signature

```
postSiebelForm()
```

### Parameters

None.

### Return Type

boolean

### Throws

```
com.stc.common.collabService.CollabDataException
```

---

## reset

### Description

Clears all headers and request data from memory.

### Signature

```
reset()
```

### Parameters

None.

### Return Type

boolean

### Overrides

reset in class `com.stc.jcsre.SimpleETDImpl`

### Throws

None.

---

## setDeleteSource

### Description

This method sets the value of `deleteSource`. This method is not currently used.

### Signature

```
setDeleteSource(delSource)
```

### Parameters

Name	Type	Description
<code>delSource</code>	<code>java.lang.String</code>	The value of <code>deleteSource</code> .

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setExecuteSource

### Description

This method sets the value of `executeSource`. This method is not currently used.

### Signature

`setExecuteSource (execSource)`

### Parameters

Name	Type	Description
execSource	java.lang.String	The value of executeSource.

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setIntegrationObjectName

### Description

This method sets the value of **IntegrationObjectName**, which specifies the integration object you want to operate on. This is used as part of the Siebel message set in **SWExtData**.

### Signature

`setIntegrationObjectName (intgName)`

### Parameters

Name	Type	Description
intgName	java.lang.String	The value of IntegrationObjectName.

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setQuerySource

### Description

This method sets the value of **querySource**. This method is not currently used.

### Signature

`setQuerySource (qrySource)`

### Parameters

Name	Type	Description
qrySource	java.lang.String	The value of qrySource.

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setSEWExtCmd

### Description

This method sets **SWExtCmd**, which is the command requested to be performed on the Siebel service.

### Signature

`setSEWExtCmd (cmd)`

### Parameters

Name	Type	Description
cmd	java.lang.String	Command, usually set to EXECUTE.

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setSWExtData

### Description

This method sets **SWExtData**, which specifies the Siebel message.

### Signature

`setSWEExtData(xmlData)`

### Parameters

Name	Type	Description
xmlData	java.lang.String	The Siebel message.

### Return Type

void

### Overrides

None.

### Throws

None.

### Additional Information

See [Siebel XML Messages](#) on page 53.

## setSWEExtSource

### Description

This method sets `SWEExtSource`, which specifies the service that is being requested. This service must match the service name in your Siebel server's `eai.cfg` file.

### Signature

`setSWEExtSource(source)`

### Parameters

Name	Type	Description
source	java.lang.String	The requested service.

### Return Type

void

### Overrides

None.

### Throws

None.

---

## setUpdateSource

### Description

This method sets the value of `updateSource`. This method is not currently used.

### Signature

```
setUpdateSource(updSource)
```

### Parameters

Name	Type	Description
<code>updSource</code>	<code>java.lang.String</code>	The value of <code>updateSource</code> .

### Return Type

`void`

### Overrides

None.

### Throws

None.

---

## setURL

### Description

This method sets the URL for the Siebel Web Engine.

### Signature

```
setURL(urlString)
```

### Parameters

Name	Type	Description
<code>urlString</code>	<code>java.lang.String</code>	The value of the URL.

### Return Type

`void`

### Overrides

None.

### Throws

`java.net.MalformedURLException`

---

## setXmlData

### Description

This method sets `xmlString`, which is used as part of `SWEEExtData`.

### Signature

```
setXmlData(xmlData)
```

### Parameters

Name	Type	Description
xmlData	java.lang.String	The Siebel message.

### Return Type

void

### Overrides

None.

### Throws

None.

# Index

## A

AcceptType parameter 164  
 AllowCookies parameter 164  
 Assigning ETDs to Event Types 72, 113  
 Asynchronous Garbage Collection parameter 160  
 Autorun 21

## B

Business Integration Manager (BIM) 13  
 Business Service 13

## C

CGI Data Section 177  
   EnvEnd 177  
   EnvInBody 177  
   EnvsAsProps 177  
   ReadChunksize 177  
   WriteChunksize 178  
 Changing the User Name 134  
 Class Garbage Collection parameter 160  
 class parameter 163  
 CLASSPATH Append From Environment Variable parameter 158  
 CLASSPATH Override parameter 158  
 CLASSPATH Prepend parameter 157  
 ClientID 176  
 Collaboration 12, 79, 121, 144, 146, 149  
   Rules 79, 121, 144, 145  
   Service 144  
 components, e\*Way 17  
 configuration  
   CGI Data Section 177–178  
   CGI Web Server 175–176  
   connector 163  
   General Settings 162  
   HTTP 164  
   HTTP Authentication 167  
   JVM Settings 157–161  
   Log Section 179  
   Proxies 165–166  
   Siebel Configuration 173–174  
   SSL 168–172  
   configuration definition files  
     httpclient.def 17  
     siebel2000.def 17  
   configuration files  
     eai.cfg 32  
     eapps.cfg 32, 35  
   configuration parameters  
     AcceptType 164  
     AllowCookies 164  
     Asynchronous Garbage Collection 160  
     class 163  
     Class Garbage Collection 160  
     CLASSPATH Append From Environment Variable 158  
     CLASSPATH Override 158  
     CLASSPATH Prepend 157  
     ContentType 164  
     DefaultURL 164  
     Disable JIT 160  
     Encrypted password 174  
     Garbage Collection Activity Reporting 160  
     HttpProxyHost 165  
     HttpProxyPort 165  
     HttpsProtocolImpl 168  
     HttpsProxyHost 165  
     HttpsProxyPort 166  
     Initial Heap Size 159  
     JNI DLL Absolute Pathname 157  
     KeyManagerAlgorithm 171  
     KeyStore 170  
     KeyStorePassword 170  
     KeyStoreType 170  
     Maximum Heap Size 159  
     Maximum Stack Size for JVM Threads 159  
     Maximum Stack Size for Native Threads 159  
     PassWord 166  
     PassWord (HTTP Authentication) 167  
     Property.Tag 163  
     Provider 168  
     Remote debugging port number 161  
     Report JVM Info and all Class Loads 160  
     Rollback Wait Interval 162  
     SSLServerSocketFactoryImpl 169  
     SSLSocketFactoryImpl 169  
     Suspend option for debugging 161  
     SWEExtCmd 173  
     SWEExtSource 173  
     TrustManagerAlgorithm 172  
     TrustStore 171  
     TrustStorePassword 171  
     TrustStoreType 171  
     type 163  
     UseHttpAuthentication 167  
     UseProxy 165



- User Name 173
- UserName (HTTP Authentication) 167
- UserName (Proxies) 166
- UseSSL 168
- X509CertificateImpl 169
- configuration procedures
  - e\*Way 133
  - e\*Way Connection 138
- ContentType parameter 164
- conventions, writing in document 10

## D

- DefaultURL parameter 164
- DELETE Workflow Template 49, 91
- Disable JIT parameter 160
- DLL files
  - sweiis.dll 34, 36

## E

- e\*Gate API Kit 17
- e\*Way
  - Components 17
  - configuration 133
  - creating 131
  - Installation 21
  - Properties 132
  - Schedules 134
  - Startup Options 134
  - troubleshooting 144
- e\*Way Connection
  - configuration procedure 138
- EAI Siebel Adapter 13, 15
- EAI XML Converter 13, 14, 15
- eai.cfg file 32
- eapps.cfg file 32, 35
- Encrypted password parameter 174
- EnvEnd 177
- EnvInBody 177
- EnvsAsProps 177
- Event 12, 146
- Event Type 72, 113
- Event Type Definition (ETD) 66, 72, 108, 113
- Event Type Definition (ETD) Editor 72, 114
- EXECUTE Workflow Template 51, 93

## G

- Garbage Collection Activity Reporting parameter 160
- General Settings configuration 162
- getDeleteSource method 183

- getExecuteSource method 183
- getHttpResult method 184
- getQuerySource method 184
- getResponseHeaderString method 184
- getResultData method 185
- getSBYN\_DELETE\_SOURCE method 185
- getSBYN\_EXECUTE\_SOURCE method 186
- getSBYN\_QUERY\_SOURCE method 187
- getSBYN\_UPDATE\_SOURCE method 186
- getSWEExtCmd method 187
- getSWEExtData method 187
- getSWEExtSource method 188
- getTAG\_SIEBEL\_EXECUTE\_DELETE\_PREFIX method 189
- getTAG\_SIEBEL\_EXECUTE\_QUERY\_PREFIX method 188
- getTAG\_SIEBEL\_EXECUTE\_UPSERT\_PREFIX method 189
- getTAG\_SIEBEL\_MSG\_SUFFIX method 190
- getUpdateSource method 190
- getURL method 191
- getXmlData method 191

## H

- Host 175
- httpClient.def file 17
- HttpProxyHost parameter 165
- HttpProxyPort parameter 165
- HttpsProtocolImpl parameter 168
- HttpsProxyHost parameter 165
- HttpsProxyPort parameter 166

## I

- IBM HTTP Server 13, 31, 41
- Initial Heap Size parameter 159
- initialize method 191
- INSERT/UPDATE Workflow Template 50, 92
- Installation 21
- Installation procedure
  - sample schema 27
  - Siebel Web Engine 32
- InstallShield 21
- Intelligent Queue (IQ) 74, 116, 144
- Internet Information Service (IIS) 13, 16, 31, 38, 43
- iPlanet Web Server 13, 31, 40

## J

- Java methods 183–199
  - getDeleteSource 183
  - getExecuteSource 183

getHttpRequest 184  
 getQuerySource 184  
 getResponseHeaderString 184  
 getResultData 185  
 getSBYN\_DELETE\_SOURCE 185  
 getSBYN\_EXECUTE\_SOURCE 186  
 getSBYN\_QUERY\_SOURCE 187  
 getSBYN\_UPDATE\_SOURCE 186  
 getSWEExtCmd 187  
 getSWEExtData 187  
 getSWEExtSource 188  
 getTAG\_SIEBEL\_EXECUTE\_DELETE\_PREFIX 189  
 getTAG\_SIEBEL\_EXECUTE\_QUERY\_PREFIX 188  
 getTAG\_SIEBEL\_EXECUTE\_UPSERT\_PREFIX 189  
 getTAG\_SIEBEL\_MSG\_SUFFIX 190  
 getUpdateSource 190  
 getURL 191  
 getXmlData 191  
 initialize 191  
 login 192  
 logoff 193  
 postSiebelForm 193  
 reset 194  
 setDeleteSource 194  
 setExecuteSource 194  
 setIntegrationObjectName 195  
 setQuerySource 195  
 setSEWExtCmd 196  
 setSWEExtData 196  
 setSWEExtSource 197  
 setUpdateSource 198  
 setURL 198  
 setXmlData 199  
 Java Object Classes  
   Siebel2000 181  
 JMS Connection Section  
   ClientID 176  
   Host 175  
   Port 175  
   Queue 176  
   RequestReply 175  
   Timeout 175  
   Topic 176  
   TopicRequest 176  
 JNI DLL Absolute Pathname parameter 157  
 JVM Settings configuration 157–161

**K**

KeyManagerAlgorithm parameter 171  
 KeyStore parameter 170

KeyStorePassword parameter 170  
 KeyStoreType parameter 170

**L**

library files  
   stdole2.tlb 20  
   stdole32.tlb 20  
 Log Section 179  
   LogFile 179  
   Trace 179  
 LogFile 179  
 logging options 136  
 login method 192  
 logoff method 193

**M**

Maximum Heap Size parameter 159  
 Maximum Stack Size for JVM Threads parameter 159  
 Maximum Stack Size for Native Threads parameter 159  
 methods, Java 183–199  
 Microsoft Internet Information Service (IIS) 13, 16, 31, 38, 43  
 monitoring thresholds 137  
 mscgi.properties  
   CGI Data Section 177  
   JMS Connection Section 175  
   Log Section 179  
 Multi-Mode e\*Way 17  
 Multi-Mode e\*Way configuration  
   General Settings 162  
   JVM Settings 157–161  
 MUX ASP 12, 16, 43

**P**

Participating Host 144  
 PassWord parameter 166  
 PassWord parameter (HTTP Authentication) 167  
 Port 175  
 POST Workflow Template 52, 94  
 postSiebelForm method 193  
 procedures  
   configuration, e\*Way 133  
   configuration, e\*Way Connection 138  
   installation, sample schema 27  
   installation, Siebel Web Engine 32  
 Properties, e\*Way 132  
 Property.Tag parameter 163  
 Provider parameter 168

## Q

QUERY Workflow Template 50, 92  
 Queue 176  
 Queues 74, 116

## R

ReadChunksize 177  
 Remote debugging port number parameter 161  
 Report JVM Info and all Class Loads parameter 160  
 RequestReply 175  
 reset method 194  
 Rollback Wait Interval parameter 162

## S

sample schema  
     description 81, 123  
     installation 27  
 Schedules 134  
 SEND Workflow Template 51, 93  
 SEND/RECEIVE Workflow Template 52, 94  
 setDeleteSource method 194  
 setExecuteSource method 194  
 setIntegrationObjectName method 195  
 setQuerySource method 195  
 setSEWExtCmd method 196  
 setSWEExtData method 196  
 setSWEExtSource method 197  
 Setting Startup Options or Schedules 134  
 setUpdateSource method 198  
 setURL method 198  
 setXmlData method 199  
 Siebel EAI Toolkit 13  
 Siebel Integration Objects 13  
 Siebel Object Manager 14  
 Siebel Web Engine (SWE) 13  
     installation 32  
 Siebel Web Server Extension (SWSE) 14, 32, 34, 35,  
 36, 77, 119  
 Siebel Web Service Extension (SWSE) 13, 32  
 Siebel2000 Class 181  
 siebel2000.def file 17  
 SSLServerSocketFactoryImpl parameter 169  
 SSLSocketFactoryImpl parameter 169  
 Startup Options 134  
 stdole2.tlb file 20  
 stdole32.tlb file 20  
 Suspend option for debugging parameter 161  
 SWEExtCmd parameter 173  
 SWEExtSource parameter 173  
 sweiis.dll file 34, 36

## T

Timeout 175  
 Topic 176  
 TopicRequest 176  
 Trace 179  
 Transport Adapters 13  
 troubleshooting the e\*Way 144  
 TrustManagerAlgorithm parameter 172  
 TrustStore parameter 171  
 TrustStorePassword parameter 171  
 TrustStoreType parameter 171  
 type parameter 163

## U

UseHttpAuthentication parameter 167  
 UseProxy parameter 165  
 User name 134  
 User Name parameter 173  
 UserName parameter (HTTP Authentication) 167  
 UserName parameter (Proxies) 166  
 UseSSL parameter 168

## W

Web servers  
     IBM HTTP Server 13, 31, 41  
     iPlanet 13, 31, 40  
     Microsoft Internet Information Service (IIS 13,  
     16, 31, 38, 43  
 Workflow Process Designer 13  
 Workflow Templates  
     DELETE 49, 91  
     EXECUTE 51, 93  
     INSERT/UPDATE 50, 92  
     POST 52, 94  
     QUERY 50, 92  
     SEND 51, 93  
     SEND/RECEIVE 52, 94  
 WriteChunksize 178

## X

X509CertificateImpl parameter 169