

SeeBeyond ICAN Suite

eGate Integrator User's Guide

Release 5.0



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, and e*Way are the registered trademarks of SeeBeyond Technology Corporation in the United States and select foreign countries; the SeeBeyond logo, e*Insight, and e*Xchange 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.

© 2003 by SeeBeyond Technology Corporation. All Rights Reserved. This work is protected as an unpublished work under the copyright laws.

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

Version 20031014183612.

Contents

List of Figures	11
-----------------	----

List of Tables	19
----------------	----

Chapter 1

Introduction	21
---------------------	-----------

Purpose and Scope	21
-------------------	----

Intended Audience	21
-------------------	----

Organization of Information	21
-----------------------------	----

Writing Conventions	22
---------------------	----

Supporting Documents	23
----------------------	----

Online Documents	23
------------------	----

The SeeBeyond Web Site	23
------------------------	----

Chapter 2

System Overview	24
------------------------	-----------

Introduction	24
--------------	----

Integration Model	25
-------------------	----

System Architecture	27
---------------------	----

Repository	28
------------	----

Environments	28
--------------	----

User Interfaces	29
-----------------	----

Enterprise Designer	29
---------------------	----

Enterprise Manager	31
--------------------	----

Chapter 3

Enterprise Designer	32
----------------------------	-----------

Overview	32
----------	----

Menu Bar	34
File Menu	34
Tools Menu	34
View Menu	34
Window Menu	35
Help Menu	35
Toolbar	36
Browser Buttons	36
Enterprise Explorer	37
Project Explorer	37
Project Explorer Icons	38
Context Menus	39
Repository Menu	39
Project Menu	40
Connectivity Map Menu	41
Environment Explorer	42
Environment Explorer Icons	43
Context Menus	44
Repository Menu	44
Environment Menu	45
Logical Host Menu	46
Enterprise Designer Editors	48
Connectivity Map Editor	49
OTD Editor	50
Java Collaboration Editor	51
XSLT Collaboration Editor	52
Environment Editor	53
Deployment Editor	54

Chapter 4

Creating Projects	55
Overview	55
Project Components	56
Starting Enterprise Designer	57
Using the Project Explorer	59
Using the Connectivity Map Editor	60
Connectivity Map Editor Toolbar	61
Creating and Configuring Logical Components	62
Services	62
External Applications	62
Schedulers	63
Component Connections	64
Configuring a Connection	65
Modifying a Configuration Property	67

Defining Constants and Variables	68
Web Services	71
Example Web Services Project	73
Creating the OTDs	73
Using the Connectivity Map	75

Chapter 5

Object Type Definitions	77
Overview	77
The <i>Bean-like</i> Interface	77
OTD Types	78
Externally-Defined OTDs	78
User-Defined OTDs	78
Building OTDs	78
Using the OTD Wizard	79
Navigation Buttons	80
DTD-Based OTDs	81
Using the DTD Wizard	81
User-Defined OTDs	85
Using the User-Defined OTD Wizard	85
Editing the OTD Properties	87
Node Properties	87
Element Properties	88
Field Properties	89
Specifying the Node Type	90
Specifying Delimiters	91
Delimiter Properties	92
Precedence	95
Creating a Delimiter List	96
WSDL-Based OTDs	99
Using the WSDL Wizard	99
WSDL OTD Operation	103
WSDL Operation Elements	103
WSDL OTD Structure	103
XSD-Based OTDs	104
Using the XSD Wizard	104
Using the OTD Editor	108
Node Management	110
Using the OTD Tester	111
The <i>Generic</i> Interface	114

Chapter 6

Java Collaboration Definitions	115
Overview	115
Using the Java Collaboration Wizard	116
Navigation Buttons	116
Creating a Java Collaboration Definition	117
New Web Service	118
Existing Web Service	122
Using the Java Collaboration Editor	124
Java Toolbar Icons	125
Business Rules Editor	127
Commands	127
Business Rules Toolbar Icons	128
Business Rules Tree	129
Business Rules Designer	130
Business Rules Designer Toolbar Icons	131
Dialog Boxes	132
Java Method Palette	134
Java Method Boxes	134
Java Source Editor	135
Java Methods	136
Boolean Methods	136
Comparison Methods	137
Math Methods	139
Object Methods	141
Object Method Dialog Boxes	142
String Methods	144
Array Operation Methods	147
Array Operation Method Dialog Boxes	148
Operator Methods	149
Java Encapsulation	151
Creating a Modified Java Collaboration Definition	151
Merging Two Versions of a Java Collaboration Definition	152
Setting Up a Java Collaboration Definition Variable	153
Creating a Variable	153
Invoking Variable Constructor	156
Displaying Method Classes	158
Using <i>Try-Catch</i>	159
Adding and Using Third-Party Java Classes	162
Adding Class Instances to a Collaboration	166
Validating Java Collaborations	168
Debugging Java Collaborations	168
Creating Log Entries	169
Creating Alerts	172

Chapter 7

XSLT Collaboration Definitions	176
Overview	176
Using the XSLT Collaboration Wizard	177
The Wizard Interface	177
Creating an XSLT Collaboration Definition	178
New Web Service	179
Existing Web Service	181
Using the XSLT Collaboration Editor	182
XSLT Toolbar Icons	183
XSLT Method Palette	184
XSLT Method Boxes	184
XSLT Methods	185
Operator Methods	185
String Methods	188
Number Methods	191
Boolean Methods	193
Nodes Methods	195
XSLT Encapsulation	198
Creating a Modified XSLT Collaboration Definition	198
Merging Two Versions of a XSLT Collaboration Definition	199

Chapter 8

Environments and Project Deployment	200
Overview	200
Environmental Constants	202
Deployment Profiles	203
Using the Deployment Editor	204
Creating a Deployment Profile	205
Activating and Deactivating Deployments	208
Mapping Variables	210
Deploying Projects to Third-Party Servers	211
BEA WebLogic	211
IBM WebSphere	214

Chapter 9

Logical Hosts	217
Overview	217

Integration Servers	217
Message Servers	217
Management Agent	217
Bootstrap	218
Integration Servers	218
Configuring a Logical Host	219
Logical Host Startup Parameters	219
Modifying the Logical Host Startup Configuration File	220
Configuring the Base Port Number	223
Starting the Logical Host	224
Starting the Logical Host as a Windows Service	225
Starting the Logical Host Manually on a Windows System	226
Starting the Logical Host on a UNIX System	227
Starting the Logical Host on a Red Hat Linux System	227
Starting the Logical Host on an HP NonStop Server	228

Chapter 10

Repository Tools	229
Overview	229
Impact Analysis	230
Command Buttons	231
Using the Impact Analyzer	231
Printing Impact Analysis Results	232
Version Control	233
Viewing History of a Project Component	233
Checking In a Project or Environment Component	234
Checking Out a Project or Environment Component	235
Repository Backup and Restoration	236
Backing Up a Repository	236
Restoring a Repository	236
Project Export and Import	238
Exporting a Project	238
Enterprise Designer Method	238
Command-Line Method	239
Importing a Project	240
Enterprise Designer Method	240
Command-Line Method	241

Chapter 11

Security	242
Overview	242
Configuration User Management	243
Roles	243

User Management Interface	244
Environment User Management	246
Message Server Security	247
Message Server Settings	247
JMS Client Security	247
ACL Management	248
SSL/HTTPS	250
Overview	250
Certificates and Keys	250
The Keytool Utility	250
Installation and Configuration	251
LDAP	254
Overview	254
Referencing and Accessing Information	255
Implementing LDAP	256
Configuring the ICAN Suite to Use LDAP	256
Configuration User Management	256
Runtime User Management	257

Chapter 12

Enterprise Manager	258
Overview	258
Starting Enterprise Manager	259
Home	261
Monitor Server	261
Administration	265
Downloads	268
Documentation	269

Chapter 13

Managing Projects	270
Overview	270
Accessing the Monitor Server	271
Monitoring Collaborations	272
Viewing Alerts	278
Viewing Logs	280
Integration Server Level	280
Collaboration Level	281
Setting Log Levels	282

Indoubt Transaction Editing	283
<hr/>	
Chapter 14	
Debugging and Logging	285
Overview	285
Debugging	285
Logging	285
The Java Debugger	286
Enabling the Debugger	286
Invoking the Java Debugger	288
Setting Breakpoints	290
Inspecting and Editing the Source Code	292
Stepping Into, Over, or Out	293
Inspecting Java Threads and Methods	294
Inspecting a Local Variable or Method	295
Saving and Resuming Debug Sessions	298
eGate Logs	299
Log File System	299
Log File Management	299
Log File Properties	300
Logical Host	300
Integration Server	300
Log Files and Locations	301
Repository	301
Logical Host	302
Management Agent	302
Integration Server	302
Message Server	302
Enterprise Designer	302
Glossary	303
e*Gate 4.x Terms in eGate 5.0	306
Index	308

List of Figures

Figure 1	SeeBeyond ICAN Suite	24
Figure 2	eGate Integrator	25
Figure 3	eGate Integrator Implementation Model	26
Figure 4	Typical eGate Integrator System	27
Figure 5	Enterprise Designer	29
Figure 6	Connectivity Map Editor	30
Figure 7	SeeBeyond Enterprise Manager Login	31
Figure 8	SeeBeyond Enterprise Designer	32
Figure 9	Enterprise Explorer: Project Explorer View	37
Figure 10	Repository Menu	39
Figure 11	Project Menu	40
Figure 12	Connectivity Map Menu	41
Figure 13	Enterprise Explorer: Environment Explorer View	42
Figure 14	Repository Menu	44
Figure 15	Environment Menu	45
Figure 16	Logical Host Menu	46
Figure 17	Connectivity Map Editor	49
Figure 18	OTD Editor	50
Figure 19	Java Collaboration Editor	51
Figure 20	XSLT Collaboration Editor	52
Figure 21	Environment Editor	53
Figure 22	Deployment Editor	54
Figure 23	Login Dialog Box	57
Figure 24	SeeBeyond Enterprise Designer	58
Figure 25	Project Explorer	59
Figure 26	Connectivity Map Window	60
Figure 27	Linking JMS Topics	60
Figure 28	Service Component	62
Figure 29	External Application Drop-Down Menu	62
Figure 30	Selected External Applications in Toolbar	63
Figure 31	Scheduler Component	63
Figure 32	Connection Icons in a Connectivity Map	64

List of Figures

Figure 33	Default Configuration Dialog Box	65
Figure 34	Editing a Default Configuration Field	67
Figure 35	Project Variable Creation	68
Figure 36	Project Constant Creation	69
Figure 37	Variables and Constants Object Group	69
Figure 38	Connector Properties	70
Figure 39	SeeBeyond UDDI Repository	71
Figure 40	Example Web Service WSDL File	71
Figure 41	Microsoft Visual Studio Example	72
Figure 42	Web Service Example (OTD 1)	73
Figure 43	Web Service Example (OTD 2)	73
Figure 44	Web Service Example (OTD 3)	74
Figure 45	Web Service Example (Connectivity Map)	75
Figure 46	Web Service Example (WSDL File)	76
Figure 47	Web Service Example (UDDI Repository)	76
Figure 48	OTD Wizard Selection Dialog	79
Figure 49	OTD Wizard Selection: DTD Wizard	81
Figure 50	Select DTD File(s) Dialog Box	82
Figure 51	Select Document Elements Dialog Box	83
Figure 52	Select OTD Options Dialog Box	84
Figure 53	OTD Wizard Selection: User-Defined OTD	85
Figure 54	Enter OTD Name	86
Figure 55	User-Defined OTD Node Properties	87
Figure 56	User-Defined OTD Element Properties	88
Figure 57	User-Defined OTD Field Properties	89
Figure 58	Node Type Property Options	90
Figure 59	Delimiter List Editor	92
Figure 60	Terminal Type Property Example	93
Figure 61	Optional Property (Example 1)	94
Figure 62	Optional Property (Example 2)	94
Figure 63	Example User-Defined OTD	96
Figure 64	Activate Field	96
Figure 65	Delimiter List Editor (1)	97
Figure 66	Context Menu	97
Figure 67	Delimiter List Editor (2)	97
Figure 68	Delimiter Specified	98
Figure 69	OTD Wizard Selection: WSDL Wizard	99
Figure 70	WSDL Wizard: Select WSDL Location	100

Figure 71	WSDL Wizard: Select WSDL File	101
Figure 72	WSDL Wizard: Select OTD Options	102
Figure 73	OTD Wizard Selection: XSD Wizard	104
Figure 74	XSD Wizard: Select XSD File(s)	105
Figure 75	Select Document Elements Dialog Box	106
Figure 76	Select OTD Options Dialog Box	107
Figure 77	OTD Editor	108
Figure 78	OTD Tester	111
Figure 79	Test Panel Data Display	111
Figure 80	Select Data File	112
Figure 81	Object Elements and Values	112
Figure 82	Data Display: Refresh Icon	113
Figure 83	Status Data Display	113
Figure 84	Initial Wizard Dialog	117
Figure 85	New Web Service: Operation Name	118
Figure 86	New Web Service: Input Message	119
Figure 87	New Web Service: Output Message	120
Figure 88	New Web Service: Auxiliary OTD	121
Figure 89	Existing Web Service: Select Operation	122
Figure 90	Existing Web Service: Select OTD	123
Figure 91	Java Collaboration Editor	124
Figure 92	Business Rules Editor	127
Figure 93	Business Rules Designer: Addition Method	130
Figure 94	Import Static Field Dialog Box	132
Figure 95	New Array Dialog Box	133
Figure 96	Java Method Palette Dialog Box	134
Figure 97	Expanded Method Box	134
Figure 98	Collapsed Method Box	134
Figure 99	Java Source Editor	135
Figure 100	Method Palette: Boolean Methods	136
Figure 101	Method Palette: Comparison Methods	137
Figure 102	Method Palette: Math Methods	139
Figure 103	Method Palette: Object Methods	141
Figure 104	CAST Dialog Box	142
Figure 105	InstanceOf Dialog Box (1)	142
Figure 106	Find Class Dialog Box	143
Figure 107	InstanceOf Dialog Box (2)	143
Figure 108	Method Palette: String Methods	144

Figure 109	Method Palette: Array Operation Methods	147
Figure 110	Array Access Dialog Box	148
Figure 111	Array Assign Dialog Box	148
Figure 112	Method Palette: Operator Methods	149
Figure 113	Java Collaboration Definition Context Menu	151
Figure 114	Version Control - Create Diff Dialog Box	151
Figure 115	Version Control - Merge Changes Dialog Box	152
Figure 116	Create a Variable Dialog Box	153
Figure 117	Find Class Dialog Box	154
Figure 118	Java Collaboration Definition with Variable	155
Figure 119	Call New Constructor Dialog Box	156
Figure 120	Java Collaboration Definition with Constructor	157
Figure 121	Variable Context Menu	158
Figure 122	Method Selection List Box	158
Figure 123	Try Icon	159
Figure 124	Catch Context Menu	159
Figure 125	Create a New Exception Variable Dialog Box	160
Figure 126	Find Class Dialog Box	160
Figure 127	Catch SQLException Rule	161
Figure 128	Exception Message	161
Figure 129	Project Context Menu: New File ...	162
Figure 130	File Selection Dialog Box	162
Figure 131	Import JAR File Icon	163
Figure 132	Add/Remove Jar File Dialog Box (1)	163
Figure 133	Add/Remove Jar File Dialog Box (2)	164
Figure 134	Call Java Method Dialog Box	164
Figure 135	Using the Third-Party Java Method	165
Figure 136	Call New Constructor Dialog Box	166
Figure 137	Constructor Example 1	167
Figure 138	Constructor Example 2	167
Figure 139	Validating a Collaboration Definition	168
Figure 140	Logger Field	169
Figure 141	Logging Menu	169
Figure 142	Logging Level/Method Selection Window	170
Figure 143	Create Literal Dialog Box	170
Figure 144	Pass Log Message to Object Argument	171
Figure 145	Alerter Field	172
Figure 146	Empty File Test	173

Figure 147	Alert Menu	174
Figure 148	Alert Severity Selection Window	174
Figure 149	Create Literal Dialog Box	174
Figure 150	Pass Alert Message to Object Argument	175
Figure 151	XSLT Collaboration Wizard Dialog Box	178
Figure 152	New Web Service: Operation Name	179
Figure 153	New Web Service: Input Message	180
Figure 154	New Web Service: Output Message	180
Figure 155	Existing Web Service: Select Operation	181
Figure 156	XSLT Collaboration Editor	182
Figure 157	XSLT Method Palette Dialog Box	184
Figure 158	Expanded Method Box	184
Figure 159	Collapsed Method Box	184
Figure 160	Method Palette: Operator Methods	185
Figure 161	Method Palette: String Methods	188
Figure 162	Method Palette: Number Methods	191
Figure 163	Method Palette: Boolean Methods	193
Figure 164	Method Palette: Nodes Methods	195
Figure 165	XSLT Collaboration Definition Context Menu	198
Figure 166	Version Control - Create Diff Dialog Box	198
Figure 167	Version Control - Merge Changes Dialog Box	199
Figure 168	Environment Editor	200
Figure 169	Repository Context Menu	201
Figure 170	Environment Context Menu	201
Figure 171	Logical Host Context Menu	201
Figure 172	Environmental Constants Editor	202
Figure 173	eGate Integrator Implementation Model	203
Figure 174	Deployment Editor Window	204
Figure 175	Example Deployment Profile (1)	205
Figure 176	Example Deployment Profile (2)	206
Figure 177	Example Deployment Profile (3)	207
Figure 178	Activate Dialog Box	208
Figure 179	Logical Host Context Menu - Apply	208
Figure 180	Activate Dialog Box	209
Figure 181	Deployment Profile Mappings	210
Figure 182	Project Variable Value Entry	210
Figure 183	WebLogic Deployment (1)	211
Figure 184	WebLogic Deployment (2)	212

Figure 185	WebLogic Deployment Verification	213
Figure 186	WebSphere Deployment (1)	214
Figure 187	WebSphere Deployment (2)	215
Figure 188	WebSphere Deployment Verification	216
Figure 189	Integration Server (J2EE Compatible)	218
Figure 190	Example logical-host.properties File	220
Figure 191	Logical Host Properties Dialog Box	223
Figure 192	Logical Hosts	224
Figure 193	Install as Service Script	225
Figure 194	Windows Services List	225
Figure 195	Uninstall as Service Script	226
Figure 196	Impact Analyzer Dialog Box	230
Figure 197	Print Dialog Box	232
Figure 198	Printed Component Data	232
Figure 199	Version Control - History Dialog Box	233
Figure 200	Version Control Dialog Box	234
Figure 201	Version Control - Check Out Dialog Box	235
Figure 202	Enter File Name Dialog Box	238
Figure 203	Message Dialog Box	239
Figure 204	Select File Dialog Box	240
Figure 205	File Destination Dialog Box	241
Figure 206	Message Dialog Box	241
Figure 207	User Management Dialog Box	244
Figure 208	User Management - Add User	245
Figure 209	Environment Context Menu	246
Figure 210	JMS IQ Manager Properties	247
Figure 211	JMS Client Security Properties	247
Figure 212	ACL Management Dialog Box	248
Figure 213	ACL Add Users Dialog Box	249
Figure 214	ACL Management Dialog Box - Read and Write Access	249
Figure 215	LDAP Directory Tree (Traditional Naming)	254
Figure 216	LDAP Directory Tree (Internet Naming)	255
Figure 217	Security Configuration Template Properties Dialog Box	257
Figure 218	Enterprise Manager Login	259
Figure 219	Enterprise Manager GUI	259
Figure 220	Monitor Launch Window	261
Figure 221	Monitor Page - Projects	262
Figure 222	Monitor Page - Environment - Alerts	263

List of Figures

Figure 223	Monitor Page - Environment - List	263
Figure 224	Monitor Page - Environment - Log Props	264
Figure 225	Upload Product License	265
Figure 226	Upload eGate SAR File	265
Figure 227	Upload Product Manifest	266
Figure 228	Choose File Dialog Box	266
Figure 229	Products Available to Upload Page	267
Figure 230	Products Available to Download Page	268
Figure 231	Documentation Tab	269
Figure 232	Monitor Launch Window	271
Figure 233	Example Project Connectivity Map	272
Figure 234	Monitor - Details - List Collaborations	272
Figure 235	Starting and Stopping Collaborations - Details Panel	273
Figure 236	Starting and Stopping Collaborations - Explorer	273
Figure 237	Pending and Processed Messages	274
Figure 238	Monitor - Project Explorer	275
Figure 239	Active/Non-active Collaborations	275
Figure 240	Pending and Processed Messages	276
Figure 241	Logging Details	276
Figure 242	Connectivity Map Details: Components	277
Figure 243	Connectivity Map View - Zoom In/Out	277
Figure 244	Collaboration Alerts	278
Figure 245	Alert Status	278
Figure 246	Alert Details	279
Figure 247	Integration Server Log Messages	280
Figure 248	Integration Server Log Messages - Filtered	280
Figure 249	Collaboration Log File	281
Figure 250	Search on Keyword	281
Figure 251	Logging Properties Page	282
Figure 252	Resetting Log Levels	282
Figure 253	Message Server Details - Controls Tab	283
Figure 254	Indoubt Transaction List	284
Figure 255	Indoubt Transaction List - Transaction Selected	284
Figure 256	Integration Server Properties Dialog Box	286
Figure 257	Logical Host Context Menu	287
Figure 258	Integration Server Context Menu	288
Figure 259	Java Debugger	288
Figure 260	File Menu	289

List of Figures

Figure 261	Attach to JVM Dialog Box	289
Figure 262	Collaboration Source Code Display	289
Figure 263	Breakpoint Example	290
Figure 264	Debug Menu	290
Figure 265	Stop in Method Dialog Box	291
Figure 266	Choose Exception Dialog Box	291
Figure 267	Break on Exception Dialog Box	291
Figure 268	Breakpoint Indicator	292
Figure 269	Stepping Into, Over, and Out Commands	293
Figure 270	Java Thread and Method Display	294
Figure 271	Local Variables Tab	295
Figure 272	Evaluate Local Variable	296
Figure 273	Evaluate Method	297
Figure 274	Save Debugger Session Dialog Box	298
Figure 275	Resume Debugger Session Dialog Box	298
Figure 276	Recirculating Log File Stack	299

List of Tables

Table 1	Writing Conventions	22
Table 2	File Menu Options	34
Table 3	Tools Menu Options	34
Table 4	View Menu Options	34
Table 5	Window Menu Options	35
Table 6	Help Menu Options	35
Table 7	Enterprise Designer Toolbar Icons	36
Table 8	Browser Buttons	36
Table 9	Project Icons	38
Table 10	Repository Menu Options	39
Table 11	Project Menu Options	40
Table 12	Connectivity Map Menu Options	41
Table 13	Environment Icons	43
Table 14	Repository Menu Options	44
Table 15	Environment Menu Options	45
Table 16	Logical Host Menu Options	46
Table 17	Connectivity Map Toolbar Icons	61
Table 18	Default Configuration Toolbar Buttons	66
Table 19	OTD Wizard Navigation Buttons	80
Table 20	Node Properties	87
Table 21	Element Properties	88
Table 22	Field Properties	89
Table 23	Node Type Property Options	90
Table 24	Escaped Delimiters	91
Table 25	Delimiter Properties and Value Options	92
Table 26	OTD Editor Toolbar Icons	109
Table 27	Wizard Navigation Buttons	116
Table 28	Java Toolbar Icons	125
Table 29	Business Rules Toolbar Buttons	128
Table 30	Rules for Placement of Subnodes	129
Table 31	Business Rules Designer Toolbar Icons	131
Table 32	Java Boolean Methods	136

List of Tables

Table 33	Java Comparison Methods	137
Table 34	Java Math Methods	139
Table 35	Java Object Methods	141
Table 36	Java String Methods	144
Table 37	Java Array Operation Methods	147
Table 38	Java Operator Methods	149
Table 39	Wizard Navigation Buttons	177
Table 40	XSLT Toolbar Icons	183
Table 41	XSLT Operator Methods	185
Table 42	XSLT String Methods	188
Table 43	XSLT Number Methods	191
Table 44	XSLT Boolean Methods	193
Table 45	XSLT Nodes Methods	195
Table 46	Environmental Constants Editor Icons	203
Table 47	Deployment Toolbar Buttons	204
Table 48	Command Arguments for bootstrap	219
Table 49	Logical Host Properties file	222
Table 50	Navigation Buttons	231
Table 51	Realm Element Attributes	256
Table 52	Log Properties for Logical Hosts	300
Table 53	Log Properties for Integration Servers	300
Table 54	eGate 5.0 Terms	306

Introduction

This chapter introduces you to this *eGate Integrator User's Guide*, its general purpose and scope, and its organization. It also provides sources of related documentation and information.

1.1 Purpose and Scope

The *eGate Integrator User's Guide* provides general information about the features and operation of SeeBeyond® eGate Integrator 5.0.

Note: *Any operation explanations provided in this document are generic, for reference purposes only, and do not necessarily address the specifics of setting up individual eGate Projects.*

1.2 Intended Audience

This guide is intended for experienced PC users who have the responsibility of helping to set up and maintain a fully functioning ICAN Suite system. This person must also understand any operating systems on which eGate will be installed (Windows or UNIX) and must be thoroughly familiar with Windows-style GUI operations.

1.3 Organization of Information

This document provides information about eGate Integrator 5.0 and includes the following chapters:

- **Chapter 1 “Introduction”** describes the purpose of *eGate Integrator User's Guide* includes writing conventions and a list of related documents.
- **Chapter 2 “System Overview”** provides an overview of the general structure, architecture, and operation of the eGate system.
- **Chapter 3 “Enterprise Designer”** provides a detailed overview of the Enterprise Designer, including its structure and operation.

- **Chapter 4 “Creating Projects”** explains how to create a Connectivity Map and use the Configuration Editor to modify eWay and JMS connections between Connectivity Map components.
- **Chapter 5 “Object Type Definitions”** describes how to create Object Type Definitions (OTDs).
- **Chapter 6 “Java Collaboration Definitions”** overviews the Java Collaboration Definition creation process.
- **Chapter 7 “XSLT Collaboration Definitions”** looks at XSLT Collaboration Definitions.
- **Chapter 8 “Environments and Project Deployment”** explains how to create and activate deployment profiles.
- **Chapter 9 “Logical Hosts”** explains how to configure and start Logical Hosts.
- **Chapter 10 “Repository Tools”** looks at the Project analysis tools available in the Enterprise Designer.
- **Chapter 11 “Security”** discussed the various security features in the ICAN Suite.
- **Chapter 12 “Enterprise Manager”** provides a detailed overview of the Enterprise Manager, including its structure and operation.
- **Chapter 13 “Managing Projects”** describes the eGate management and monitoring tools available with the Enterprise Manager.
- **Chapter 14 “Debugging and Logging”** provides troubleshooting tips and describes the eGate logs

In addition, refer to the [Glossary](#) on page 303 for a list of eGate-related terminology.

1.4 Writing Conventions

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

Table 1 Writing Conventions

Text	Convention	Example
Buttons, File, Icon, and Menu Names	Bold text	Click OK to save and close. From the File menu, select Exit . Select the logicalhost.exe file.
Command Line Code	Courier font	<code>java -jar EnterpriseDesigner.jar</code>
Hypertext Links	Blue text	For more information, see “Writing Conventions” on page 22 .
Notes	Bold Italic text	<i>Note: If a toolbar button is dimmed, you cannot use it with the selected component.</i>

1.5 Supporting Documents

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

- *eGate Integrator Installation Guide*
- *eGate Integrator JMS Reference Guide*
- *eGate Integrator Release Notes*
- *eGate Integrator Tutorial*
- *SeeBeyond ICAN Suite Deployment Guide*
- *SeeBeyond ICAN Suite Primer*

For information on a specific add-on product (for example, an eWay Intelligent Adapter), see the User's Guide for that product. A complete list of eGate-related documentation is included in the *SeeBeyond ICAN Suite Primer*.

1.6 Online Documents

The documentation for the SeeBeyond ICAN Suite is distributed as a collection of online documents. These documents are viewable with the Acrobat Reader application from Adobe Systems. Acrobat Reader can be downloaded from:

<http://www.adobe.com>

1.7 The SeeBeyond Web Site

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

<http://www.seebeyond.com>

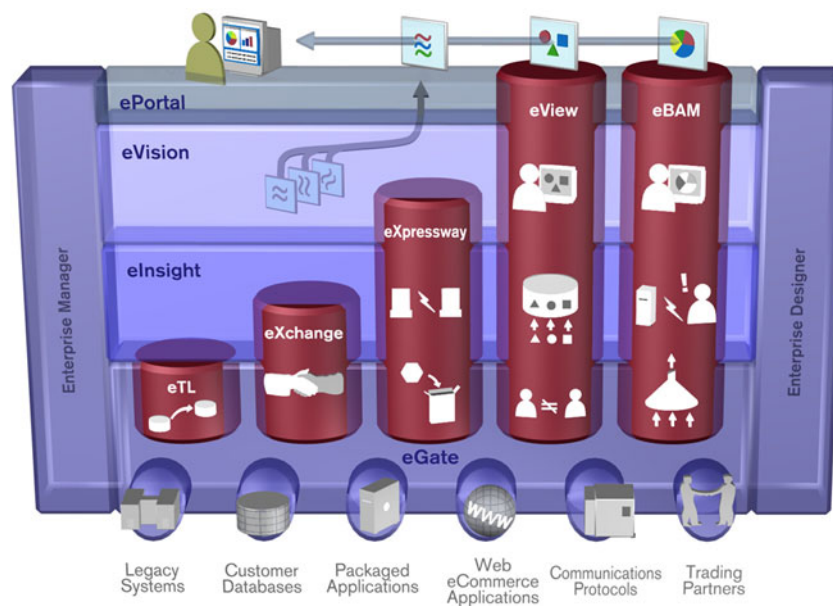
System Overview

This chapter provides an overview of the conceptual operation and general architecture of the eGate Integrator system.

2.1 Introduction

SeeBeyond's Integrated Composite Application Network (ICAN) Suite provides businesses with a comprehensive, unified eBusiness infrastructure to connect, integrate, and manage enterprise-wide software applications running on various computer systems. The full ICAN Suite is depicted in Figure 1.

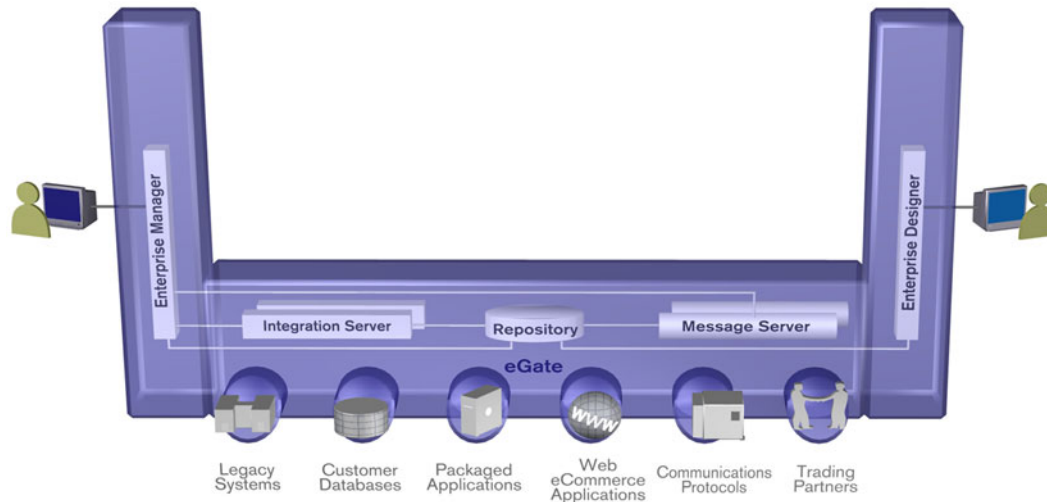
Figure 1 SeeBeyond ICAN Suite



SeeBeyond's eGate Integrator provides the "backbone" for the ICAN suite, integrating the various components of the Suite and all other connected components of the business enterprise. As shown in Figure 2, eGate Integrator includes the Enterprise Manager and Enterprise Designer, which provide graphical user interfaces for managing, configuring, and controlling the entire ICAN Suite and the business

processes running therein. See [Enterprise Manager](#) on page 31 and [Enterprise Designer](#) on page 29.

Figure 2 eGate Integrator



Other major constituents of eGate Integrator shown in Figure 2 are the Integration Server, the JMS IQ Manager, and the Repository, all of which will be described briefly later in this chapter. The flexibility of the eGate system allows the option of deploying it across a distributed network of hardware platforms, if desired, and running it on any combination of SeeBeyond, BEA WebLogic, and IBM WebSphere servers.

eGate Integrator can communicate with and link multiple applications and databases across a variety of different operating systems. eGate performs with a wide variety of hardware, message standards, operating systems, databases, and communication protocols in both real-time and batch (scheduled) integration modes.

2.2 Integration Model

SeeBeyond addresses application integration by means of an eGate Project, which contains the business logic required to solve the specific problem. The Project contains the various logical components and supporting information required to perform the routing, processing, and caching of messages containing the relevant data from one application to another. All Project information is stored in the Repository.

Projects are created using tools contained within the Enterprise Designer and, once deployed, can be run and monitored using Enterprise Manager. Projects can also be set up to be run from the business process level using the SeeBeyond eInsight Business Process Manager, if that product is also installed.

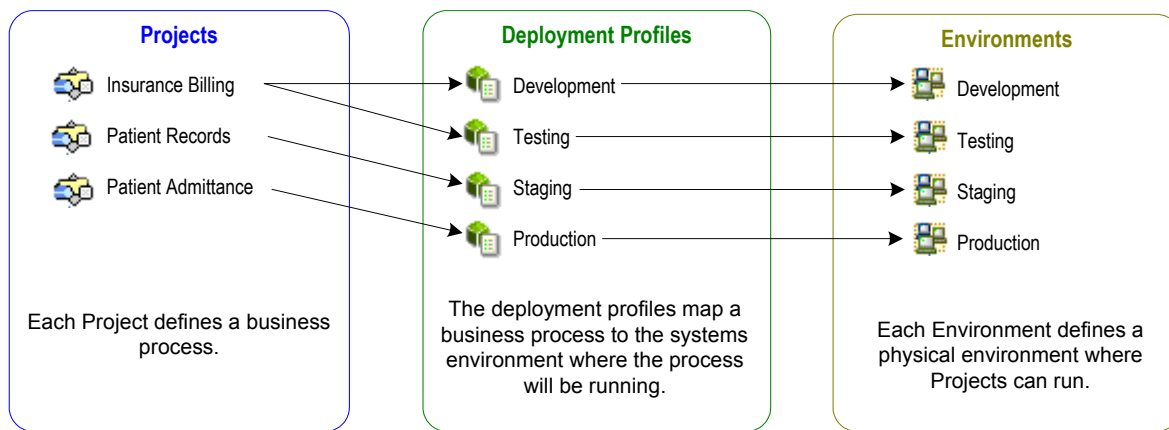
Projects are run within Logical Hosts, which are individual, runtime instances of eGate Integrator. Logical Hosts are defined within Environments, which represent the physical resources required to implement the Project. Projects are mapped to the individual Environments by means of Deployment Profiles, which are defined within

the Enterprise Designer and become part of the Project. Activating the Deployment Profile deploys the Project to the associated Environment.

This structure of Projects, Environments, and Deployment Profiles isolates each implementation into logical and physical components. This provides you with extensive flexibility and efficiency in designing eGate Integrator implementations. For example, once you build your Projects and Environments, you have the flexibility to change each component without having to make changes to the other component.

The finished Project, of course, will run in your production Environment; separate Environments, having the same structure as the production Environment, should be created for development and testing. You may also want some additional Environments, such as staging. The following figure illustrates the eGate Integrator implementation model using a healthcare-related example.

Figure 3 eGate Integrator Implementation Model



In the figure above, any of the Projects can be deployed to any of the Environments via the mapping defined in the deployment profiles. The example in the figure above shows that the patient admittance Project is already in the production phase and therefore was deployed using the production deployment profile. The patient records Project is in the staging phase and was therefore deployed to the staging Environment using the staging deployment profile. The insurance billing Project is still being developed and tested, and therefore it is deployed to development and testing via the development and testing profiles.

In broad outline, an eGate Integrator implementation includes the following steps:

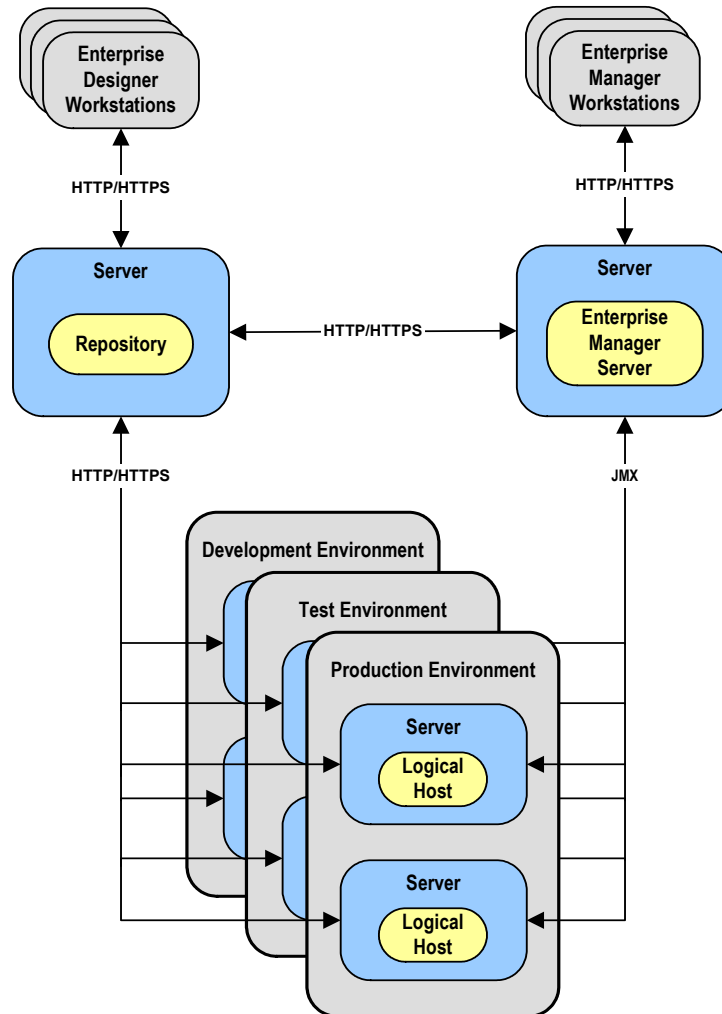
- 1 **Design your Project.**
- 2 **Define your Environments.**
- 3 **Create your Deployment Profiles.**
- 4 **Deploy the eGate Project.**

These implementation steps are all accomplished using the eGate Enterprise Designer, which is introduced in [Enterprise Designer](#) on page 29 and developed further in subsequent chapters.

2.3 System Architecture

eGate Integrator employs a flexible architecture that is ideal for distributed computing environments. As a result, the various components of an eGate Integrator system can reside on the same hardware platform (assuming adequate system resources), or be distributed across several different hardware platforms in the enterprise network. Figure 4 shows an example system implementation that is highly distributed.

Figure 4 Typical eGate Integrator System



2.3.1 Repository

The setup, components, and configuration information for the elements of a Project are stored in the Repository. The Repository also stores all of the product binary files that are required at runtime by the Logical Hosts. The components and configurations are downloaded to the Logical Host during the initial bootstrap process and as needed after design-time configuration changes are made.

As shown in Figure 4, a single Repository serves the entire enterprise. This common Repository is used for development, testing, and production purposes. Communication between the Repository and other eGate components can be configured to use either HTTP or HTTPS. The Enterprise Designer and Enterprise Manager clients can communicate with the Repository and Enterprise Manager servers through a firewall.

2.3.2 Environments

An eGate Environment represents the physical system required to implement a Project. It consists of a collection of Logical Hosts, capable of hosting components of the ICAN Suite, along with information about external systems involved in the implementation.

- **Logical Hosts**

Each Environment contains one or more Logical Hosts. A Logical Host contains one or more **integration servers**, which are the engines that run eGate Collaborations and eWays, and one or more **message servers**, which manage JMS topics (publish-and-subscribe messaging) and queues (point-to-point messaging).

- **External Systems**

An external system is a representation of a real, physical system that exists within the specific Environment, with configuration properties for locating and accessing that system.

In the example system shown in Figure 4, the production environment is split across two hardware platforms, each running a single Logical Host. Separate environments for development and testing should duplicate the structure of the production environment. The test environment should be supported by hardware similar to that supporting the production environment, to allow performance and load testing to give representative throughput results. The hardware supporting the development environment, however, does not usually have the same performance requirements as that supporting the test and production environments.

An eGate Project is created within the development environment, then migrated to the test environment, and finally to the production environment. This migration path is a necessary and highly critical practice in implementing a working system.

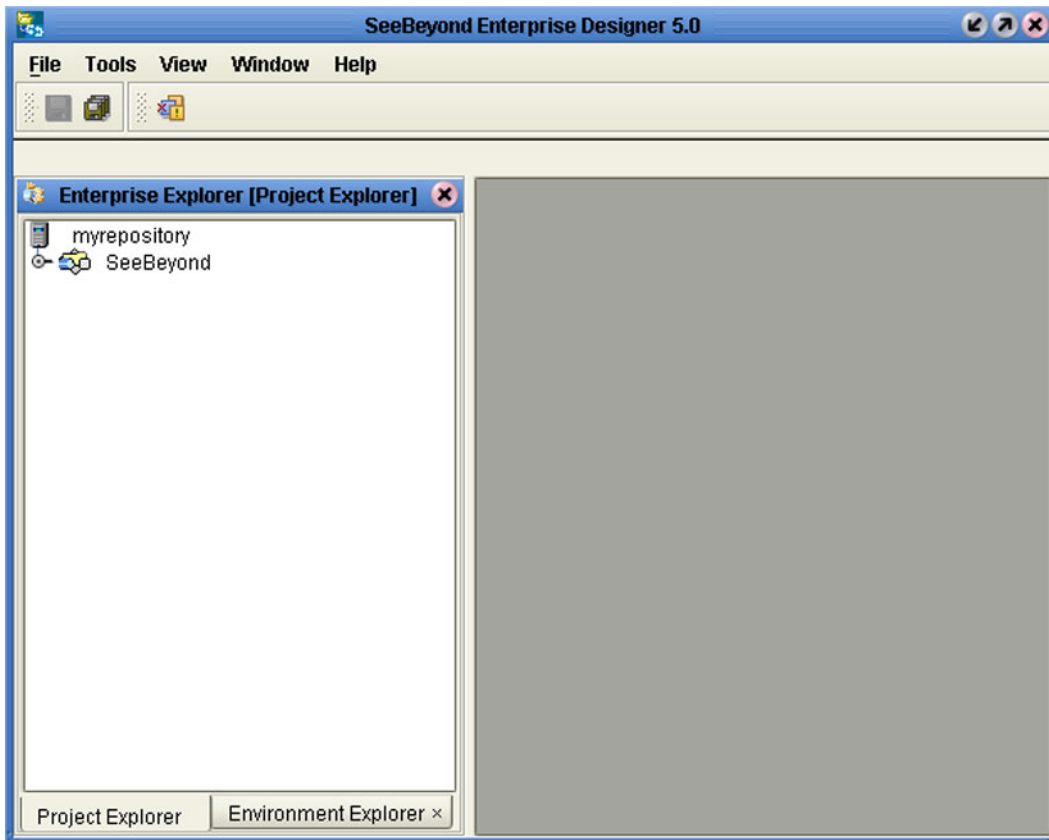
Note again that there is no requirement for the components shown in Figure 4 to run on separate systems; all could run on a single system, provided that resources (CPU, memory, and disk) are sufficient to support the concurrent usage.

2.4 User Interfaces

2.4.1 Enterprise Designer

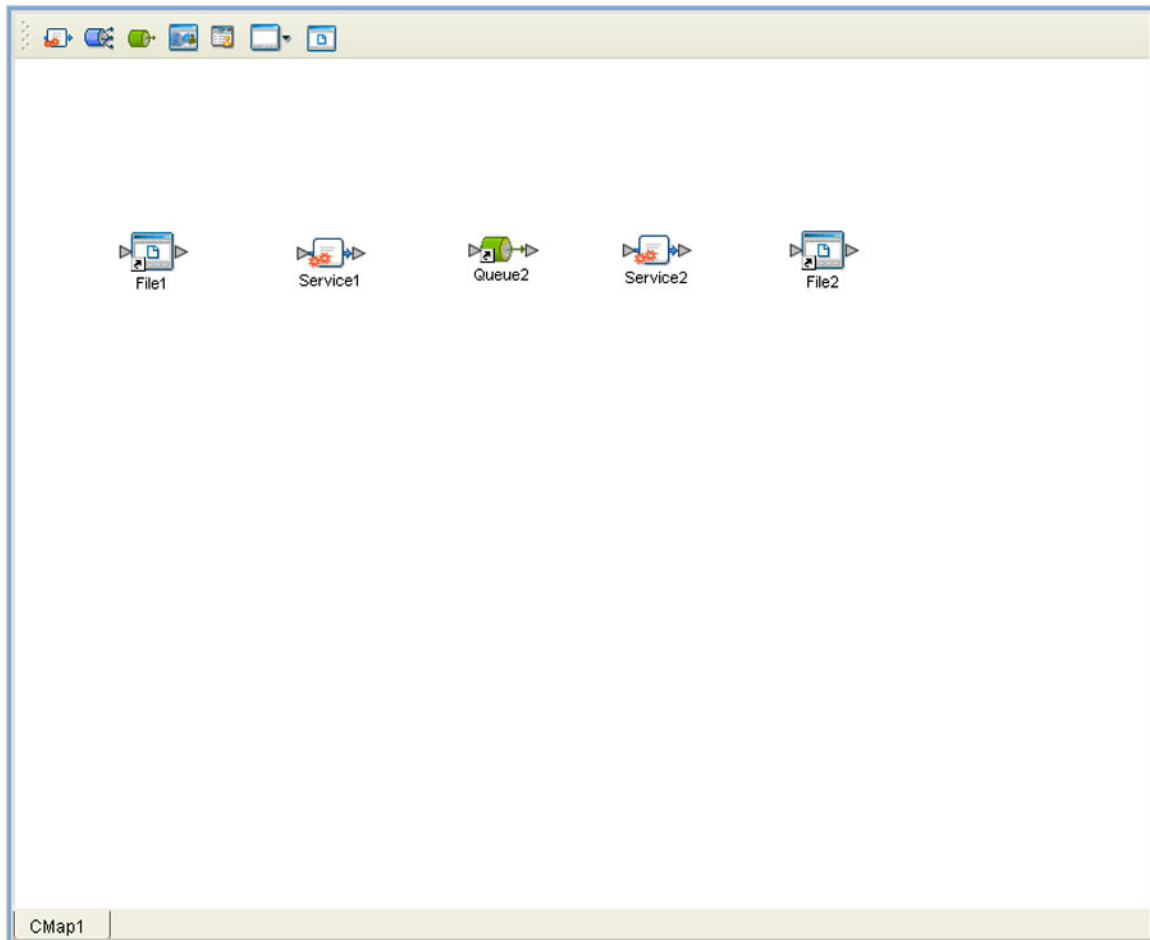
The Enterprise Designer graphical user interface (GUI) is used to create and configure the logical components and physical resources of an eGate Project. Through this GUI (see Figure 5), you can develop Projects to process and route data through an eGate Integrator system.

Figure 5 Enterprise Designer



The major features of the Enterprise Designer are the Enterprise Explorer on the left, and an editor panel on the right—which is initially blank. The Enterprise Explorer follows the familiar Windows Explorer format, displaying a tree structure. The editor panel displays a variety of editors, depending upon what component is selected in the Enterprise Explorer. The Connectivity Map Editor (see Figure 6) provides a graphic example of one of these, in which logical components of a Project are created and connected.

Figure 6 Connectivity Map Editor



The features and usage of the Connectivity Map Editor are described in [Creating Projects](#) on page 55. Other editors are displayed for creating and modifying Object Type Definitions, Collaboration Definitions, Deployment Profiles, and other Project components.

The Enterprise Designer also includes the design-time functionality for other ICAN products, such as eInsight and eXchange. For more information on using other ICAN products in the Enterprise Designer, see the product documentation for those products.

For more information on the Enterprise Designer, see [Enterprise Designer](#) on page 32 .

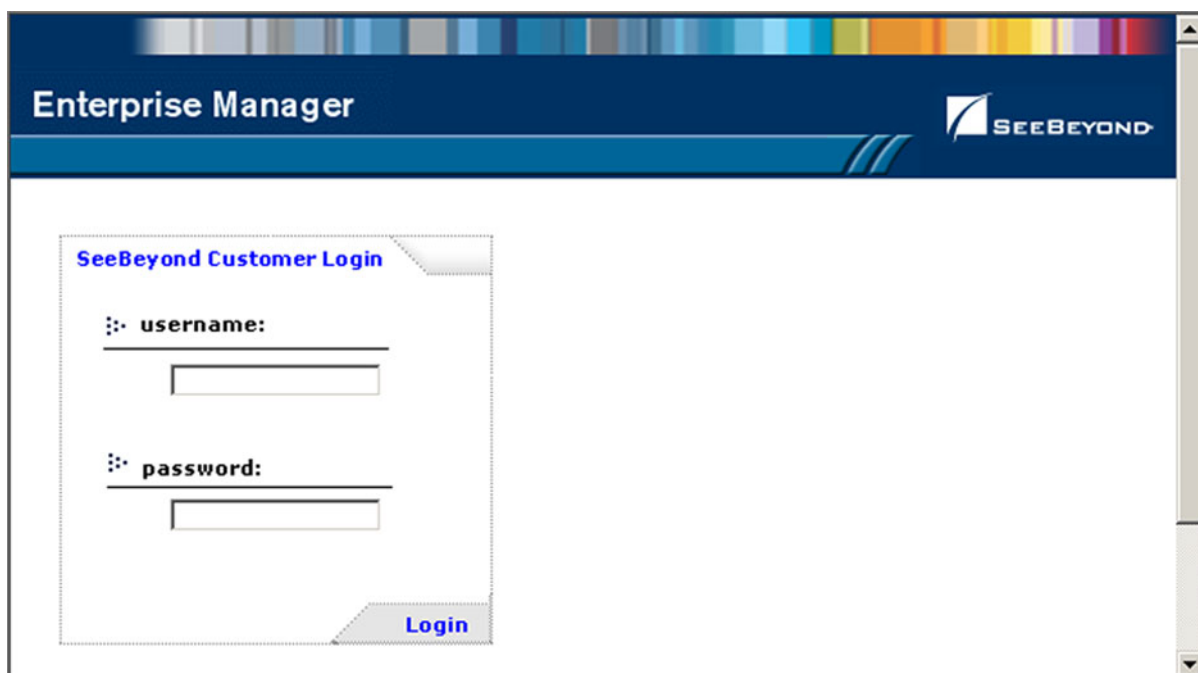
2.4.2 Enterprise Manager

The Enterprise Manager is a Web-based application you use for:

- Managing and monitoring eGate runtime components.
- Installing ICAN Suite products into the Repository.
- Downloading and installing products from the Repository.
- Accessing other Web-based ICAN Suite products.
- Accessing ICAN Suite product documentation.

The Enterprise Manager (see Figure 7) is accessed via Microsoft Internet Explorer.

Figure 7 SeeBeyond Enterprise Manager Login



The screenshot shows a web browser window displaying the SeeBeyond Enterprise Manager login page. The page features a dark blue header with the text "Enterprise Manager" on the left and the "SEE BEYOND" logo on the right. Below the header, there is a white box titled "SeeBeyond Customer Login". Inside this box, there are two input fields: "username:" and "password:", each with a small icon to its left. Below the password field is a "Login" button.

For more information on the Enterprise Manager, see [Enterprise Manager](#) on page 258.

For more information on Web Services capability, see [Web Services](#) on page 71.

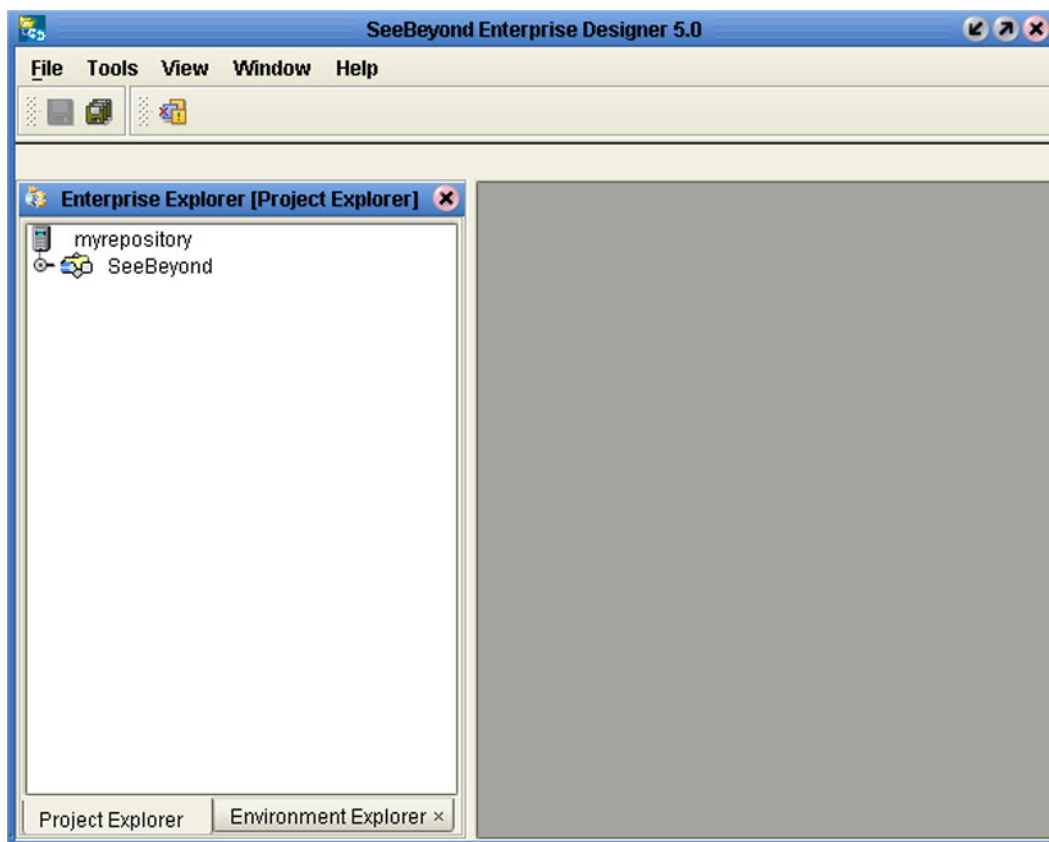
Enterprise Designer

This chapter describes the various features of the Enterprise Designer.

3.1 Overview

The Enterprise Designer graphical user interface (GUI) is used to create and configure the logical components and physical resources of an eGate Project. Through this GUI (see Figure 8), you can develop Projects to process and route data through an eGate Integrator system.

Figure 8 SeeBeyond Enterprise Designer



The major features of the Enterprise Designer are the Enterprise Explorer on the left, and an editor panel on the right—which is initially blank. The Enterprise Explorer follows the familiar Windows Explorer format, displaying a tree structure. The Enterprise Explorer provides two views of the ICAN system, described in the following sections of this chapter:

- **Project Explorer** on page 37
- **Environment Explorer** on page 42

The editor panel displays a variety of editors, depending upon what component is selected in the Enterprise Explorer. These editors are described in the following sections of this chapter:

- **Connectivity Map Editor** on page 49
- **OTD Editor** on page 50
- **Java Collaboration Editor** on page 51
- **XSLT Collaboration Editor** on page 52
- **Environment Editor** on page 53
- **Deployment Editor** on page 54

The Enterprise Designer also contains the customary graphical interface features, including the following:

- **Menu Bar** on page 34 describes the options contained in the individual menus.
- **Toolbar** on page 36 describes the functionality of the toolbar icons.
- **Browser Buttons** on page 36 describes the browser buttons that appear throughout the Enterprise Designer, in various wizards and dialog boxes.

The procedure for invoking the Enterprise Designer is described in **Starting Enterprise Designer** on page 57.

3.2 Menu Bar

The menu bar provides access to a variety of options for managing your Project. The individual menus are described in the following tables.

3.2.1 File Menu

Table 2 File Menu Options

Option	Function
Restore	Displays a dialog box with which you can locate and select a Repository archive file to restore in the Enterprise Designer. See Restoring a Repository on page 236.
Backup	Displays a dialog box with which you can select a location on your computer to save a copy of the Repository archive. See Backing Up a Repository on page 236.
Save	Saves changes to the selected Project.
Save All	Saves changes to all Projects.
Exit	Closes the Enterprise Designer.

3.2.2 Tools Menu

Table 3 Tools Menu Options

Option	Function
Impact Analysis	Displays a dialog box with which you can view how one component of a Project impacts other components. See Impact Analysis on page 230.
Update Center	Displays a dialog box with which you can check for program updates. See the <i>eGate Integrator Installation Guide</i> .

3.2.3 View Menu

Table 4 View Menu Options

Option	Function
Environment Explorer	Activates the Environment Explorer tab on the Enterprise Explorer. See Environment Explorer on page 42.
Project Explorer	Activates the Project Explorer tab on the Enterprise Explorer. See Project Explorer on page 37.

3.2.4 Window Menu

Table 5 Window Menu Options

Option	Function
Cascade	Displays all open windows so that each window slightly overlaps the others in the Project Editor.
Tile	Displays all open windows in a stacked tile pattern.
Horizontal Layout	Displays all open windows from top to bottom.
Vertical Layout	Displays all open windows from left to right.
Minimize All	Minimizes all open windows so that only the title bar displays at the bottom of the Project Editor.
Restore All	Returns minimized windows to their original position on the Project Editor.
Close All	Closes all open windows.




3.2.5 Help Menu

Table 6 Help Menu Options

Option	Function
Contents	Displays the online help for eGate Integrator.
Help Set	Displays additional online help options.

3.3 Toolbar






Table 7 Enterprise Designer Toolbar Icons

Icon	Function
	Save saves changes to the selected Project (inactive if no changes have been made).
	Save All saves changes to all Projects (inactive if no changes have been made).
	Displays the Impact Analyzer dialog box, which allows you to view how one component of a Project impacts other components.

3.4 Browser Buttons

The following buttons are used throughout the Enterprise Designer, in wizards and file selection dialog boxes. They correspond to standard Windows browser buttons.

Table 8 Browser Buttons

Button	Function
	Up One Level returns you to the parent folder or directory.
	Home returns you to the root folder or directory.
	Create New Folder creates a new folder under the current folder.
	List displays folder/file names only.
	Details displays details of the folders or files (name, type, date last modified, etc.).

3.5 Enterprise Explorer

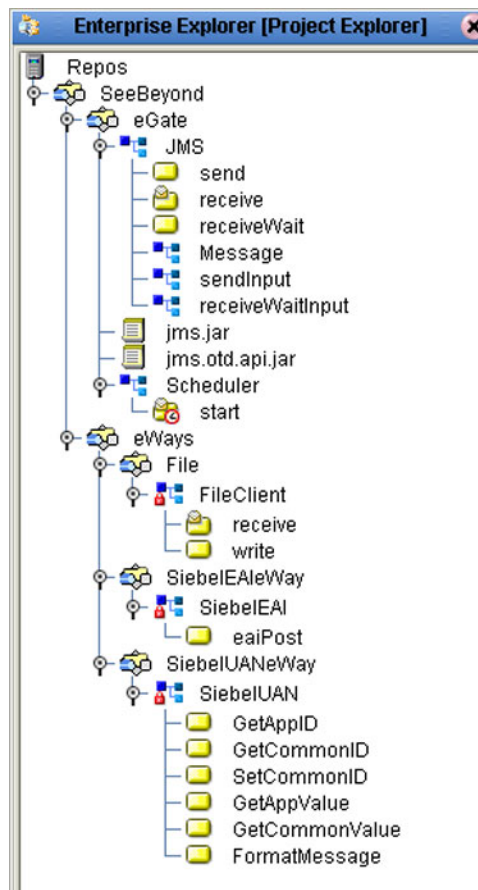
The Enterprise Explorer organizes the components of a Project into tabs that display different views of an eGate system.

- **Project Explorer** on page 37 deals with logical components.
- **Environment Explorer** on page 42 deals with physical resources, including the Logical Host and Integration Server.

3.5.1 Project Explorer

A Project consists of logical constructs and configurations designed to solve some or all of a business problem. The **Project Explorer** tab includes folders and icons that represent the names and contents of Projects. Some example components of a Project are shown in Figure 9.

Figure 9 Enterprise Explorer: Project Explorer View












Salient features of the Project Explorer are:

- **Project Explorer Icons** on page 38
- **Context Menus** on page 39

3.5.2 Project Explorer Icons

The icons described in Table 9 appear in the Project Explorer.

Table 9 Project Icons

Icon	Description
	Represents the Repository , which is the database where all Projects and contents are saved.
	Represents the Project or subproject.
	Represents a Connectivity Map , which contains the business logic and information about the data transmission. .
	Represents a Project variable or constant .
 	Represents an Object Type Definition (OTD) file. A lock displayed in the lower-left corner indicates that the OTD is currently checked into the version control system.
	Represents a Java Collaboration Definition file. A lock is displayed in the lower-left corner for Collaboration Definitions that are currently checked into the version control system.
	Represents an XSLT Collaboration Definition file. A lock is displayed in the lower-left corner for Collaboration Definitions that are currently checked into the version control system.
	Represents a Deployment Profile , which specifies how Project components will be deployed in an Environment.

3.5.3 Context Menus

Right-clicking on a component in the Project Explorer displays a context menu for that component. Included here are descriptions of options for the following component context menus:

- [Repository Menu](#) on page 39
- [Project Menu](#) on page 40
- [Connectivity Map Menu](#) on page 41

Repository Menu

Figure 10 Repository Menu

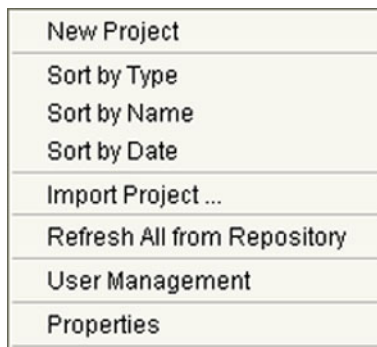


Table 10 Repository Menu Options

Option	Function
Project	Adds a new Project to the Project Explorer tab.
Sort by Type	Places Project components in order by grouping component types.
Sort by Name	Places Projects and Project components in alphabetical order.
Sort by Date	Places Projects in order by creation date from oldest Project to newest.
Import Project	Displays a dialog box, which you can use to locate and select a Repository archive file to import into the Enterprise Designer.
Refresh All from Repository	Updates Enterprise Designer with Project/Environment configurations stored in the Repository. (Open editors are not refreshed.)
User Management	Displays the User Management dialog box, where you can manage user access to the Repository with options for adding, modifying, and deleting users. See Configuration User Management on page 243.
Properties	Displays a dialog box (the exact name of the dialog box mirrors the name of your Repository). Use this dialog box to manage Enterprise Designer users and their access to the Repository. See Configuration User Management on page 243.

Project Menu

Figure 11 Project Menu

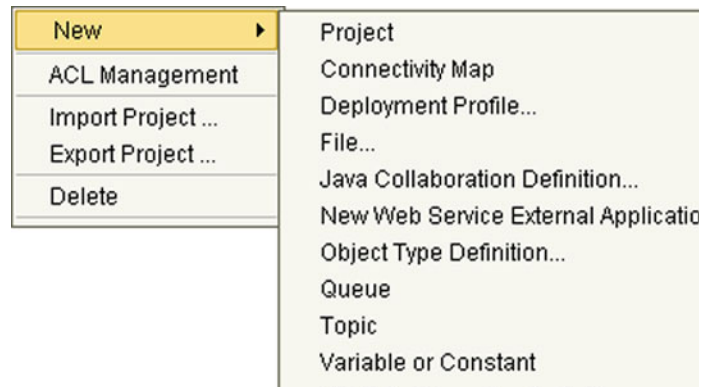


Table 11 Project Menu Options

Option	Option	Function
New	Project	Adds a Subproject folder to the selected Project.
	Connectivity Map	Adds a Connectivity Map to the Project. See Using the Connectivity Map Editor on page 60.
	Deployment Profile	Displays a dialog box with which you can assign a Deployment Profile to the selected Project. See Using the Deployment Editor on page 204.
	File	Displays a dialog box with which you can create an external file to use with the Project.
	Java Collaboration Definition	Displays the Java Collaboration Definition Wizard , with which you can create a Java Collaboration Definition. See Using the Java Collaboration Wizard on page 116.
	New Web Service ...	Adds a third-party Web service application to the Project Explorer. See Example Web Services Project on page 73.
	Object Type Definition	Displays the OTD Wizard , with which you can create an Object Type Definition (OTD) file. See Using the OTD Wizard on page 79 for more information.
	Queue	Adds a queue to your Project.
	Topic	Adds a topic to your Project.
	Variable or Constant	Displays a dialog box with which you can add a constant or variable icon to your Project.
ACL Management		Displays the ACL Properties dialog box, with which you can assign read and/or write privileges to users for the selected Project. See ACL Management on page 248.
Import Project		Displays a dialog box with which you can locate and select a Project file to import into the Enterprise Designer.
Export Project		Displays a dialog box with which you can select a location on your computer to save a copy of the Project.

Table 11 Project Menu Options

Option	Option	Function
Delete		Deletes the selected Project.

Connectivity Map Menu

Figure 12 Connectivity Map Menu

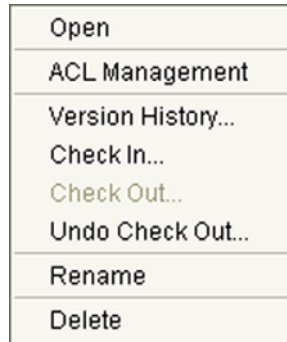


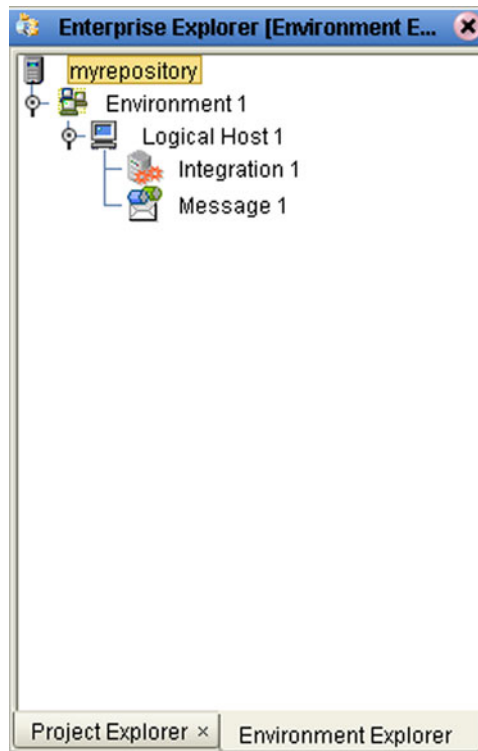
Table 12 Connectivity Map Menu Options

Command	Function
Open	Opens the Connectivity Map Editor for the selected Connectivity Map. See Using the Connectivity Map Editor on page 60.
ACL Management	Displays the ACL Properties dialog box, with which you can assign read and/or write privileges to users for the selected Project. See ACL Management on page 248.
Version History	Displays a dialog box with which you can track the version history for OTDs and Collaboration Definitions. Version control allows users to maintain multiple versions of the same OTD and Collaboration Definition files. See Version Control on page 233 for more information.
Check In	Displays a dialog box, with which you can check in the current version of a Project. Refer to Checking In a Project or Environment Component on page 234 for more details.
Check Out	Displays a dialog box with which you can check out a version of a Project. See Checking Out a Project or Environment Component on page 235 for more information.
Undo Check Out	Reverses the Check Out command, returns you to the previous state.
Rename	Allows you to rename the selected Connectivity Map.
Delete	Displays a dialog box in which you confirm that you want to delete the selected Connectivity Map. Clicking Yes then deletes the Connectivity Map.

3.5.4 Environment Explorer

An Environment consists of Logical Hosts capable of hosting eGate components and information about external systems which may be involved with an eGate configuration.

Figure 13 Enterprise Explorer: Environment Explorer View











Salient features of the Project Explorer are:

- [Environment Explorer Icons](#) on page 43
- [Context Menus](#) on page 44

3.5.5 Environment Explorer Icons

The icons described in Table 13 appear in the Environment Explorer.

Table 13 Environment Icons

Icon	Function
	Represents the Repository , which is the database where all Projects and contents are saved.
	Represents the Environment , which contains Logical Hosts and information about external systems.
	Represents a Logical Host , which contains the software and other installed components that are required at runtime.
	Represents an external file .
	Represents an Environmental constant, which you can use to automate eWay and message destination configuration changes.
	Represents a Scheduler component of an Environment, which you can use to set data transfer to occur at set intervals.
	Represents an Integration Server .
	Represents a JMS IQ Manager or third-party message server , which is used to store and forward eGate system messages.

3.5.6 Context Menus

Right-clicking on a component in the Environment Explorer displays a context menu for that component. Included here are descriptions of options for the following component context menus:

- [Repository Menu](#) on page 44
- [Environment Menu](#) on page 45
- [Logical Host Menu](#) on page 46

Repository Menu

Figure 14 Repository Menu



Table 14 Repository Menu Options

Option	Function
New Environment	Displays a dialog box with which you can create a new Environment.
Save Changes to Repository	Saves all changes made in the Environment Explorer.
Refresh All from Repository	Updates Enterprise Designer with Project/Environment configurations stored in the Repository. (Open editors are not refreshed.)

Environment Menu

Figure 15 Environment Menu

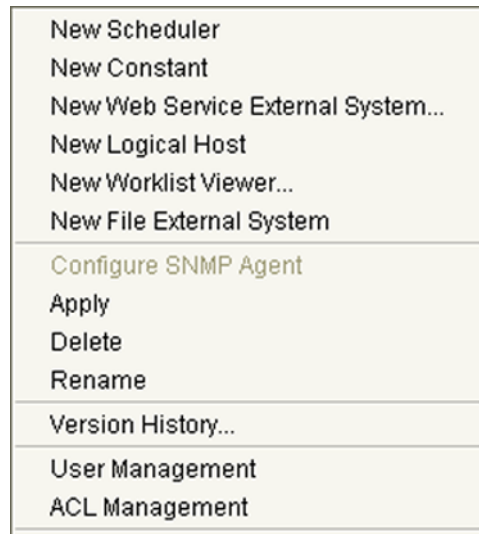


Table 15 Environment Menu Options

Option	Function
New Scheduler	Displays a dialog box with which you can add a new scheduling component to the selected Environment.
New Constant	Displays a dialog box with which you can add a constant to the Environment.
New Web Service ...	Adds a third-party Web service application to the Project Explorer. See Example Web Services Project on page 73.
New Logical Host	Adds a new Logical Host to the selected Environment.
New Worklist Viewer	This option is present only when eInsight Business Process Manager is installed. See the <i>eInsight Business Process Manager User's Guide</i> for information.
New File External System	Displays a dialog box with which you can add a new external file to the selected Environment.
Configure SNMP Agent	Displays a dialog box with which you can modify agent properties.
Apply	Saves changes to the selected Environment.
Delete	Displays a dialog box in which you confirm that you want to delete the selected Environment. Clicking Yes then deletes the Environment.
Rename	Allows you to rename the selected Environment.
Version History	Displays a dialog box with which you can track the version history for Environments. Version control allows users to maintain multiple versions of the same Environment. See Version Control on page 233 for more information.

Table 15 Environment Menu Options

Option	Function
User Management	Displays a dialog box with which you can manage Repository access. See Configuration User Management on page 243.
ACL Management	Displays the ACL Properties dialog box, with which you can assign read and/or write privileges to users for the selected Project. See ACL Management on page 248.

Logical Host Menu

Figure 16 Logical Host Menu

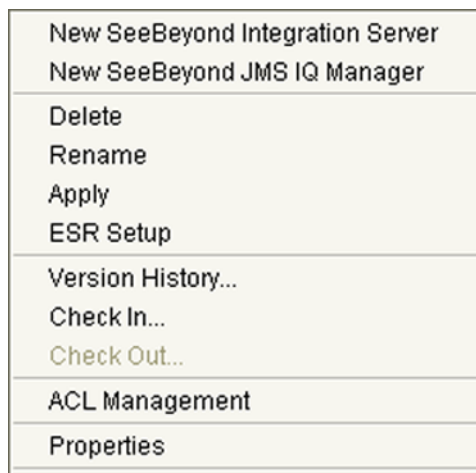


Table 16 Logical Host Menu Options

Option	Function
New SeeBeyond Integration Server	Adds a new SeeBeyond integration server to the selected Logical Host.
New SeeBeyond JMS IQ Manager	Adds a new SeeBeyond JMS IQ Manager to the selected Logical Host.
Delete	Displays a dialog box in which you confirm that you want to delete the selected Logical Host. Clicking Yes then deletes the Logical Host.
Rename	Allows you to rename the selected Logical Host.
Apply	Saves changes to the selected Logical Host.
ESR Setup	Displays a dialog box with which you can select emergency software releases (ESRs) to add to the Environment.
Version History	Displays a dialog box with which you can track the version history for Logical Hosts. Version control allows users to maintain multiple versions of the same Logical Host. See Version Control on page 233 for more information.

Table 16 Logical Host Menu Options

Option	Function
Check In	Displays a dialog box, with which you can check in the current version of an Environment. Refer to Checking In a Project or Environment Component on page 234 for more details.
Check Out	Displays a dialog box with which you can check out a version of an Environment. See Checking Out a Project or Environment Component on page 235 for more information.
ACL Management	Displays the ACL Properties dialog box, with which you can assign read and/or write privileges to users for the selected Environment See ACL Management on page 248.
Properties	Displays a dialog box with which you can modify the default settings for the selected Logical Host.

Note: If you are using BEA WebLogic and/or IBM WebSphere, the Integration Servers and JMS Message Servers for these products will also appear in the context menu.

3.6 Enterprise Designer Editors

This part of the Enterprise Designer is empty when you start a new Project. As you proceed through the development process, different editors are displayed in this area. These are described briefly as follows:

- [Connectivity Map Editor](#) on page 49
- [OTD Editor](#) on page 50
- [Java Collaboration Editor](#) on page 51
- [XSLT Collaboration Editor](#) on page 52
- [Environment Editor](#) on page 53
- [Deployment Editor](#) on page 54

Additional facilities are also displayed here; for example, see [The Java Debugger](#) on page 286.

Note: See the [eGate Tutorial](#) for detailed information and instructions for setting up a Project.

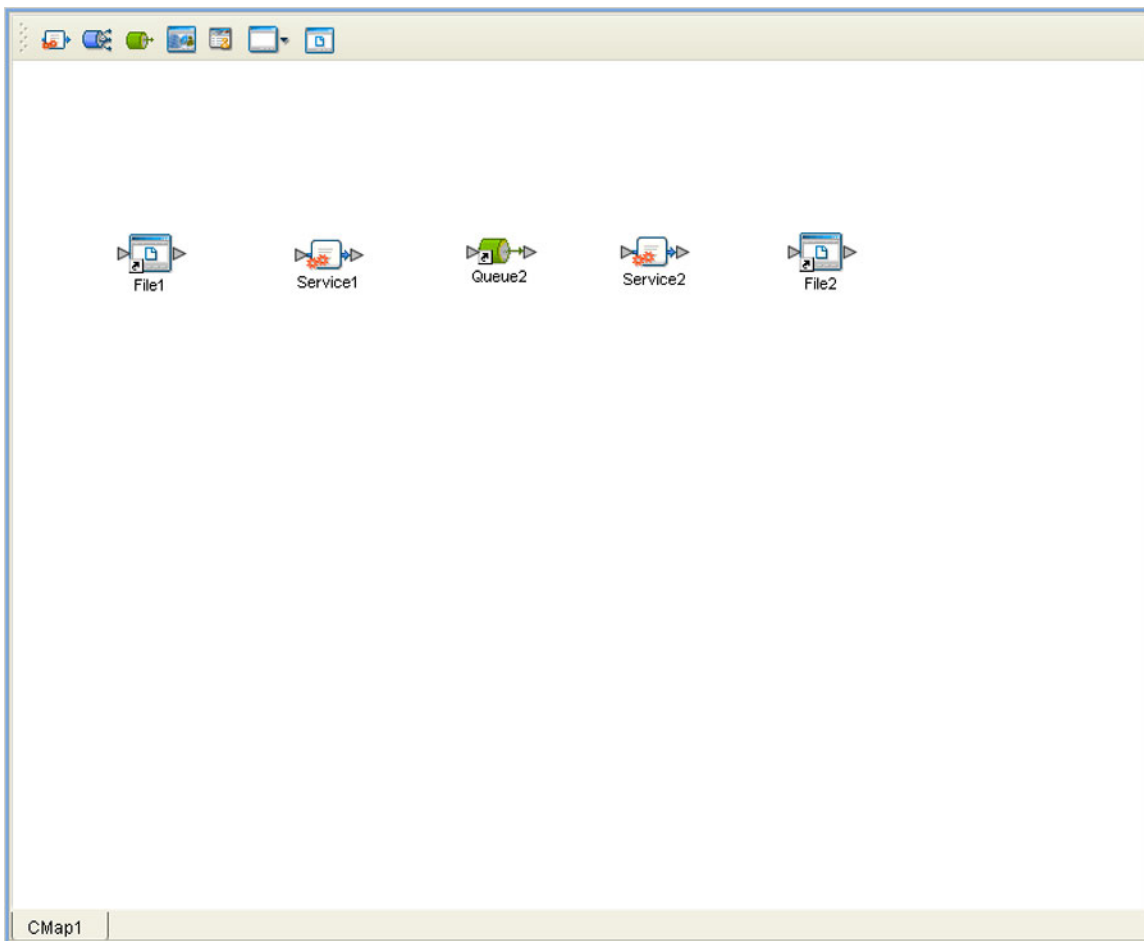
3.6.1 Connectivity Map Editor

A Connectivity Map is a graphical representation of your Project, containing the various logical components comprising the Project and the links between them. The Connectivity Map Editor, shown in Figure 17, allows you to create your Project by simply dragging and dropping icons onto a Project canvas and then connecting them to form data paths. You then can configure the components by means of dialog boxes that are displayed by clicking on the component icons.

Note: *It is best to create your Collaboration Definitions before using the Connectivity Map to connect components.*

See [Using the Connectivity Map Editor](#) on page 60 for detailed information.

Figure 17 Connectivity Map Editor

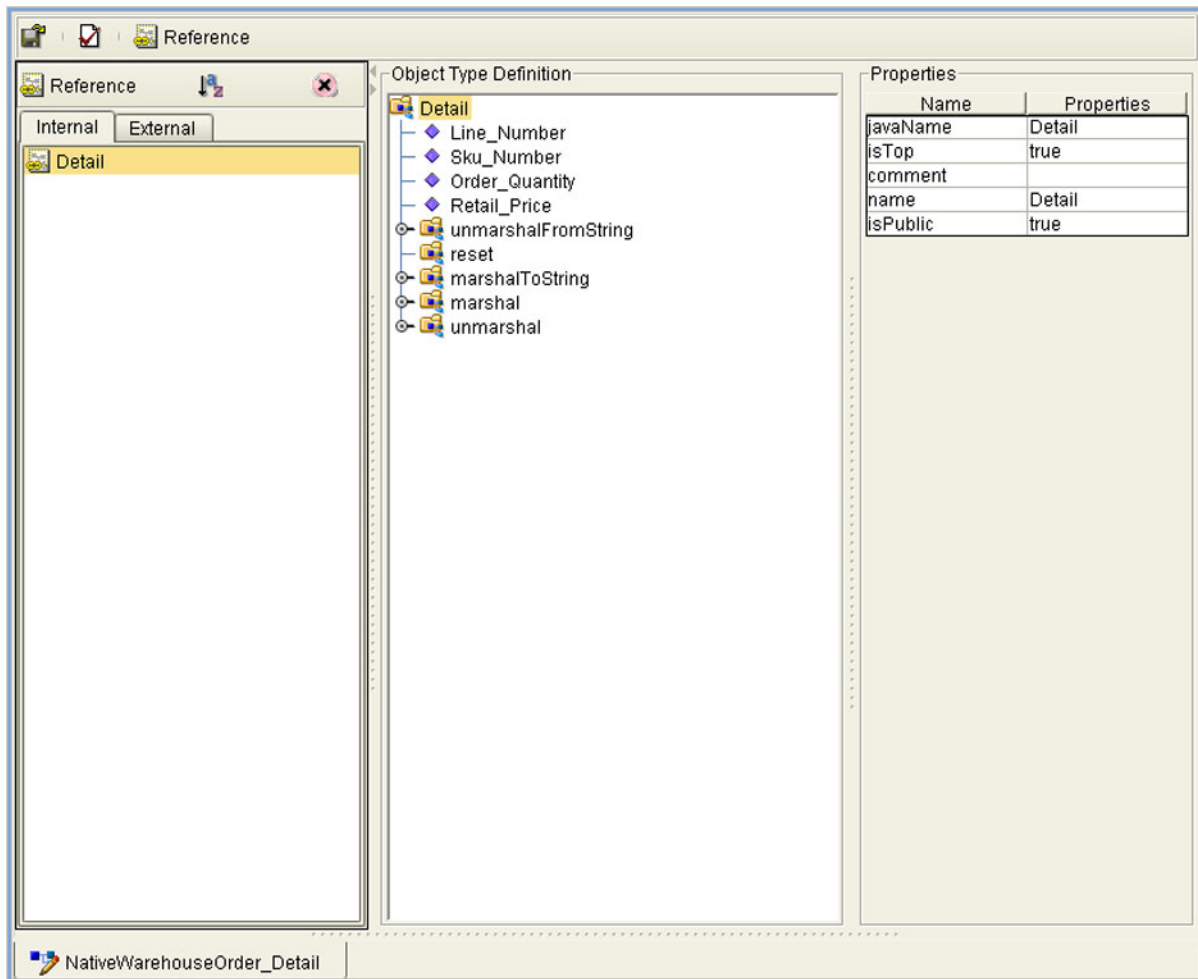


3.6.2 OTD Editor

The OTD Editor window, as shown in Figure 18, displays the source files used to create the Object Type Definitions (OTDs) to use with a Project. You use an OTD wizard tool to compile OTD files and add them to the **Project Explorer** tab.

See [Using the OTD Editor](#) on page 108 for detailed information.

Figure 18 OTD Editor

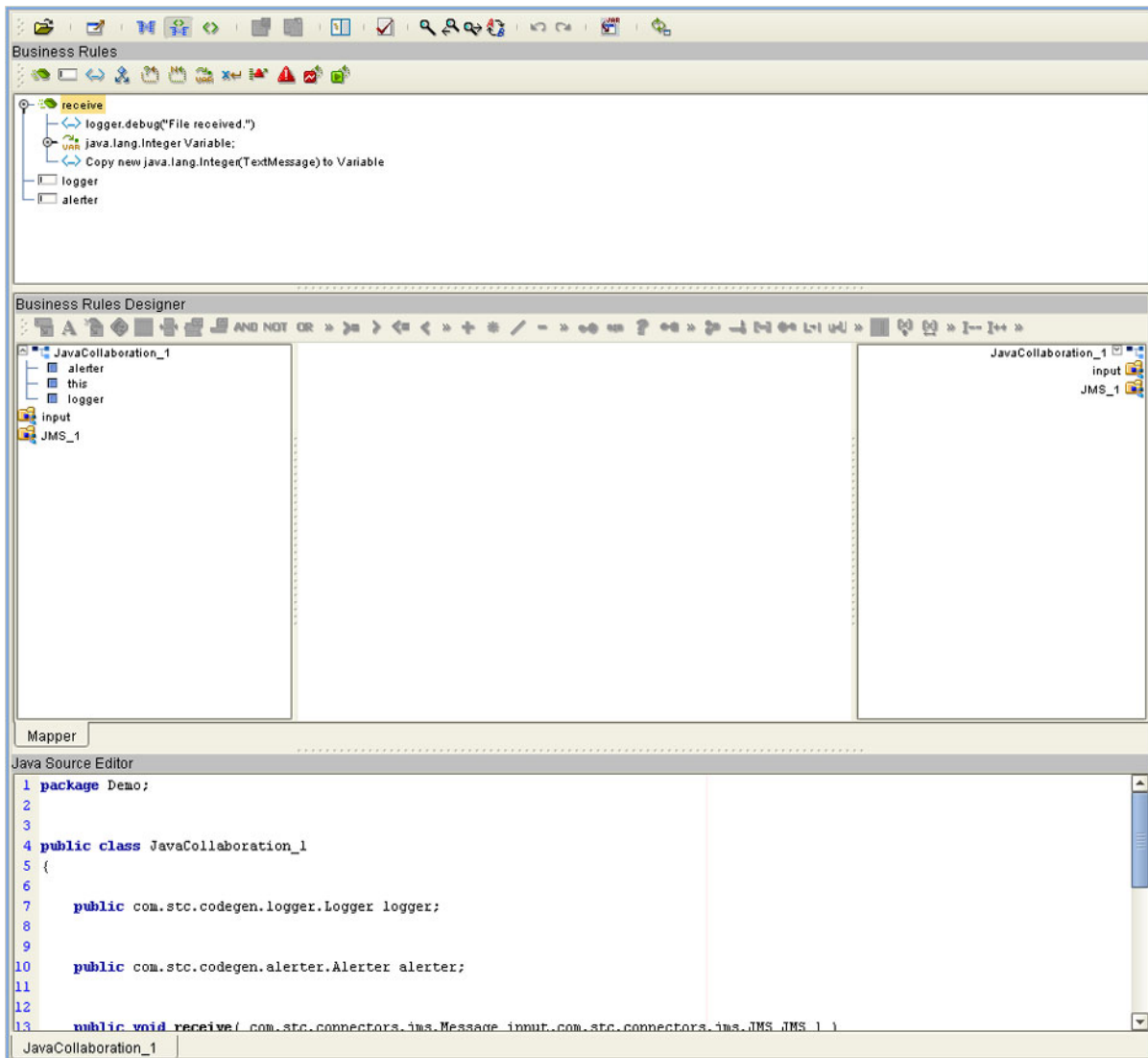


3.6.3 Java Collaboration Editor

The Java Collaboration Editor window, as shown in Figure 19, displays a Java-based Collaboration Definition that you want to include in a Project. You use a Java wizard tool to compile Collaboration Definition files and add them to the **Project Explorer** tab.

See [Using the Java Collaboration Editor](#) on page 124 for detailed information

Figure 19 Java Collaboration Editor

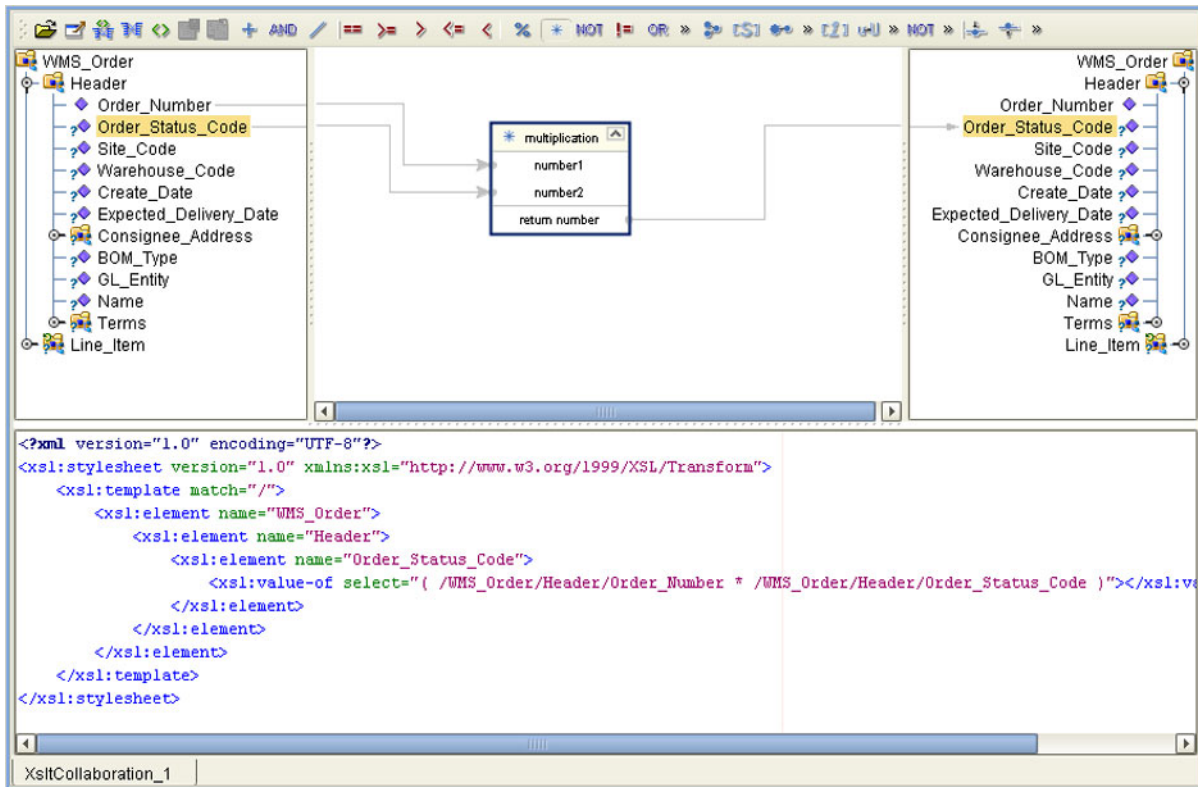


3.6.4 XSLT Collaboration Editor

The XSLT Collaboration Editor window, as shown in Figure 20, displays the XSLT Collaboration Definitions that you need to map together and include in the Project. You use a XSLT wizard tool to compile Collaboration Definition files and add them to the Project Explorer tab.

See [Using the XSLT Collaboration Editor](#) on page 182 for detailed information

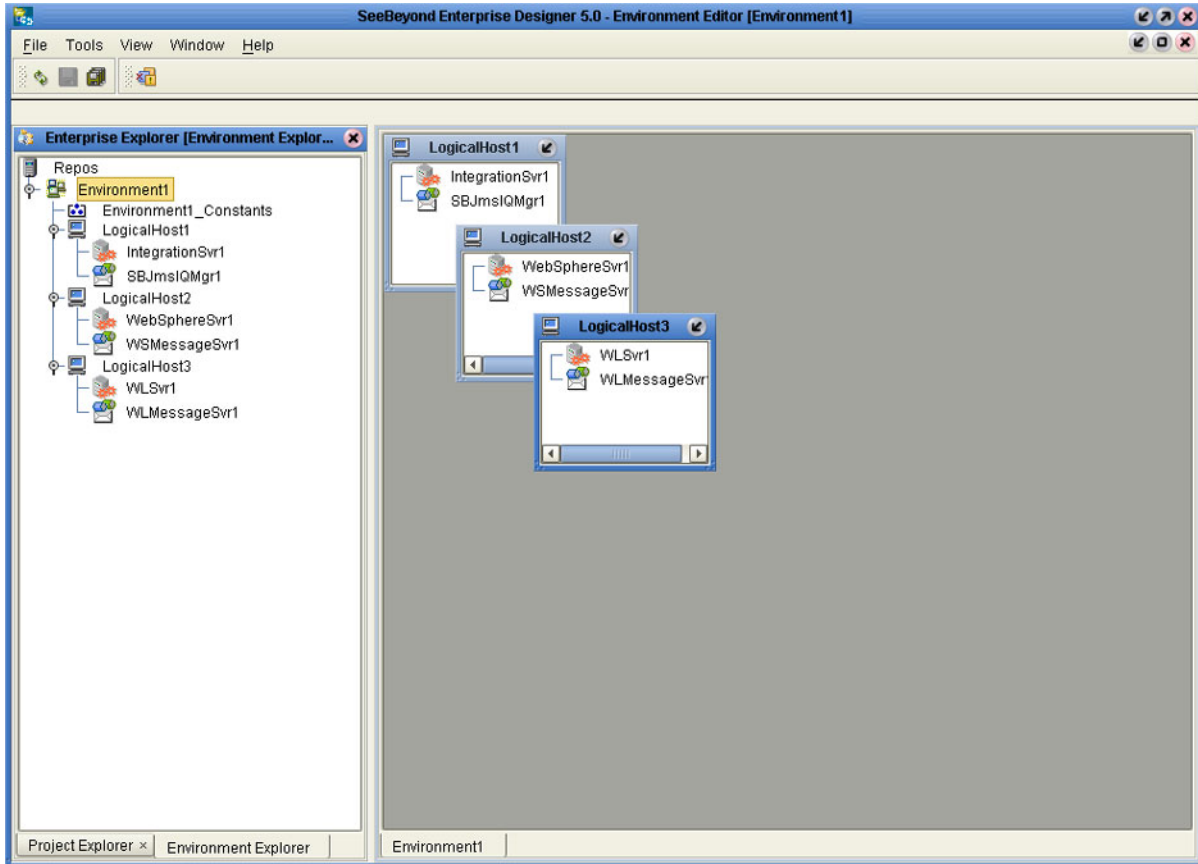
Figure 20 XSLT Collaboration Editor



3.6.5 Environment Editor

Clicking on an Environment icon in the Environment Explorer invokes the Environment Editor, which provides a canvas in which you can create and customize an Environment. Here you can see the various components (Logical Hosts, servers, and external systems) included in the selected Environment. An environment containing example Logical Hosts is shown in Figure 21.

Figure 21 Environment Editor

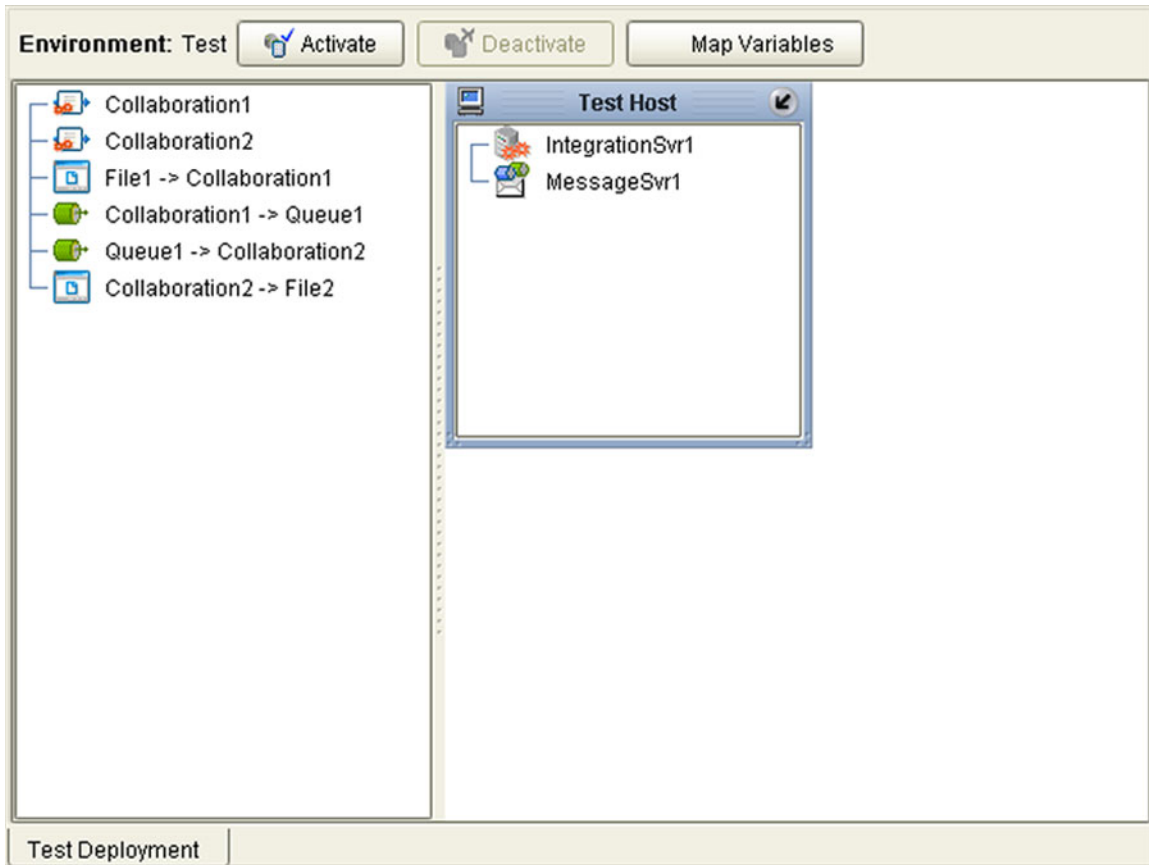


Note: Unlike changes to Project-related configuration properties, changes to Environment-related properties do not require redeployment.

3.6.6 Deployment Editor

The Deployment Editor, as shown in Figure 22, contains information about how Project components will be deployed in an Environment. See [Using the Deployment Editor](#) on page 204 for detailed information

Figure 22 Deployment Editor



Creating Projects

This chapter describes the use of the Enterprise Designer in defining your Project.

4.1 Overview

An eGate Project represents the logical system designed to solve either all or part of a business problem. Projects are created using tools contained within the Enterprise Designer (see [Enterprise Designer](#) on page 32). They are deployed to specific Logical Hosts in specific Environments by means of Deployment Profiles (see [Environments and Project Deployment](#) on page 200). An end-to-end example of Project implementation is given in the *eGate Integrator Tutorial*.

A typical Project contains:

- Object Type Definitions (OTDs), that define object encoding rules.
- Collaborations, based on Collaboration Definitions, that process data.
- Message destinations, including topics and queues, that store data.
- Connections, including eWays and JMS Clients, that specify logical connections.
- Proxies for external applications.
- Connectivity Maps, that display the business logic.
- Deployment Profiles, that map the Project to specific Environments.

If you are also using eInsight Business Process Manager, an eGate Project is related to an Activity in an eInsight business process. Components developed for use in one Project can be used in another, and a Project can internally reference another Project.

4.2 Project Components

As previously mentioned, an eGate Project represents the logical system designed to solve either all or part of a business problem. Projects are created using tools contained within the Enterprise Designer such as the Connectivity Map Editor. Various components of a Project shown in the figure include:

- **External Application Proxies**

The basic purpose of eGate Integrator is to facilitate the interchange of data between external business applications. These business applications are collectively referred to as External Applications, and are represented in the Project by logical proxies for the specific applications involved. An External Application can be identified with an ERP application such as SAP or PeopleSoft, a DBMS such as Oracle or SQL, or with a particular communications protocol, such as TCP/IP or HTTPS.

- **Collaborations**

A logical operation taking place between a Message Destination and an External Application, two Message Destinations, or two External Applications, having a publication and subscription relationship with those entities. The operation is defined by a Collaboration Definition, which can be encoded in either Java or XSLT.

- **Message Destinations**

A Message Destination is a container for stored data, and can follow either the topic or queue JMS model. A **topic** is a message destination that conforms to the publish-and-subscribe messaging paradigm. A **queue** is a message destination that conforms to the point-to-point messaging paradigm.

- **Logical Connections**

An **eWay Intelligent Adapter** provides the logical connection between external systems (applications or protocols) and a service. The identity of the External Application placed in a Connectivity Map dictates the type of eWay adapter assigned to the link between it and a Service such as a Collaboration, and can be accessible via multiple transport mechanisms.

A **JMS Client Connection** specifies the connection properties for the linked service (for example, publisher or subscriber).

Behind the scenes, and not explicitly shown in a Connectivity Map, are other Project components, such as:

- **Collaboration Definitions**

A Collaboration Definition defines the logical operation taking place in the related Collaboration. It is created in either the Java Configuration Editor or the XSLT Configuration Editor, and is based on an Object Type Definition.

- **Object Type Definitions**

Object Type Definitions (OTDs) are sets of rules that define the encoding of an object. They describe messages that are propagated through eGate, and the methods available for operating on them, and also interactions with external APIs.

4.3 Starting Enterprise Designer

To start the Enterprise Designer

- 1 Start the Enterprise Designer using one of these two methods:
 - If you are connecting to a Repository on an HP NonStop Server, open a command prompt and change directories to `C:\ICAN50\edesigner\bin`, and type the following command:

```
runed hostname port rep_name
```

where:

hostname is the TCP/IP host name of the server where you installed the Repository—not the name of the Repository itself,

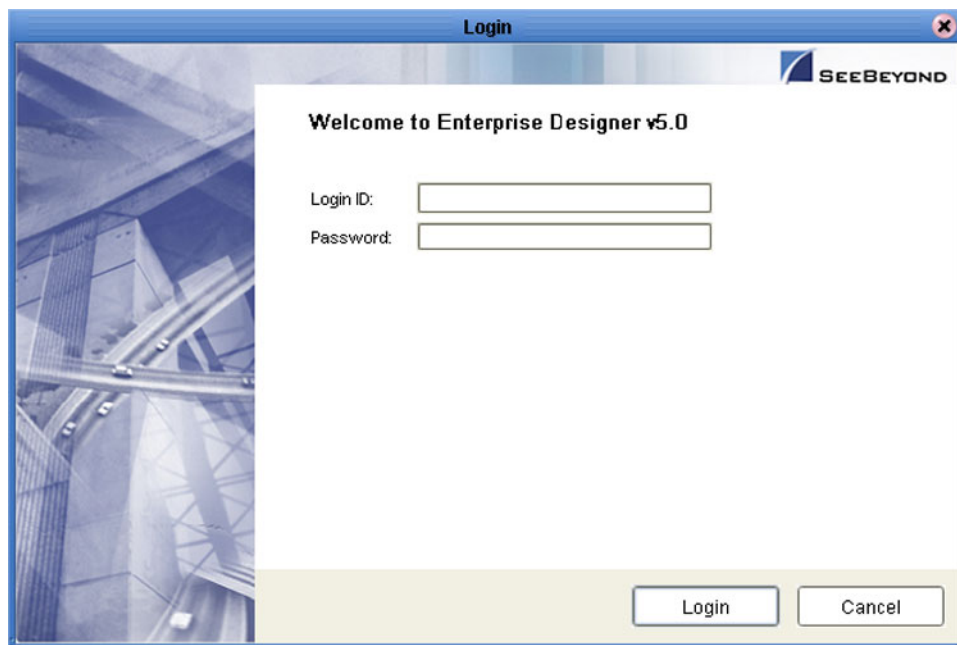
port is the port number of the Repository, and

rep_name is the name of the Repository.

This displays the dialog box shown in Figure 23.

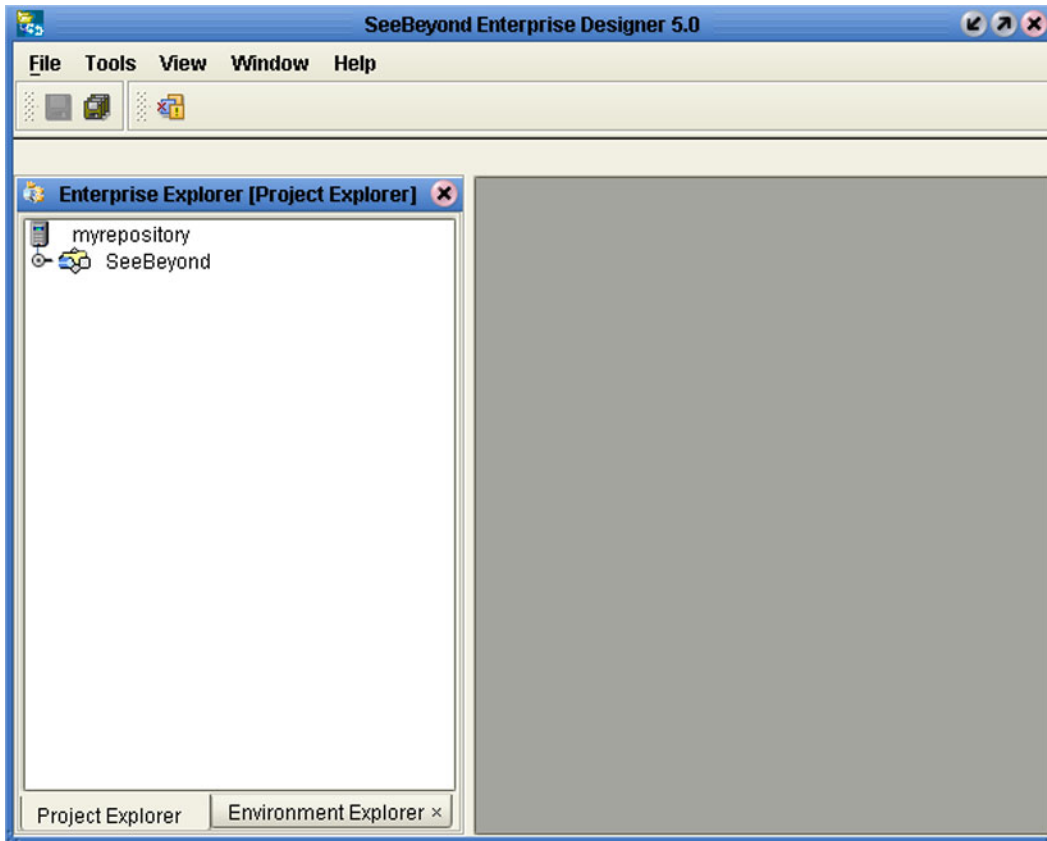
- If you are connecting to a Repository on any other platform, run the batch file `C:\ICAN50\edesigner\bin\runed.bat` to display the *Login* dialog box shown in Figure 23.

Figure 23 Login Dialog Box



- 2 Click in the *Login ID* text box, and enter your login ID.
- 3 Tab to the *Password* text box, and enter your password.
- 4 Click **Login** to complete the login process and display the Enterprise Designer GUI shown in Figure 24.

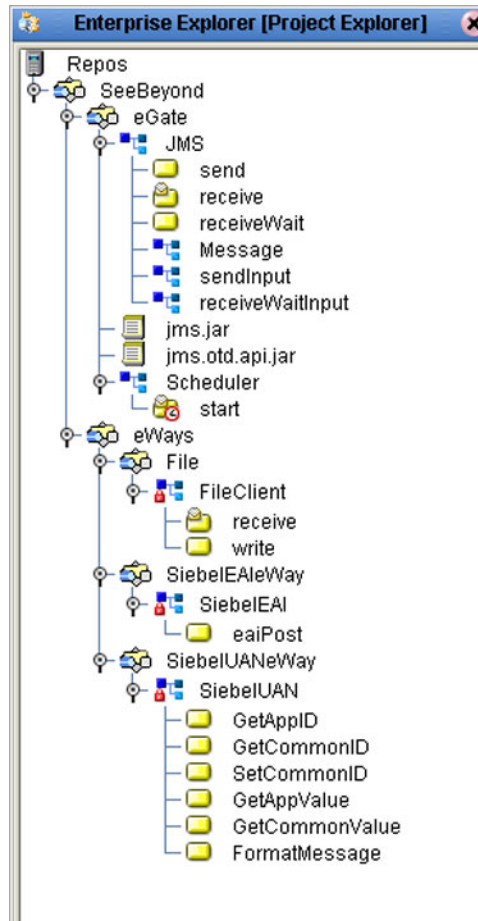
Figure 24 SeeBeyond Enterprise Designer



The editor panel—which is initially blank—displays a variety of editors, depending upon what component is selected in the Enterprise Explorer. The use of these editors is described in the following sections of this chapter.

4.4 Using the Project Explorer

Figure 25 Project Explorer

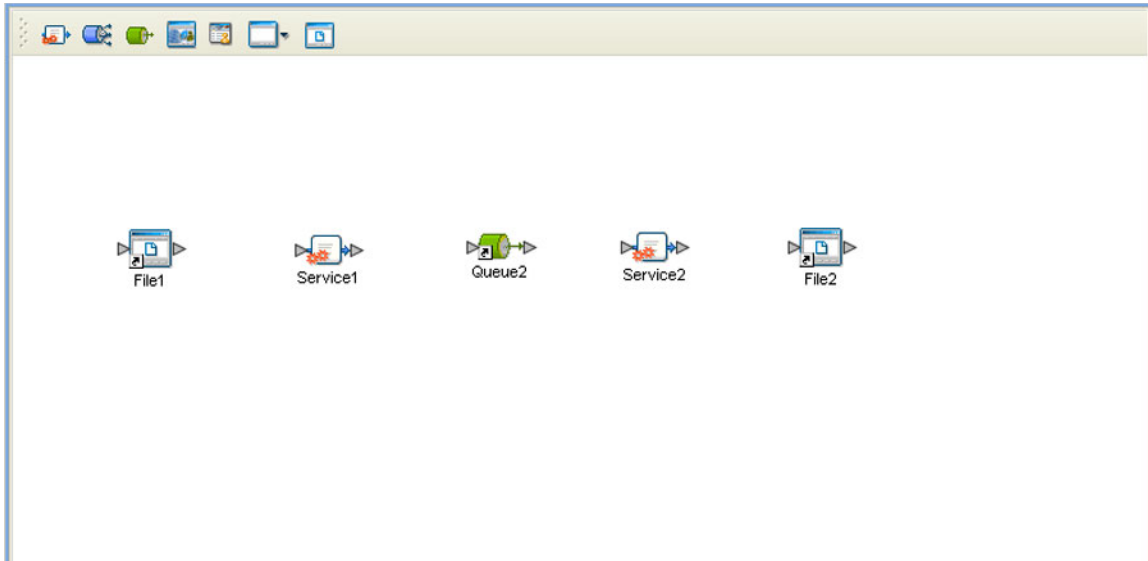


Note: Any changes made to the names of the Collaborations should be done when the Collaboration is created. If the name is changed later, the Collaboration should be opened, regenerated if applicable, and saved again. For safe measure, this should also be done before creating the Connectivity Map and Deployment Profile.

4.5 Using the Connectivity Map Editor

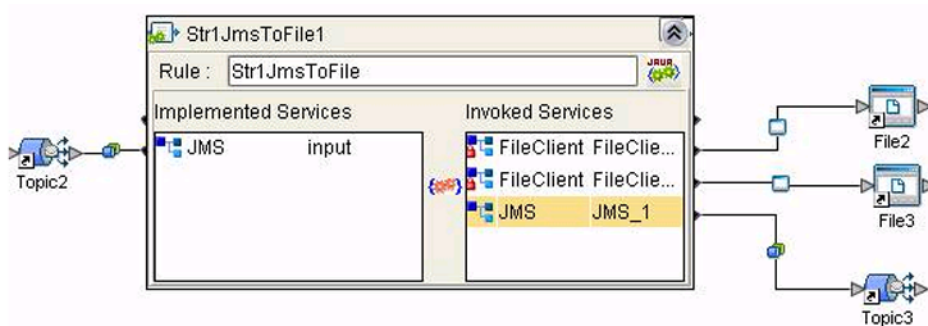
When you create a new Connectivity Map in the Enterprise Explorer, the editor panel displays the Connectivity Map Editor (see Figure 26). To define your Project, you simply drag icons from the toolbar to the workspace, or canvas, to populate the Connectivity Map with the necessary components. You subsequently link the components by dragging the cursor from one to the other.

Figure 26 Connectivity Map Window



When there are multiple destinations, as with a JMS topic, the Connectivity Map Editor cannot resolve which output port to connect to which destination. Because of this, the Collaboration definition must be created first, and the connections must be drawn by opening the Collaboration Binding box in Connectivity Map (see Figure 27).









Figure 27 Linking JMS Topics



4.5.1 Connectivity Map Editor Toolbar

Each toolbar icon in the Connectivity Map Editor is described in Table 17.

Table 17 Connectivity Map Toolbar Icons

Icon	Component	Function
	Service	A logical component that provides the framework for a process or Collaboration. See Service Component on page 62.
	Queue	A Message Destination that conforms to the point-to-point messaging paradigm, having one sender and one receiver. See the <i>eGate Integrator JMS Reference Guide</i> for information.
	Topic	A Message Destination that conforms to the publish/subscribe messaging paradigm, having one sender (publisher) and multiple receivers (subscribers). See the <i>eGate Integrator JMS Reference Guide</i> for information.
	Web Service External Application	Represents a third-party Web service application external to eGate. See Example Web Services Project on page 73.
	External Applications	Represents an application external to eGate. Click the arrow beside the icon to view a list of specific applications to which you can connect. See External Application Drop-Down Menu on page 62.
	File	Represents an external file. See the <i>File eWay Intelligent Adapter User's Guide</i> for more information about the eWay used to read or write the file.
	Scheduler	Represents a scheduling component of the Connectivity Map. Use this component to set data transfer to occur at set intervals. See Scheduler Component on page 63.
	Work List Viewer	This icon is present only when eInsight Business Process Manager is installed. See the <i>eInsight Business Process Manager User's Guide</i> for information.

It is important to understand that the logical components appearing in the Connectivity Map are essentially *placeholders* that refer to the “actual” components that exist in the Repository and appear in the Project Explorer. Renaming or deleting a queue or topic in the Connectivity Map only affects the placeholder, not the object in the Repository.

Also, renaming or deleting a queue or topic in the Repository will not affect the existence or name of the associated placeholder in the Connectivity Map. The change will, however, be reflected in the *tooltips* for the placeholder. This allows you to re-assign the placeholder without disrupting the continuity of the Connectivity Map.

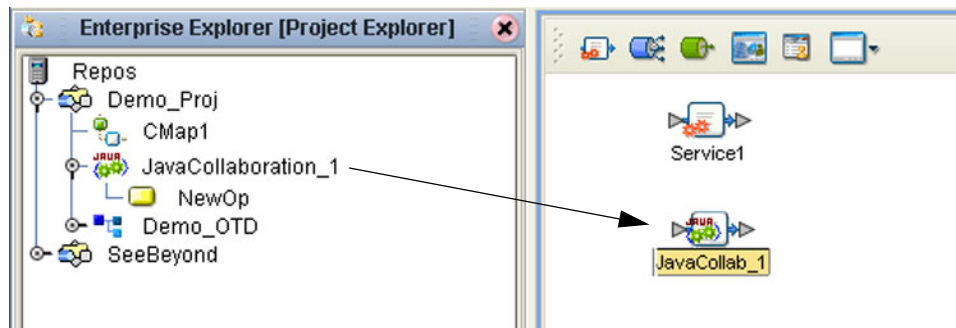
4.6 Creating and Configuring Logical Components

The drag-and-drop components include Services, queues, topics, schedulers, and external applications. Additional components, such as eWays and JMS Clients, are placed automatically when you link the components you have placed manually.

4.6.1 Services

A Service provides a framework for a process or a Collaboration, which contains the information required to execute a set of business rules. Dragging a Collaboration from the Project Explorer to the Service icon in the Connectivity Map defines the Service as a Collaboration.

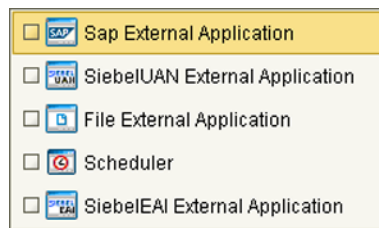
Figure 28 Service Component



4.6.2 External Applications

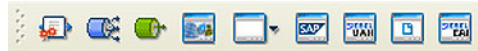
External Applications are logical representations of external software applications that are being integrated by the eGate system. These are linked to a Service by means of an eWay. Clicking the drop-down arrow beside the External Application icon displays a menu showing those applications corresponding to eWays that have been purchased and installed, plus the Scheduler (see Figure 29).

Figure 29 External Application Drop-Down Menu



Checking the individual External Applications adds the selected icon to the toolbar (see Figure 30).

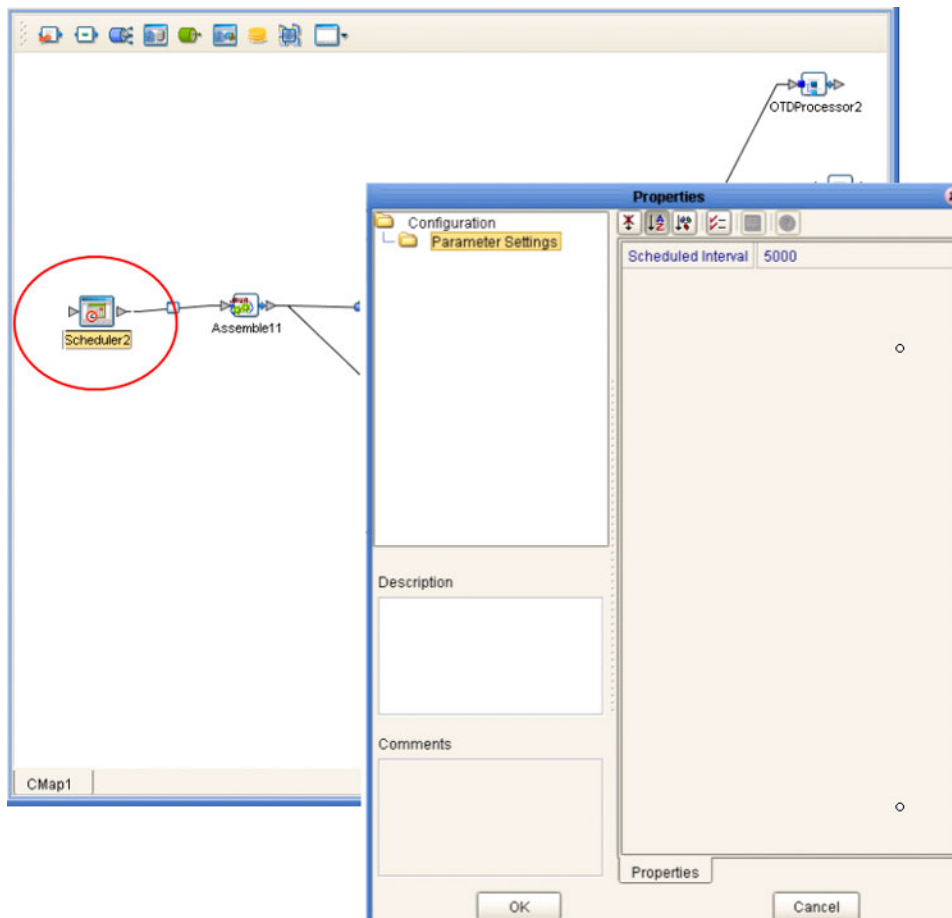
Figure 30 Selected External Applications in Toolbar



4.6.3 Schedulers

A Scheduler allows a service to be performed at a prescribed interval. The interval can be static, or can be made dynamic by using a Project variable for the interval value.

Figure 31 Scheduler Component

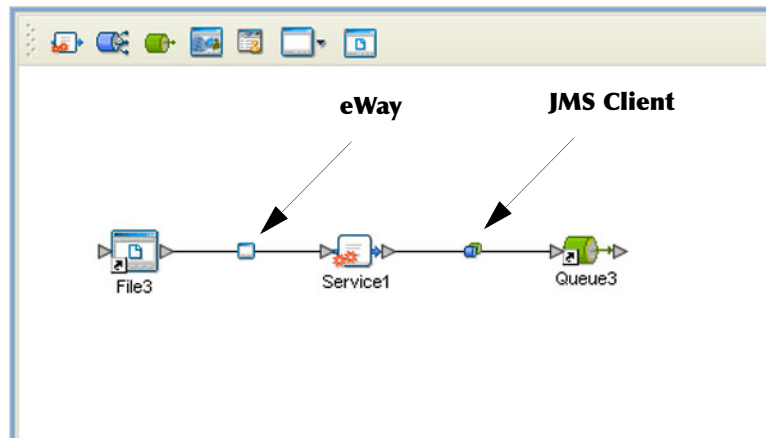


As an example, depicted in Figure 31, a scheduler can be used to invoke a Collaboration with an interval of 5 seconds (5000 milliseconds).

4.7 Component Connections

When you link two components on a Connectivity Map, the Enterprise Designer places either an eWay or JMS Client connection icon on the link, depending upon the type of components you are linking (see Figure 32).

Figure 32 Connection Icons in a Connectivity Map



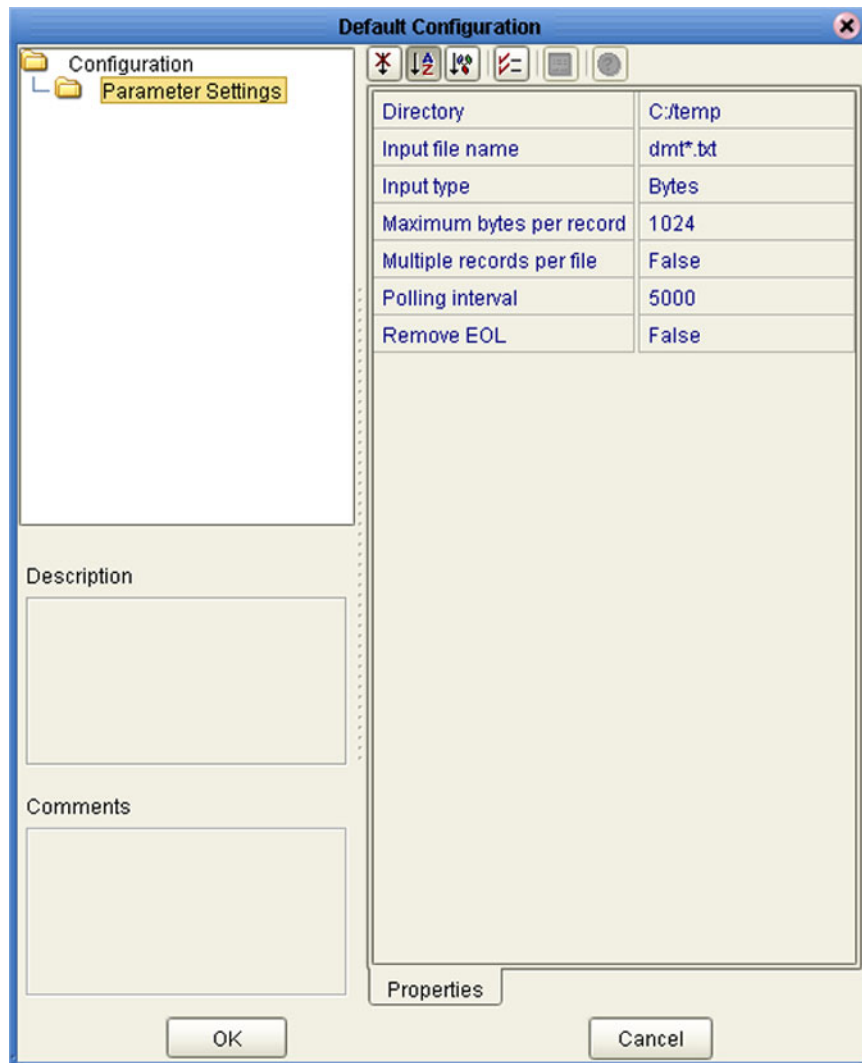
- When you link an external application with a Collaboration, the Enterprise Designer automatically adds an eWay Connection icon to the link. The eWay enables communication and movement of data between the external application and the eGate system. The eWay configuration specifies the logical connection properties for the link. See the individual eWay Intelligent Adapter User's Guides for specific information.
- When you link a Service with a Message Destination (queue or topic), the Enterprise Designer adds a JMS Client Connection icon. The JMS Client configuration specifies the logical connection properties for the linked Service. See the *eGate Integrator JMS Reference Guide* for information.

Note: You must configure the JMS properties, since no default values are assigned. Failure to do so will result in an error at a later time.

4.7.1 Configuring a Connection

Double-clicking an eWay or JMS Client properties handle icon in the Connectivity Map displays the Default Configuration dialog box. As an example, Figure 33 shows a dialog box that lists the configuration properties for a File eWay.

Figure 33 Default Configuration Dialog Box









Note: The first time you double-click an eWay or JMS Client icon, you will see a Templates dialog box. Here, you must designate an eWay to be inbound or outbound, or a Service to be a publisher or subscriber. Clicking OK will then display the Default Configuration dialog box.

The constituent parts of the Default Configuration dialog box are:

- The **Configuration Tree** includes folders that contain configuration and connection properties for the selected eWay or message destination.
- The **Editor Toolbar** contains a series of buttons used to sort and modify the information listed in the Properties folder, as described in Table 18.
- The **Properties** folder lists the default properties for the selected eWay or JMS Client. See, respectively, the individual eWay Intelligent Adapter User’s Guides and the *eGate Integrator JMS Reference Guide* for information.
- The **Description** box contains a brief description of the contents of the item currently selected in the Configuration Tree.
- The **Comments** box lists additional information about the item selected in the Configuration Tree.

Table 18 Default Configuration Toolbar Buttons

Button	Command	Function
	Unsorted	Displays configuration properties in their default order.
	Sort by Name	Sorts configuration properties alphabetically by name.
	Sort by Type	Displays configuration properties by property type.
	Show Editable Properties Only	Displays only the properties of an eWay or message destination that can be modified.
	Customizer	Displays the Customizer dialog box, which you can use to customize the selected eWay or message destination.
	Help	Displays the online help documentation for the Configuration Editor.

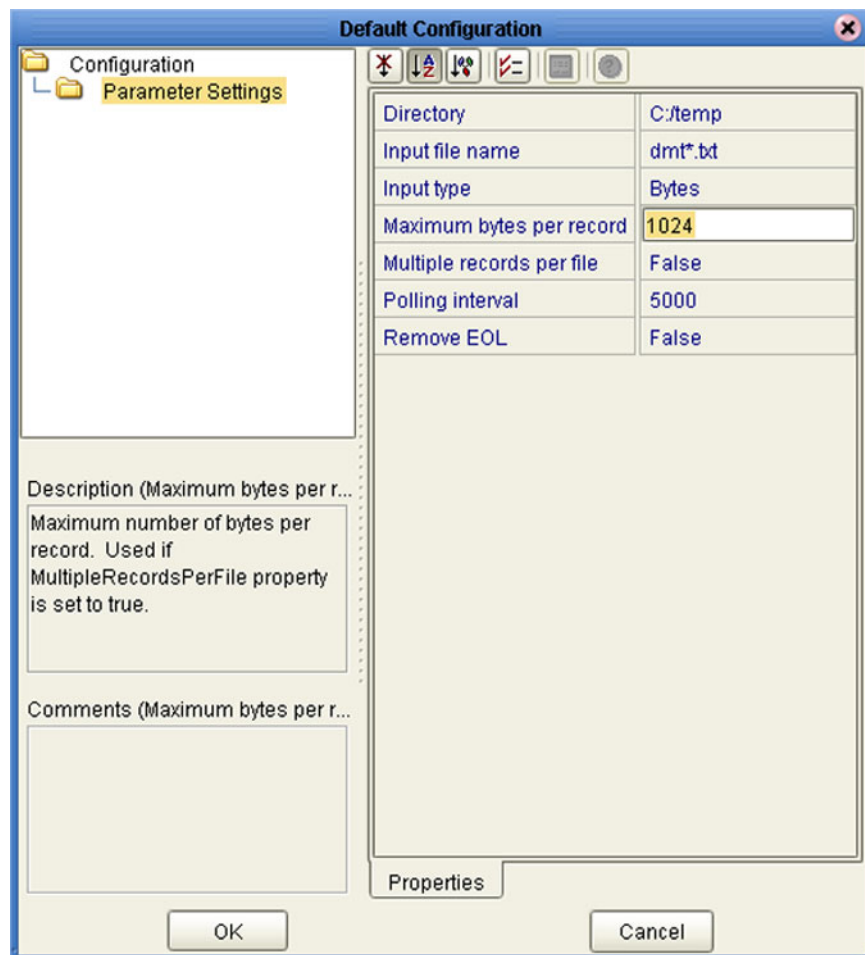
4.7.2 Modifying a Configuration Property

The properties initially listed in the Configuration Properties folder are the default settings for the selected eWay or JMS Client. This section describes how to change the default configuration values, using an inbound eWay as an example.

To change a configuration property

- 1 Double-click on an eWay or JMS Client properties handle icon in a Connectivity Map to display the Default Configuration dialog box.
- 2 Click on the right column of a row to place the row in edit mode. For example, click on the *Maximum bytes per record* field as shown in Figure 34, which highlights the current value.

Figure 34 Editing a Default Configuration Field



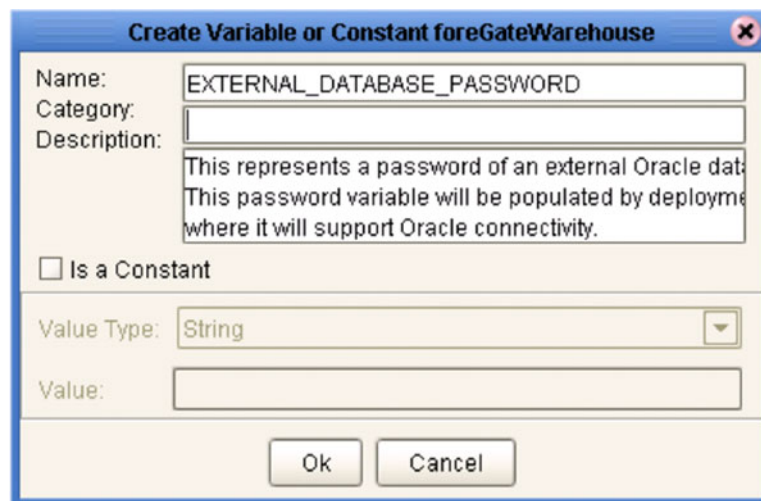
- 3 Enter a new value in the field. (Note that in this example, the parameter is not used unless you allow multiple records per file, as explained in the Description box).
- 4 Click **OK** to save the new settings.

4.8 Defining Constants and Variables

You can define variables and constants for a specific Project. Variables function as placeholders, having values that are determined when you create a specific Deployment Profile (see [Mapping Variables](#) on page 210). Project variable values can be literals or Environmental constants.

For example, a project variable is defined to represent a password of a database user in a target environment (see Figure 35). System managers will assign an actual value to this variable in the deployment profile editor. The value of the assigned project variable—an Environment constant—is then used to connect the database in the target environment.

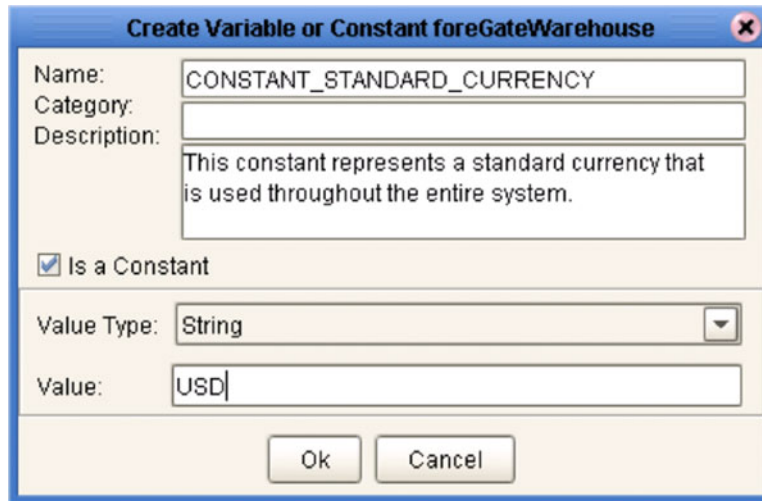
Figure 35 Project Variable Creation



The screenshot shows a dialog box titled "Create Variable or Constant foreGateWarehouse". The "Name" field contains "EXTERNAL_DATABASE_PASSWORD". The "Description" field contains the text: "This represents a password of an external Oracle dat", "This password variable will be populated by deployme", and "where it will support Oracle connectivity.". The "Is a Constant" checkbox is unchecked. The "Value Type" dropdown is set to "String". The "Value" field is empty. The dialog has "Ok" and "Cancel" buttons at the bottom.

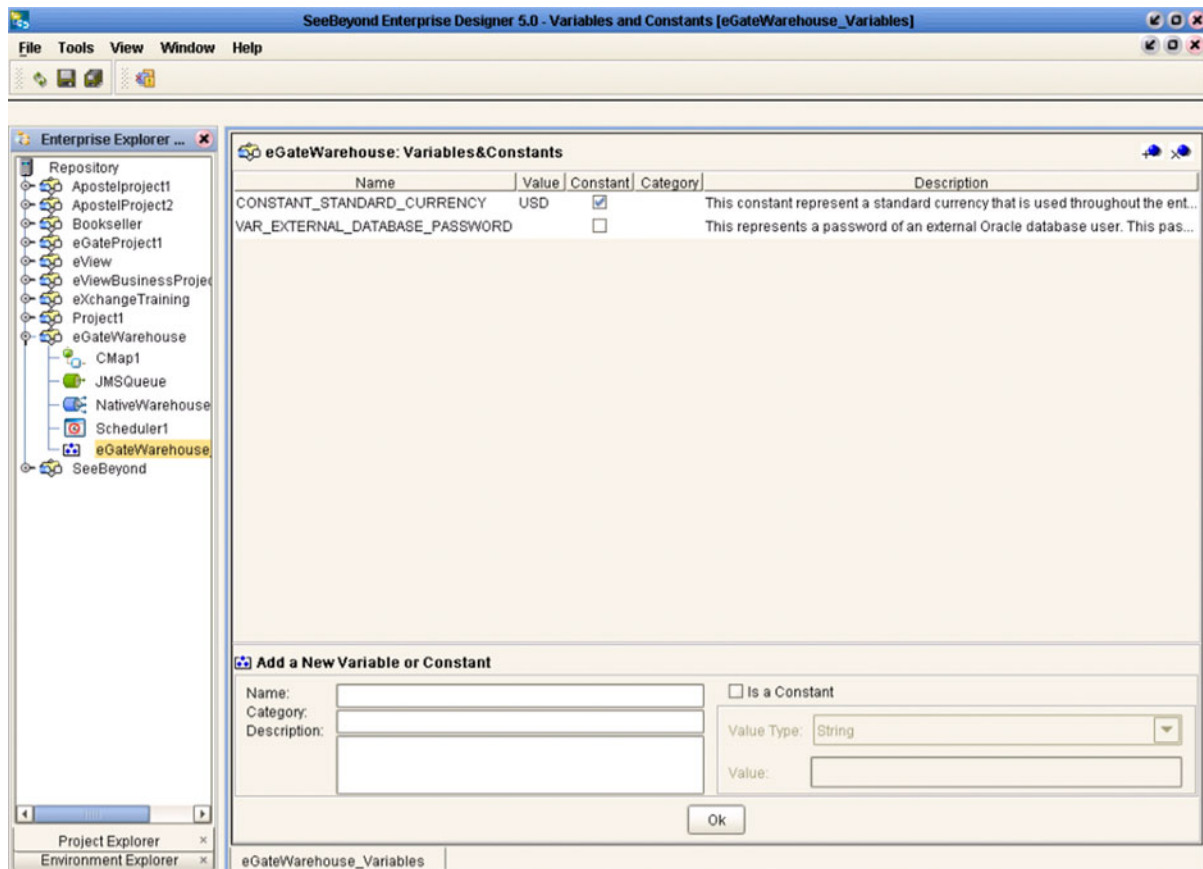
A constant is a name-value pair; when you create a constant you assign a permanent value to it, which cannot be overridden. An example of this would be a standard currency used globally throughout the system (see Figure 36).

Figure 36 Project Constant Creation



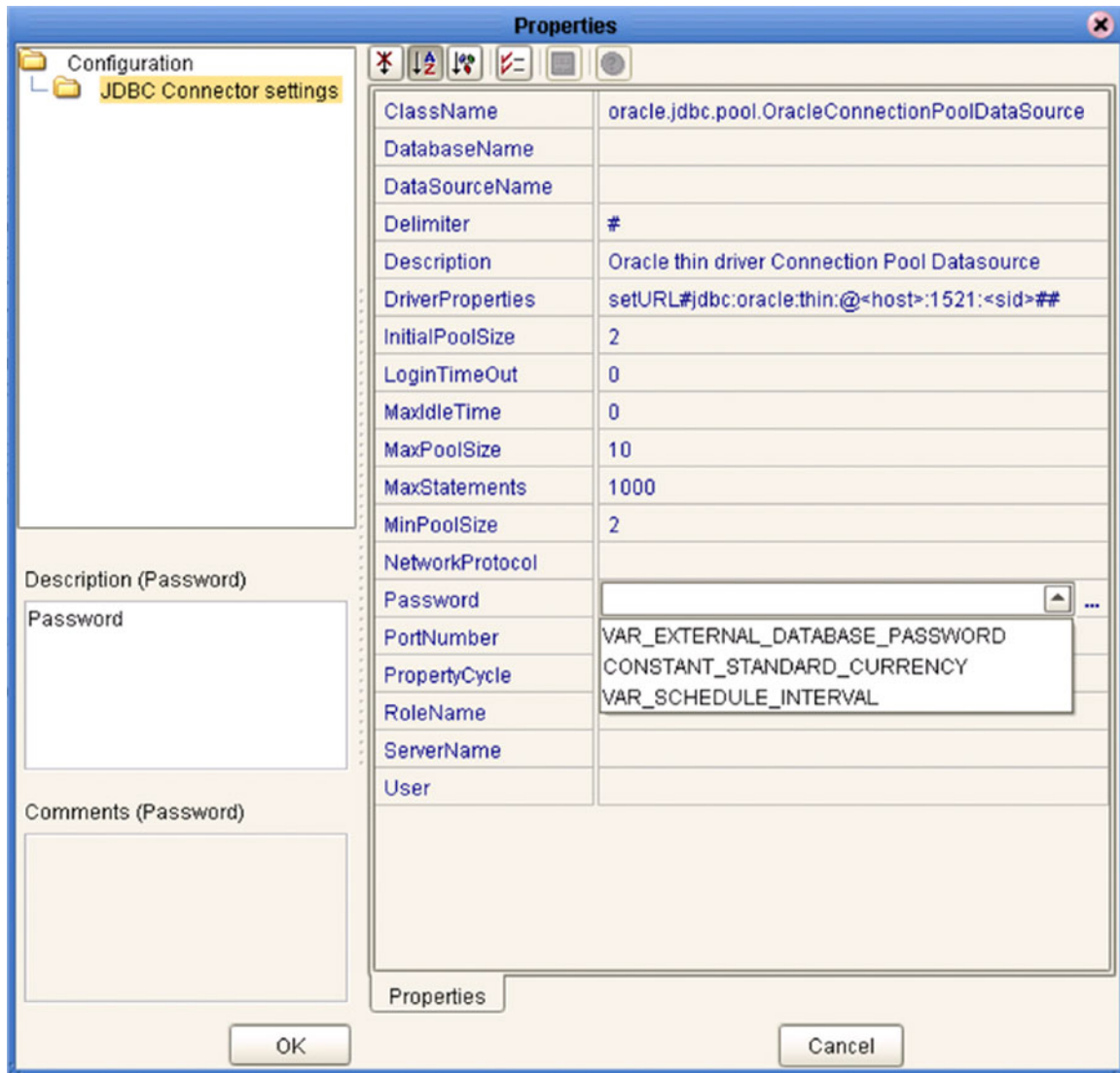
These constants and variables are automatically added to a Variables and Constants object group within the Project (see Figure 37).

Figure 37 Variables and Constants Object Group



The Project variables and constants can be referenced as properties within the Project. For example, the password variable described previously can be selected from the Variables and Constants object group to provide the Password property in the JDBC Connector settings dialog box (see Figure 38).

Figure 38 Connector Properties



See [Mapping Variables](#) on page 210 for information on setting the values for Project variables.

4.9 Web Services

In general, all ICAN objects that expose themselves as a Web Service (such as an eInsight business process) are presented in the SeeBeyond UDDI Repository (see Figure 39). The URL of this repository is:

```
http://ICAN_Suite_host_name:enterprise_manager_installation_port/  
stcuddi
```

Figure 39 SeeBeyond UDDI Repository



Environment	Service Name	WSDL
Environment2	BusinessProcess1	http://art2k:10000/repository/MyRepository/data/uddocs/Environment2/BusinessProcess1/BusinessProcess1.wsdl
Environment2	BusinessProcess1	http://art2k:10000/repository/MyRepository/data/uddocs/Environment2/BusinessProcess1/BusinessProcess2.wsdl

Each entry in the UDDI Repository includes:

- The ICAN environment name.
- The actual (Web) Services name.
- The location of the Web Service’s WSDL file.

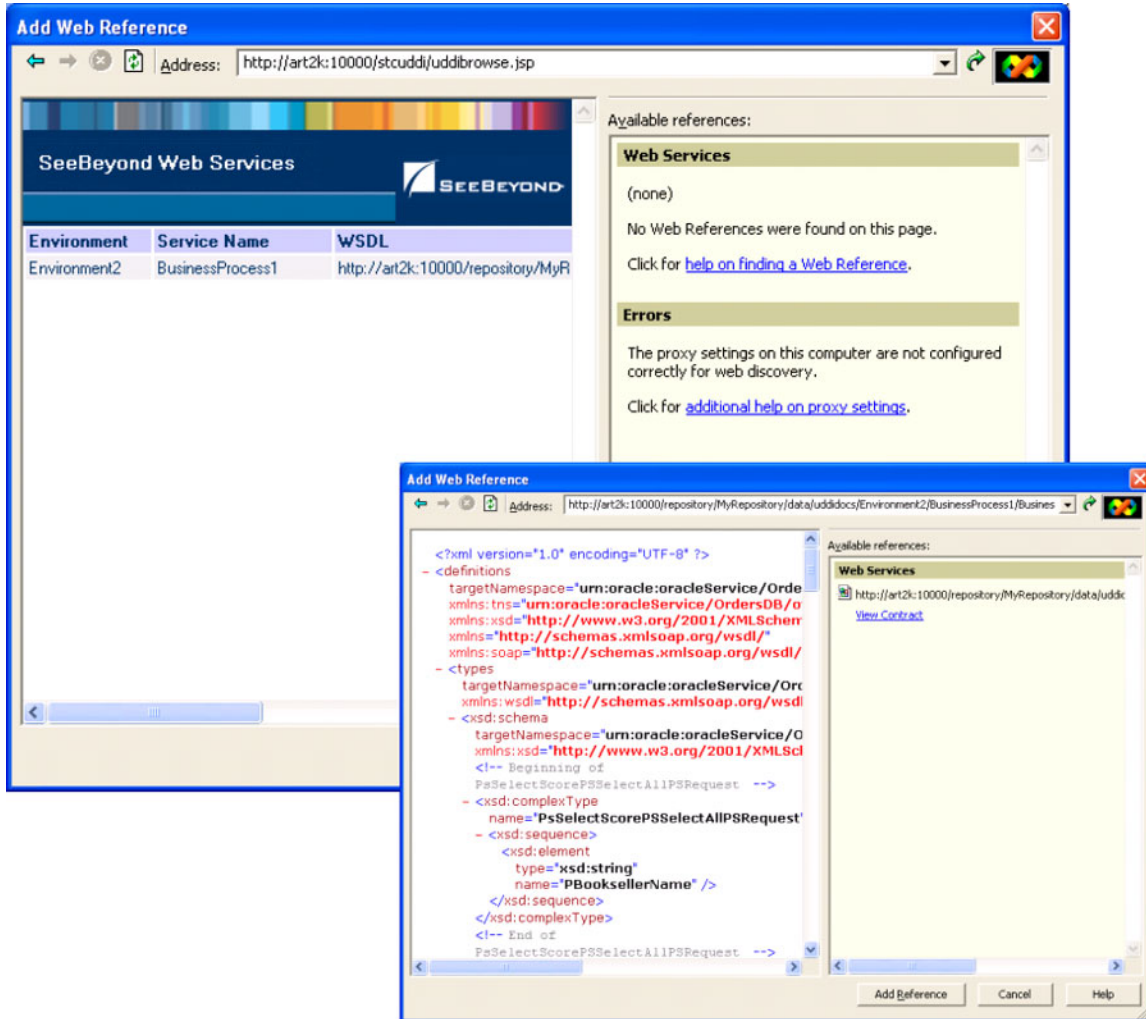
By selecting an entry its WSDL file is displayed, as shown in Figure 40.

Figure 40 Example Web Service WSDL File

```
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions targetNamespace="urn:oracle:oracleService/OrdersDB/otdGetCreditScore" xmlns:tns="urn:oracle:oracleService/OrdersDB/otdGetCreditScore"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
- <types targetNamespace="urn:oracle:oracleService/OrdersDB/otdGetCreditScore" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/">
- <xsd:schema targetNamespace="urn:oracle:oracleService/OrdersDB/otdGetCreditScore" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <!-- Beginning of PsSelectScorePSSelectAllPSRequest -->
- <xsd:complexType name="PsSelectScorePSSelectAllPSRequest">
- <xsd:sequence>
  <xsd:element type="xsd:string" name="PBooksellerName" />
</xsd:sequence>
</xsd:complexType>
<!-- End of PsSelectScorePSSelectAllPSRequest -->
<!-- Beginning of PsSelectScorePSSelectOnePSResponseType -->
- <xsd:complexType name="PsSelectScorePSSelectOnePSResponseType">
- <xsd:sequence>
  <xsd:element type="xsd:decimal" name="CREDIT_SCORE" />
</xsd:sequence>
</xsd:complexType>
<!-- End of PsSelectScorePSSelectOnePSResponseType -->
<!-- Beginning of PsSelectScorePSSelectMultiplePSResponseType -->
- <xsd:complexType name="PsSelectScorePSSelectMultiplePSResponseType">
- <xsd:sequence>
  <xsd:element type="xsd:long" name="rowCount" />
  <xsd:element type="xsd:decimal" name="CREDIT_SCORE" />
</xsd:sequence>
</xsd:complexType>
<!-- End of PsSelectScorePSSelectMultiplePSResponseType -->
<!-- Beginning of PsSelectScorePSSelectAllPSResponseType -->
- <xsd:complexType name="PsSelectScorePSSelectAllPSResponseType">
- <xsd:sequence>
  <xsd:element type="xsd:long" name="rowCount" />
```

The SeeBeyond UDDI Repository can be used in a third party tool, for example Microsoft Visual Studio (see Figure 41). In this example, a so-called *Web reference* (to the UDDI repository) is added to a C# project.

Figure 41 Microsoft Visual Studio Example



eGate Integrator can exchange data with Internet and Web Services applications using the Web Services Description Language (WSDL). This language is XML-based and is used to define Web services and describe how to access them. The WSDL OTD Wizard is used to build OTDs that are used in the Project Collaborations (see [WSDL-Based OTDs](#) on page 99).

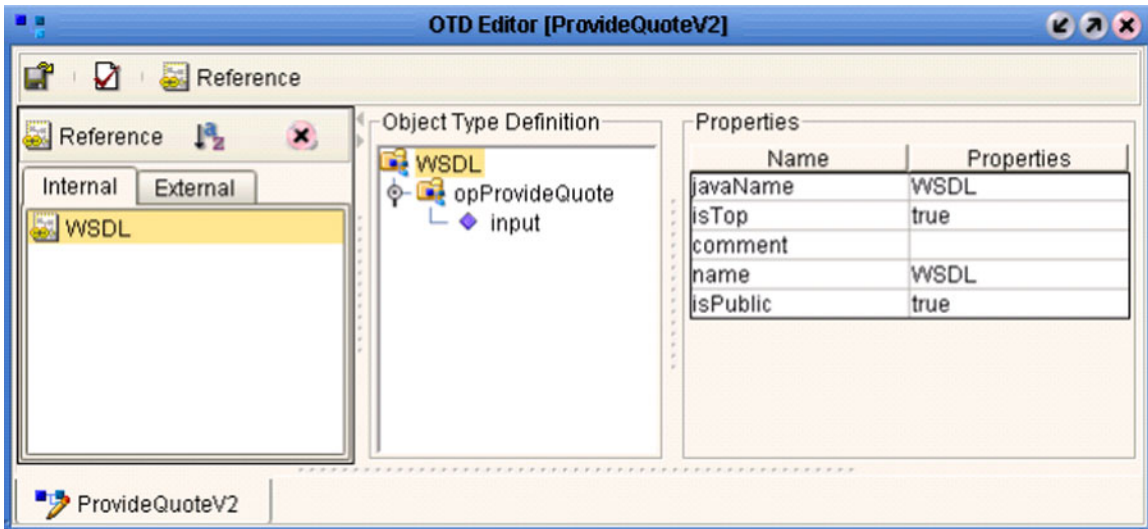
4.9.1 Example Web Services Project

As an illustrative example, we will show a project called "prgProvideStockQuote".

Creating the OTDs

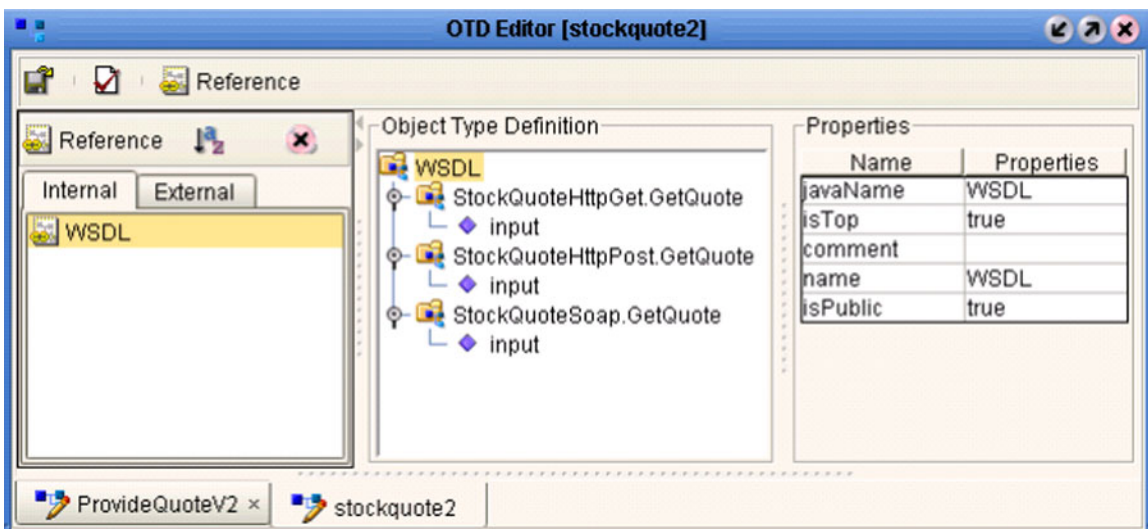
Three OTDs are required for this Project. First is an OTD based on the WSDL called **ProvideQuoteV2.wsdl** (see Figure 42).

Figure 42 Web Service Example (OTD 1)



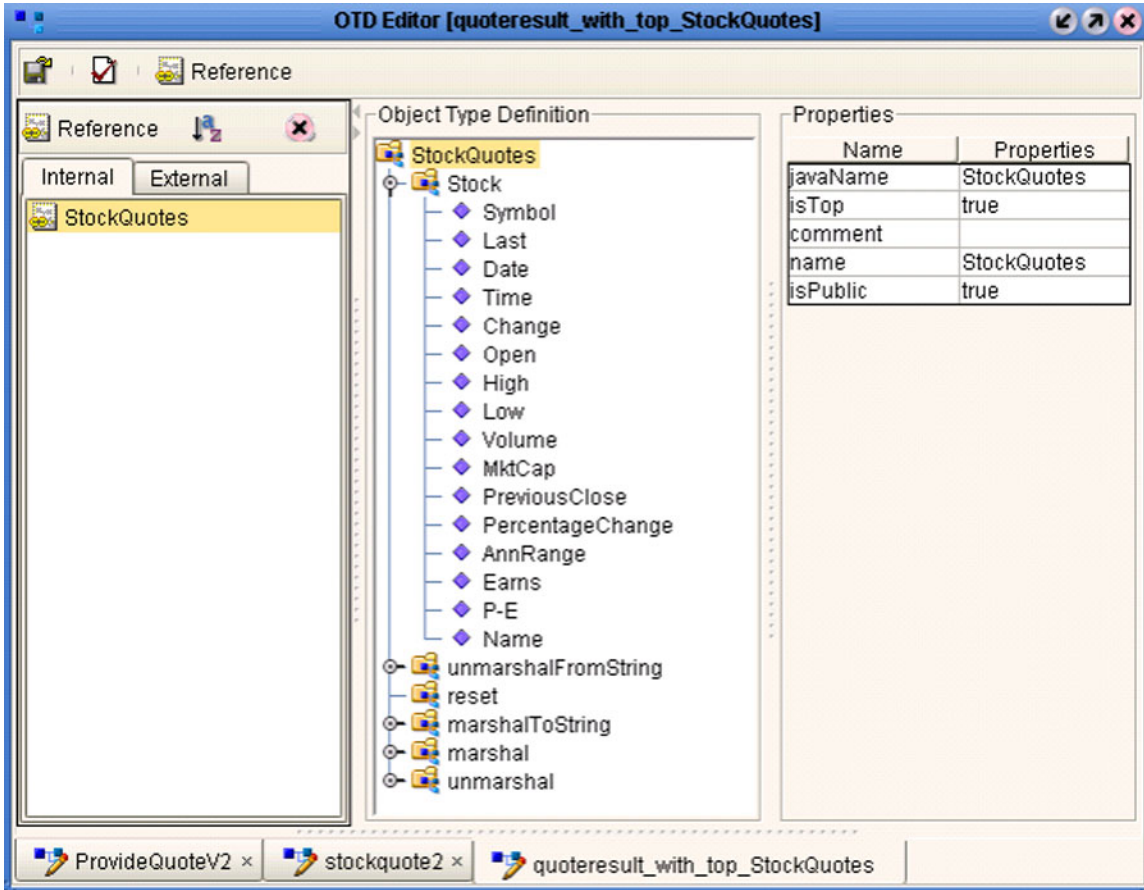
Second is an OTD based on the WSDL for calling the external stock quote Web Service, based on the **stockquote2.wsdl** (see Figure 43).

Figure 43 Web Service Example (OTD 2)



Third is an OTD for unmarshaling the result we'll be receiving from the external Web Service. The result is in XML and a DTD file has been provided for the Web Service called **quoteresult.dtd** (see Figure 44).

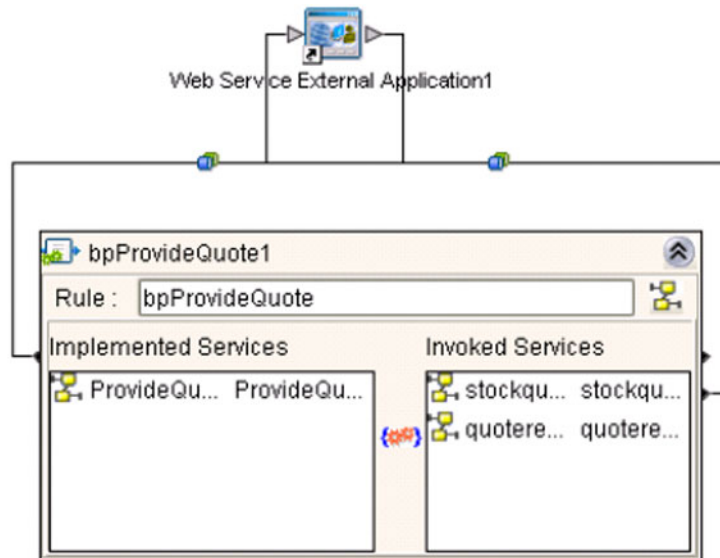
Figure 44 Web Service Example (OTD 3)



Using the Connectivity Map

In this example, an eInsight business process is imported into the eGate Connectivity Map, and a Web Service External Application is added to the business process (see Figure 45).

Figure 45 Web Service Example (Connectivity Map)



The resulting WSDL file is shown in Figure 46.

Figure 46 Web Service Example (WSDL File)

```
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions targetNamespace="http://seebeyond/quoteservice" xmlns:tns="http://seebeyond/quoteservice"
  xmlns:sbypnx="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
- <types targetNamespace="http://seebeyond/quoteservice">
  - <xsd:schema targetNamespace="http://seebeyond/quoteservice"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <xsd:element type="xsd:string" name="QuoteRequest" />
    <xsd:element type="xsd:string" name="QuotePrice" />
    <xsd:element type="xsd:string" name="QuoteCompany" />
    </xsd:schema>
  </types>
- <message name="msgProvideQuoteRequest">
  <part name="TICKER" element="tns:QuoteRequest" />
</message>
- <message name="msgQuoteResponse">
  <part name="price" element="tns:QuotePrice" />
  <part name="company" element="tns:QuoteCompany" />
</message>
- <portType name="ptProvideQuote">
  - <operation name="opProvideQuote">
    <input message="tns:msgProvideQuoteRequest" />
    <output message="tns:msgQuoteResponse" />
  </operation>
</portType>
- <binding name="ptProvideQuoteBinding" type="tns:ptProvideQuote">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http" />
  - <operation name="opProvideQuote">
    <soap:operation soapAction="{http://seebeyond/quoteservice}ptProvideQuote::opProvideQuote" />
    - <input>
      <soap:body use="literal" />
    </input>
    - <output>
      <soap:body use="literal" />
    </output>
  </operation>
</binding>
- <service name="bpProvideQuoteService">
  - <port name="ptProvideQuoteBindingService" binding="tns:ptProvideQuoteBinding">
    <soap:address location="http://localhost:18003/WebServiceExt/service/ptProvideQuote" />
  </port>
</service>
</definitions>
```

Upon activation, the deployed WSDL file is displayed in the UDDI Repository as shown in Figure 47.

Figure 47 Web Service Example (UDDI Repository)

Environment	Service Name	WSDL
Environment1	bpProvideQuote	http://localhost:12000/repository/Repository/data/uddidocs/Environment1/bpProvideQuote/bpProvideQuote.wsdl

Object Type Definitions

This chapter describes the OTD creation process. The Enterprise Designer includes two tools, the OTD Wizard and OTD Editor, to help you create and customize OTDs.

5.1 Overview

An Object Type Definition (OTD) is a description of a complex hierarchical data structure that can be accessed and manipulated by your code in a Collaboration. OTDs typically have a specific external representation format that is used to store and transport the OTD contents through the parts of a eGate Project. The OTD defines both the run-time structure and the external representation.

At run time, an OTD instance is accessed either directly from Java in a Java Collaboration, using the so-called *bean-like* accessors, or from BPEL using XPath expressions. In Java, the nodes comprising the hierarchy of the data structure have an interface similar to Java beans: each node has a set of properties with *get* and *set* methods to manipulate them (see [The Bean-like Interface](#) on page 77). There is also a so-called generic interface (see [The Generic Interface](#) on page 114).

Typically, a collaboration will receive a message containing the external representation of a particular OTD. It will use the *unmarshal* method of an instance of that OTD to parse the data and make it accessible through the hierarchical data structure. Then it will perform some operation: for example, copying parts of the data to another OTD instance. Finally, it will invoke the *marshal* method on the other OTD instance to render the contents of its data structure as a single, serialized data stream for further transport.

5.1.1 The *Bean-like* Interface

The hierarchical data structure of an OTD is represented at run time by a set of generated Java classes. These classes follow the Java bean rule. They have a set of zero or more properties, each of which has a specific type and a given name, and may be optional and/or repeating. In contrast to regular bean properties, a OTD node property has two distinct names: a display name, that can be a virtually-arbitrary string, and a Java name that is the accessor basename.

For example, if a node has a property with Java name "X", then the implementing class for that node will have a method "getX". The Java name is normally derived from the display name, modified to suit the restrictions on Java identifiers, and supplied automatically by eGate.

5.1.2 OTD Types

Externally-Defined OTDs

Externally-defined OTDs are based on formats or standards external to eGate Integrator, such as Document Type Definition (DTD), Web Services Definition Language (WSDL), XML Schema Definition (XSD), and various proprietary formats such as SAP BAPI. Some of these OTDs are *messagable*, others are API-based. Externally defined OTDs are read-only.

User-Defined OTDs

User-defined OTD are native to eGate Integrator. You can create a User-defined OTD from scratch using the User-Defined OTD Wizard. User-defined OTDs are read/write—you can add or delete nodes or edit their properties, and you can work with internal and external templates.

5.1.3 Building OTDs

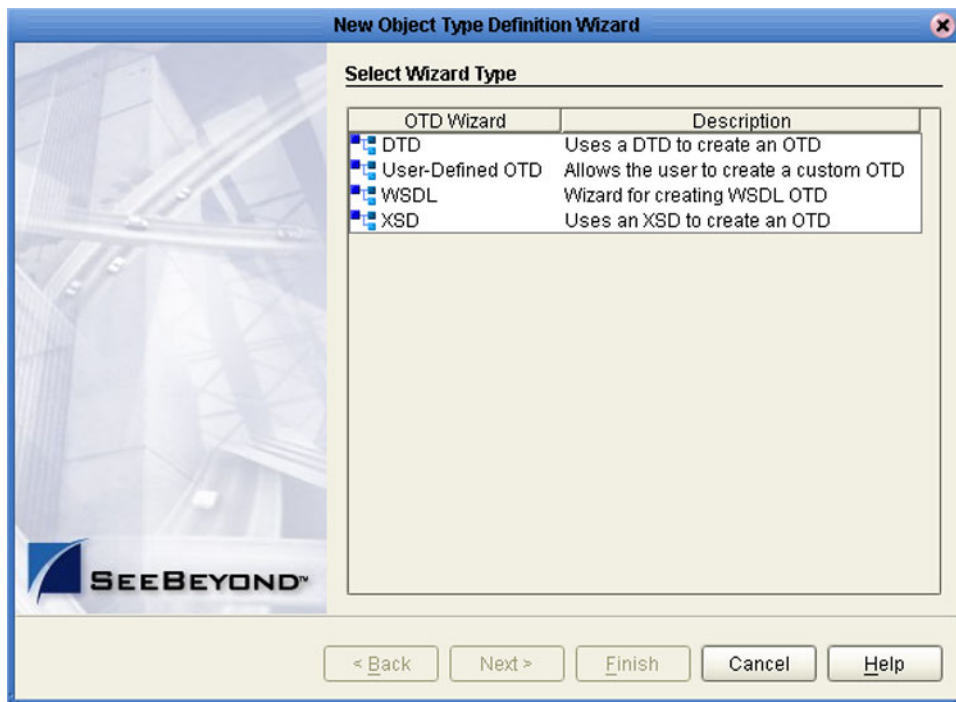
Wizards are provided in the Enterprise Designer GUI to guide you through the OTD building process. These Wizards call back-end builders that actually implement the building of the code, based on the provided information.

5.2 Using the OTD Wizard

Right-click on a Project in the Enterprise Explorer to display the Project context menu, then select **New Object Type Definition** to display the OTD Wizard, shown in Figure 48. The initial dialog allows you to select a specific OTD Wizard. These Wizards are respectively described in:

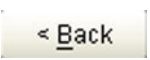




- [Using the DTD Wizard](#) on page 81
- [Using the User-Defined OTD Wizard](#) on page 85
- [Using the WSDL Wizard](#) on page 99
- [Using the XSD Wizard](#) on page 104

Figure 48 OTD Wizard Selection Dialog



5.2.1 Navigation Buttons

Table 19 OTD Wizard Navigation Buttons

Button	Function
	Returns to the previous step in the wizard. This button is disabled on the first step.
	Goes to the next step in the wizard. This button is disabled on the last step.
	Saves all OTD settings and closes the wizard. This button is only enabled on the last step.
	Closes the wizard without saving the OTD.
	Displays the online help documentation for the OTD Wizard dialog box.

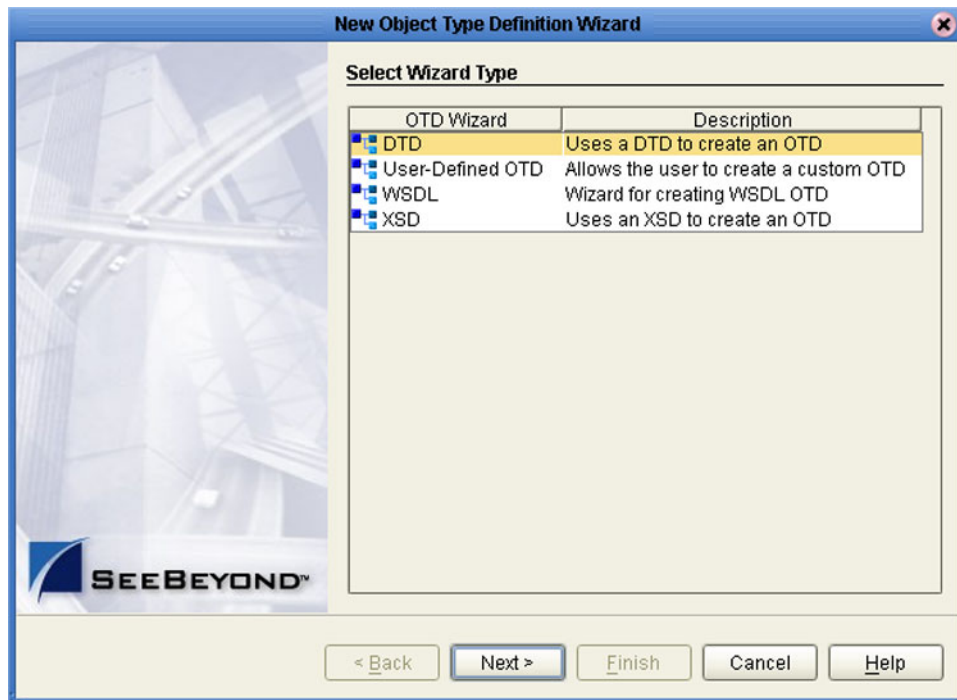
5.3 DTD-Based OTDs

5.3.1 Using the DTD Wizard

To create an OTD file from a DTD file

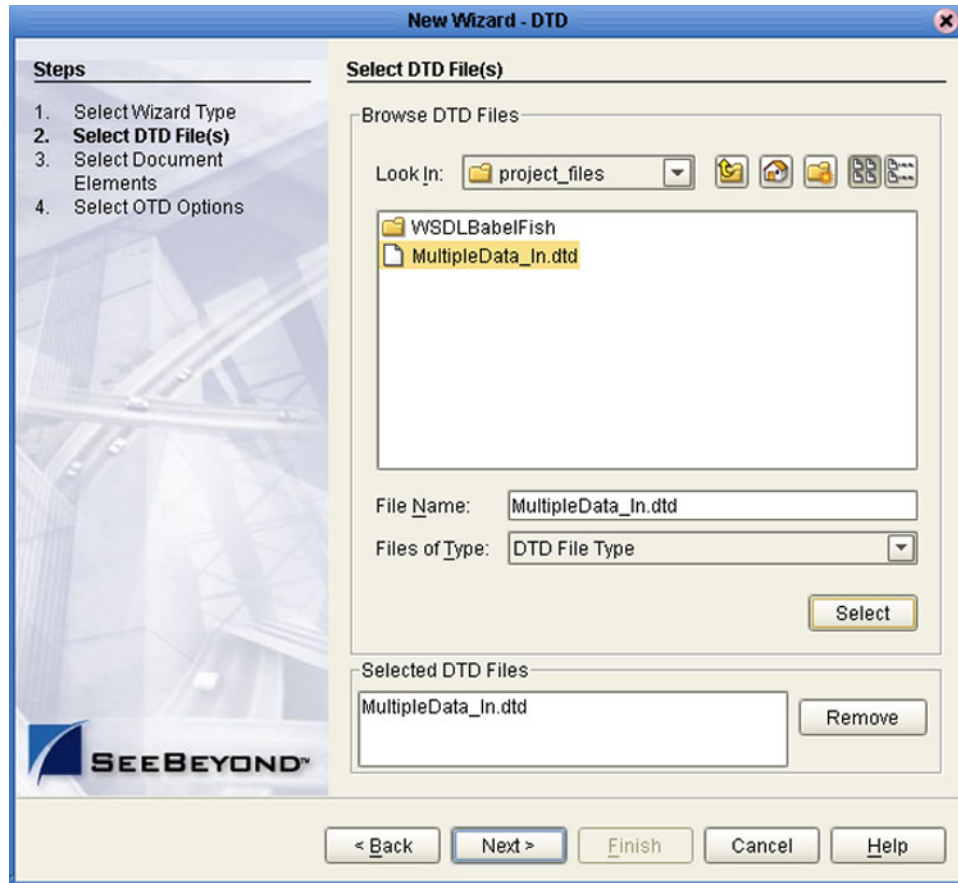
- 1 Select **DTD** from the *Select an Object Type Definition Wizard* list (see Figure 49) to create an OTD file from a Data Type Definition (DTD) file.

Figure 49 OTD Wizard Selection: DTD Wizard



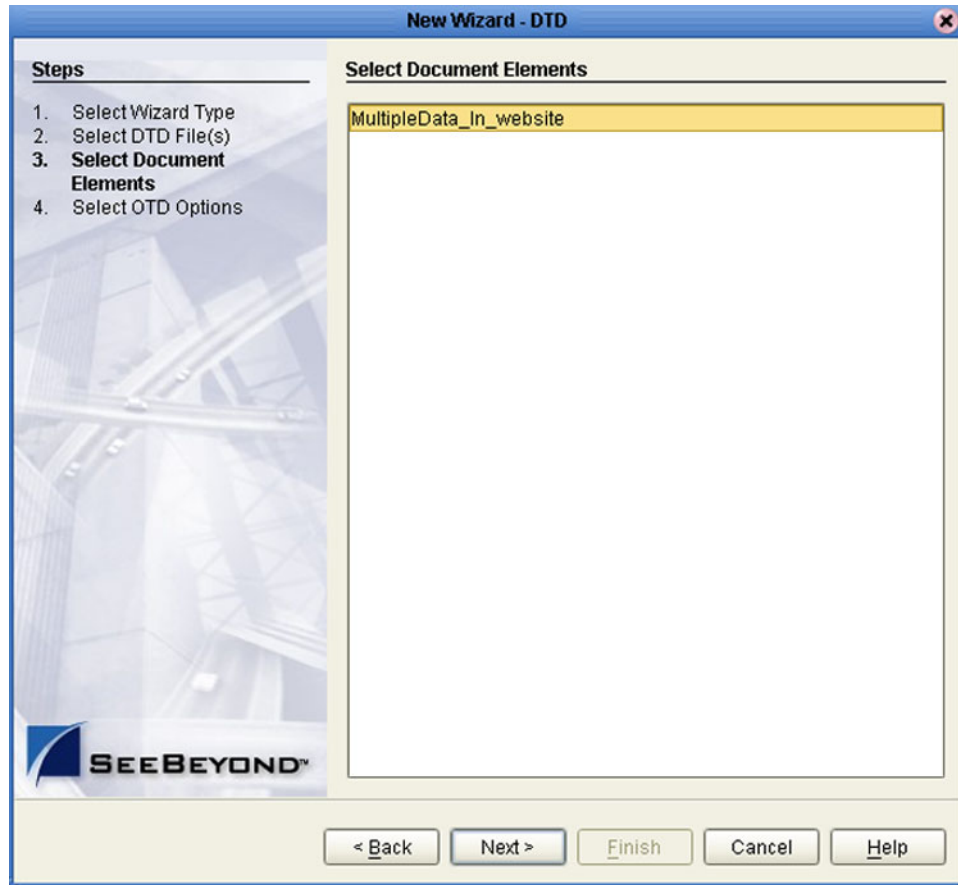
- 2 Click **Next** to display the Select DTD File(s) dialog box, shown in Figure 50.

Figure 50 Select DTD File(s) Dialog Box



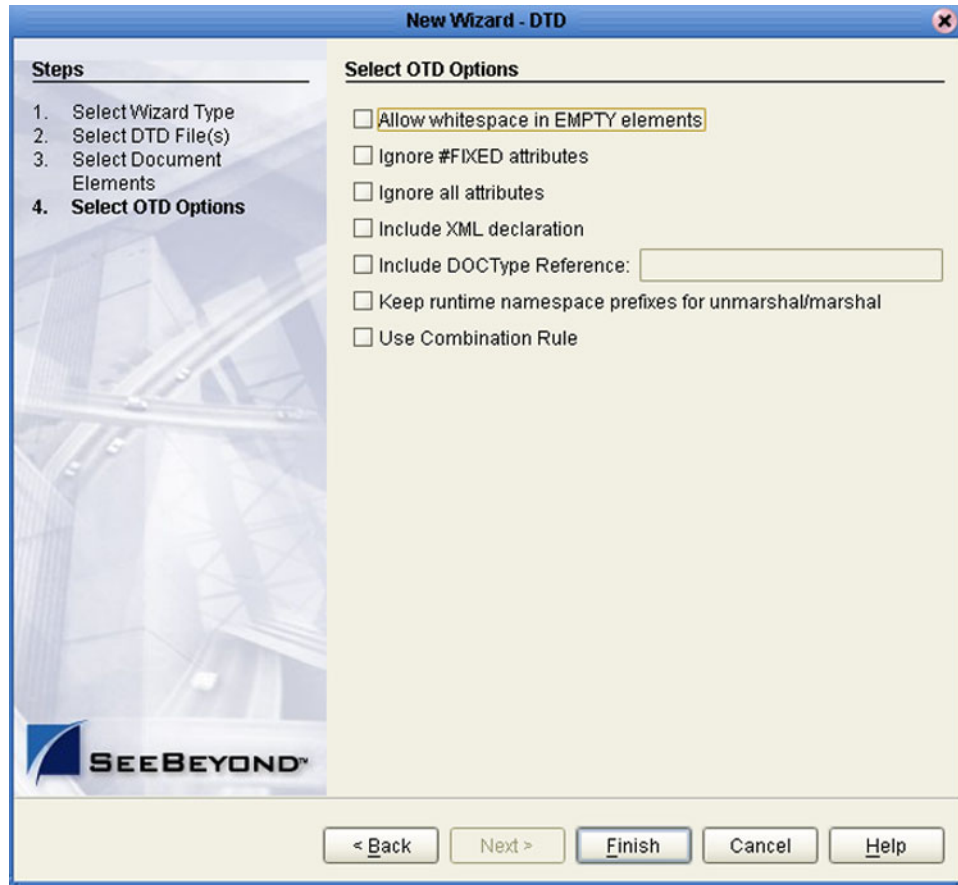
- 3 In the *Look in* drop-down list, navigate to the DTD file or files that you want to use to create the OTD. Click **Select** to add the files to the *List of Selected DTDs*.
- 4 Click **Next** to display the *Select Document Elements* dialog box, shown in Figure 51.

Figure 51 Select Document Elements Dialog Box



- 5 Select the elements of the document that you want to include in the OTD.
- 6 Click **Next** to display the *Select OTD Options* dialog box, shown in Figure 52.

Figure 52 Select OTD Options Dialog Box



- 7 Select the check boxes next to the OTD options you want to enable, and click **Finish** to add the OTD to the Enterprise Designer with the selected OTD options.

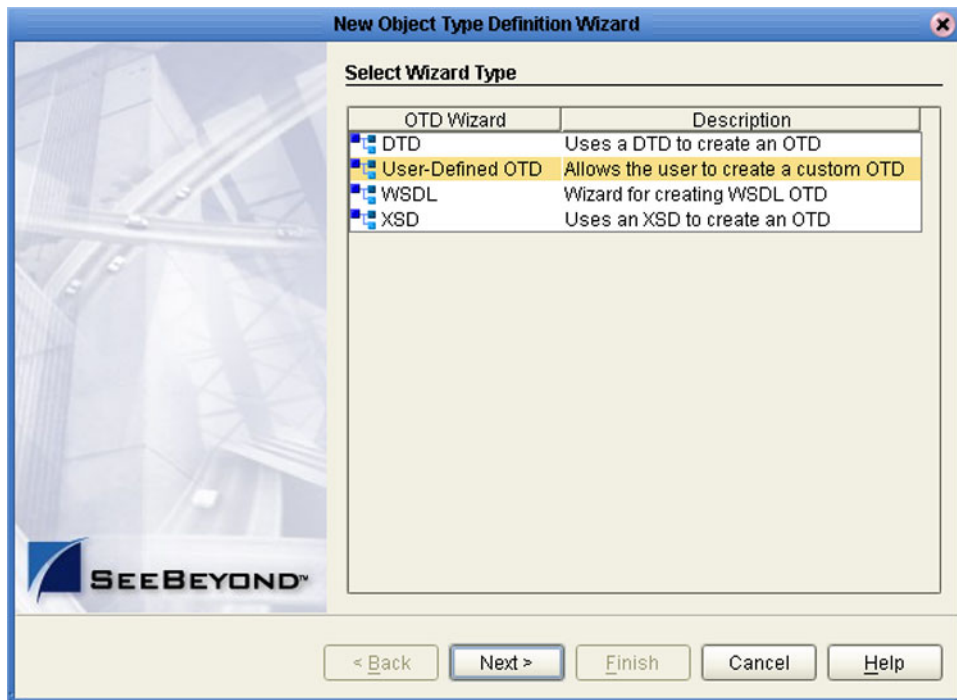
5.4 User-Defined OTDs

5.4.1 Using the User-Defined OTD Wizard

To create a User-Defined OTD

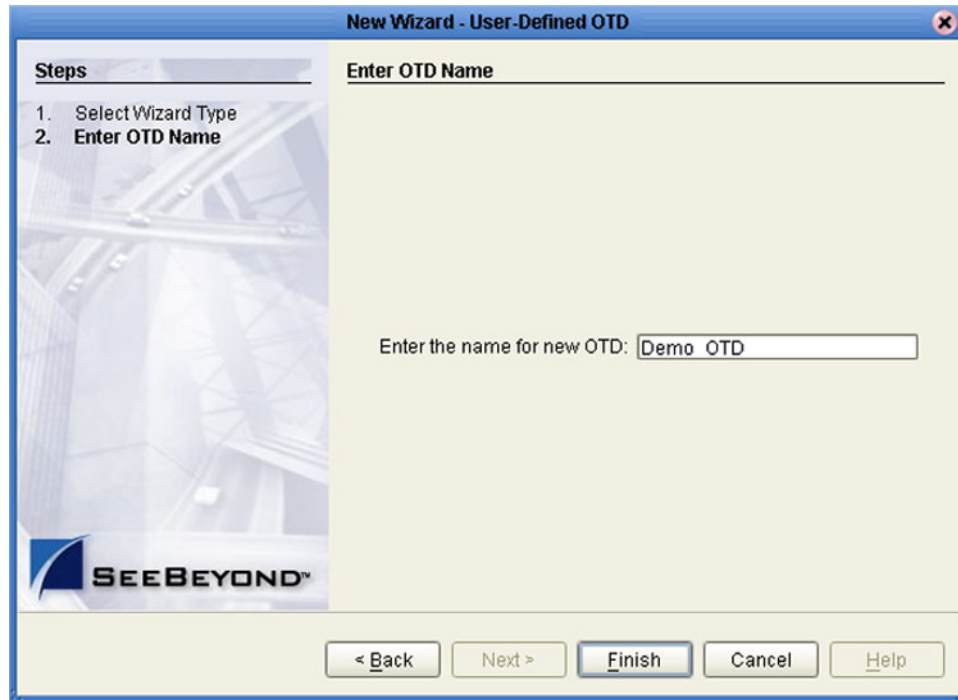
- 1 Select **User-Defined OTD** from the *Select an Object Type Definition Wizard* list (see Figure 53) to create an OTD file without using a source file.

Figure 53 OTD Wizard Selection: User-Defined OTD



- 2 Click **Next** to display the *Enter OTD Name* dialog box, shown in Figure 54.

Figure 54 Enter OTD Name



- 3 Enter a name for the OTD in the *Enter the name for new OTD* text box, then click **Finish** to add the OTD to the Enterprise Designer.

5.4.2 Editing the OTD Properties

Node Properties

Once a new User-Defined OTD has been created, the properties of the root node will be as shown in Figure 55.

Figure 55 User-Defined OTD Node Properties

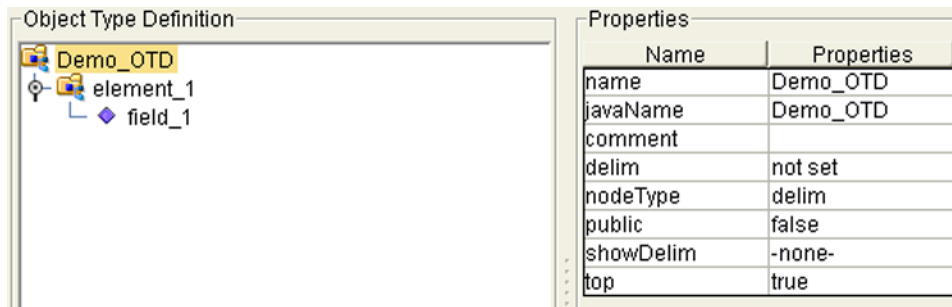


Table 20 Node Properties

Name	Description
name	Node display name (see The Bean-like Interface on page 77).
javaName	Property accessor basename (see The Bean-like Interface on page 77).
comment	Free-form text (no run-time effect).
delim	Delimiter specification (see Specifying Delimiters on page 91).
nodeType	Governs the marshal/unmarshal format (see Specifying the Node Type on page 90).
public	(for future use, not currently active)
showDelim	If nodeType is delimited,
top	Flag on root node: support marshal/unmarshal (T/F).

Important: Do not modify the javaName property.

Element Properties

The set of properties associated with the element level is shown in Figure 56.

Figure 56 User-Defined OTD Element Properties

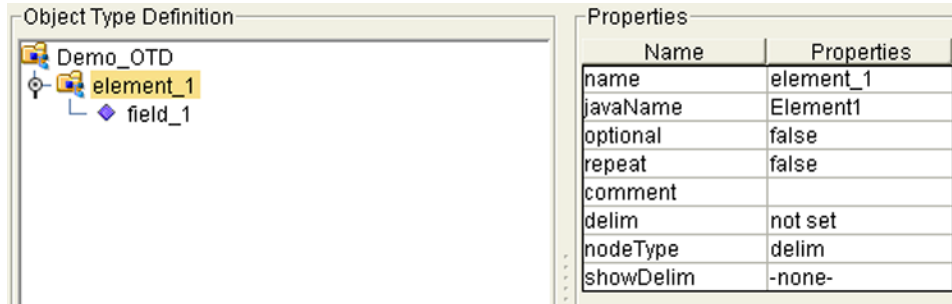


Table 21 Element Properties

Name	Description
name	Element display name.
javaName	Property accessor basename.
optional	Flag: Can the element be absent? (T/F) Not applicable to root, or child of a choice Node.
repeat	Flag: Can the node appear multiple times? (T/F) Not applicable to root, or child of a choice Node.
comment	Free-form text (no run-time effect).
delim	Delimiter specification (see Specifying Delimiters on page 91).
nodeType	Governs the marshal/unmarshal format.
showDelim	If nodeType is delimited,

Field Properties

The set of properties associated with the field level is shown in Figure 57.

Figure 57 User-Defined OTD Field Properties

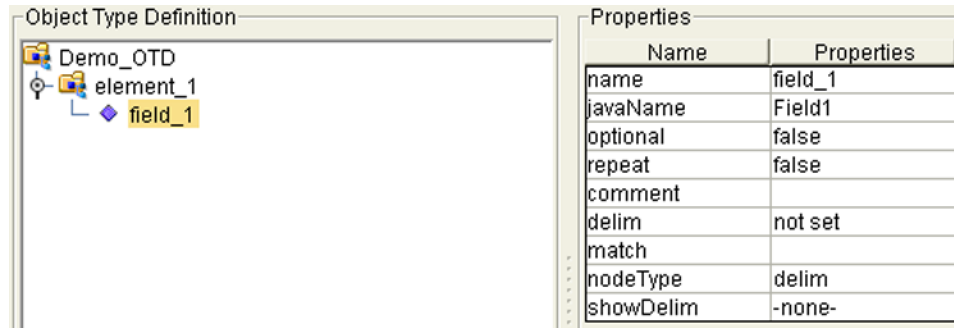


Table 22 Field Properties

Name	Description
name	Field display name.
javaName	Property accessor basename.
optional	Flag: Can the element be absent? (T/F) Not applicable to root, or child of a choice Element.
repeat	Flag: Can the node appear multiple times? (T/F) Not applicable to root, or child of a choice Element.
comment	Free-form text (no run-time effect).
delim	Delimiter specification (see Specifying Delimiters on page 91).
match	If nodeType is delimited, performs exact match to the data.
nodeType	Governs the marshal/unmarshal format.
showDelim	If nodeType is delimited,

5.4.3 Specifying the Node Type

Double-clicking in the nodeType properties field activates the field for editing. Click the button to display the drop-down menu (see Figure 58). Descriptions of the property options are listed in Table 23.

Figure 58 Node Type Property Options

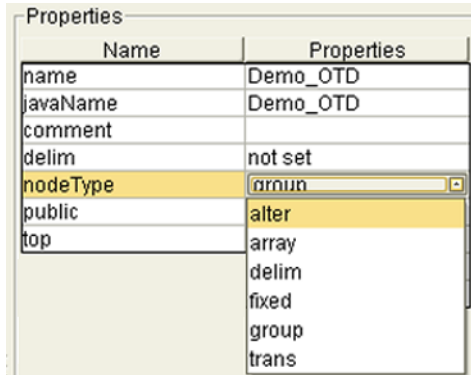


Table 23 Node Type Property Options

Option	Description	Element	Field	Internal
alter	Alternate ; repeating node, selects one child or the other. Always one child present after unmarshal operation. Applies to elements only.	Yes	No	choice
array	Delimited structure. If repeated, occurrences are separated by the <i>repeat</i> delimiter. The last occurrence may be terminated by a <i>normal</i> delimiter.	Yes	Yes	simple or group
delim	Delimited structure. Like array, but repeated occurrences are separated by a <i>normal</i> delimiter.	Yes	Yes	simple or group
fixed	Fixed length, specified by non-negative integer, or zero to indicate end of parent node data.	Yes	Yes	simple or group
group	Provides organizational grouping for purposes such as repetition. Only applies to elements, not fields.	Yes	No	group
trans	Transient ; appears only in internal tree as a scratch pad field (does not appear in external data representation). Can only have trans nodeTypes as children.	Yes	Yes	choice, simple, or group

5.4.4 Specifying Delimiters

Any node can define a set of delimiters to be used in the external data representation for itself and its descendents in the hierarchical data structure. If a node defines a delimiter list, this negates any effect of any ancestor's delimiter list on itself and its descendents. We typically specify the list on the root node.

For example, if you want to parse the following data:

```
a^b|c^d|e
```

you might create a User-Defined OTD as follows:

- demo-otd
 - ♦ element1
 - ♦ field1
 - ♦ field2
 - ♦ element2
 - ♦ field3
 - ♦ field4
 - ♦ field5

The delimiter list for this OTD will be specified on the *demo-otd* element, so that it applies to the entire OTD, and will have two levels:

- Level 1
 - ♦ Delimiter |
- Level 2
 - ♦ Delimiter ^

Level 1's delimiter applies to the two elements and field5, and level 2's delimiter applies to fields 1 through 4.

Delimiter lists can be much more complex than this very simple example. For instance, you can create multiple delimiters of different types at any given level. Also, you can specify a delimiter list on any node within the OTD, not only the root node as shown in the example.

The following escaped delimiters are allowed.

Table 24 Escaped Delimiters

Delimiter	Description
\n	Newline
\r	Carriage return
\t	Tab
\\	Backslash

Delimiter Properties

Delimiters are defined using the Delimiter List Editor (see Figure 59).

Figure 59 Delimiter List Editor

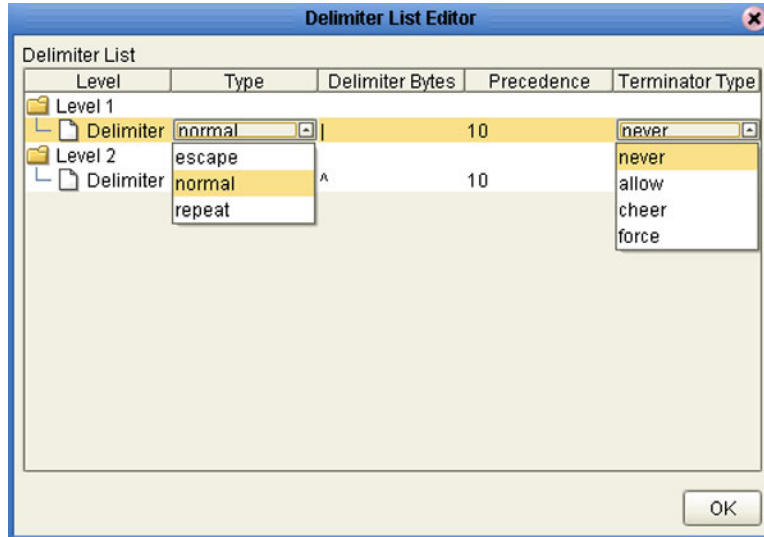


Table 25 Delimiter Properties and Value Options

Property	Option	Description
Level		Child level beneath defining node.
Type	escape	Escape sequence.
	repeat	Array delimiter/separator.
	normal	Terminator.
Delimiter Bytes		Delimiter (single or multiple characters).
Precedence		See Precedence on page 95.
Terminator Type	never	Do not allow on input, do not emit on output (pure separator).
	allow	Allow on input, do not emit on output.
	cheer	Allow on input, always emit on output.
	force	Require on input, always emit on output (pure terminator).

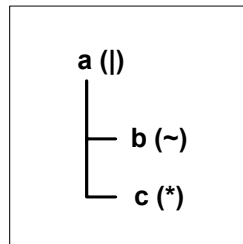
Table 25 Delimiter Properties and Value Options

Property	Option	Description
Optional	never	Do not allow on input, do not emit on output (empty field between delimiters implies zero length data field).
	allow	Skip empty field if present; if absent, do not delimit on output.
	favor	Skip empty field if present; if absent, do not delimit on output.
	force	Require empty, delimited field on input; always delimit on output.

Terminator Type Property Options

Consider the tree structure shown in Figure 60, where the node **a** has a pipe (|) as its delimiter, the sub-node **b** has a tilde (~) as its delimiter, and sub-node **c** has an asterisk (*) as its delimiter.

Figure 60 Terminal Type Property Example



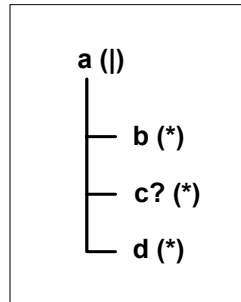
Option	Input	Output
never	c	c
allow	c or c*	c
cheer	c or c*	c*
force	c*	c*

Optional Property Options

Consider the tree structures shown in Figure 61 and Figure 62, where the node **a** has a pipe (|) as its delimiter, and the sub-nodes **b**, **c**, and **d** all have asterisks (*) as their delimiters.

- **Example 1:** Sub-node **c** is *optional*. (Sub-node **c** and sub-node **d** must have different values for the *match* parameter.)

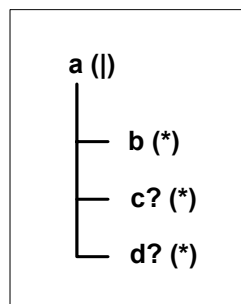
Figure 61 Optional Property (Example 1)



Option	Input	Output
never	b*d 	b*d
allow	b**d 	b*d
cheer	b**d 	b**d
force	b**d 	b**d

- **Example 2:** Both sub-node **c** and sub-node **d** are *optional*.

Figure 62 Optional Property (Example 2)



Option	Input	Output
never	b 	b
allow	b , b* , or b** 	b
cheer	b , b* , or b** 	b**
force	b** 	b**

Precedence

Precedence (see [Figure 67 on page 97](#)) indicates the priority of a certain delimiter, relative to the other delimiters. By default, all delimiters are at precedence 10, which means they are all considered the same; fixed fields are hard-coded at precedence 10. Delimiters on parent nodes are not considered when parsing the child fields; only the child's delimiter (or if it is a fixed field, its length).

Changing the precedence of a delimiter will cause them to be applied to the input data-stream in different ways. For example:

- *root node*
 - ♦ element (type delim, delimiter = "^", repeat)
 - ♦ field1 (type fixed, length = 5)
 - ♦ field2 (type fixed, length = 8, optional)

Although this will parse 'abcde12345678^zyxvuABCDEFGH', it will *not* parse the text 'abcde^zyxvuABCDEFGH' even though the second fixed field is optional. The reason is that the element's delimiter is ignored within the fixed field because they have the same precedence. If you want the element's delimiter to be examined within the fixed field data, you must change its precedence, for example:

- *root node*
 - ♦ element (type delim, delimiter = "^", repeat, **precedence = 11**)
 - ♦ field1 (type fixed, length = 5)
 - ♦ field2 (type fixed, length = 8, optional)

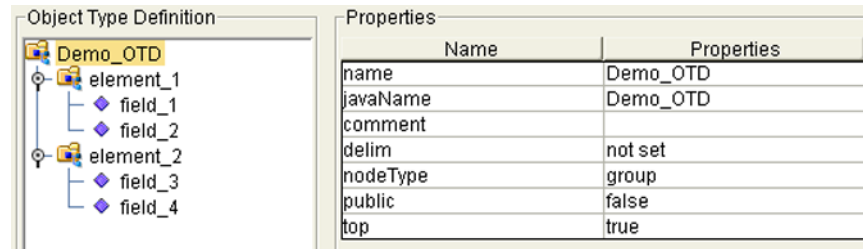
This will successfully parse the text 'abcde^zyxvuABCDEFGH'.

5.4.5 Creating a Delimiter List

To create a delimiter list

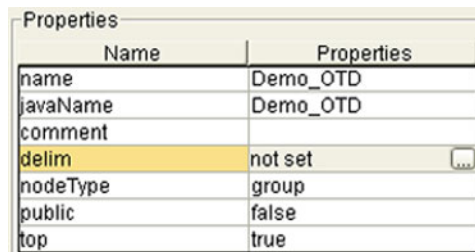
- 1 Create the structure of this example OTD in the User-Defined OTD editor as shown in Figure 63.

Figure 63 Example User-Defined OTD



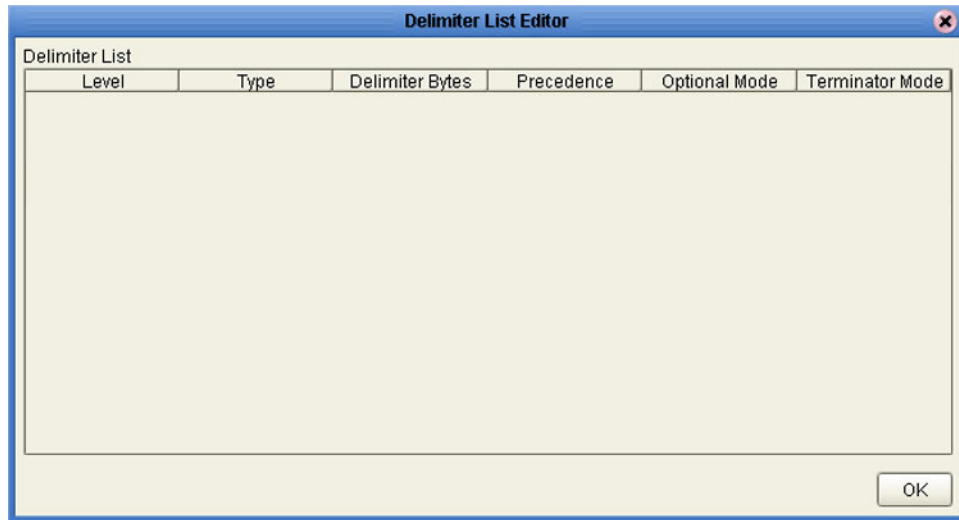
- 2 Select the node to which the delimiter list will apply.
- 3 Double-click on the value of the **delim** property to activate the property field for editing (see Figure 64). Initially it appears as *not set*.

Figure 64 Activate Field



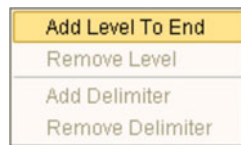
- 4 Click on the ellipsis (...) button to display the Delimiter List Editor, which is blank when the delimiter list is *not set* (see Figure 65).

Figure 65 Delimiter List Editor (1)



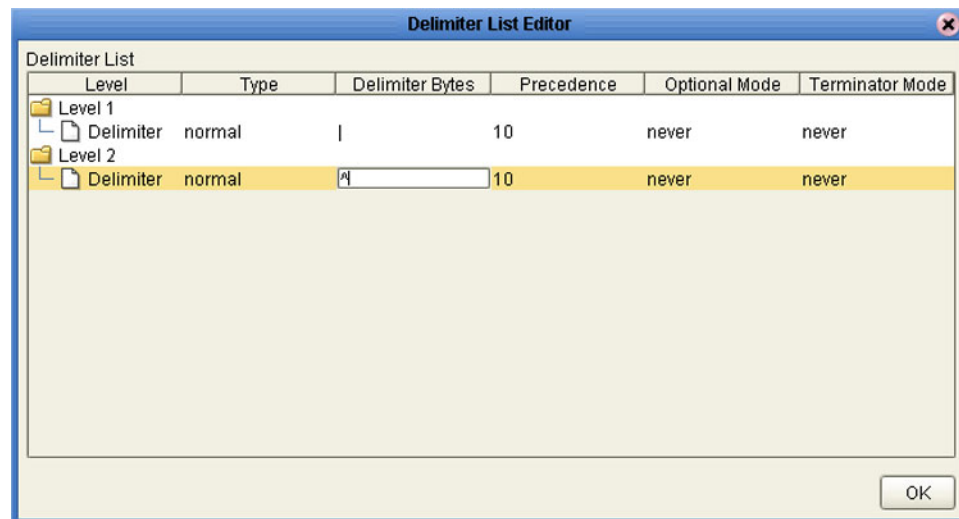
- 5 Right-click anywhere within the table area to invoke the context menu (see Figure 66).

Figure 66 Context Menu



- 6 Levels can be added and removed in the delimiter list, and delimiters can be added or removed from under a given level, using the context menu options. Using the **Add Level To End** option, you can create the delimiter list shown in Figure 67.

Figure 67 Delimiter List Editor (2)



- Click the **OK** button to complete the edit. The **delim** property's value will now appear as *specified*, as shown in Figure 68.

Figure 68 Delimiter Specified

repeat	raise
comment	
delim	specified
nodeType	group
nonRootType	

- Test the OTD by invoking the OTD Tester, as described in [Using the OTD Tester](#) on page 111. Entering the data `a^b | c^d | e` will result in a successful parse, as shown in Figure 77.

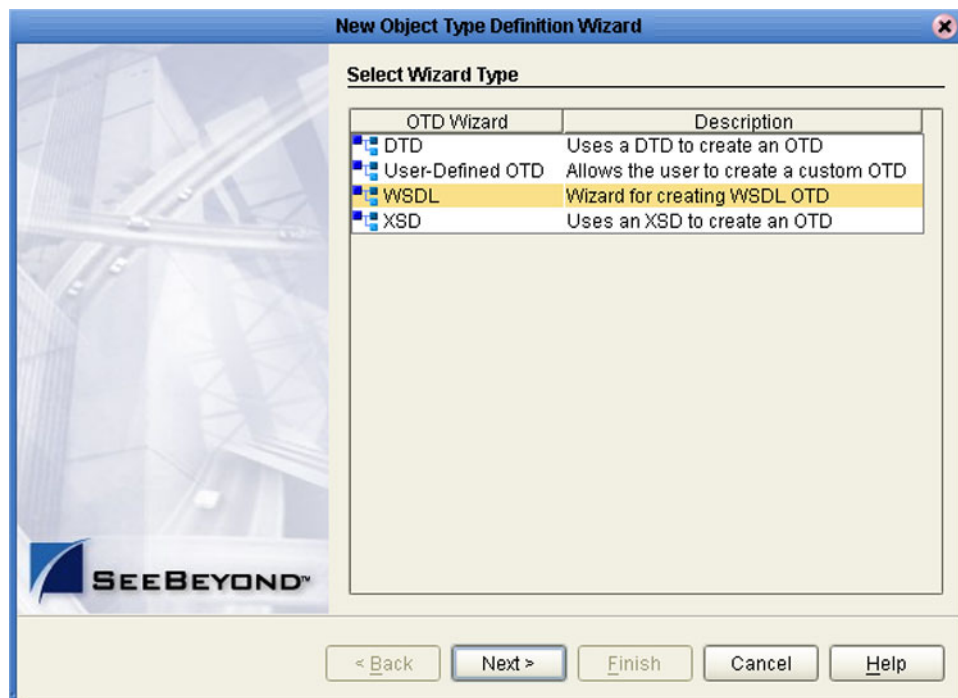
5.5 WSDL-Based OTDs

5.5.1 Using the WSDL Wizard

To create an OTD file from a WSDL file

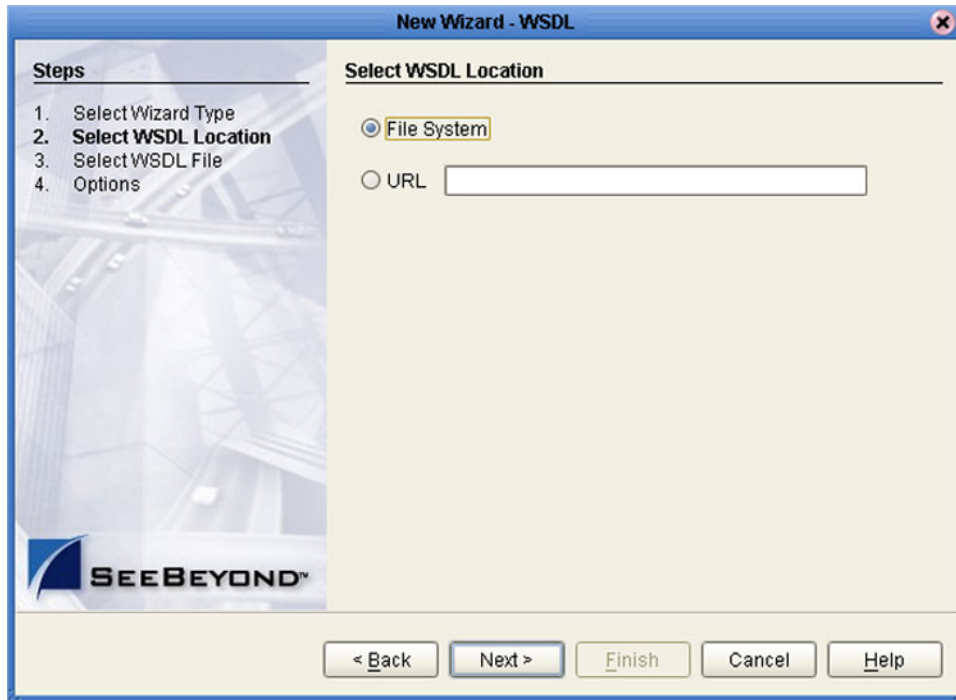
- 1 Select **WSDL** from the *Select an Object Type Definition Wizard* list (see Figure 69) to create an OTD file from an WSFL file.

Figure 69 OTD Wizard Selection: WSDL Wizard



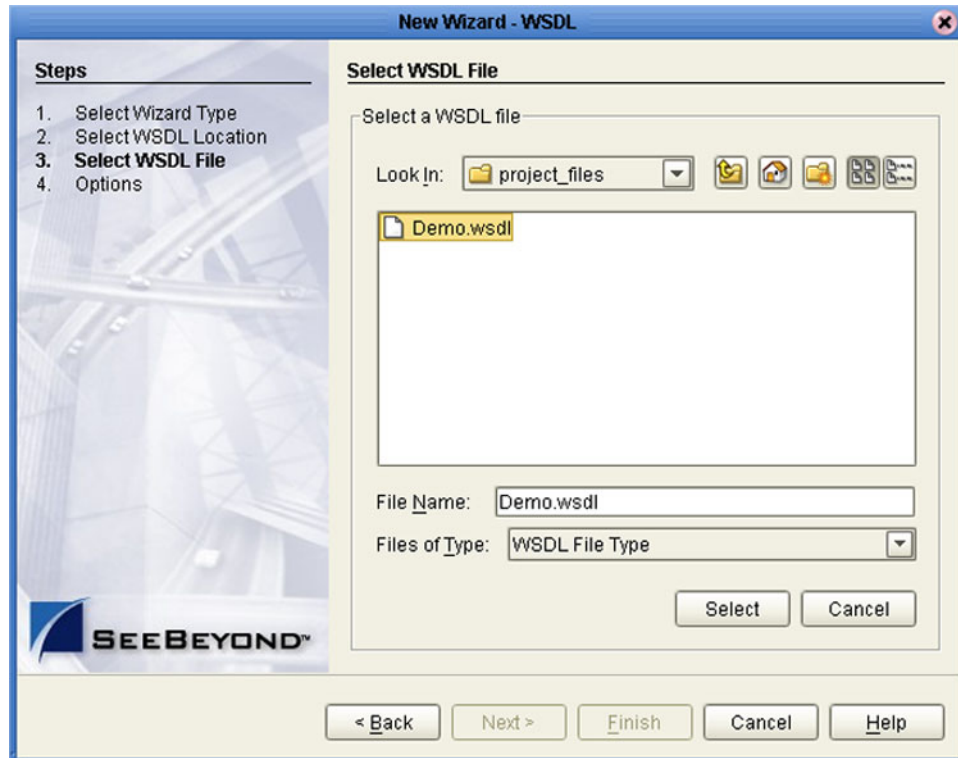
- 2 Click **Next** to display the *Select WSDL File Location* dialog, shown in Figure 70

Figure 70 WSDL Wizard: Select WSDL Location



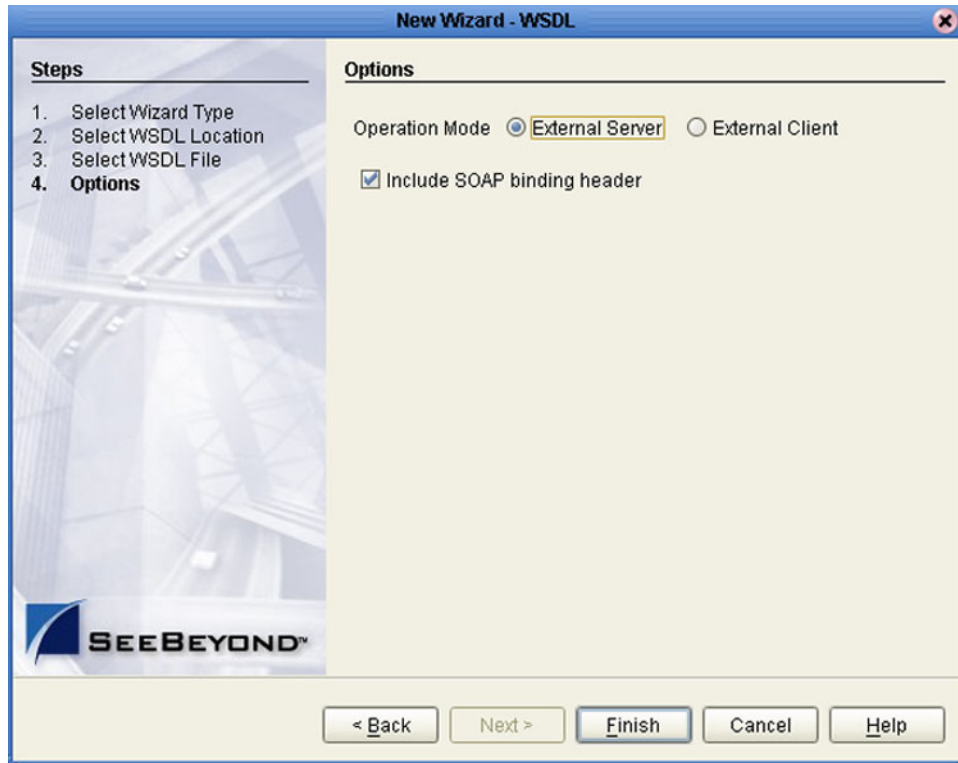
- 3 Select **File System** or enter a **URL**, depending upon where your WSDL file is located.
- 4 Click **Next** to display the *Select WSDL File* dialog, shown in Figure 71.

Figure 71 WSDL Wizard: Select WSDL File



- 5 In the *Look in* drop-down list, navigate to the WSDL file or files that you want to use to create the OTD. Click **Select** to add the files to the *List of Selected WSDLs*.
- 6 Click **Next** to display the *Select OTD Options* dialog, shown in Figure 72.

Figure 72 WSDL Wizard: Select OTD Options



- 7 Select the check boxes next to the OTD options you want to enable, and click **Finish** to add the OTD to the Enterprise Designer with the selected OTD options.

5.5.2 WSDL OTD Operation

WSDL Operation Elements

To tie your messages together as a request-response pair corresponding to a method call, you must define operations using the WSDL **<operation>** element. A WSDL operation specifies which message is the *input* and which message is the *output*.

Inside the WSDL file's **<operation>** element, you specify your **<input>** and **<output>** elements. Each element refers to the corresponding message by its fully qualified name. The collection of all WSDL operations (that is, methods) exposed by your service is called a **portType** and is defined using the WSDL **<portType>** element.

The **<operation>** element is a child of **<portType>**. You can name the **<portType>** whatever you want. The port type **name** attribute provides a unique name among all the PortTypes defined within the enclosing WSDL file. Each WSDL operation is named via the **name** attribute.

Each operation within a WSDL OTD (like its WSDL file counterpart) uses one of the following operation modes for communication:

- **One-way:** The server receives a message from the client; also referred to as “fire and forget.”
- **Request-response:** The server receives a message from the client and sends a correlated message back

WSDL OTD Structure

The WSDL OTD has the following basic structure:

```
Root Node
  PortType_XXX
    Operation_XXX
      Input_XXX
      Output_XXX
  PortType_XXX
    Operation_XXX
      Input_XXX
      Output_XXX (and so on)
```

Where **XXX** is the name for each element given in the original WSDL file.

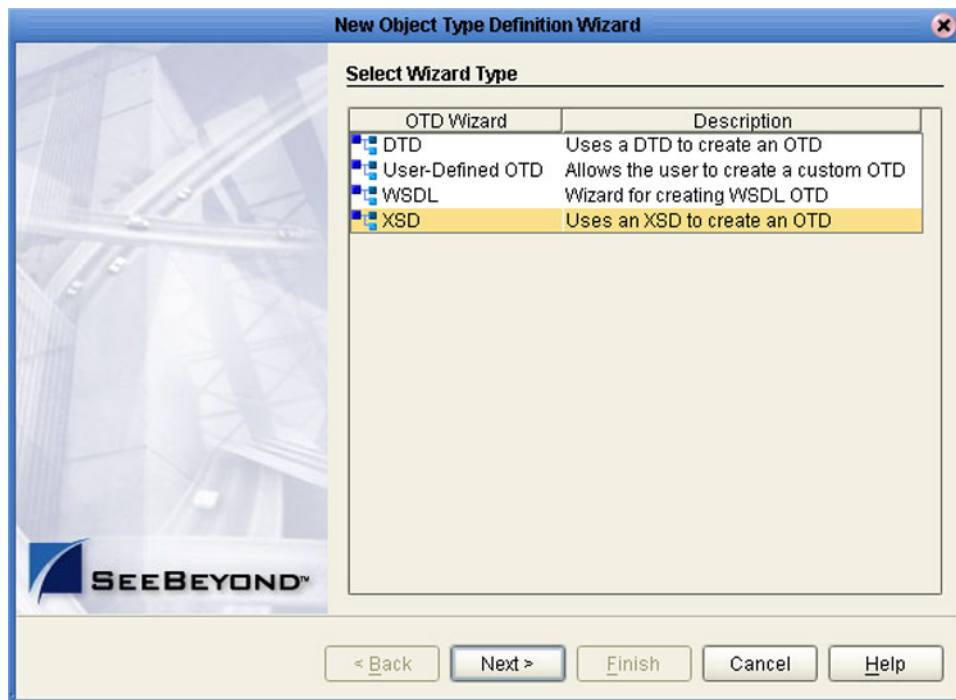
5.6 XSD-Based OTDs

5.6.1 Using the XSD Wizard

To create an OTD file from an XSD file

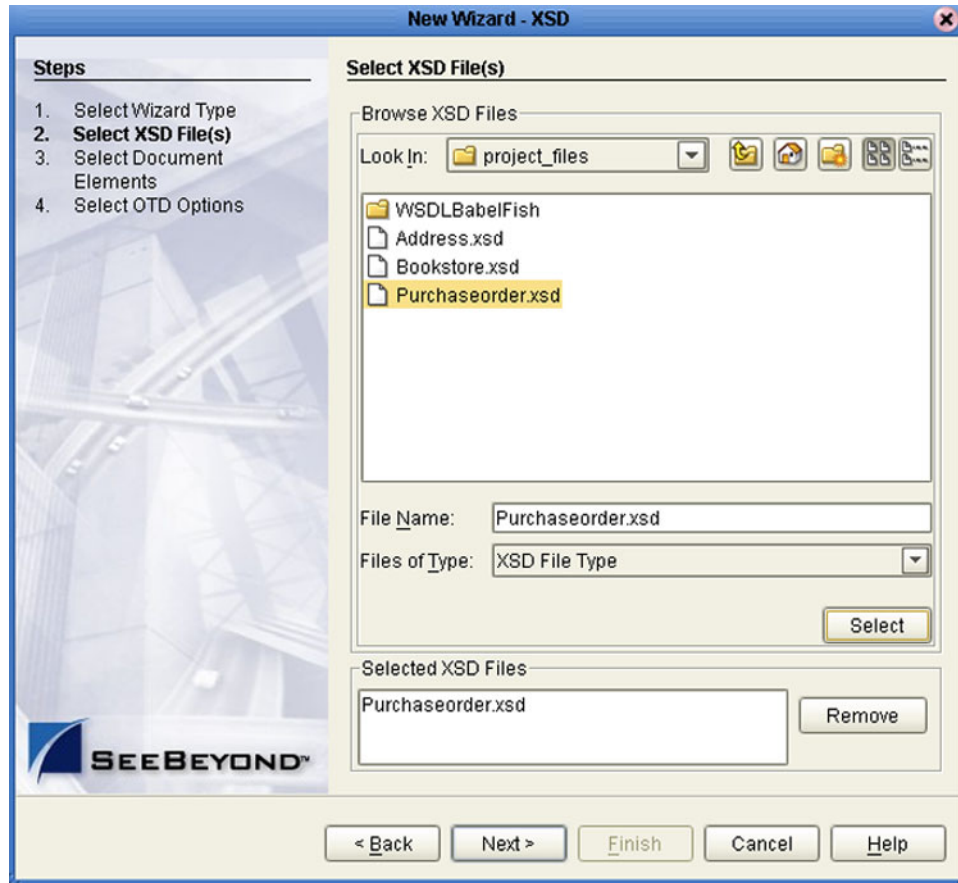
- 1 Select **XSD** from the *Select an Object Type Definition Wizard* list (see Figure 73) to create an OTD file from an XSD file.

Figure 73 OTD Wizard Selection: XSD Wizard



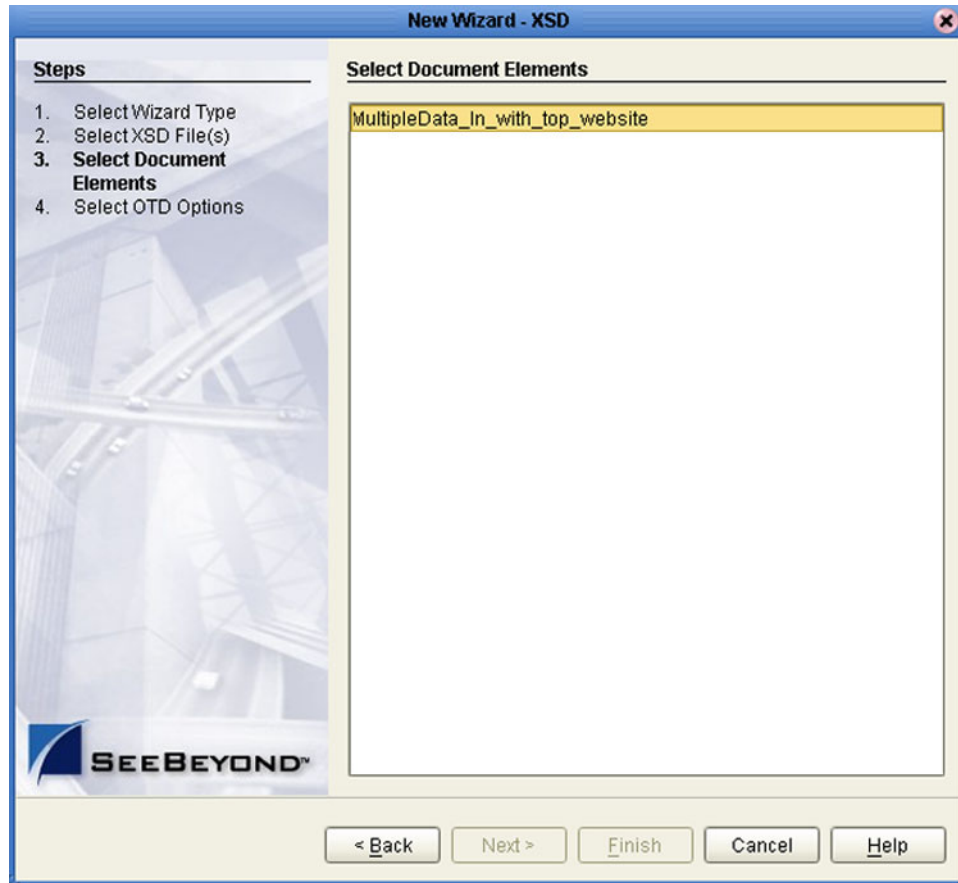
- 2 Click **Next** to display the Select XSD File(s) dialog box, shown in Figure 74.

Figure 74 XSD Wizard: Select XSD File(s)



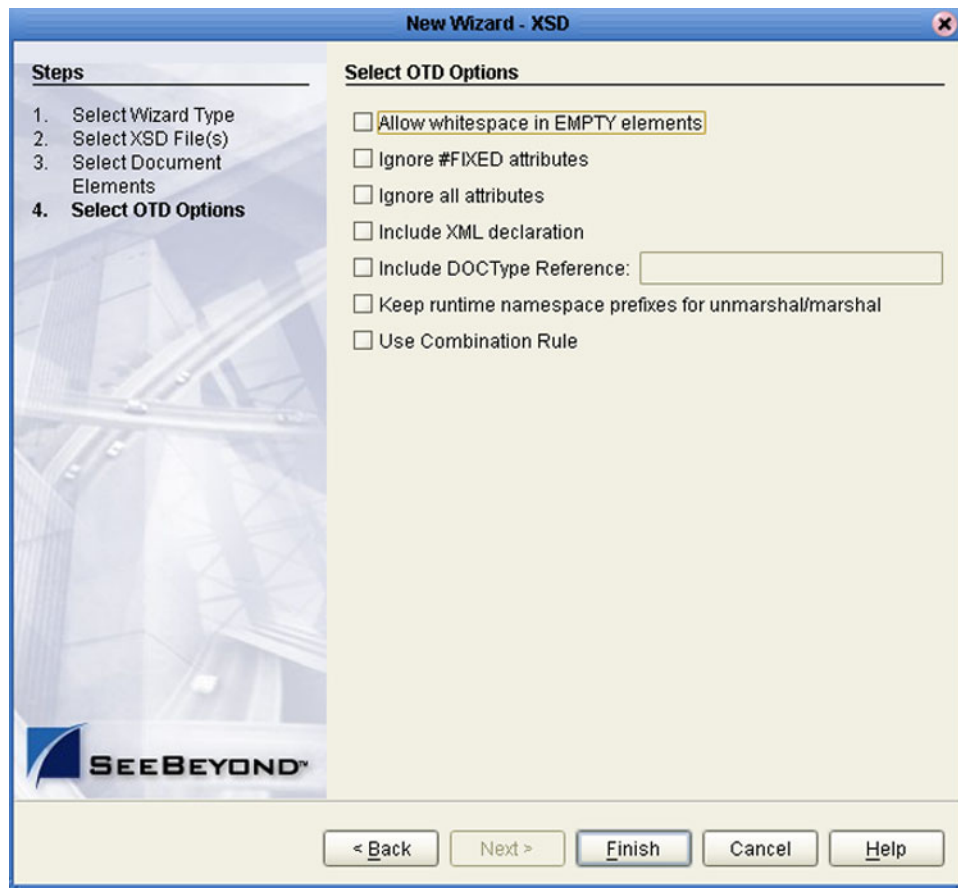
- 3 In the *Look in* drop-down list, navigate to the XSD file or files that you want to use to create the OTD. Click **Select** to add the files to the *List of Selected XSDs*.
- 4 Click **Next** to display the *Select Document Elements* dialog box, shown in Figure 51.

Figure 75 Select Document Elements Dialog Box



- 5 Select the elements of the document that you want to include in the OTD.
- 6 Click **Next** to display the *Select OTD Options* dialog box, shown in Figure 52.

Figure 76 Select OTD Options Dialog Box



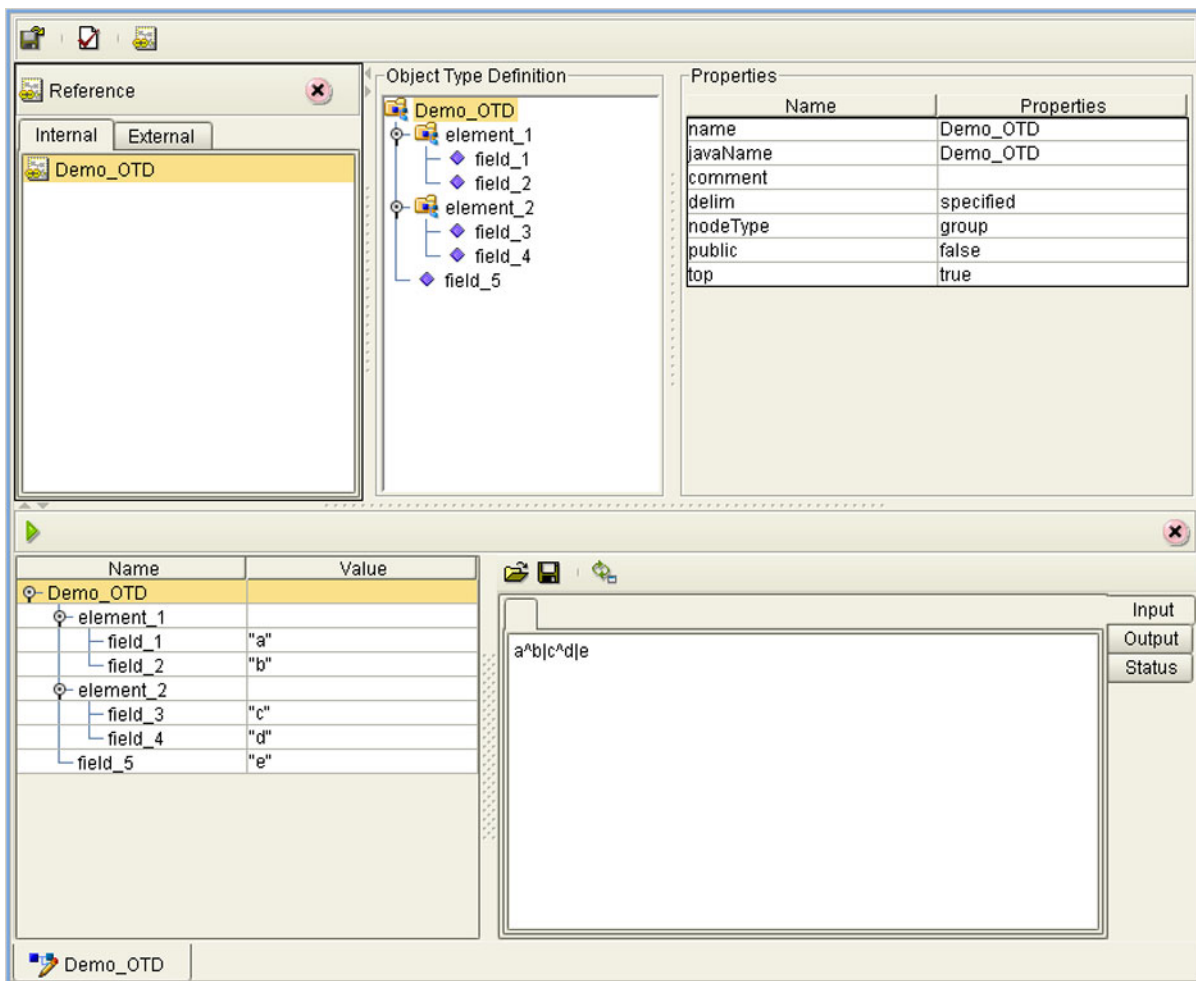
- 7 Select the check boxes next to the OTD options you want to enable, and click **Finish** to add the OTD to the Enterprise Designer with the selected OTD options.

5.7 Using the OTD Editor

After you create an OTD file using the OTD Wizard, the OTD Editor appears in the editor panel of the Enterprise Designer, as shown in Figure 77. You can also invoke the OTD Editor by selecting **Open** in the context menu for an existing OTD in the Project Explorer. OTDs are saved to the Project automatically.

Important: If you delete an OTD in the Project Explorer, any Collaboration Definitions that have been built using that OTD will be affected. It is recommended that you run the Impact Analyzer before attempting to delete any OTDs (see [Impact Analysis](#) on page 230).









Figure 77 OTD Editor



Major features of the OTD Editor interface are:

- **Reference**
This area contains internal and external templates for the OTD file.
- **Object Type Definition**
This area displays each field and element included in the OTD file.
- **Properties**
This area displays details about the OTD file or field selected in the *Object Type Definition* list.
- **Tester**
This area displays in the bottom part of the window when you click **Tester**. Use this area to perform tests on the contents of the OTD.
- **Toolbars**
Several toolbars appear in the OTD Editor, containing icons as described in Table 26.

Table 26 OTD Editor Toolbar Icons

Icon	Command	Function
	Save as New Name in Repository	Saves current OTD under a new name in the Repository.
	Tester	Displays/refreshes the Tester area.
	Toggle Reference Tab Panel	Displays/hides the Reference area.
	Sort by Name	Sorts list alphabetically by name.
	Run Tester	Runs the tester with the entered values.
	Open	Displays file browser.
	Save	Saves displayed file.
	Refresh	Repopulates the OTD object elements with the values from the data display panel.

5.7.1 Node Management

The OTD Editor allows you to:

- **Add** nodes and elements to an OTD.
- **Delete** nodes and elements from an OTD.

When a node is *deleted*, both the node and its associated 'children' (data elements) are deleted.

- **Prune** nodes in an OTD.

When a node is *pruned*, only its associated 'children' (data elements) are deleted, while the node itself is preserved. Pruning can only be performed on nodes.

These commands are accessed from the node context menu.

5.8 Using the OTD Tester

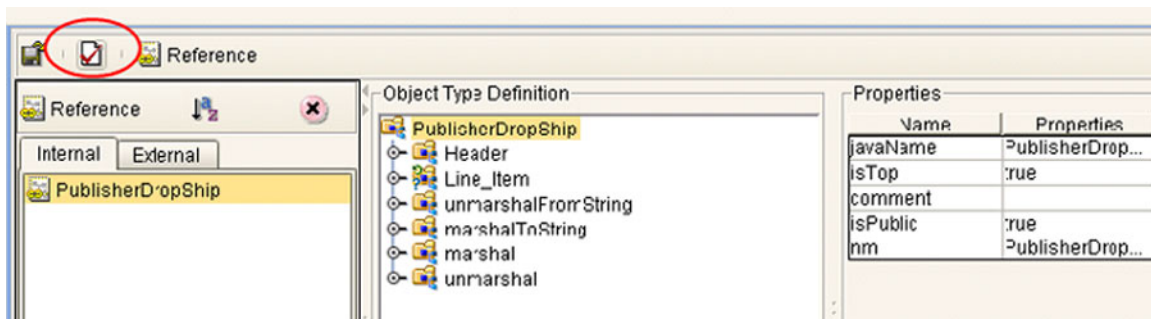
The OTD tester provides a facility to verify the correctness of OTDs, for example to:

- Prevent data errors at runtime.
- Verify that all required data elements are available.
- Verify that all used data formats are correct.

To use the OTD tester

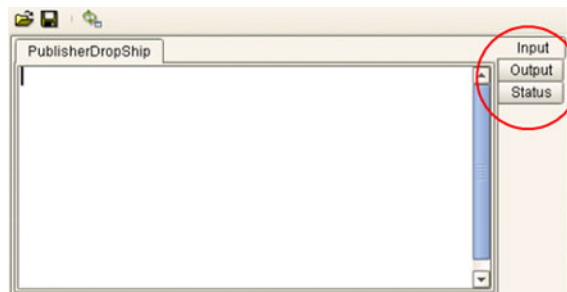
- 1 Open or create an OTD.
- 2 Click the **Tester** icon (see Figure 78).

Figure 78 OTD Tester



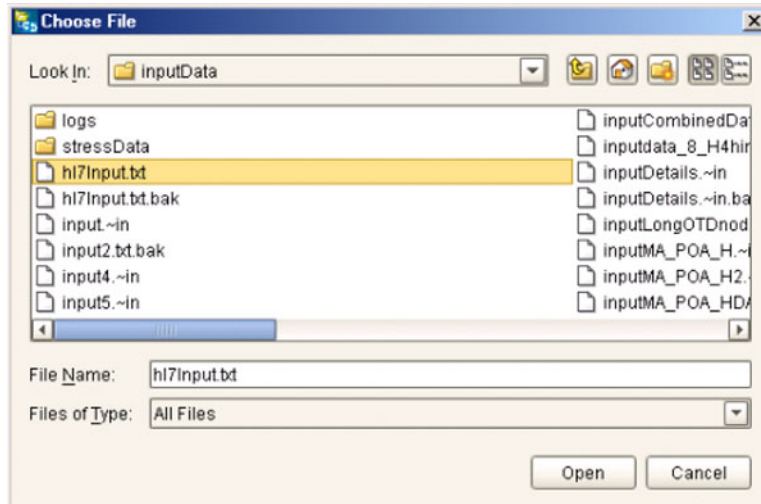
A test panel will appear below the OTD detail area of the editor. Note that there are three data display modes, selectable by tabs (see Figure 79). The Input tab is selected by default.

Figure 79 Test Panel Data Display



- 3 You can provide the input test data either by selecting a data file (see Figure 80), or by entering the data manually.

Figure 80 Select Data File



- 4 Click the **Run Tester** icon (green arrow) to test the selected OTD.
- 5 Verify the output by checking the values for each element for correctness (see Figure 81).

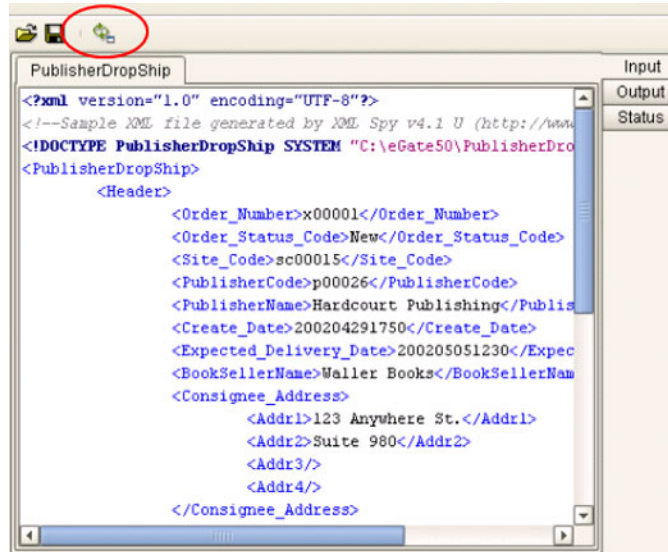
Figure 81 Object Elements and Values

Name	Value
☐ PublisherDropShip	
☐ header	
- name	""
- order_Number	"x00001"
- order_Status_Code	"New"
- site_Code	"sc00015"
- publisherCode	"p00026"
- publisherName	"Hardcourt Publi..."
- create_Date	"200204291750"
- expected_Delivery	"200205051230"
- bookSellerName	"Waller Books"
☐ consignee_Address	
- bom_type	""
- gl_entity	"GLN"
☐ terms	
☐ line_item	
- length	1
☐ [0]	
- value	"500"
- counter	"0"
- itemCode	"ISBN000139298"
- itemDescription	"King James Bib..."
- qty	"100"
- cost	"5.00"

- 6 You can save your input test data to a file for re-use by selecting the **Input** data display and clicking the **Save** icon.

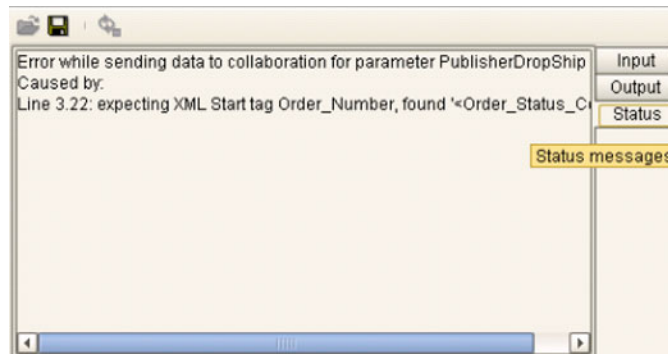
- 7 You can also change your test data in the Input data display, then re-test the OTD by clicking the **Refresh** icon (see Figure 82) to repopulate your OTD object elements with the new values.

Figure 82 Data Display: Refresh Icon



- 8 If there are errors in your input data, the **Status** data display is automatically invoked, showing the appropriate error messages (see Figure 83).

Figure 83 Status Data Display



5.8.1 The *Generic* Interface

The **OtdNode** interface describes the generic access provided by any non-leaf, implemented by a generated class for a single OTD node. The OTD metadata is provided separately by an object that implements the **OtdMeta** interface.

- The *meta ()* method gets the run-time metadata for this node in the OTD's grammar. (This is really a static method by nature, because the grammar for the OTD is not tied to any particular instance of it, but Java interfaces do not allow static methods in interface definitions, and this gives more freedom of implementation.)
- The *has (child)* method tests whether given child is present. For repeated children, it tests against the collection as a whole.
- The *has (child, index)* method tests whether given instance of a repeated child is present. This method may not be called for a non-repeating child node, and is only useful if the pertinent OTD builder supports transient “holes” in a repeating child's set of instances.
- The *size (child)* method returns one past the last valid index for a repeating child. For a non-repeating node, this should return 1. If the OTD does not permit holes in repetition, then this value is the same as returned by the *count()* method.
- The *count (child)* method returns the number of repetition instances of the given child. For a non-repeating node, this should return 1.
- For a **choice** node, the *choice* method returns the index of the valid child. A choice node has a number of children, only one of which can occur below any given parent instance. If there is no current valid child, it returns -1.
- The *get (child)* method retrieves the given child instance. This method may only be called for a non-repeating child node.
- The *get (child, index)* method retrieves the given child instance. This method may not be called for a non-repeating child node.
- The *set (child, value)* method stores the given child instance. This method may only be called for a non-repeating child node.
- The *set (child, index, value)* method stores the given child instance. This method may not be called for a non-repeating child node.
- The *remove (child)* method removes the given child instance. The child must be optional. The implementation is allowed (but not required) to throw an exception if the child is not optional.
- The *remove (child, index)* method removes the given occurrence of a child instance. The child must be repeated. The implementation is allowed (but not required) to throw an exception if the child is absent. A subsequent *has (child, index)* call with the same arguments should return false. If the implementation does not support holes, an attempt to remove the non-ultimate occurrence may throw an exception or replace the value with the same default value used by *set (child, index)* to fill unassigned occurrences.

Java Collaboration Definitions

This chapter describes the process for creating Java Collaboration Definitions.

6.1 Overview

Collaborations use Collaboration Definitions to define how data should be routed between Project components. Collaborations also define how databases should be queried in response to requests and how APIs to one or more applications should be invoked. Collaborations are used when data translation is required.

The Enterprise Designer includes two tools, the Java Collaboration Wizard and Java Collaboration Editor, that are used to create and customize your Java Collaboration Definitions. You must have OTDs available to use as the foundation for creating a Java Collaboration Definition. See [Object Type Definitions](#) on page 77 for more details.

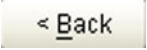

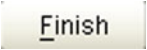


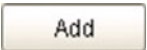

Important: *If you delete an OTD in the Project Explorer, any Java Collaboration Definitions that have been built using that OTD will be affected. It is recommended that you run the Impact Analyzer before attempting to delete any OTDs (see [Impact Analysis](#) on page 230).*

Note: *Any changes made to the names of the Collaborations should be done when the Collaboration is created. If the name is changed later, the Collaboration should be opened, regenerated if applicable, and saved again. For safe measure, this should also be done before creating the Connectivity Map and Deployment Profile.*

6.2 Using the Java Collaboration Wizard

6.2.1 Navigation Buttons

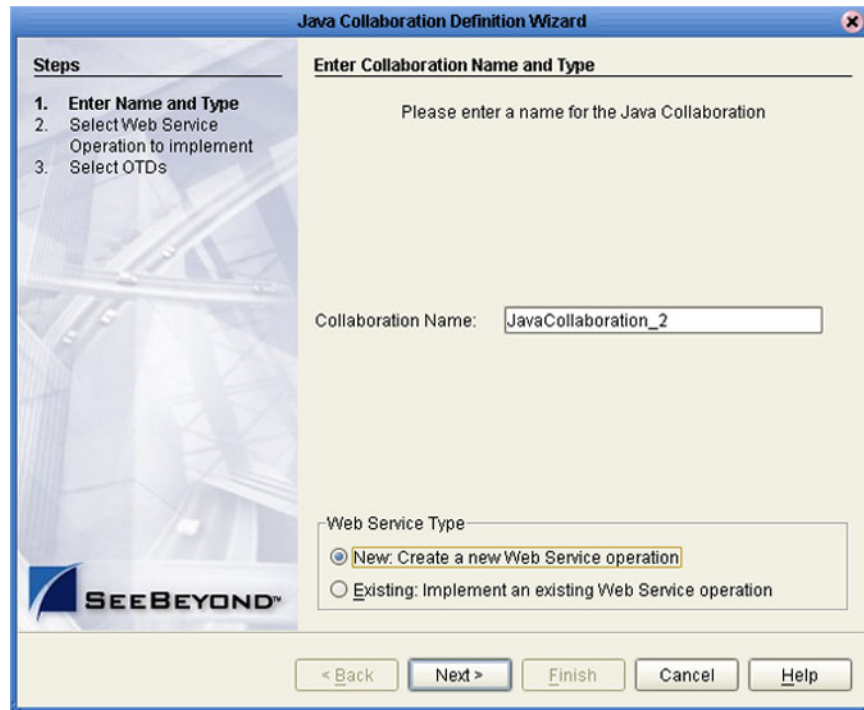
Table 27 Wizard Navigation Buttons

Button	Function
	Returns to the previous step in the wizard. This button is disabled on the first step.
	Goes to the next step in the wizard. This button is disabled on the last step.
	Saves all Collaboration Definition settings and closes the wizard. This button is only enabled on the last step.
	Closes the wizard without saving the Collaboration Definition.
	Displays the online help documentation for the Collaboration Definition Wizard dialog box.
	Adds a selected Object Type Definition to the Collaboration Definition.
	Removes a selected Object Type Definition from the Collaboration Definition.

6.2.2 Creating a Java Collaboration Definition

- 1 Right-click on a Project in the Enterprise Explorer to display the Project context menu.
- 2 Select **New Java Collaboration Definition** to invoke the Java Collaboration Wizard.
- 3 Enter a **Name** for your Collaboration, as shown in Figure 84.

Figure 84 Initial Wizard Dialog



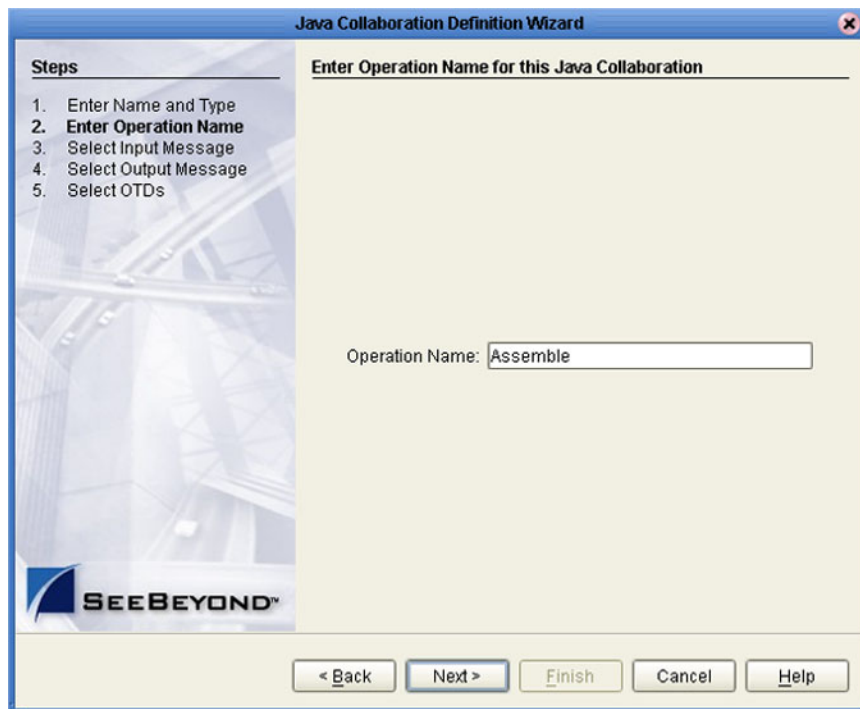
- 4 Select a Web service, which can be either:
 - A New Web Service.
 - An Existing Web Service (for example, an eInsight process or an OTD).
- 5 Click **Next** to proceed to the next Wizard dialog, which is dependent upon your Web Service selection.

New Web Service

If you selected a New Web Service, you will be presented with the following set of Wizard dialogs.

- 1 Enter an operation name, as shown in Figure 85. This will become the *method* that can be used to invoke the Java Collaboration as a Web service.

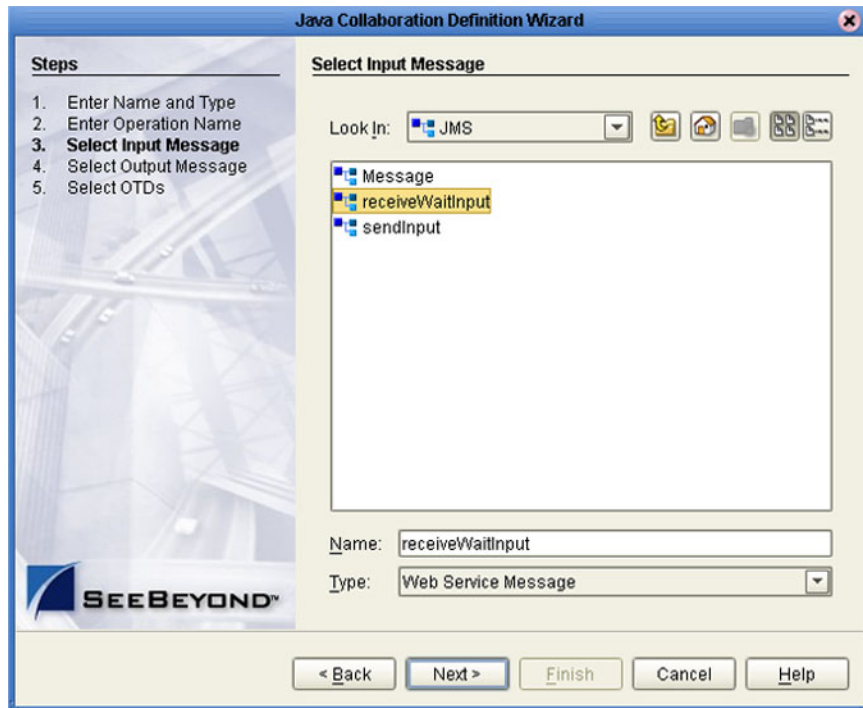
Figure 85 New Web Service: Operation Name



- 2 Click **Next** to proceed to the next Wizard dialog.

- 3 Select the input Web service message, as shown in Figure 86.

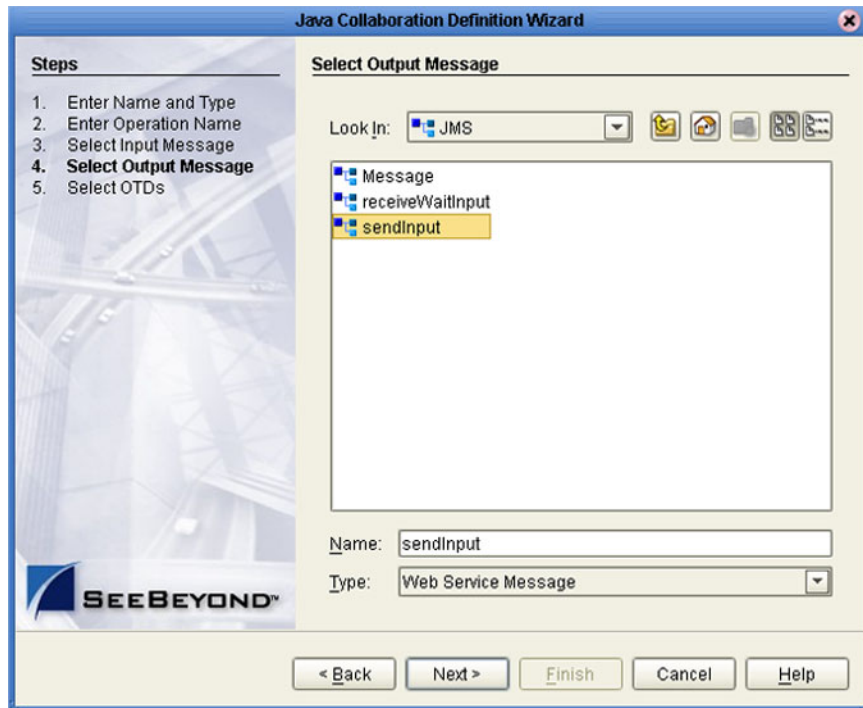
Figure 86 New Web Service: Input Message



- 4 Click Next to proceed to the next Wizard dialog.

- 5 Select the output Web service message, as shown in Figure 87.

Figure 87 New Web Service: Output Message

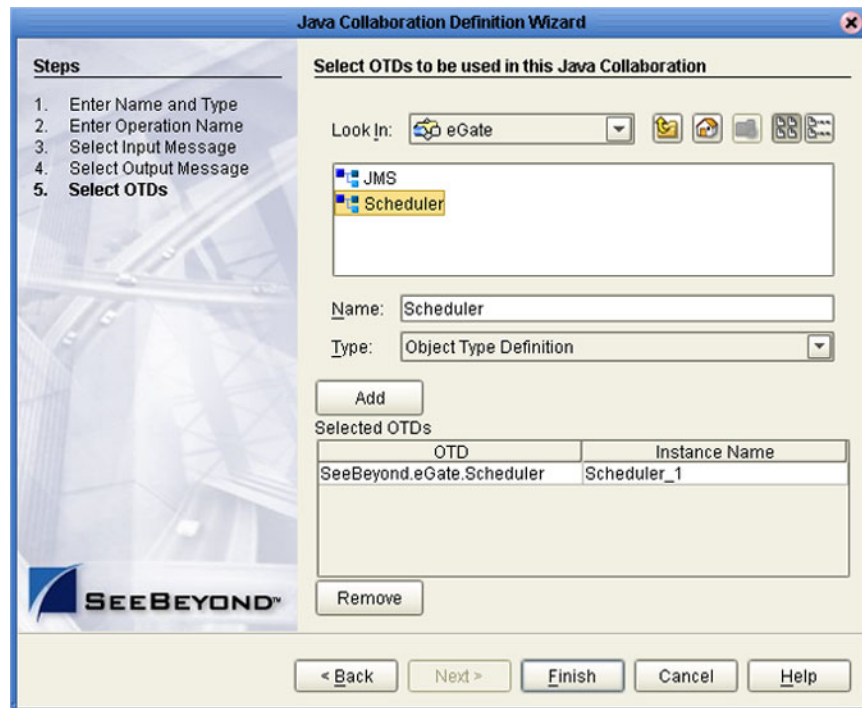


- 6 Click Next to proceed to the next Wizard dialog.

- 7 Select an auxiliary OTD, as shown in Figure 88. This step is optional, and is intended to support additional functionality such as a database lookup.

Note: Use caution here, since you may already have OTDs selected in the preceding steps.

Figure 88 New Web Service: Auxiliary OTD



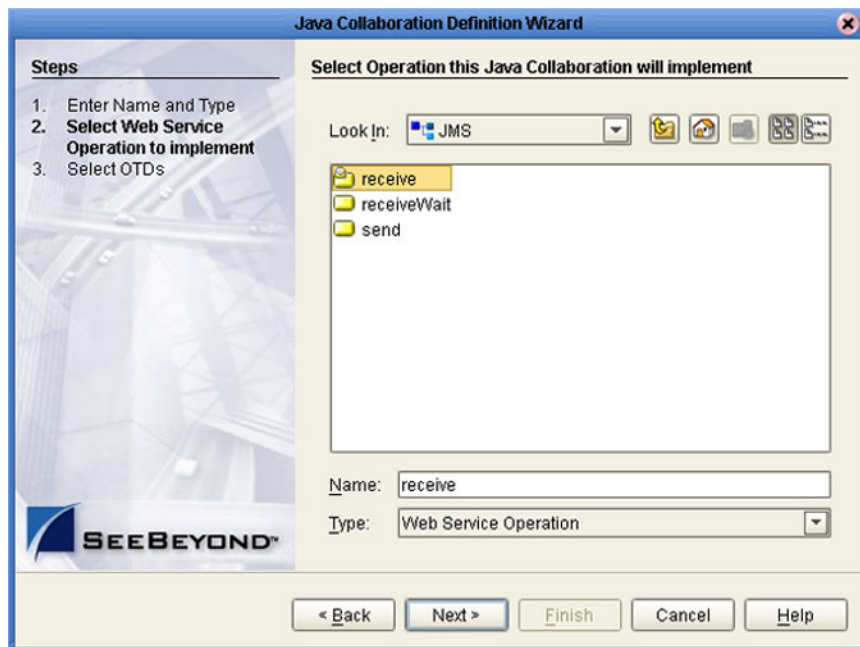
- 8 Click **Add** to add the OTD to the Collaboration Definition.
- 9 Click **Next** to proceed to the Java Collaboration Editor.

Existing Web Service

If you selected an Existing Web Service, you will be presented with the following set of Wizard dialogs.

- 1 Select a Web service operation, which can be either:
 - An installed ICAN Web Service (for example, a JMS *receive* Web service - see Figure 89).
 - A custom Web Service (for example, something that has been created in an eGate Project).

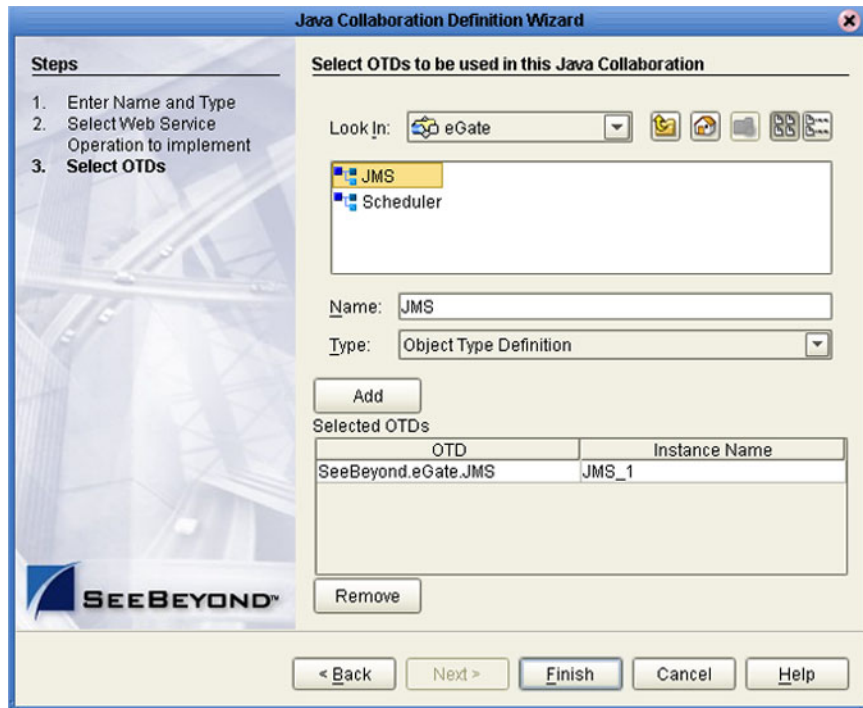
Figure 89 Existing Web Service: Select Operation



- 2 Click **Next** to proceed to the next Wizard dialog.

- 3 Select an OTD, as shown in Figure 90.

Figure 90 Existing Web Service: Select OTD

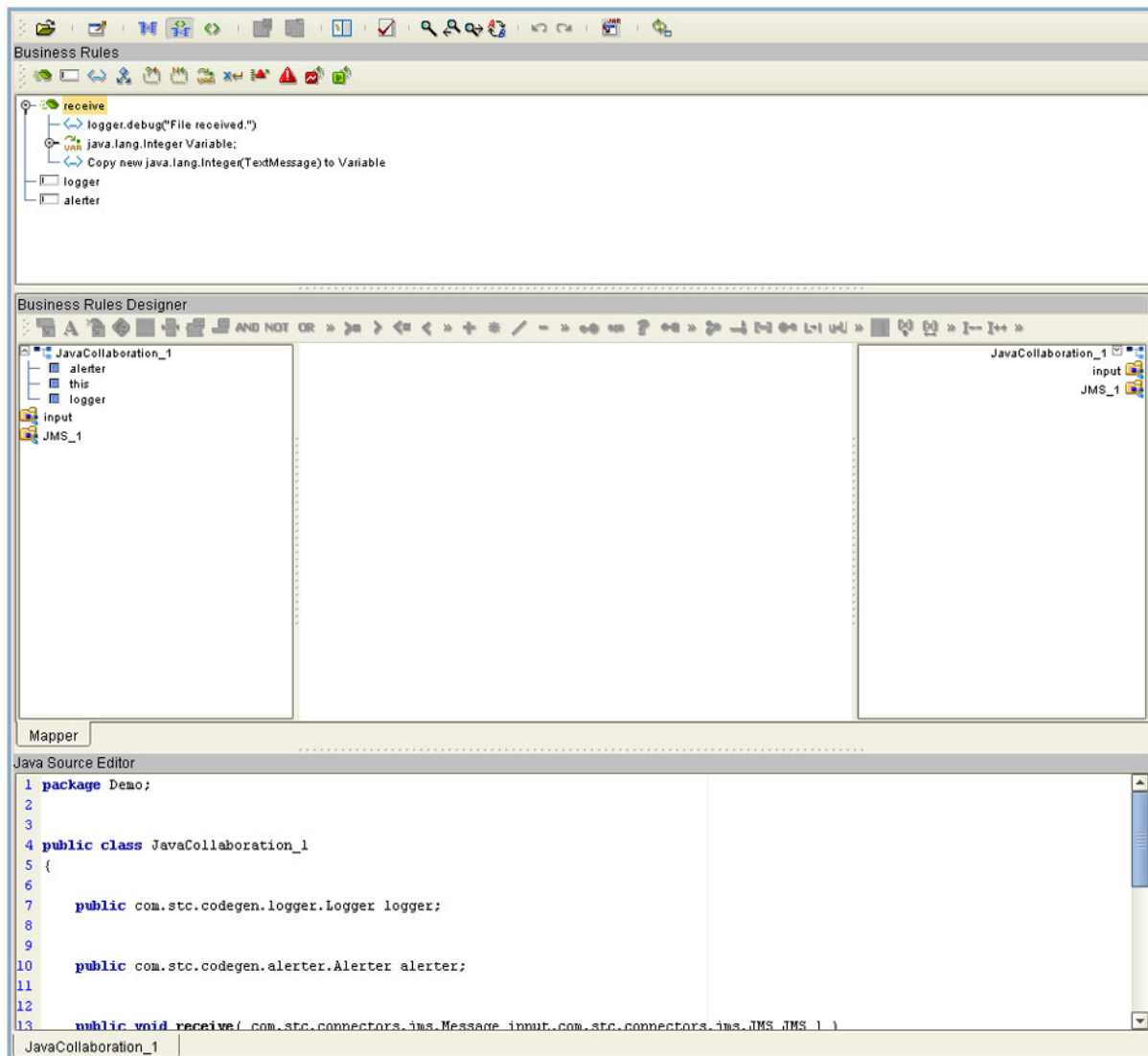


- 4 Click **Add** to add the OTD to the Collaboration Definition.
- 5 Click **Next** to proceed to the Java Collaboration Editor.

6.3 Using the Java Collaboration Editor

After you create a Java Collaboration Definition file using the Java Collaboration Definition Wizard, the Java Collaboration Editor appears in the editor panel of the Enterprise Designer (see Figure 91).

Figure 91 Java Collaboration Editor



The features of the Editor are described in the following sections:

- [Java Toolbar Icons](#) on page 125.
- [Business Rules Editor](#) on page 127.
- [Business Rules Designer](#) on page 130.
- [Java Source Editor](#) on page 135.

6.3.1 Java Toolbar Icons

Table 28 Java Toolbar Icons












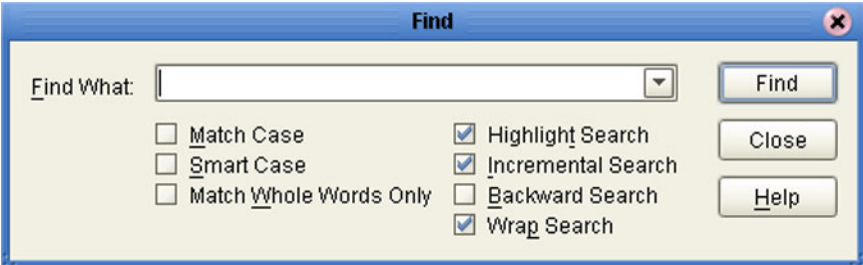








Icon	Command	Function
	Import a Local File	Displays the Open dialog box, which you can use to locate and select a Java Collaboration Definition to import. When you import a file, any previously generated code or rules are deleted. The imported code does not get appended to the existing Collaboration Rules.
	Export Java Rule to a Local File	Displays the Save dialog box, which you can use to save the selected Java Collaboration Definition to a file.
	Standard Mode	Displays the Business Rules Editor and Business Rules Designer only (default setting).
	Advanced Mode	Displays the Business Rules Editor, Business Rules Designer, and Java Source Editor.
	Source Code Mode	Displays the Business Rules and Java Source Editors only.
	Commit Changes	Saves changes made in the Business Rules or Java Source Editors. Re-enables the Business Rules Editor and Business Rules Designer after using the Java Source Editor.
	Roll Back Changes	Cancels changes made in the Business Rules or Java Source Editors and returns the Java Collaboration Definition to the last saved state.
	Business Rules on Left	Changes the editor layout to display the Business Rules panel to the left of the Business Rules Designer. Toggles with <i>Business Rules on Top</i> .
	Business Rules on Top	Changes the editor layout to display the Business Rules panel above the Business Rules Designer (default setting). Toggles with <i>Business Rules on Left</i> .
	Validate	Verifies that changes made to the Java code are valid.

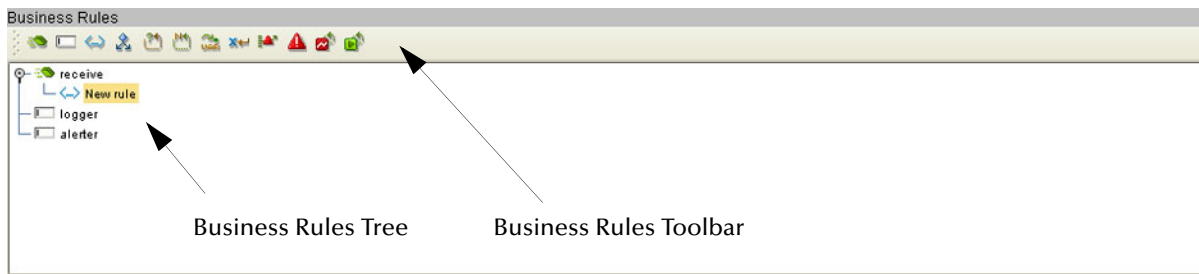
Table 28 Java Toolbar Icons

Icon	Command	Function
	Find	<p>Opens the Find dialog box, where you can enter text to search for in the Java Source Editor.</p> 
	Find Previous	Searches backward for a previous instance of the text entered in the Find dialog box.
	Find Next	Searches forward for the next instance of the text entered in the Find dialog box.
	Replace	<p>Opens the Replace dialog box, where you can enter text to search for, and to replace with, in the Java Source Editor.</p> 
	Undo	Undoes the last operation.
	Redo	Restores the last operation, if undone.
	Import JAR File	Displays the Add/Remove JAR Files dialog box, which you can use to import a .jar file.
	Refresh Collaboration	Refreshes the Collaborations from the Repository.

6.3.2 Business Rules Editor

The Business Rules Editor lists each business rule included with the Java Collaboration Definition. This editor has its own toolbar which you use to add business rules to the Collaboration Definition (see Figure 92). You add rules simply by dragging and dropping the rules into the Business Rules tree. Click the **Commit Changes** button when you are finished.

Figure 92 Business Rules Editor



Note: *The Enterprise Designer automatically disables the Business Rules Editor when you are actively using the Java Source Editor. This is a safety feature that prevents you from inadvertently making contradictory changes. To re-enable the Business Rules Editor, click the **Commit Changes** button in the main toolbar.*

Commands













The **Method** command is used when you want to create a reusable set of instructions inside a Java Collaboration for a specific purpose. Methods are implemented as Java methods, and can be enhanced by means of parameters, return values, and access types (such as public, which means they are also available to other Collaborations).

The **Field** command is used when you want to create a field within a Java Collaboration for some specific purpose, such as storing a temporary variable. As soon as the field has been created, all other rules in the Collaboration are able to read or write from it. A field can be used with the Collaboration (access type = local) or by any other Collaboration (access type = public).

The remaining commands, as represented in the toolbar, are explained in Table 29.

Business Rules Toolbar Icons

Table 29 Business Rules Toolbar Buttons

Icon	Command	Function
	Method	Displays the <i>Add Method</i> dialog box, which you can use to add a new business rule method to the Business Rules tree. The Business Rules tree includes a default method called executeBusinessRules .
	Field	Displays the <i>Create a Parameter</i> dialog box, which you can use to add a new field to the Business Rules tree.
	Rule	Adds a new (empty) rule statement to the Business Rules tree. Use this command as a safeguard regarding positioning.
	If-Then	Adds an if-then statement to the Business Rules tree.
	While	Adds a while statement to the Business Rules tree, starting a specific iteration (repetition) of business rules. You can configure the condition using drag-and-drop when the while statement is selected.
	For	Adds a for statement to the Business Rules tree, starting a specific iteration (repetition) of business rules.
	Local Variable	Displays the <i>Create a Variable</i> dialog box, which you can use to add a local variable to the Business Rules tree.
	Return	Adds a return statement to the Business Rules tree.
	Throw	Adds a throw statement for throwing exceptions.
	Try	Adds a try statement to the Business Rules tree, initiating a number of programming statements that are monitored to see whether they succeed or fail. A finally statement is added automatically; you must right-click on the rule to add a catch statement.
	Break	Breaks out of a business rule while , for , or switch loop.
	Continue	Continues the execution of business rules in a for or while loop by starting the next iteration.

Business Rules Tree

The business rules tree consists of method and field nodes. These are the top level nodes on the tree and cannot be moved. The rule, if-then, while, for, local variable, return, and try statements can be added to a method as subnodes. Statements cannot be added to fields.

The rules for placing subnodes in a method are described in Table 30.

Table 30 Rules for Placement of Subnodes

Subnode	Description
if	Place as a sibling after the last rule in the if-then-else block.
then	Place as a child after the last rule in the then code block.
else	Place as a child after the last rule in the else code block.
while	Place as a sibling after the last rule in the while node, just after the closing parenthesis.
while (code block)	The target node is represented by a code block node under the while node. The source rule becomes a child as the last rule in the block.
for	Place as a sibling after the last rule in the for node, just after the closing parenthesis.
for (code block)	The target node is represented by a code block node under the for node. The source rule becomes a child as the last rule in the block.
local variable	Place as a sibling after the last local variable block.
return	Place as a sibling after the last return block.
try	Place as a sibling after the last rule in the try-catch-finally block.
catch	Place as a child after the last rule in the catch code block.
finally	Place as a child after the last rule in the finally code block.

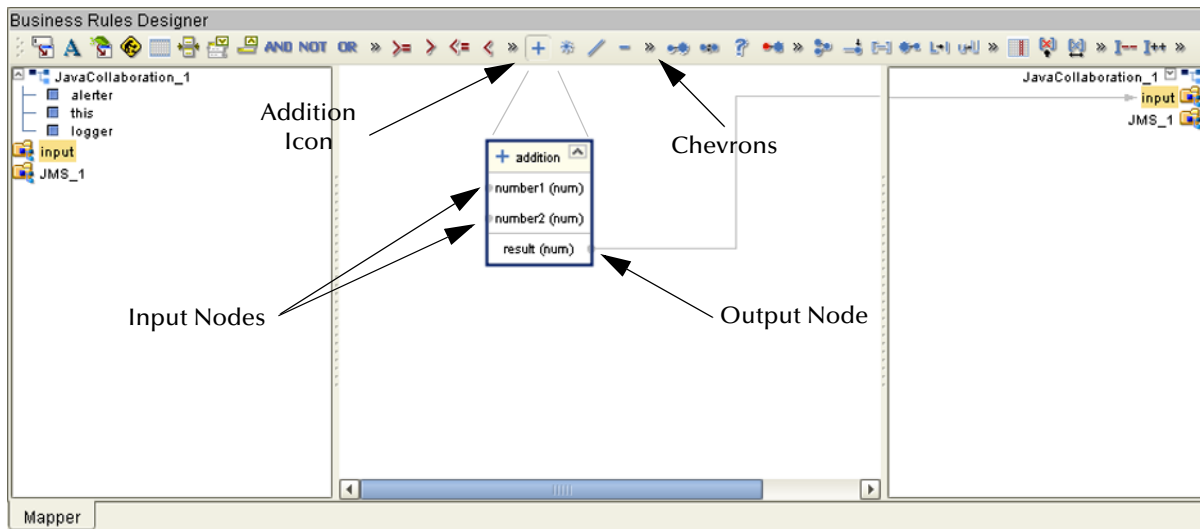
Note: When a subnode is moved to another method, or node, it becomes the first subnode listed under that node.

6.3.3 Business Rules Designer

The Business Rules Designer (see Figure 93) lists each field included in the Java Collaboration Definition. This area has its own toolbar and method palette which you use to map fields and add methods.

*Note: The Enterprise Designer automatically disables the Business Rules Designer when you are actively using the Java Source Editor. This is a safety feature that prevents you from inadvertently making contradictory changes. To re-enable the Business Rules Designer, click the **Commit Changes** button in the main toolbar.*









Figure 93 Business Rules Designer: Addition Method



Note: If you encounter an error while using the addition, subtraction, multiplication, or division operators, try manually converting the string to integers.

Business Rules Designer Toolbar Icons

Table 31 Business Rules Designer Toolbar Icons

Icon	Command	Function
	Import Static Field	Displays the <i>Import Static Field</i> dialog box (see Figure 94 on page 132), which you can use to select a field to add to the Collaboration Definition.
	Create Literal	Displays the <i>Create Literal</i> dialog box (see Figure 141 on page 169), which you can use to select a literal and enter a value to add to the Collaboration Definition.
	Call Java Method	Displays the <i>Call Java Method</i> dialog box Figure 132 on page 163 (see Figure 94 on page 132), which you can use to select a Java method to add to the Collaboration Definition.
	Call New Constructor	Displays the <i>Call New Constructor</i> dialog box (see Figure 117 on page 154), which you can use to select a Java class to the Collaboration Definition.
	New Array	Displays the <i>New Array</i> dialog box (see Figure 95 on page 133), which you can use to add an array to the Collaboration Definition.
	Auto Layout	Updates the layout of fields, literals, and methods that you have added to the middle column of the Business Rules Designer and vertically aligns method boxes.
	Expand All Methods	Displays the title and contents of fields, literals, and methods.
	Collapse All Methods	Displays the title of fields, literals, and methods only.

Dialog Boxes

Dialog boxes invoked from selected toolbar icons are shown in the following figures.

Figure 94 Import Static Field Dialog Box

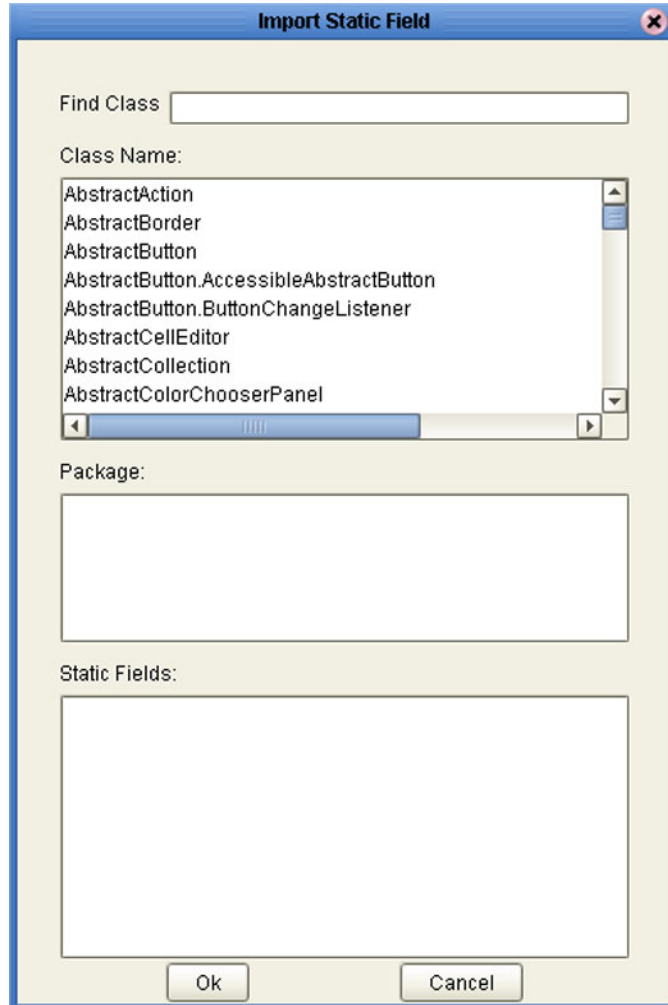
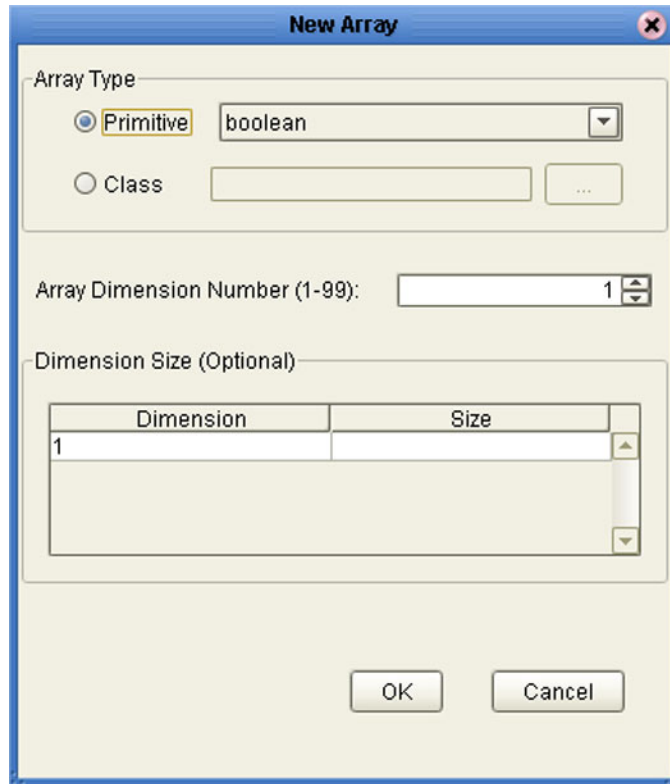


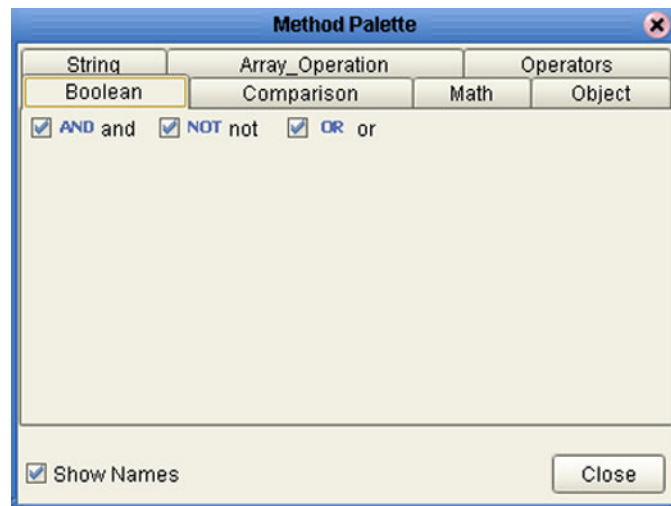
Figure 95 New Array Dialog Box



Java Method Palette

The Java Method Palette includes a series of method icons that you can drag onto the Business Rules Designer mapping area. Click the Chevrons (>>) to the right of the method groups to display the dialog box shown in Figure 96. Select a check box to add the method to the toolbar; clear a check box to remove the method from the toolbar. The methods are described in [Java Methods](#) on page 136.

Figure 96 Java Method Palette Dialog Box



Java Method Boxes

The method boxes are placed in the mapping panel of the Business Rules Designer by dragging the corresponding icon from the method palette toolbar. As shown in [Figure 93 on page 130](#), the method boxes typically have input and output nodes that you link to fields in the left and right panels, respectively. The method boxes are expanded by default (see Figure 97); you can collapse them (see Figure 98) by clicking the caret icon in the upper right corner of the box, which can be useful when the mapping area becomes crowded. Some boxes expand further as needed to provide additional arguments.

Figure 97 Expanded Method Box

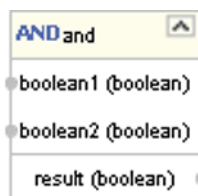


Figure 98 Collapsed Method Box

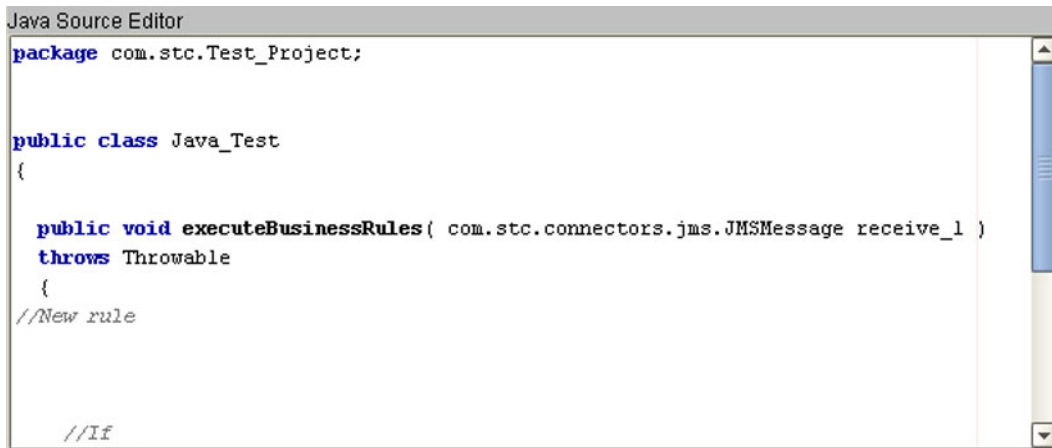


6.3.4 Java Source Editor

The Java Source Editor (see Figure 99) displays the actual Java code as you map the Collaboration Definition. At your option, you can enter and edit the raw Java code for the Collaboration Definition. Click the **Commit Changes** button in the main toolbar when you are finished.

Note: *The Java Source Editor is not displayed by default—you must click the Source Code Mode or Advanced Mode toolbar buttons to display it.*

Figure 99 Java Source Editor

A screenshot of a software window titled "Java Source Editor". The window contains a text area with Java code. The code is as follows:

```
package com.stc.Test_Project;

public class Java_Test
{
    public void executeBusinessRules( com.stc.connectors.jms.JMSMessage receive_1 )
    throws Throwable
    {
        //New rule

        //If
```

The text area has a vertical scrollbar on the right side. The code is color-coded: keywords like "package", "public", "void", "throws", and "class" are in blue, and identifiers like "com.stc.Test_Project", "Java_Test", and "receive_1" are in black. Comments are in grey.

Note: *The Enterprise Designer automatically disables the Business Rules Editor and the Business Rules Designer when you are actively using the Java Source Editor. This is a safety feature that prevents you from inadvertently making contradictory changes. You must click the **Commit Changes** button in the main toolbar to re-enable the Business Rules Editor and the Business Rules Designer.*

6.4 Java Methods

6.4.1 Boolean Methods

Figure 100 Method Palette: Boolean Methods

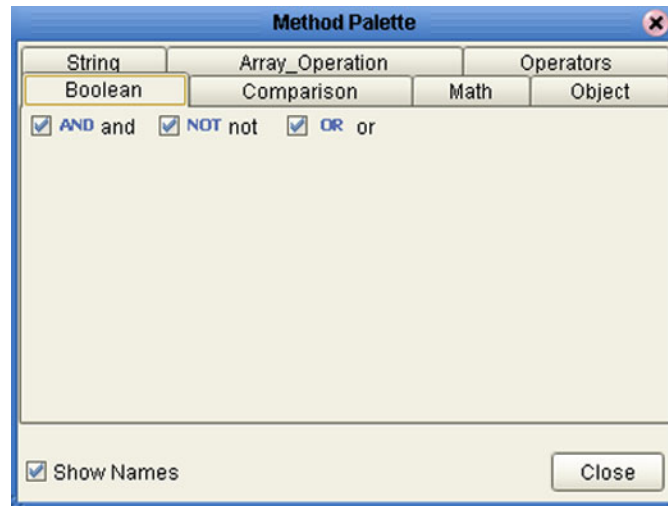


Table 32 Java Boolean Methods

Method Box	Description/Usage
	The AND method returns Boolean true if both <i>boolean1</i> and <i>boolean2</i> are true; otherwise, returns Boolean false.
	The OR method returns Boolean false if both <i>boolean1</i> and <i>boolean2</i> are false; otherwise, returns Boolean true.
	The NOT method returns the inverse of <i>boolean1</i> .

6.4.2 Comparison Methods

Figure 101 Method Palette: Comparison Methods

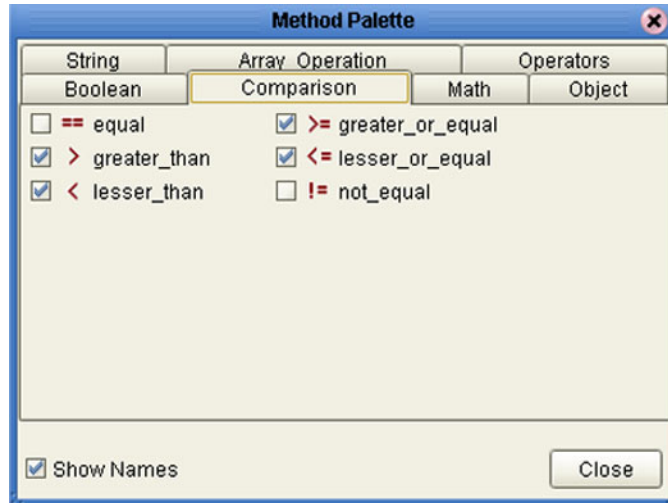
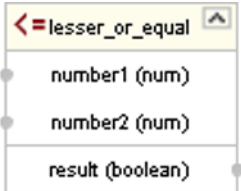
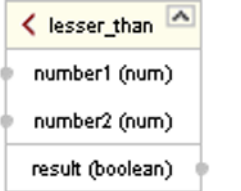
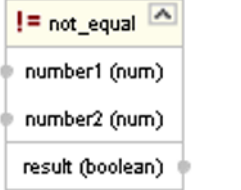


Table 33 Java Comparison Methods

Method Box	Description/Usage
	The equal method returns Boolean true if <i>number1</i> is equal to <i>number2</i> ; otherwise, returns Boolean false.
	The greater or equal method returns Boolean true if <i>number1</i> is greater than or equal to <i>number2</i> ; otherwise, returns Boolean false.
	The greater than method returns Boolean true if <i>number1</i> is greater than <i>number2</i> ; otherwise, returns Boolean false.

Table 33 Java Comparison Methods

Method Box	Description/Usage
 <p>The method box for <code>lesser_or_equal</code> shows a title bar with a left arrow and an up arrow. Below the title bar are three rows: <code>number1 (num)</code>, <code>number2 (num)</code>, and <code>result (boolean)</code>. Each row has a small grey circle on its left side, and the <code>result (boolean)</code> row has a small grey circle on its right side.</p>	<p>The lesser or equal method returns Boolean true if <i>number1</i> is less than or equal to <i>number2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for <code>lesser_than</code> shows a title bar with a left arrow and an up arrow. Below the title bar are three rows: <code>number1 (num)</code>, <code>number2 (num)</code>, and <code>result (boolean)</code>. Each row has a small grey circle on its left side, and the <code>result (boolean)</code> row has a small grey circle on its right side.</p>	<p>The lesser than method returns Boolean true if <i>number1</i> is less than <i>number2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for <code>not_equal</code> shows a title bar with an exclamation mark and an up arrow. Below the title bar are three rows: <code>number1 (num)</code>, <code>number2 (num)</code>, and <code>result (boolean)</code>. Each row has a small grey circle on its left side, and the <code>result (boolean)</code> row has a small grey circle on its right side.</p>	<p>The not equal method returns Boolean true if <i>number1</i> is not equal to <i>number2</i>; otherwise, returns Boolean false.</p>

6.4.3 Math Methods

Figure 102 Method Palette: Math Methods

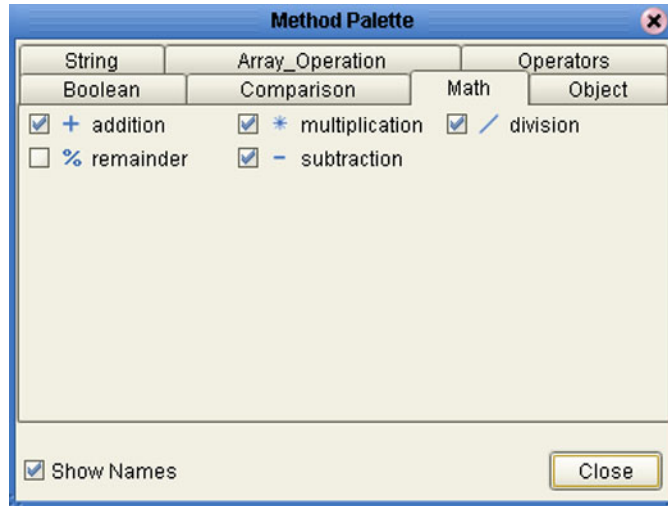


Table 34 Java Math Methods

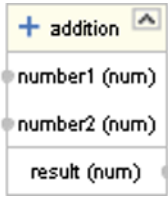
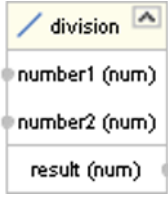
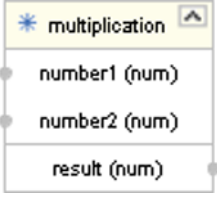
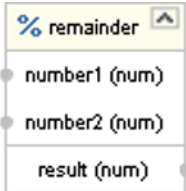
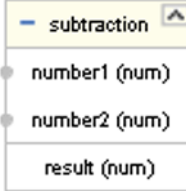
Method Box	Description/Usage
	The addition method adds the value of <i>number1</i> to the value of <i>number2</i> , returns the sum.
	The division method divides the value of <i>number1</i> by the value of <i>number2</i> , returns the quotient.
	The multiplication method multiplies the value of <i>number1</i> by the value of <i>number2</i> , returns the product.

Table 34 Java Math Methods

Method Box	Description/Usage
 <p>The remainder method box contains a title bar with a blue percentage icon and the text '% remainder' followed by a close button. Below the title bar are three input fields: 'number1 (num)', 'number2 (num)', and 'result (num)'. Each input field has a small grey dot on its left side.</p>	<p>The remainder method divides the numerical value of <i>number1</i> by the numerical value of <i>number2</i>, returns the remainder.</p>
 <p>The subtraction method box contains a title bar with a blue minus icon and the text '- subtraction' followed by a close button. Below the title bar are three input fields: 'number1 (num)', 'number2 (num)', and 'result (num)'. Each input field has a small grey dot on its left side.</p>	<p>The subtraction method subtracts the numerical value of <i>number2</i> from the numerical value of <i>number1</i>, returns the difference.</p>

6.4.4 Object Methods

Figure 103 Method Palette: Object Methods

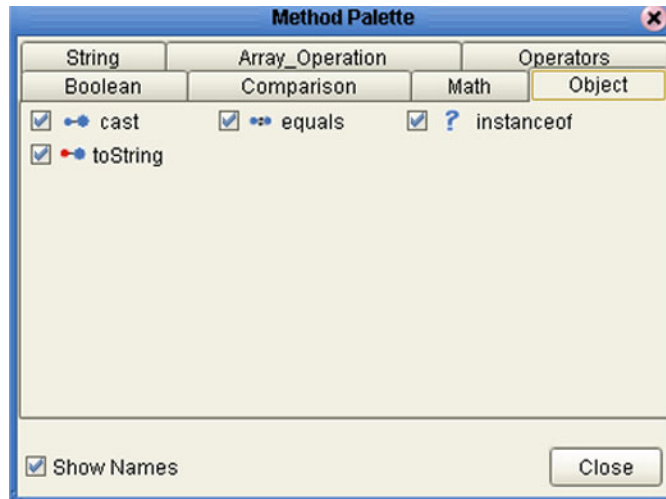


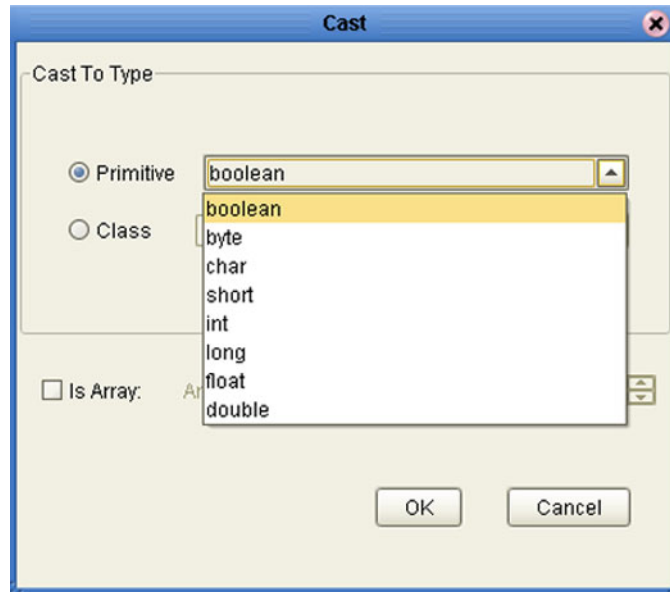
Table 35 Java Object Methods

Method Box	Description/Usage
	<p>Dragging the icon first displays the CAST dialog box (see Figure 104 on page 142), in which you specify the desired type (can be either primitive or class). The cast method then converts the given type (<i>any</i>) to the type specified in the CAST dialog box.</p>
	<p>The equals method returns Boolean true if <i>Object</i> is logically equal to <i>obj</i>, otherwise, returns Boolean false.</p>
	<p>Dragging the icon first displays the InstanceOf dialog box (see series beginning with Figure 105 on page 142), in which you specify the desired type. The instanceof method returns Boolean true if <i>Object</i> is of the specified type; otherwise, returns Boolean false .</p>
	<p>The toString method converts the object in the input field into a string.</p>

Object Method Dialog Boxes

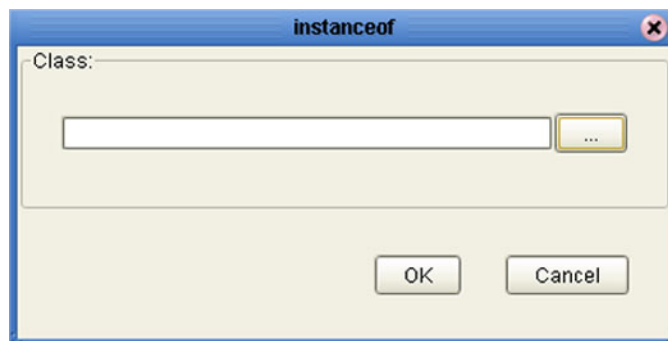
CAST Method

Figure 104 CAST Dialog Box



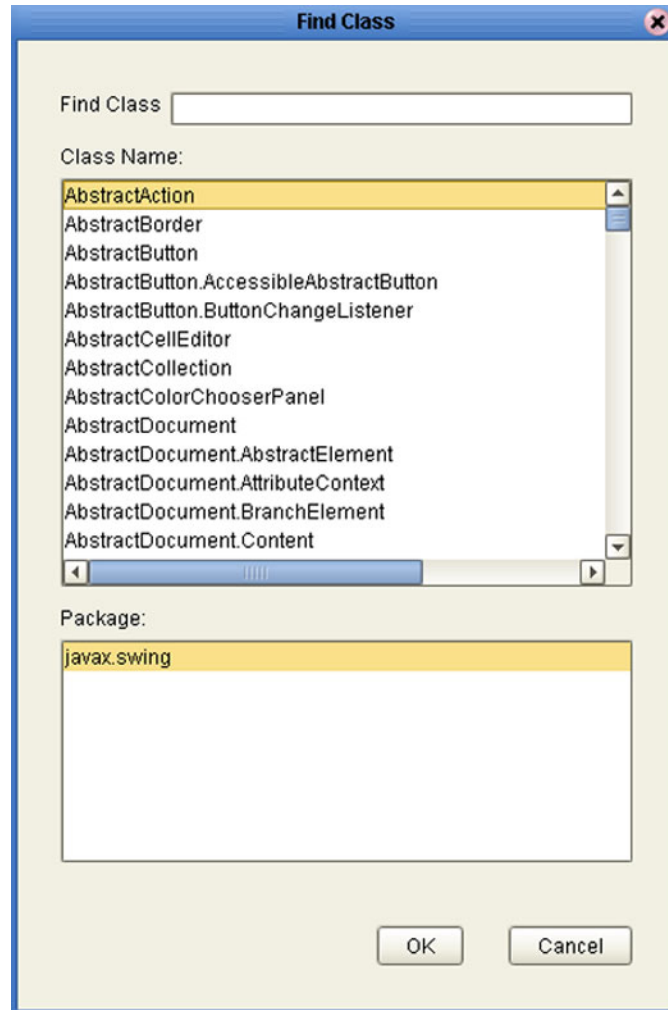
InstanceOf Method

Figure 105 InstanceOf Dialog Box (1)



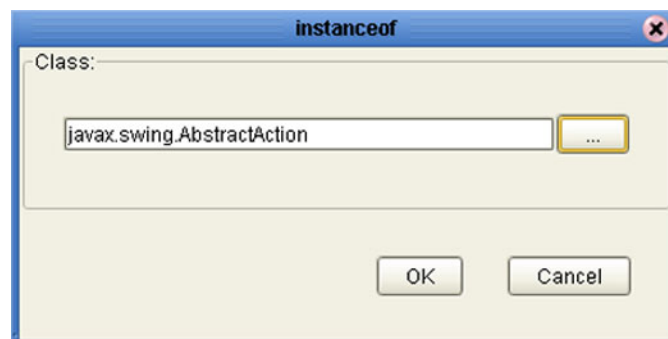
Clicking the ellipsis (...) button displays the *Find Class* dialog box (see [Figure 105 on page 142](#)), in which you select the desired class name.

Figure 106 Find Class Dialog Box



Clicking **OK** returns you to the *InstanceOf* dialog box (see Figure 107), with the field displaying *Package Name Class Name*.

Figure 107 InstanceOf Dialog Box (2)



Clicking **OK** displays the *InstanceOf* method box.

6.4.5 String Methods

Figure 108 Method Palette: String Methods

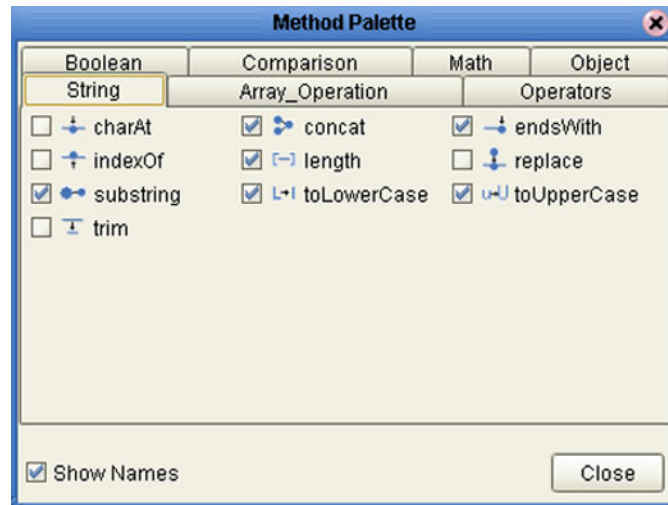


Table 36 Java String Methods

Method Box	Description/Usage
	<p>The charAt method returns the character at the specified index, where <i>index</i> is the number of characters from the beginning of <i>String</i>.</p>
	<p>The concat method returns the string created by concatenating <i>str</i> to the end of <i>String</i>.</p>
	<p>The endsWith method returns Boolean true if <i>String</i> ends with the string <i>suffix</i>; otherwise, returns Boolean false.</p>

Table 36 Java String Methods

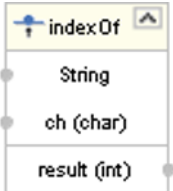



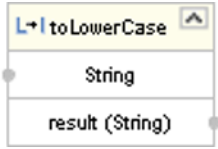
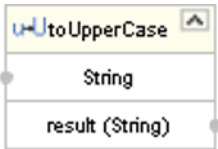
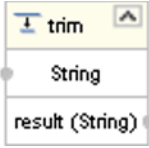
Method Box	Description/Usage
	<p>The indexOf method returns the index within <i>String</i> corresponding to the location of the specified character (<i>ch</i>), where the index represents the number of characters from the beginning of <i>String</i>.</p>
	<p>The length method returns the length (number of characters) of <i>String</i>.</p>
	<p>The replace method returns a new string in which all occurrences of <i>oldChar</i> are replaced with <i>newChar</i>.</p>
	<p>The substring method returns a string that is a substring of <i>String</i>, beginning from <i>beginIndex</i> (inclusive) and ending at <i>endIndex</i> (exclusive). The indices represent the number of characters from the beginning of <i>String</i>.</p>
	<p>The toLowerCase method converts all characters in <i>String</i> to lower case, using the rules of the default locale.</p>
	<p>The toUpperCase method converts all characters in <i>String</i> to upper case, using the rules of the default locale.</p>

Table 36 Java String Methods

Method Box	Description/Usage
 A screenshot of a Java IDE's method box for the <code>trim</code> method. The box has a yellow header with the text <code>trim</code> and a small icon. Below the header, the parameter <code>String</code> is listed with a small circle to its left. Below the parameter, the return type <code>result (String)</code> is listed with a small circle to its right.	The trim method trims leading and trailing whitespace from <i>String</i> .

6.4.6 Array Operation Methods

Figure 109 Method Palette: Array Operation Methods

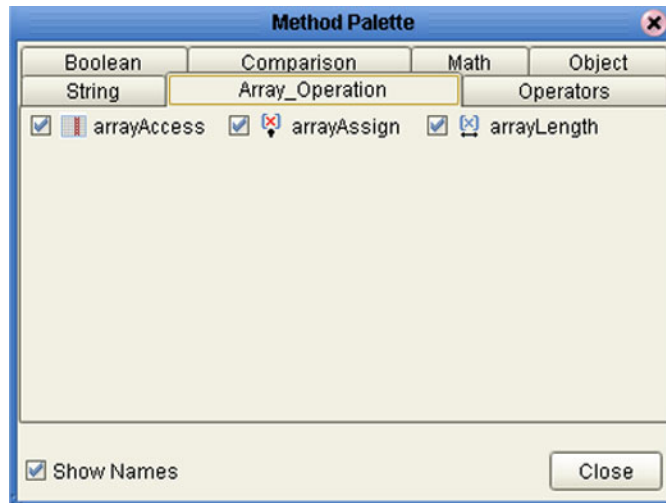
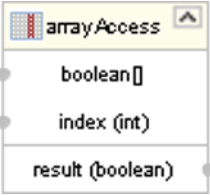

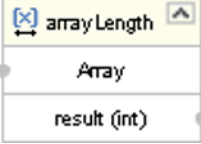


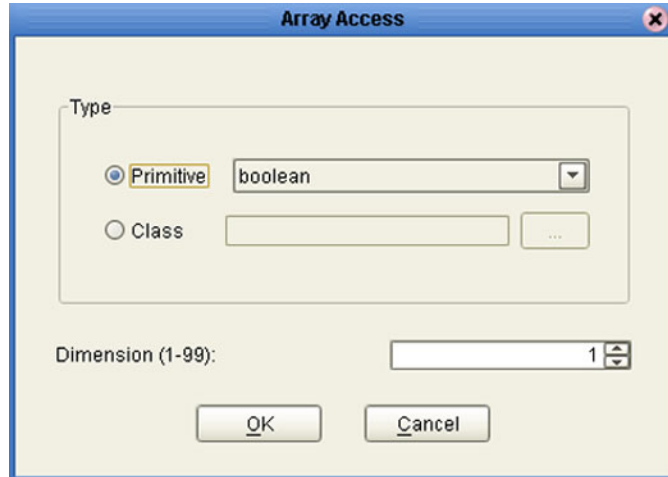
Table 37 Java Array Operation Methods

Method Box	Description/Usage
	<p>Dragging the icon first displays the Array Access dialog box (see Figure 110 on page 148), in which you specify the desired array type (can be either primitive or class) and dimension. The arrayAccess method then returns the element located at the specified index within the array, where the index represents the number of elements from the beginning of the array.</p>
	<p>Dragging the icon first displays the Array Assign dialog box (see Figure 111 on page 148), in which you specify the desired array type (can be either primitive or class) and dimension. The arrayAssign method then sets the element located at the specified index within the array, where the index represents the number of elements from the beginning of the array.</p>
	<p>The arrayLength method returns the length (number of elements) of the array.</p>

Array Operation Method Dialog Boxes

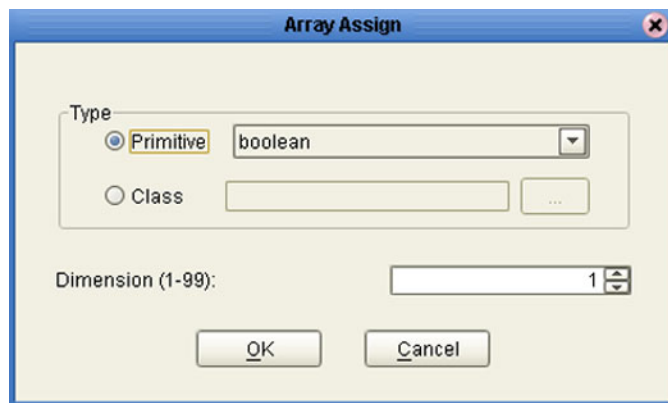
arrayAccess Method

Figure 110 Array Access Dialog Box



arrayAssign Method

Figure 111 Array Assign Dialog Box



6.4.7 Operator Methods

Figure 112 Method Palette: Operator Methods

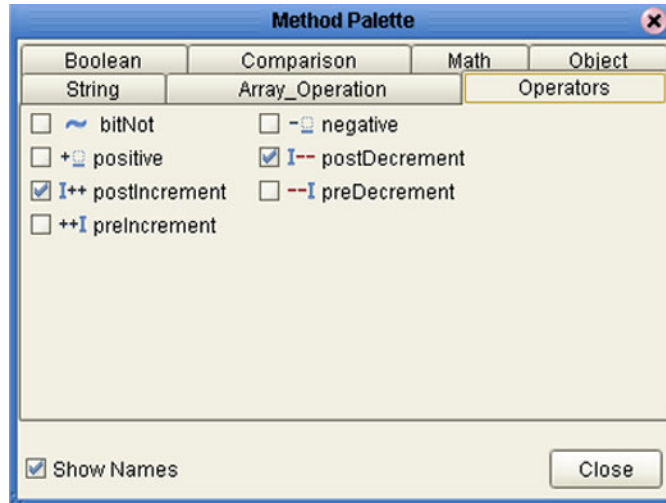
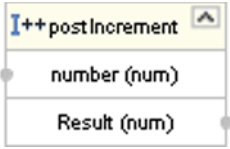
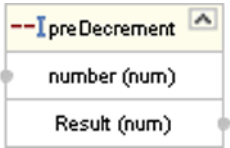
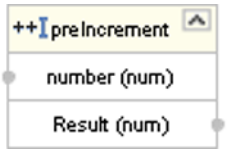


Table 38 Java Operator Methods

Method Box	Description/Usage
	The bitNot method returns the inverse of the bit value of the input number.
	The negative method returns the arithmetic negation of the input number.
	The positive method returns the value of the input number.
	The postDecrement method decrements the input number by one, after other operations.

Table 38 Java Operator Methods

Method Box	Description/Usage
	<p>The postIncrement method increments the input number by one, after other operations.</p>
	<p>The preDecrement method decrements the input number by one, before other operations.</p>
	<p>The preIncrement method increments the input number by one, before other operations.</p>

6.5 Java Encapsulation

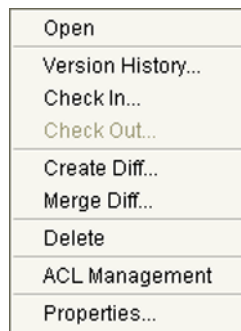
Java Collaboration Definitions can be saved to a Diff (.sdf) file. This feature allows two sites working with the same Java Collaboration Definition to seamlessly merge changes.

6.5.1 Creating a Modified Java Collaboration Definition

This section explains how to create a *different* version of a Java Collaboration Definition file.

- 1 From the Project Explorer tab, select a **Java Collaboration Definition** icon.
- 2 Right-click to display the context menu shown in Figure 113.

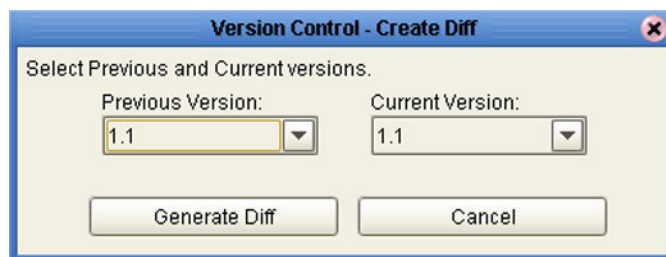
Figure 113 Java Collaboration Definition Context Menu



The options on the Java Collaboration Definition context menu are similar to those on the Connectivity Map context menu. See [Connectivity Map Menu Options](#) on page 41 for more details about the other options on this menu.

- 3 Select **Create Diff** to display the dialog box shown in Figure 114.

Figure 114 Version Control - Create Diff Dialog Box



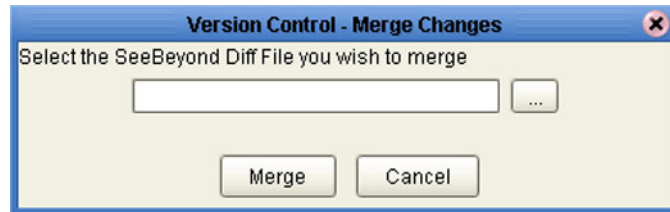
- 4 Click **Generate Diff** to display the *Specify Name ...* dialog box.
- 5 Enter a new name for the Diff file in the File Name box if you do not want to use the default file name.
- 6 Click **Save** to save the Diff file.

6.5.2 Merging Two Versions of a Java Collaboration Definition

This section explains how to merge two versions of a Java Collaboration Definition.

- 1 From the Project Explorer tab, select a **Java Collaboration Definition** icon.
- 2 Right-click to display the Java Collaboration Definition context menu.
- 3 Select **Merge Diff...** to display the dialog box shown in Figure 115.

Figure 115 Version Control - Merge Changes Dialog Box



- 4 Click the **Ellipsis (...)** button to display the *Specify Name ...* dialog box.
- 5 Locate and select the Diff file.
- 6 Click **Open** add the Diff file to the dialog box shown in Figure 115.
- 7 Click **Merge** to merge the Diff file with the corresponding Java Collaboration Definition in your Project.

6.6 Setting Up a Java Collaboration Definition Variable

The Java Collaboration Editor includes a variable creation feature. The following sections describe how to set up and assign a constructor to a variable.

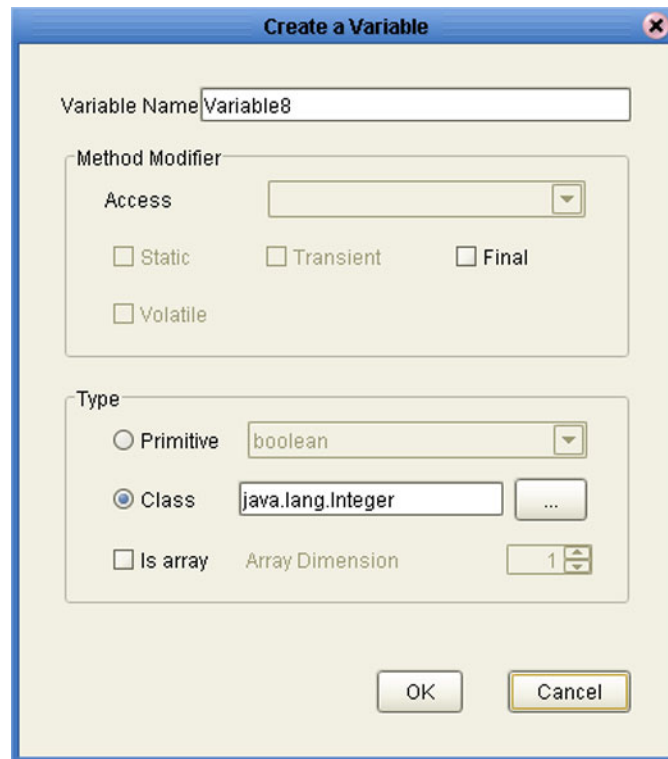
6.6.1 Creating a Variable

This section describes how to add a variable to a Java Collaboration Definition.

To create the variable

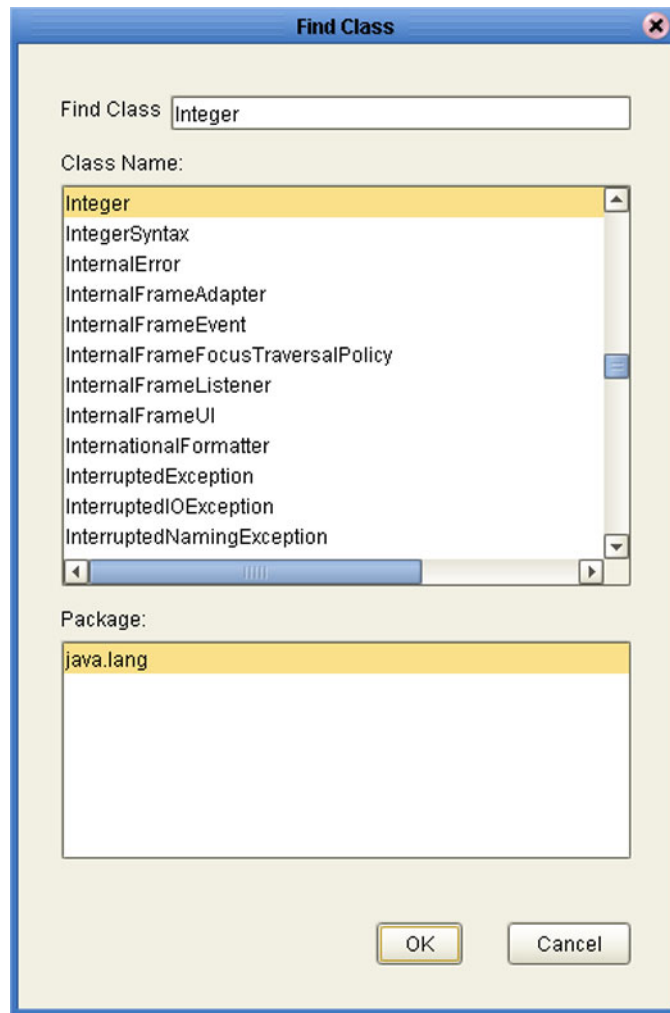
- 1 In the Business Rules area, click the **Local Variable** button to display the dialog box shown in Figure 116.

Figure 116 Create a Variable Dialog Box



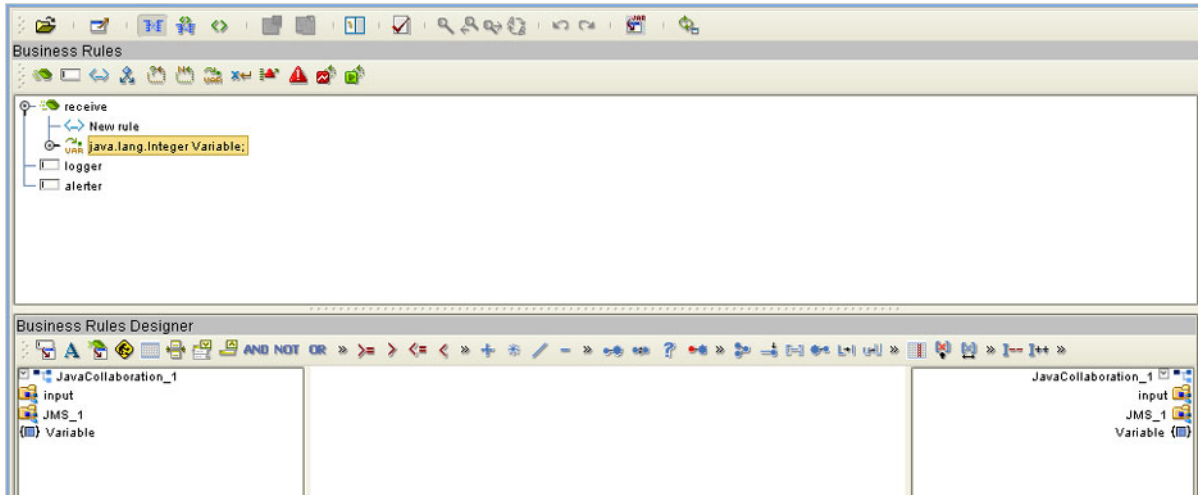
- 2 Enter a name for the variable in the **Variable Name** box.
- 3 Select the **Class** option button.
- 4 Click the **Ellipsis (...)** button to display the **Find Class** dialog box shown in Figure 117.

Figure 117 Find Class Dialog Box



- 5 Select a class from the **Class Name** list.
- 6 Click the **OK** button to add the variable, with selected class, to the Java Collaboration Editor window as shown in Figure 118.

Figure 118 Java Collaboration Definition with Variable



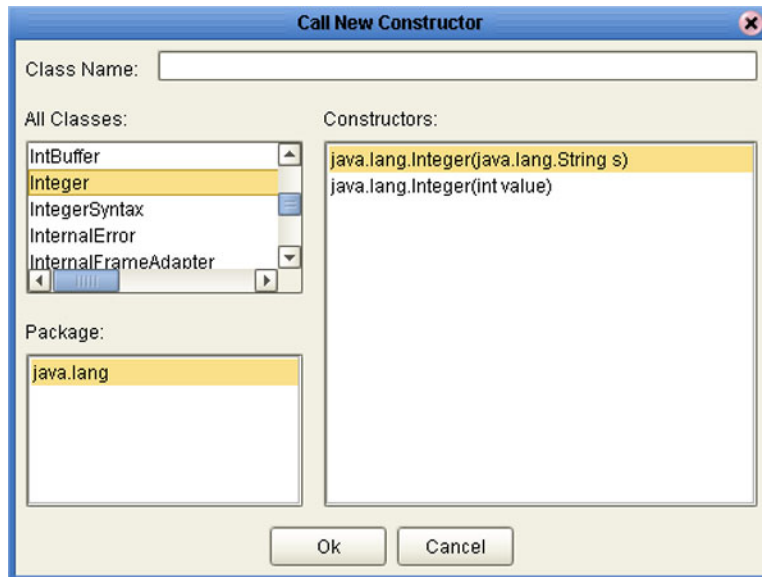
6.6.2 Invoking Variable Constructor

This section describes how to invoke a constructor for the variable.

To invoke a constructor

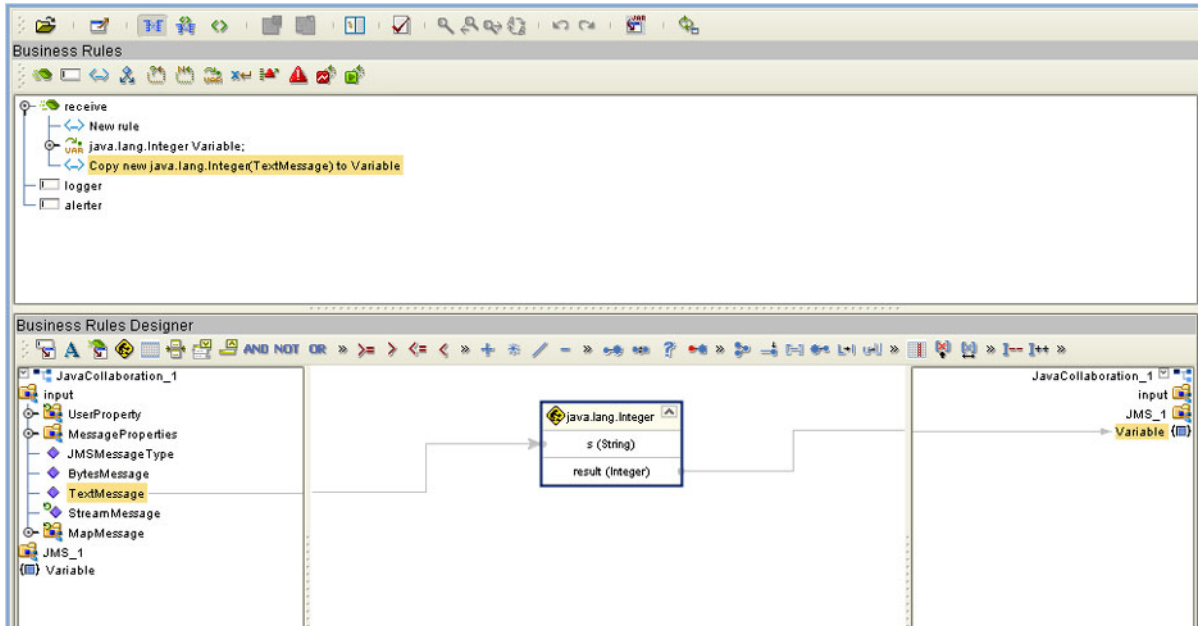
- 1 In the Business Rules Designer area, click the **Call New Constructor** button to display the dialog box shown in Figure 119.

Figure 119 Call New Constructor Dialog Box



- 2 Select the class used for the variable in the **All Classes** list.
- 3 Click **OK** to add an integer box to the Business Rules Designer area as shown in Figure 120.
- 4 Link an item in the left column through the integer to an item in the right column.

Figure 120 Java Collaboration Definition with Constructor



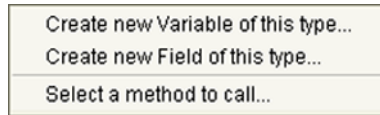
6.6.3 Displaying Method Classes

This section describes how to display a list of method classes.

To view the available method classes

- 1 Select a Variable in the Business Rules Designer area.
- 2 Right-click to display its context menu, as shown in Figure 121

Figure 121 Variable Context Menu



- 3 Click **Select a Method to Call** to display a list of method classes as shown in Figure 122.

Figure 122 Method Selection List Box



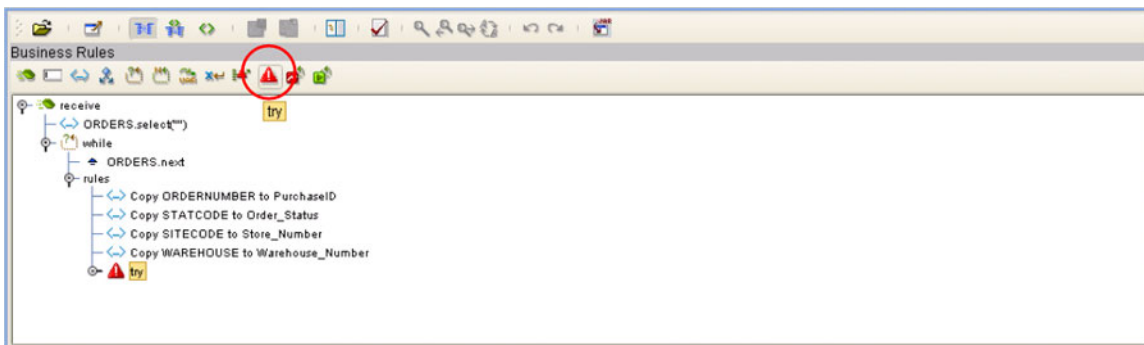
6.7 Using Try-Catch

Clicking the **try** icon in the toolbar adds a **try** statement to the Business Rules tree, initiating a number of programming statements that are monitored to see whether they succeed or fail. A **finally** statement is added automatically; if you want to perform a **try-catch**, you must use the following procedure.

To create a try-catch operation

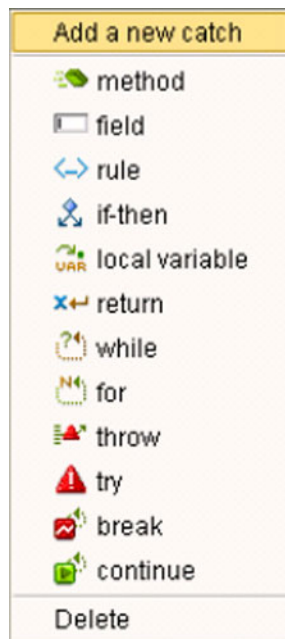
- 1 Click the **try** icon in the toolbar (see Figure 123), which adds the **try** rule to the business rules tree.

Figure 123 Try Icon



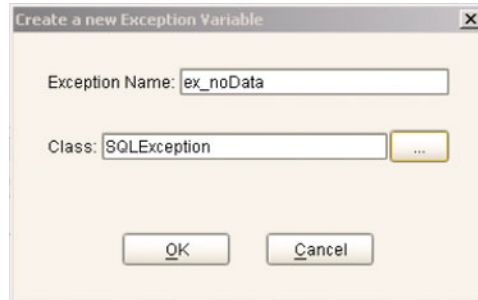
- 2 Right-click on the **try** rule to display the catch context menu (see Figure 124).

Figure 124 Catch Context Menu



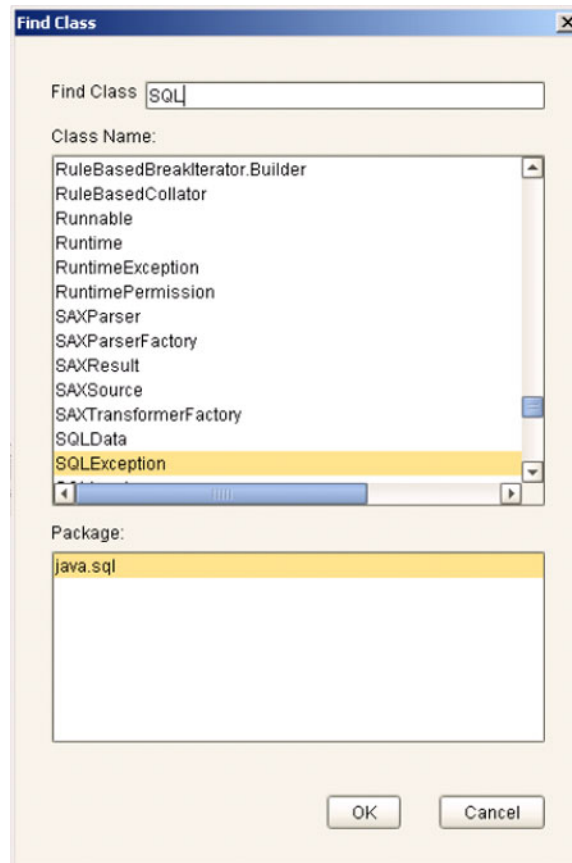
- 3 Select the **Add a new catch** option to add an exception variable, which displays a dialog box (see Figure 125).

Figure 125 Create a New Exception Variable Dialog Box



- 4 Enter a name for the exception, for example: **ex_noData**.
- 5 Click the button to the right of the Class text box to display a list of available classes (see Figure 126).

Figure 126 Find Class Dialog Box



- 6 To specify that the exception represents a database error, for example, select **SQLException** and click **OK** on both dialog boxes.

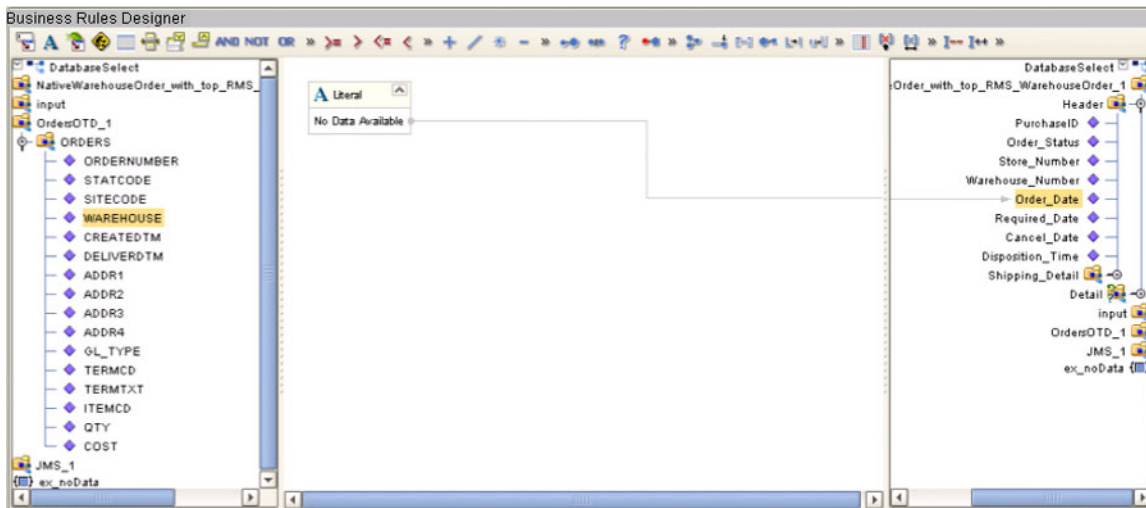
- 7 The catch rule is now added to the business rules tree (see Figure 127).

Figure 127 Catch SQLException Rule



- 8 The rule shown in Figure 127 contains a literal that represents the message that will be assigned to an outbound element Order_Date when the rule is executed (see Figure 128).

Figure 128 Exception Message



6.8 Adding and Using Third-Party Java Classes

Enterprise Designer allows you to add any file to the repository. Certain products in the ICAN Suite can then instruct the repository to distribute these files to Logical Hosts as needed. Selecting **New > File ...** in a Project context menu (see Figure 129) displays a file selection dialog box (see Figure 130).

Figure 129 Project Context Menu: New File ...

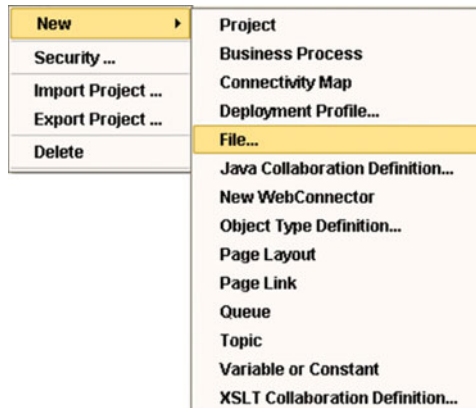
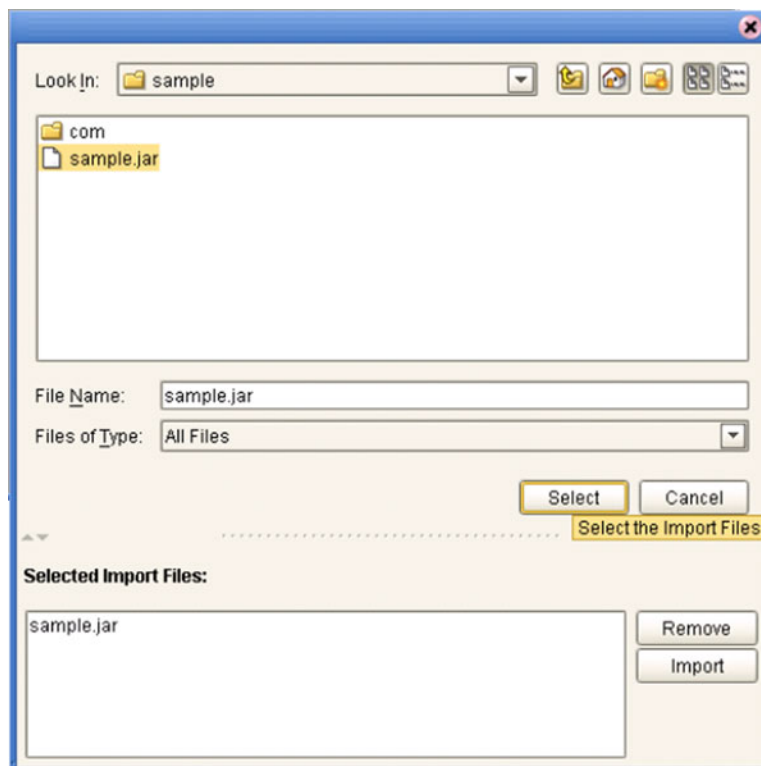


Figure 130 File Selection Dialog Box

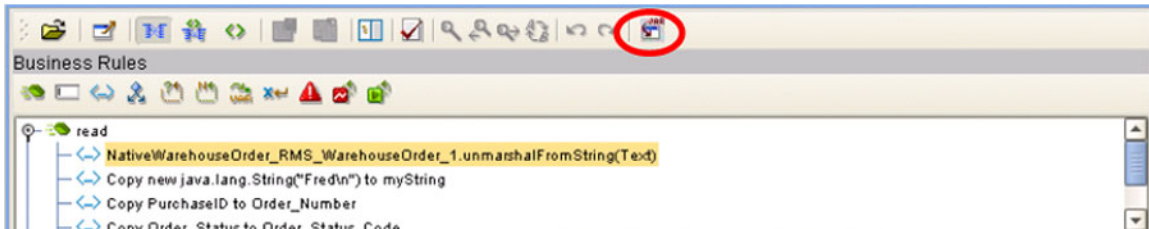


Locate the desired .jar files and click **Select** for each; then click **Import**.

To add the file(s) to a Collaboration Definition:

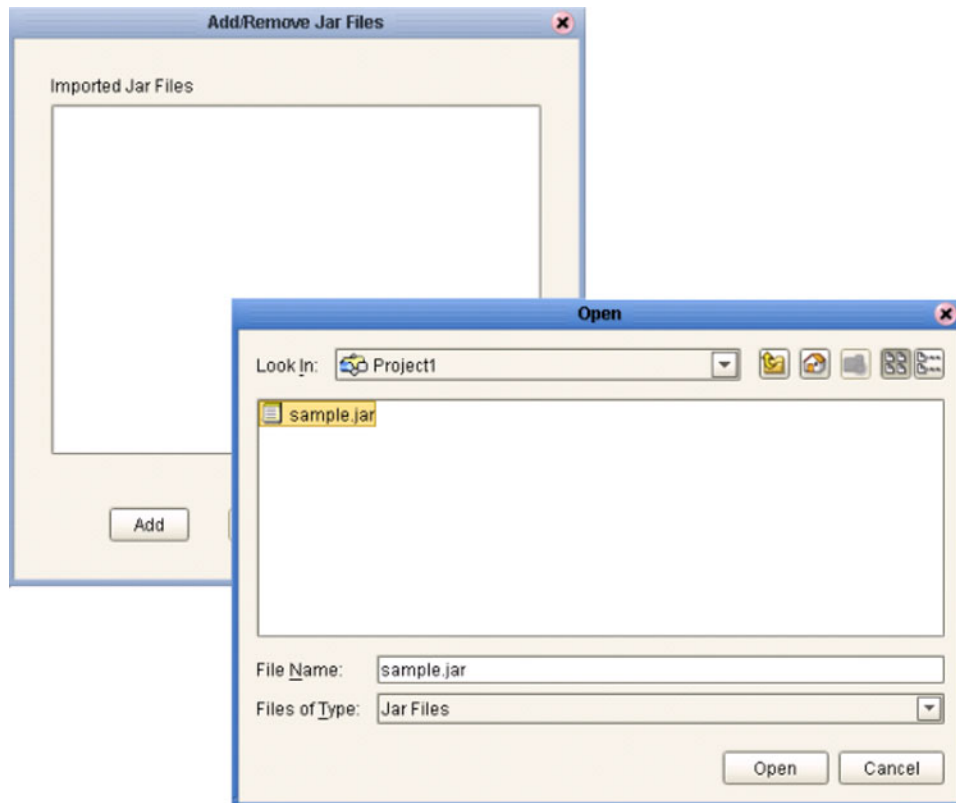
- 1 Open or create a Collaboration.
- 2 Click the **Import JAR File** icon in the toolbar (see Figure 131), which displays an *Add/Remove Jar Files* dialog box.

Figure 131 Import JAR File Icon



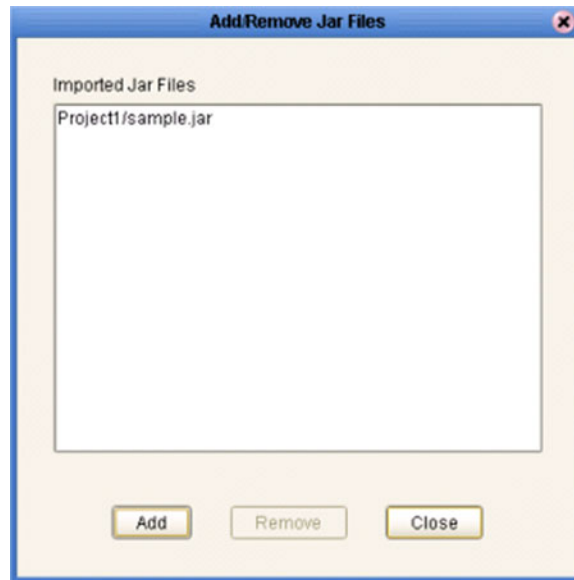
- 3 Click **Add** in the dialog box to display a file browser box (see Figure 133).
- 4 Locate the desired .jar files and click **Open** for each.

Figure 132 Add/Remove Jar File Dialog Box (1)



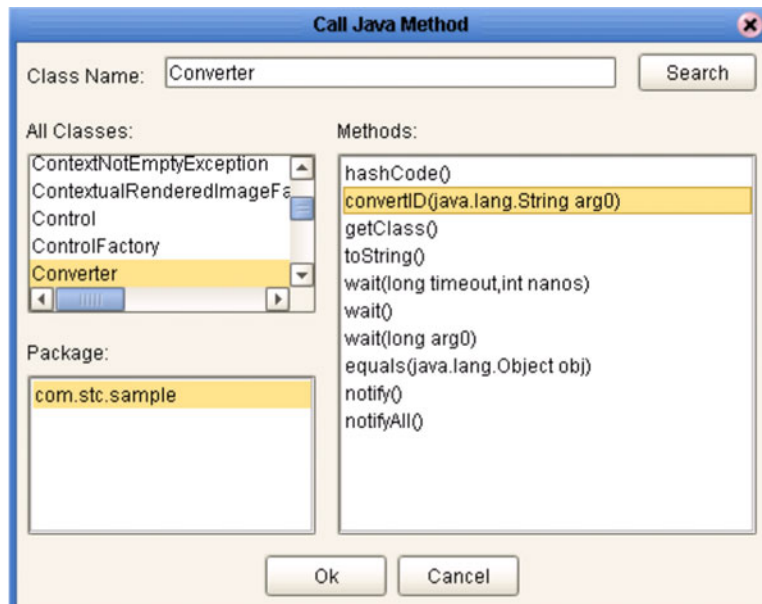
- 5 When all the .jar files are listed in the *Add/Remove Jar Files* dialog box (see Figure 133), click **Close**.

Figure 133 Add/Remove Jar File Dialog Box (2)



- 6 Click the **Call Java Method** icon in the Business Rules Designer, and a *Call Java Method* dialog box opens (see Figure 134).

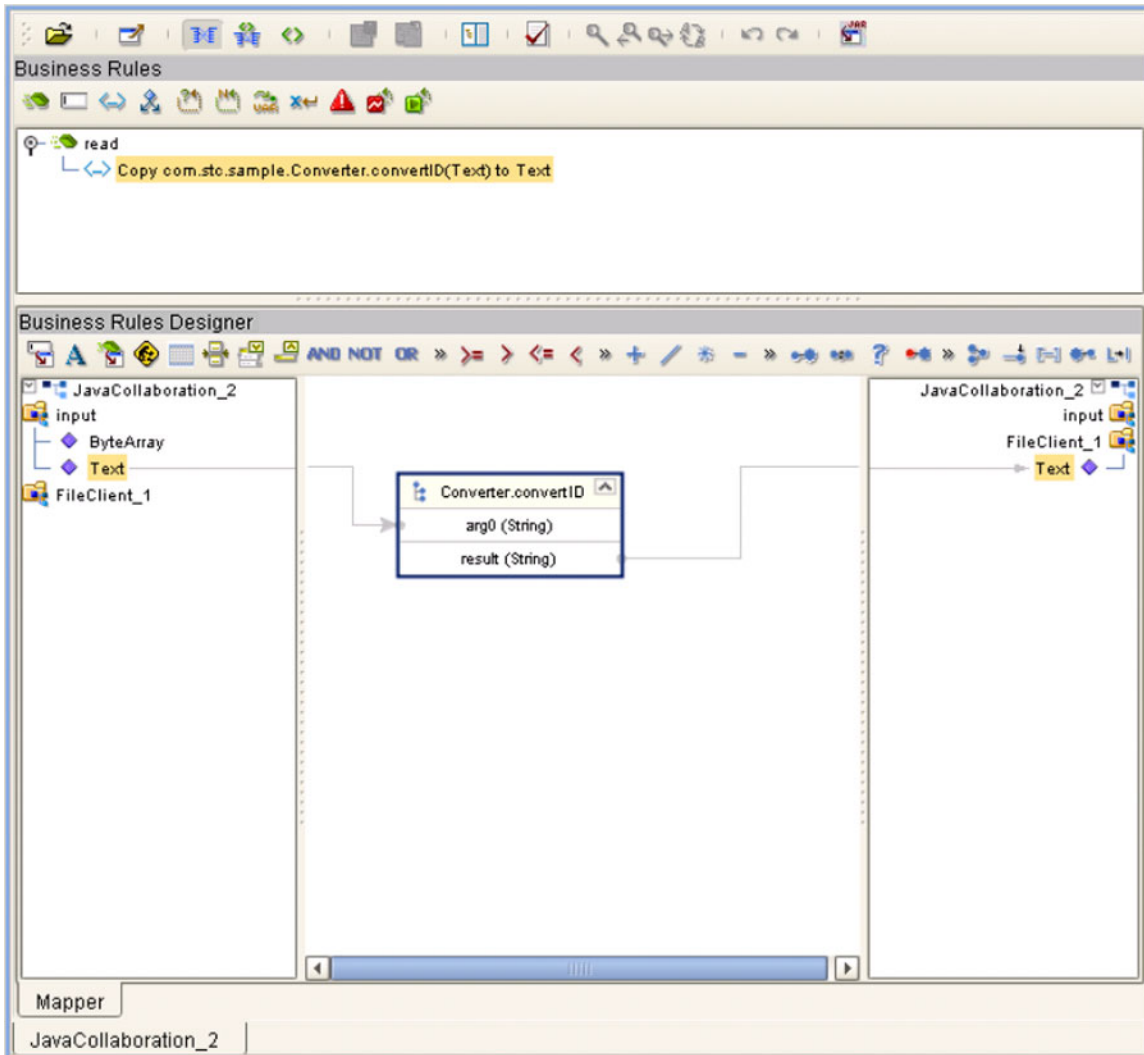
Figure 134 Call Java Method Dialog Box



- 7 Select the desired class from the **All Classes** list.
- 8 Select the desired method from the **Methods** list.
- 9 Click **Ok**.

To use the third-party method, add the business rule containing the method. In the example shown, the method is a **convertID** operation that converts text from upper case to lower case. Connect an inbound text file to the input node of the method, and the outbound node to an outbound text file (see Figure 135).

Figure 135 Using the Third-Party Java Method



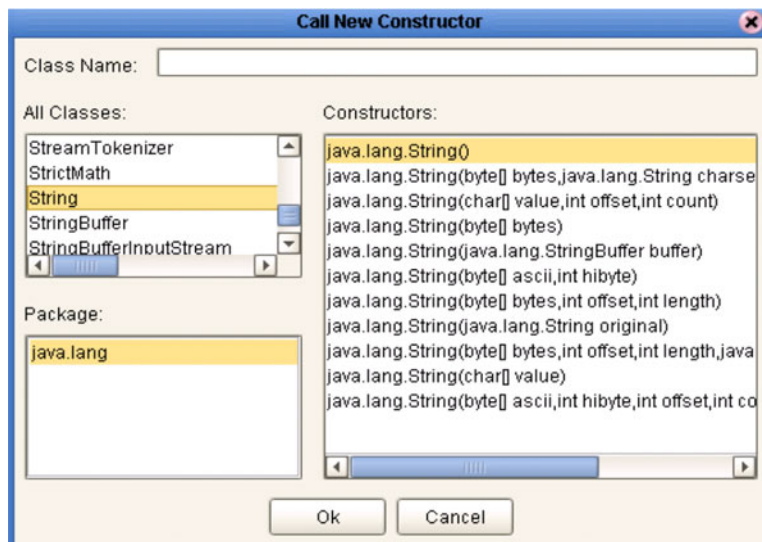
6.9 Adding Class Instances to a Collaboration

Instances of Java classes are added by means of the Java constructors, as described in the following procedure.

To add a class instance to a Collaboration

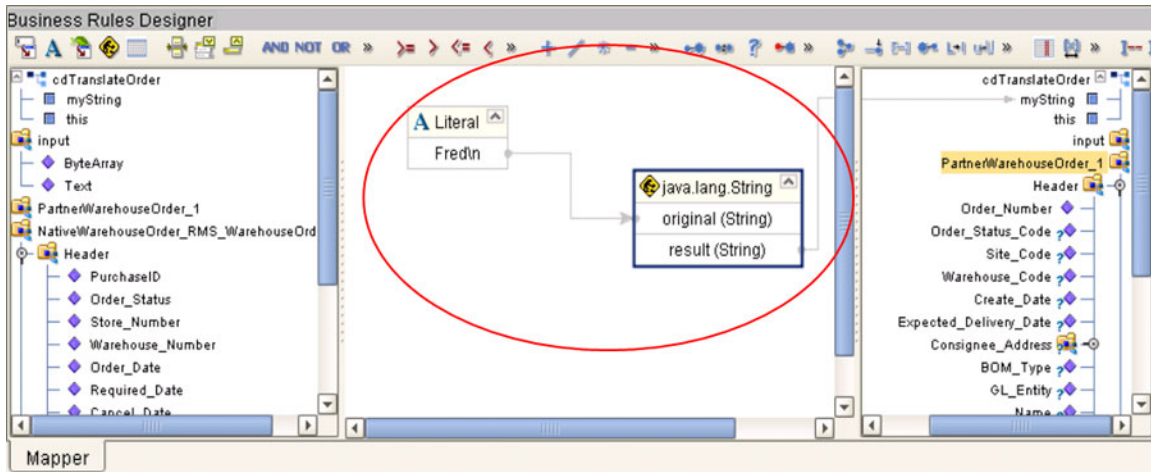
- 1 Create or open a Collaboration and create a new rule.
- 2 Click the **Constructor** icon on the toolbar (see [Business Rules Designer Toolbar Icons](#) on page 131) to display the *Call New Constructor* dialog box (see Figure 136). This dialog box presents all available classes, including any third-party classes that have been uploaded.

Figure 136 Call New Constructor Dialog Box



- 3 Select a constructor method from the Constructors list and click **OK**. This method is placed on the Business Rules Designer's canvas, and can now be used as shown in Figure 137.

Figure 137 Constructor Example 1

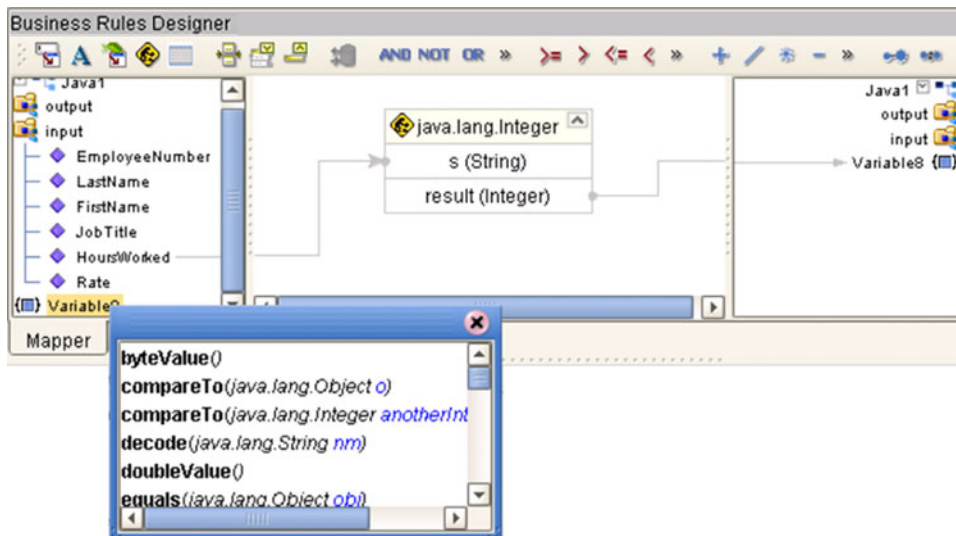


Alternative procedure to add a class instance to a Collaboration

An alternative way to invoke a constructor is to right-click on an element in the left panel of the Business Rules Designer and select a constructor method from the list box that appears, as illustrated in Figure 138. The procedure is:

- 1 Create a local string variable.
- 2 Right-click on the variable element.
- 3 Select a constructor from the list box.

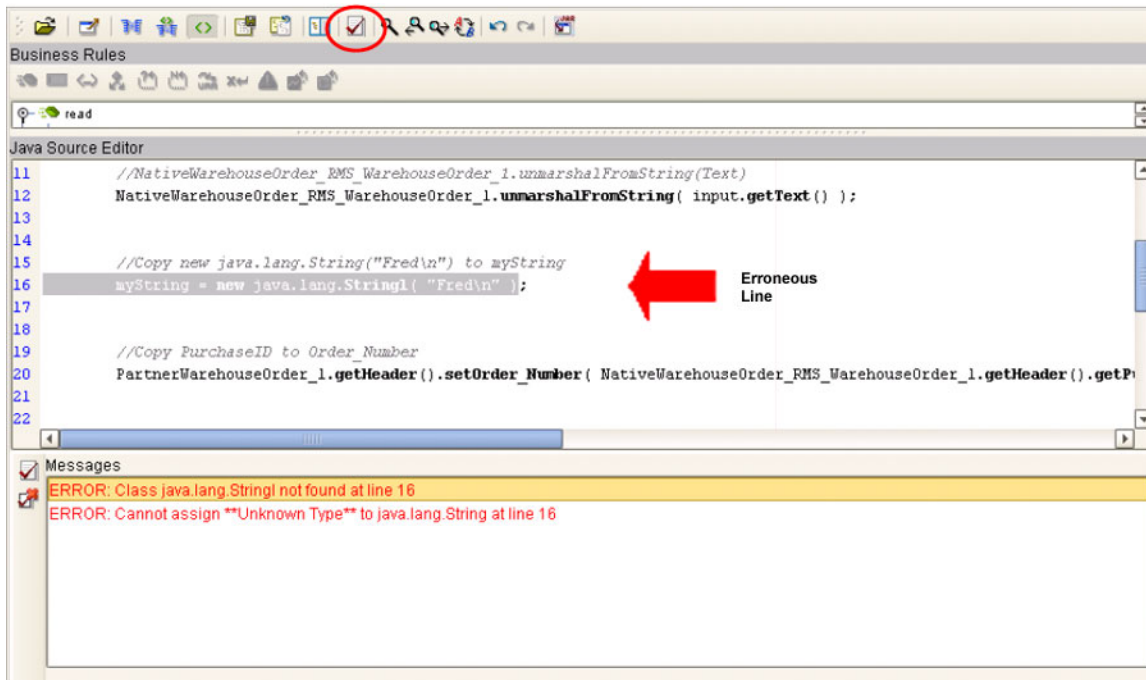
Figure 138 Constructor Example 2



6.10 Validating Java Collaborations

Clicking the **Validate** icon allows you to “precompile” the Java Collaboration Definition and display the errors in a Validate panel, as shown in Figure 139. To locate the error, double-click on the error message and the Java Source Editor will be displayed, showing the erroneous line of code.

Figure 139 Validating a Collaboration Definition



6.11 Debugging Java Collaborations

The Java Debugger enables you to debug Java Collaborations as deployed within an integration server on a Logical Host, and offers an alternative to creating logs and warnings in an Java Collaboration and subsequently inspecting them via the Enterprise Manager. See [The Java Debugger](#) on page 286.

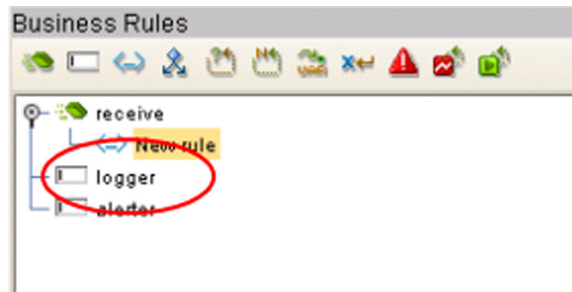
6.12 Creating Log Entries

You can create log file entries in the Java Collaboration Editor by means of the following procedure.

To create log entries using the JCE

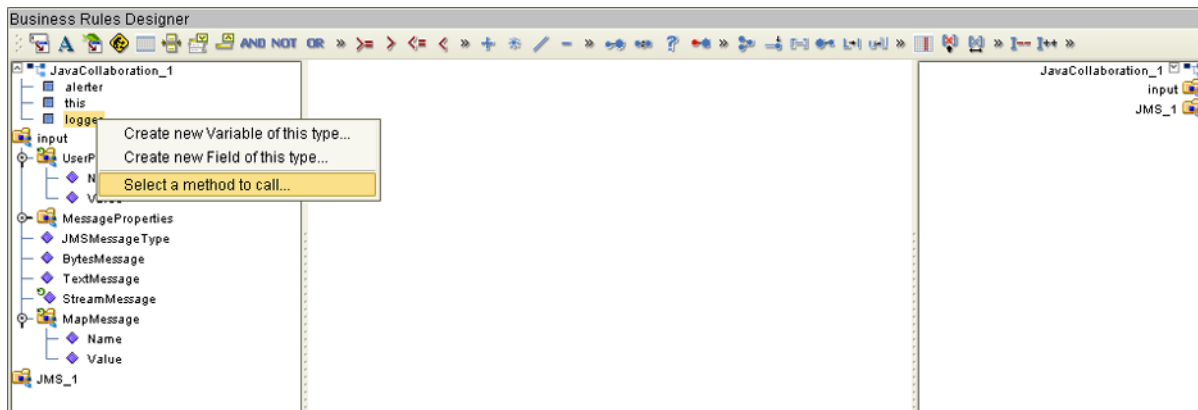
- 1 Create a new Collaboration.
A logger field is added automatically in the Business Rules pane (see Figure 140).

Figure 140 Logger Field



- 2 Add your Collaboration rules and initiate a logging event.
- 3 Right-click the logger node of the Collaboration in the left pane of the Business Rules Designer, which displays a menu (see Figure 141).

Figure 141 Logging Menu



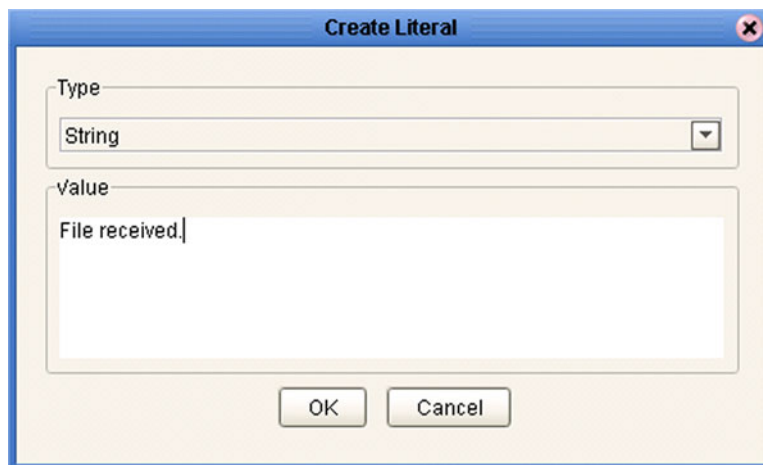
- 4 Click *Select a method to call ...*
- 5 Select the logging level with desired method from the selection window (see Figure 142). As an example, we will select **Debug**.

Figure 142 Logging Level/Method Selection Window



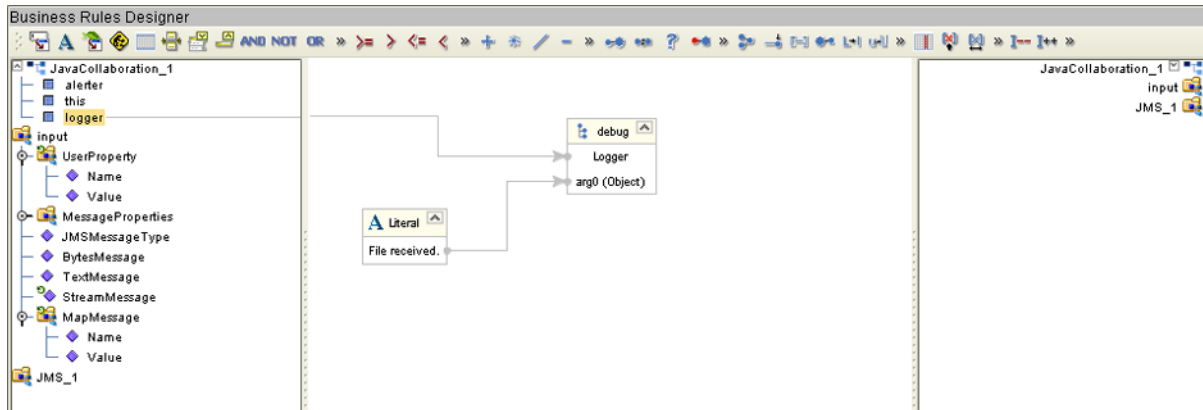
- 6 Create your log message, which can be a literal, a constant, or an OTD field name. As an example, we will create a **literal** (see Figure 143).

Figure 143 Create Literal Dialog Box



- 7 Pass the log message to the logging event by dragging the message to the argument of the logging object (see Figure 144).

Figure 144 Pass Log Message to Object Argument



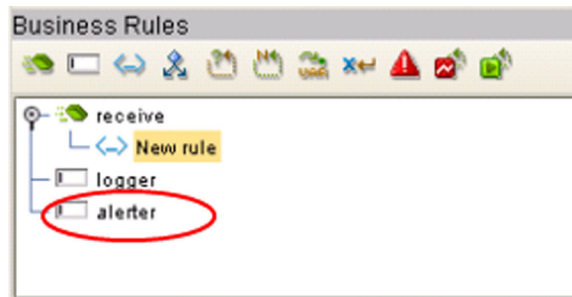
6.13 Creating Alerts

You can create alerts in the Java Collaboration Editor by means of the following procedure.

To create alerts using the JCE

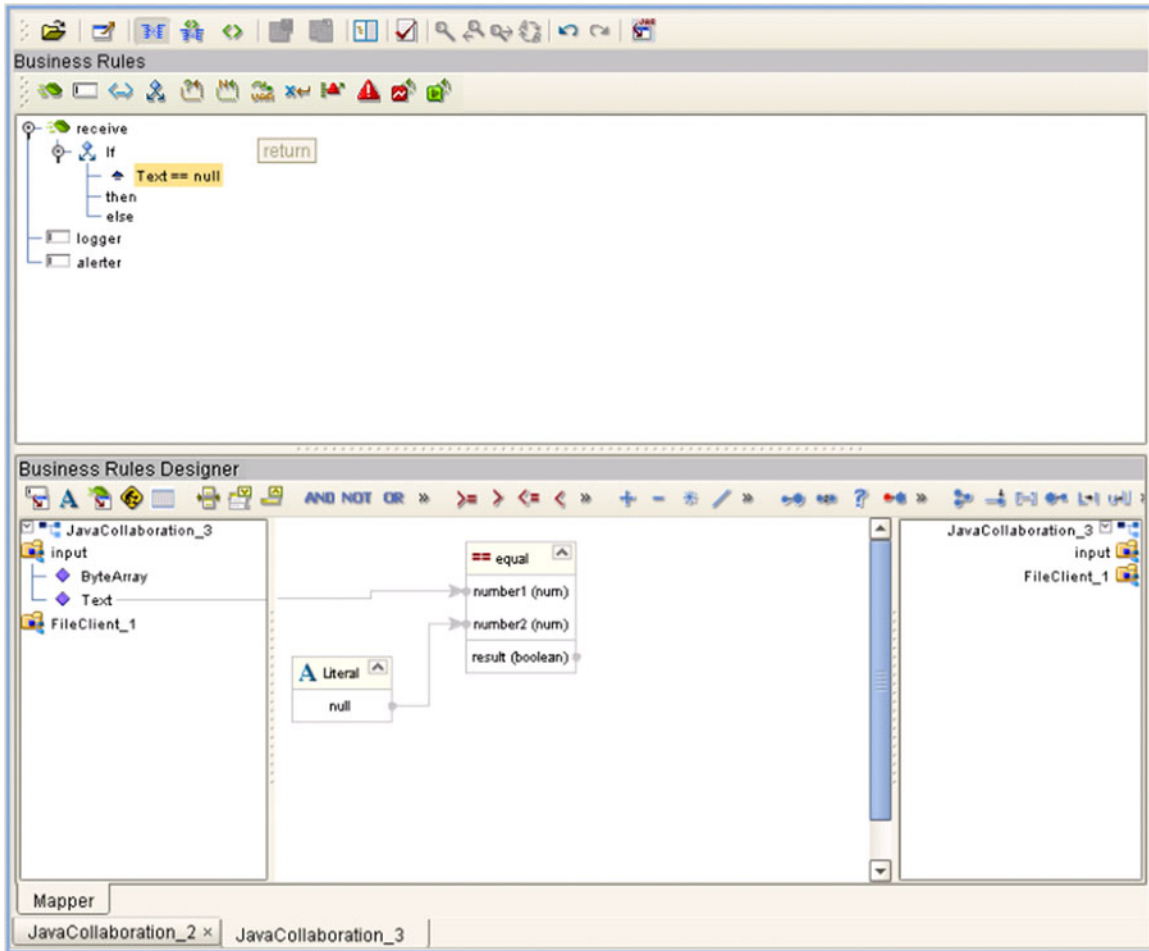
- 1 Create a new Collaboration.
An alerter “field” is added automatically in the Business Rules pane (see Figure 145).

Figure 145 Alerter Field



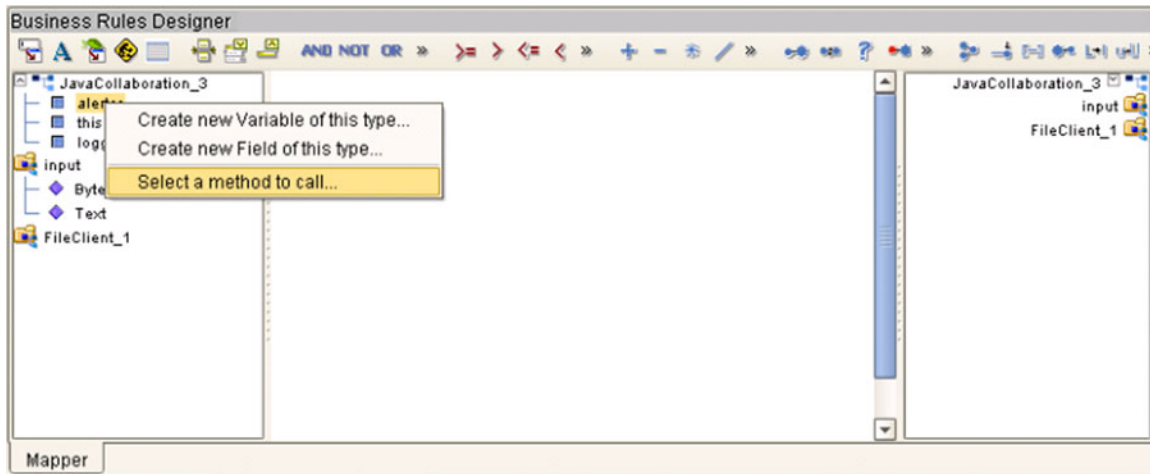
- 2 Add your business rules. As an example, we perform a test to determine whether or not a file is empty; if it is empty, we raise an alert (see Figure 146).

Figure 146 Empty File Test



- 3 Initiate an alert object.
- 4 Right-click the alerter node of the Collaboration in the left pane of the Business Rules Designer, which displays a menu (see Figure 147).

Figure 147 Alert Menu



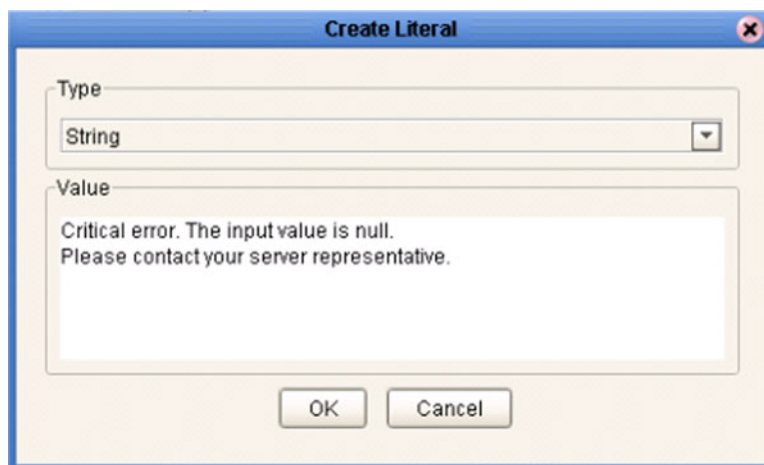
- 5 Click *Select a method to call ...*
- 6 Select the severity of the alert from the selection window (see Figure 148). As an example, we will select **Critical**.

Figure 148 Alert Severity Selection Window



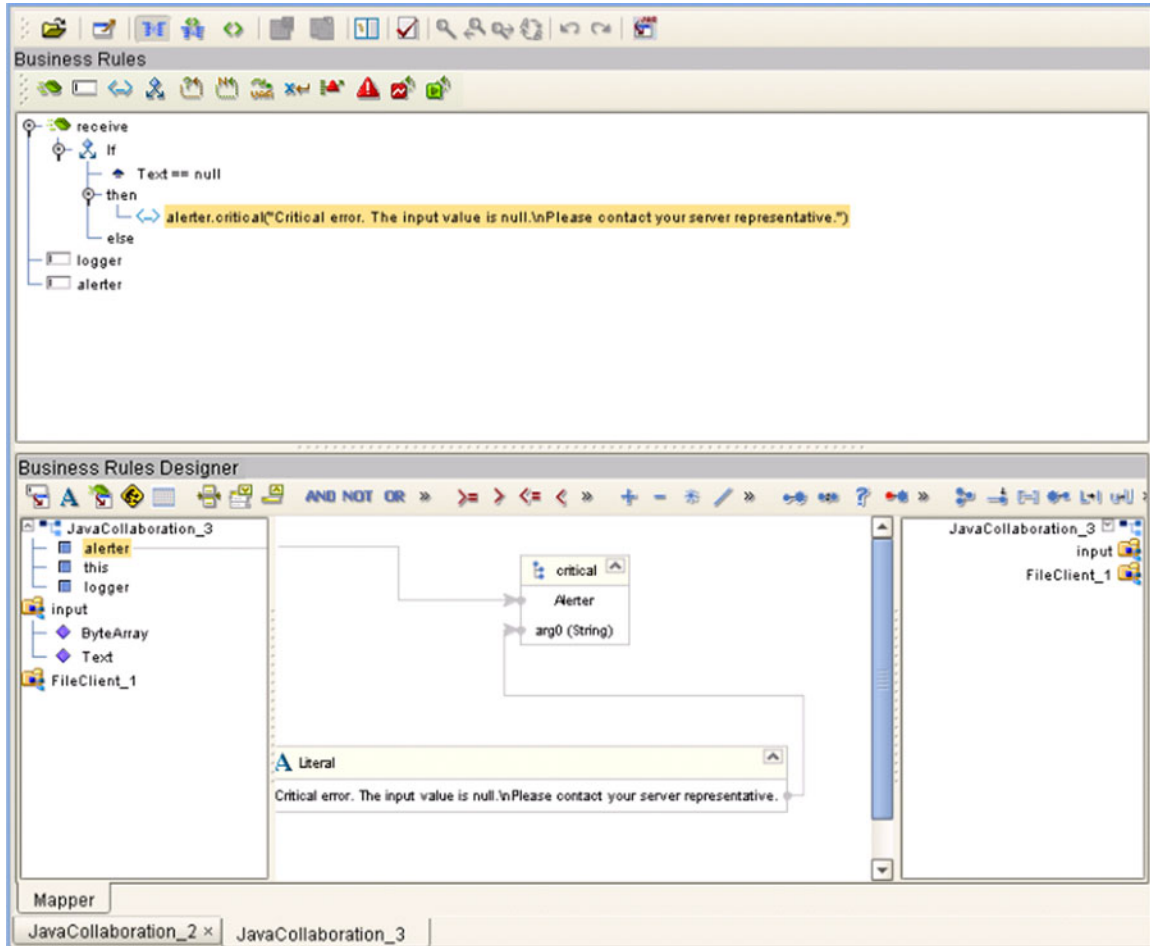
- 7 Create your alert message, which can be a literal, a constant, or an OTD field name. As an example, we will create a **literal** (see Figure 149).

Figure 149 Create Literal Dialog Box



- 8 Pass the alert message to the alert event by dragging the message to the argument of the alerter object (see Figure 150).

Figure 150 Pass Alert Message to Object Argument



XSLT Collaboration Definitions

This chapter describes the process for building XSLT Collaboration Definitions.

7.1 Overview

Collaborations use Collaboration Definitions to define how data should be routed between Project components. Collaborations also define how databases should be queried in response to requests and how APIs to one or more applications should be invoked. Collaborations are used when data translation is required.

The Enterprise Designer includes two tools, the XSLT Collaboration Wizard and XSLT Collaboration Editor, that are used to create and customize your XSLT Collaboration Definitions. You must have OTDs available to use as the foundation for creating an XSLT Collaboration Definition. See [Object Type Definitions](#) on page 77 for more details.

Important: *If you delete an OTD in the Project Explorer, any XSLT Collaboration Definitions that have been built using that OTD will be affected. It is recommended that you run the Impact Analyzer before attempting to delete any OTDs (see [Impact Analysis](#) on page 230).*

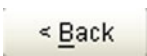

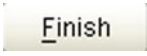


Note: *Any changes made to the names of the Collaborations should be done when the Collaboration is created. If the name is changed later, the Collaboration should be opened, regenerated if applicable, and saved again. For safe measure, this should also be done before creating the Connectivity Map and Deployment Profile.*

7.2 Using the XSLT Collaboration Wizard

7.2.1 The Wizard Interface

The XSLT Collaboration Wizard guides you through the initial phases of creating a Java Collaboration Definition, and then invokes the XSLT Collaboration Editor. The user interface is highly self-explanatory, but details of the navigation buttons are listed in Table 39 for your reference.

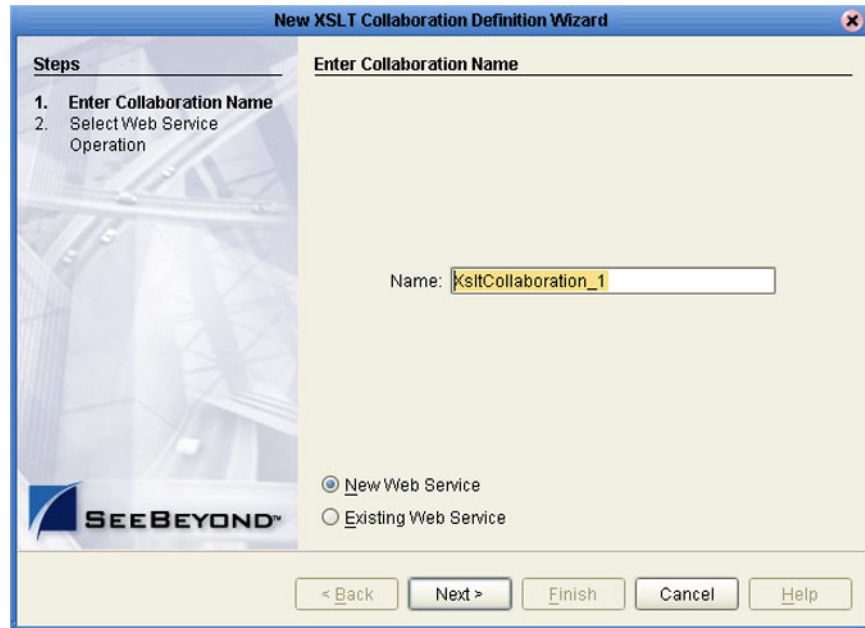
Table 39 Wizard Navigation Buttons

Button	Function
	Returns to the previous step in the wizard. This button is disabled on the first step.
	Goes to the next step in the wizard. This button is disabled on the last step.
	Saves all Collaboration Definition settings and closes the wizard. This button is only enabled on the last step.
	Closes the wizard without saving the Collaboration Definition.
	Displays the online help documentation for the Collaboration Definition Wizard dialog box.

7.2.2 Creating an XSLT Collaboration Definition

- 1 Right-click on a Project in the Enterprise Explorer to display the Project context menu.
- 2 Select **New XSLT Collaboration Definition** to invoke the XSLT Collaboration Wizard.
- 3 Enter a **Name** for your Collaboration, as shown in Figure 151.

Figure 151 XSLT Collaboration Wizard Dialog Box



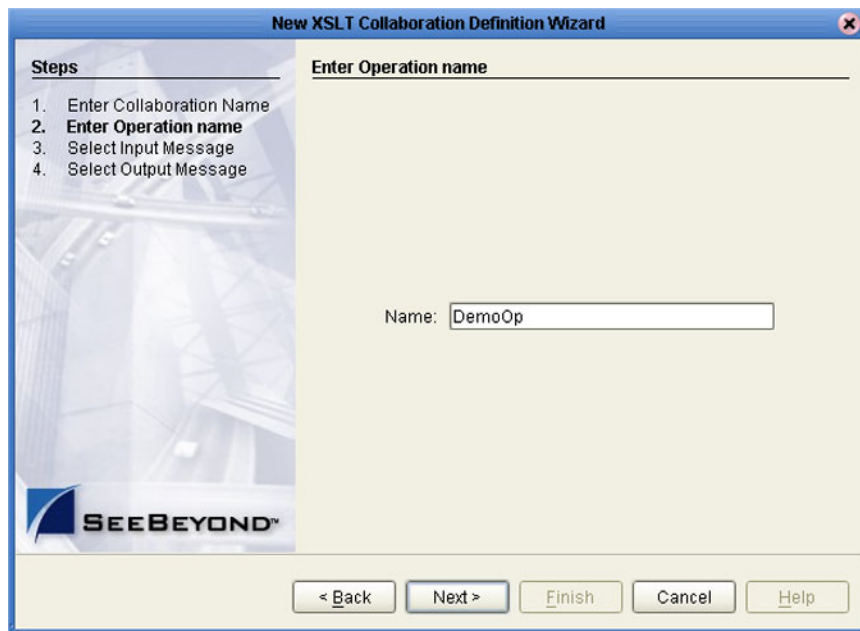
- 4 Select a Web service, which can be either:
 - A New Web Service.
 - An Existing Web Service (for example, an eInsight process or a Java Web Service Operation).
- 5 Click **Next** to proceed to the next Wizard dialog, which is dependent upon your Web Service selection.

New Web Service

If you selected a New Web Service, you will be presented with the following set of Wizard dialogs.

- 1 Enter an operation name, as shown in Figure 152. This will become the *method* that can be used to invoke the XSLT Collaboration as a Web service.

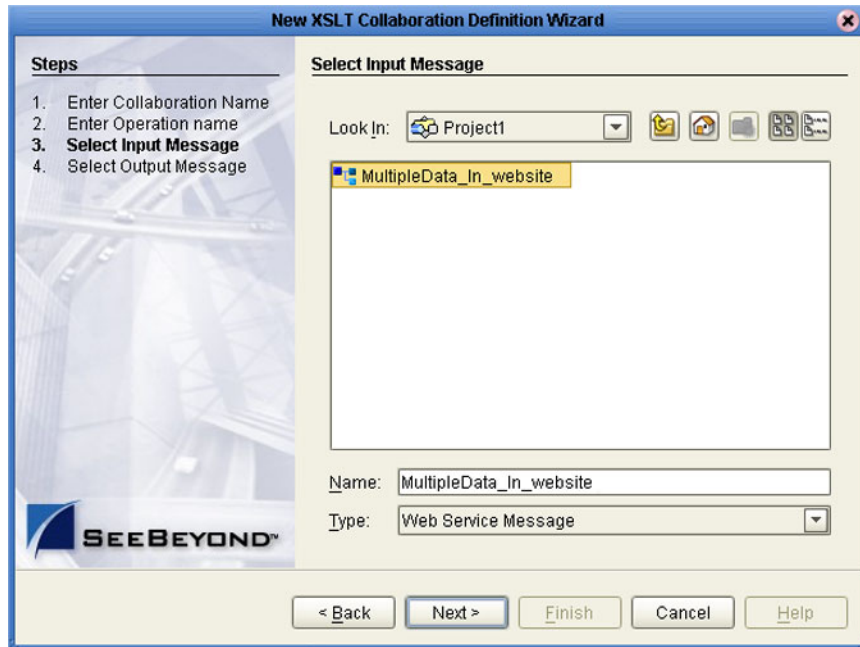
Figure 152 New Web Service: Operation Name



- 2 Click **Next** to proceed to the next Wizard dialog.

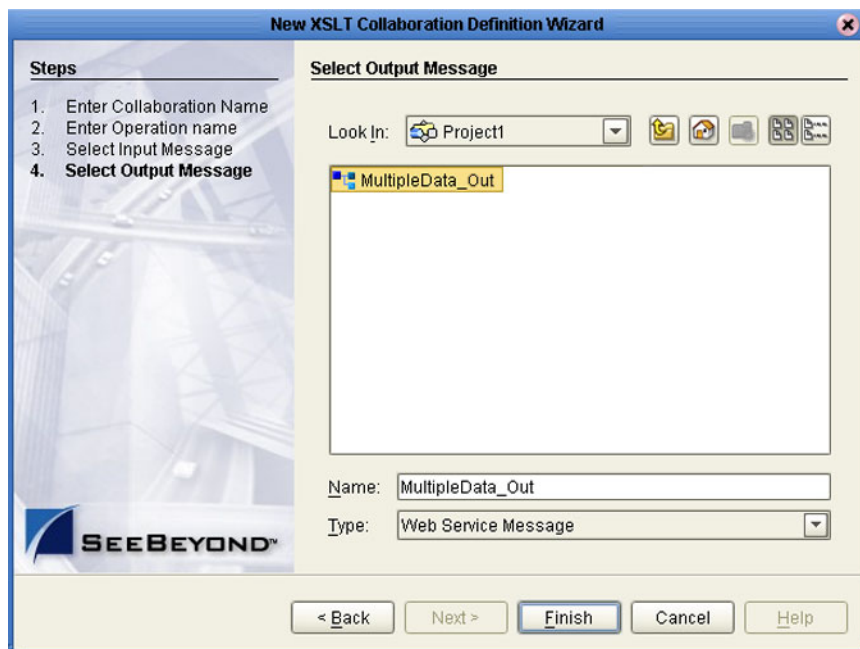
- 3 Select the input Web service message, as shown in Figure 153.

Figure 153 New Web Service: Input Message



- 4 Click **Next** to proceed to the next Wizard dialog.
- 5 Select the output Web service message, as shown in Figure 154.

Figure 154 New Web Service: Output Message



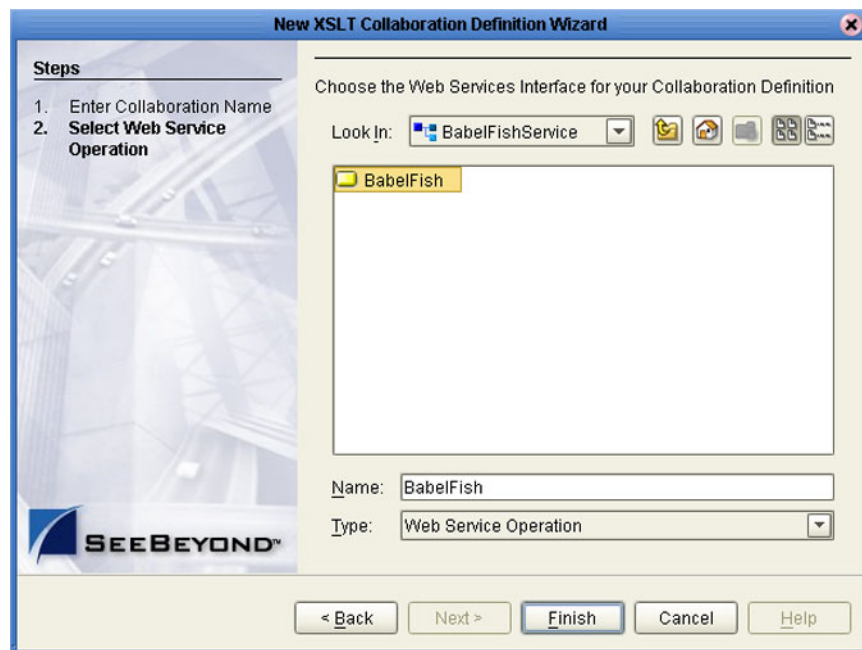
- 6 Click **Finish** to proceed to the XSLT Collaboration Editor.

Existing Web Service

If you selected an Existing Web Service, you will be presented with the Wizard dialog shown in Figure 155.

- 1 Select a Web service operation, which can be either:
 - An installed ICAN Web Service.
 - A custom Web Service (for example, something that has been created in an eGate Project).

Figure 155 Existing Web Service: Select Operation



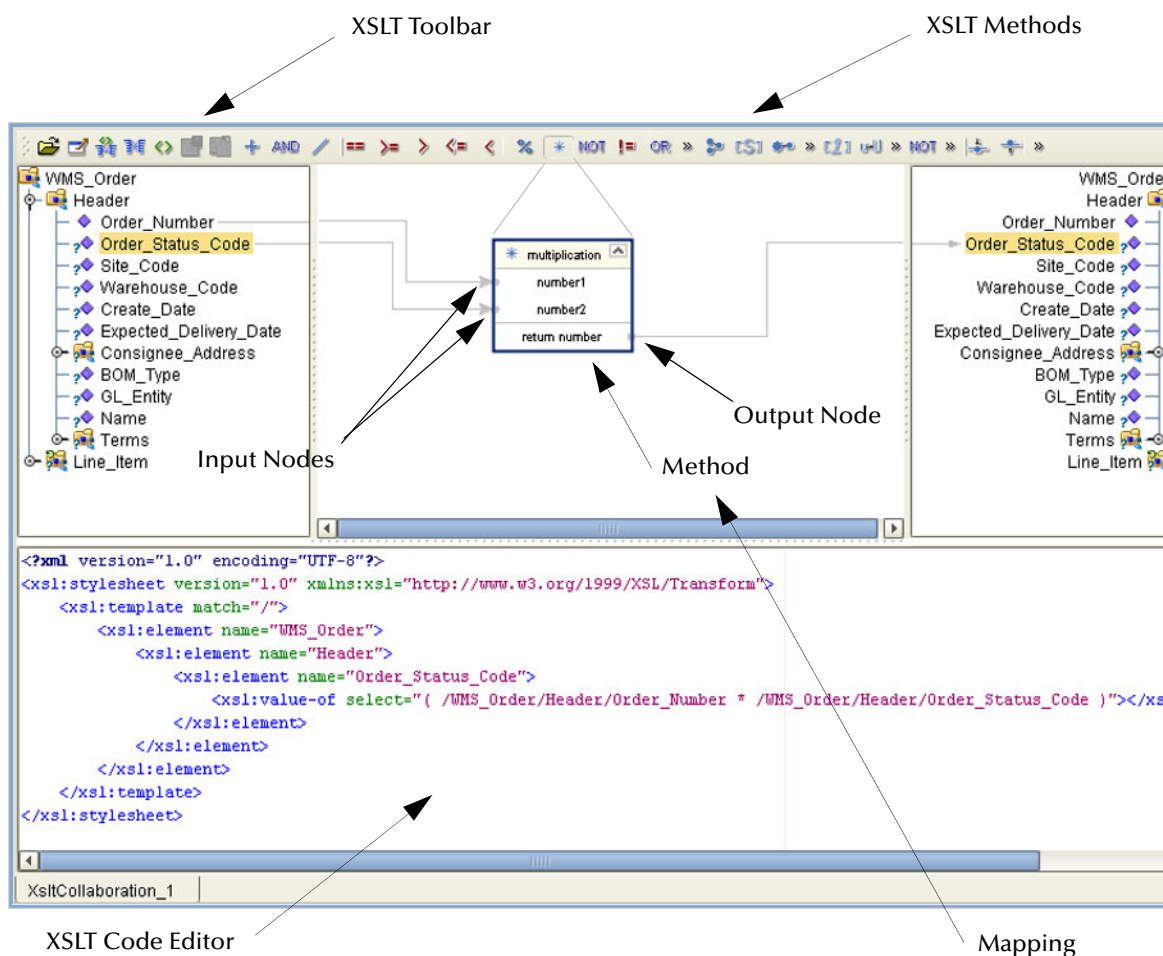
- 2 Click **Finish** to proceed to the XSLT Collaboration Editor.

7.3 Using the XSLT Collaboration Editor

After you have created an XSLT Collaboration Definition using the XSLT Collaboration Wizard, the XSLT Collaboration Editor appears in the Editor panel of the Enterprise Designer. You can also invoke the XSLT Collaboration Editor by selecting **Open** in the context menu for an existing XSLT Collaboration Definition in the Enterprise Explorer.

You can use the XSLT Collaboration Editor to customize Collaboration Definitions. The components of this window are identified in Figure 156. This figure also shows how a Multiplication method displays in the Mapping area after it is dragged down from the Method Palette area and mapped to fields in the left and right panels.








Figure 156 XSLT Collaboration Editor



The XSLT Mapping panel is used to map fields and add methods to the Collaboration Definition. At the top left of the Mapping panel is the toolbar, containing icons as described in Table 40. At the top right of the Mapping area is the XSLT Method Palette, which contains a collection of XSLT methods. The XSLT Code Editor panel allows you to view, enter and edit the XSLT code for the Collaboration Definition.

7.3.1 XSLT Toolbar Icons

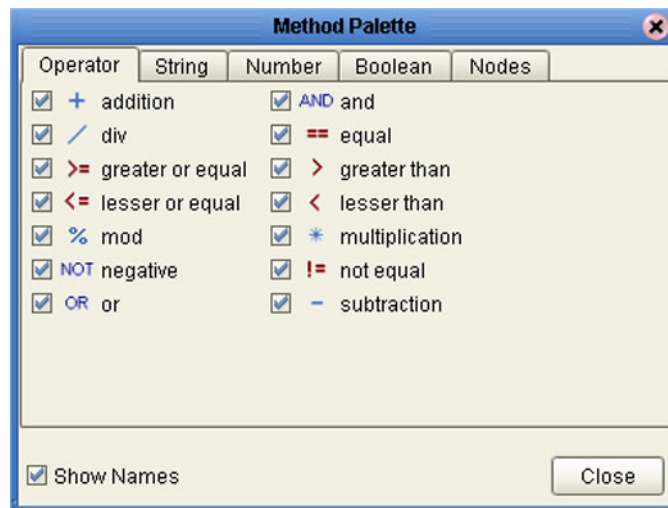
Table 40 XSLT Toolbar Icons

Icon	Command	Function
	Import XSL from a Local File	Displays the Open dialog box, which you can use to locate and select an XSL file to import.
	Save XSL to a Local File	Displays the Save dialog box, which you can use to save the selected Java Collaboration Definition to a file.
	Show Maps and Code	Displays both the Mapping and XSLT Code areas. This is the default view setting.
	Show Mapping Only	Displays the Mapping area and hides the XSLT Code area.
	Show XSLT Code Only	Displays the XSLT Code area and hides the Mapping area.
	Commit Code Changes	Commits changes made to the XSLT code since the last time it was committed. Changes will now be shown in the Mapping panel.
	Roll Back Code Changes	Cancel changes made to the XSLT code since the last time it was committed.

XSLT Method Palette

The XSLT Method Palette includes a series of method icons that you can drag onto the Mapping area. Click the Chevrons (>>) to the right of the method groups to display the dialog box shown in Figure 157. Select a check box to add the method to the toolbar; clear a check box to remove the method from the toolbar. The methods are described in detail in [XSLT Methods](#) on page 185.

Figure 157 XSLT Method Palette Dialog Box



XSLT Method Boxes

The method boxes are placed in the mapping area by dragging the corresponding icon from the method palette toolbar. As shown in [Figure 156 on page 182](#), the method boxes typically have input and output nodes that you link to fields in the left and right panels, respectively. The method boxes are expanded by default (see Figure 158); you can collapse them (see Figure 159) by clicking the caret icon in the upper right corner of the box, which can be useful when the mapping area becomes crowded. Some boxes expand further as needed to provide additional arguments.

Figure 158 Expanded Method Box

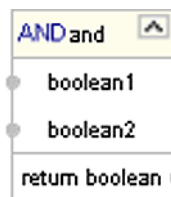
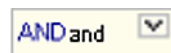


Figure 159 Collapsed Method Box



7.4 XSLT Methods

7.4.1 Operator Methods

Figure 160 Method Palette: Operator Methods

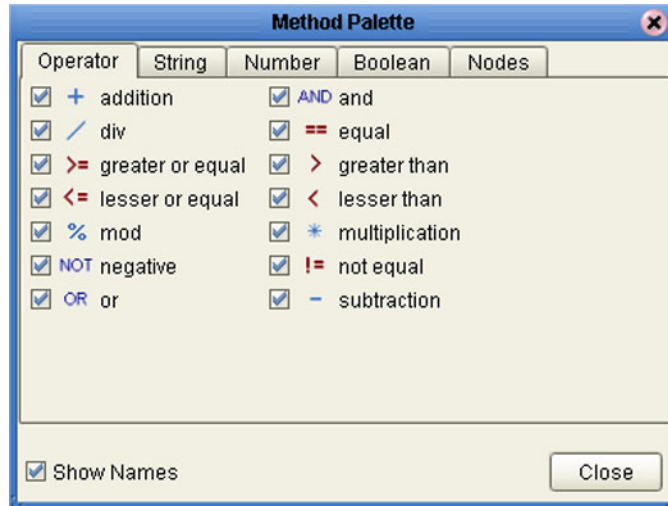


Table 41 XSLT Operator Methods

Method Box	Description/Usage
	The addition method adds the value of <i>number1</i> to the value of <i>number2</i> , returns the sum.
	The AND method returns Boolean true if both <i>boolean1</i> and <i>boolean2</i> are true; otherwise, returns Boolean false.
	The division method divides the value of <i>number1</i> by the value of <i>number2</i> , returns the quotient.

Table 41 XSLT Operator Methods

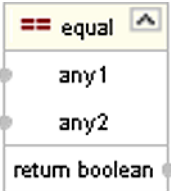
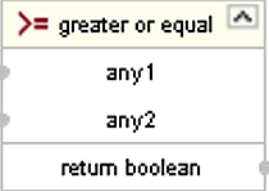
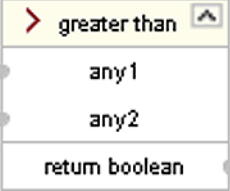
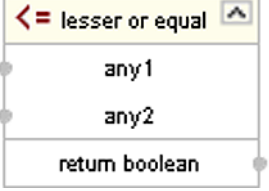
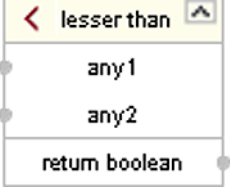
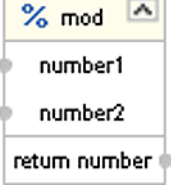
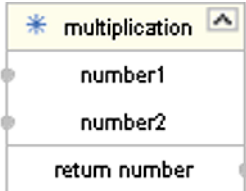
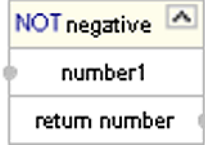
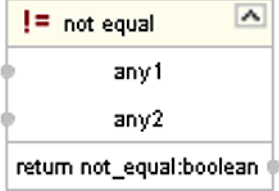
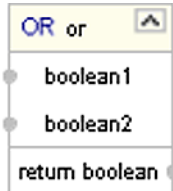
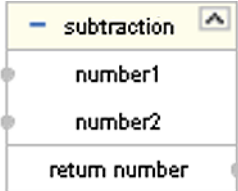
Method Box	Description/Usage
 <p>The method box for 'equal' shows a yellow header with the text '== equal' and a small upward arrow icon. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return boolean' with a small grey dot to its right.</p>	<p>The equal method returns Boolean true if <i>any1</i> is equal to <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'greater or equal' shows a yellow header with the text '>= greater or equal' and a small upward arrow icon. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return boolean' with a small grey dot to its right.</p>	<p>The greater_or_equal method returns Boolean true if <i>any1</i> is greater than or equal to <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'greater than' shows a yellow header with the text '> greater than' and a small upward arrow icon. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return boolean' with a small grey dot to its right.</p>	<p>The greater_than method returns Boolean true if <i>any1</i> is greater than <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'lesser or equal' shows a yellow header with the text '<= lesser or equal' and a small upward arrow icon. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return boolean' with a small grey dot to its right.</p>	<p>The lesser_or_equal method returns Boolean true if <i>any1</i> is less than or equal to <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'lesser than' shows a yellow header with the text '< lesser than' and a small upward arrow icon. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return boolean' with a small grey dot to its right.</p>	<p>The lesser_than method returns Boolean true if <i>any1</i> is less than <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'modulus' shows a yellow header with the text '% mod' and a small upward arrow icon. Below the header are two input fields labeled 'number1' and 'number2', each with a small grey dot to its left. At the bottom of the box is a return field labeled 'return number' with a small grey dot to its right.</p>	<p>The modulus method divides the numerical value of <i>number1</i> by the numerical value of <i>number2</i>, and returns the remainder.</p>

Table 41 XSLT Operator Methods

Method Box	Description/Usage
 <p>The method box for 'multiplication' features a yellow header with a blue asterisk icon and the text 'multiplication'. Below the header are two input fields labeled 'number1' and 'number2', each with a small grey dot on its left side. At the bottom is an output field labeled 'return number' with a small grey dot on its right side.</p>	<p>The multiplication method multiplies the value of <i>number1</i> by the value of <i>number2</i>, returns the product.</p>
 <p>The method box for 'NOT negative' features a yellow header with the text 'NOT negative'. Below the header is one input field labeled 'number1' with a small grey dot on its left side. At the bottom is an output field labeled 'return number' with a small grey dot on its right side.</p>	<p>The negative method returns the arithmetic negation of <i>number1</i>.</p>
 <p>The method box for 'not equal' features a yellow header with a red '!=' icon and the text 'not equal'. Below the header are two input fields labeled 'any1' and 'any2', each with a small grey dot on its left side. At the bottom is an output field labeled 'return not_equal:boolean' with a small grey dot on its right side.</p>	<p>The not_equal method returns Boolean true if <i>any1</i> is not equal to <i>any2</i>; otherwise, returns Boolean false.</p>
 <p>The method box for 'OR or' features a yellow header with a blue 'OR' icon and the text 'or'. Below the header are two input fields labeled 'boolean1' and 'boolean2', each with a small grey dot on its left side. At the bottom is an output field labeled 'return boolean' with a small grey dot on its right side.</p>	<p>The OR method returns Boolean false if both <i>boolean1</i> and <i>boolean2</i> are false; otherwise, returns Boolean true.</p>
 <p>The method box for 'subtraction' features a yellow header with a blue '-' icon and the text 'subtraction'. Below the header are two input fields labeled 'number1' and 'number2', each with a small grey dot on its left side. At the bottom is an output field labeled 'return number' with a small grey dot on its right side.</p>	<p>The subtraction method subtracts the numerical value of <i>number2</i> from the numerical value of <i>number1</i>, returns the difference.</p>

7.4.2 String Methods

Figure 161 Method Palette: String Methods

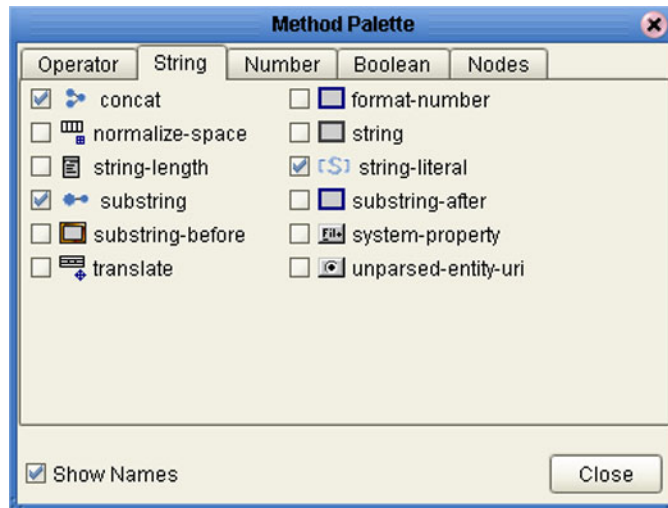


Table 42 XSLT String Methods

Method Box	Description/Usage
	<p>The concat method returns the string created by concatenating <i>string2</i> to the end of <i>string1</i>, and <i>string3*</i> (if present) to the end of <i>string2</i>.</p>
	<p>The format-number method converts its first argument (<i>number1</i>) to a string using the format pattern string specified by the second argument (<i>string2</i>) and the decimal-format named by the third argument (<i>string3</i>) or the default decimal-format, if there is no third argument.</p>
	<p>The normalize-space method returns the argument <i>string1?</i> with whitespace normalized by stripping leading and trailing whitespace and replacing sequences of whitespace characters by a single space. If the argument is omitted, it defaults to the string-value of the context node.</p>

Table 42 XSLT String Methods

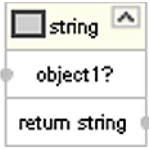
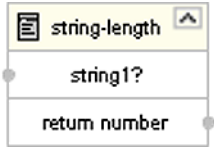

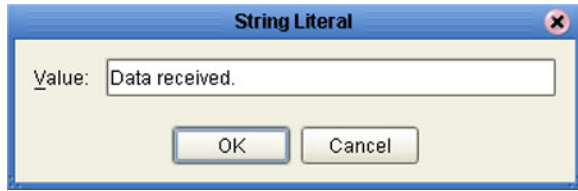
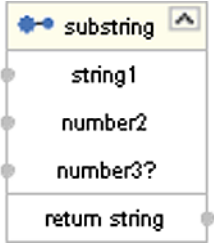
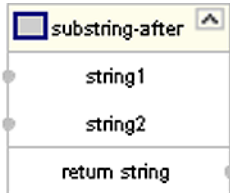
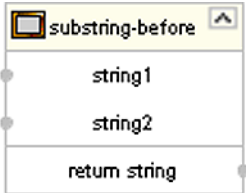
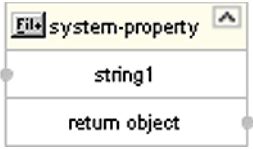

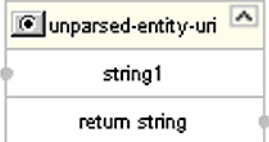
Method Box	Description/Usage
	<p>The string method returns a string representation of the input object.</p>
	<p>The string-length method returns the number of characters in the string.</p>
	<p>Dragging the icon first displays the <i>String Literal</i> dialog box, where you enter the literal value, such as “Data received”:</p> <div data-bbox="711 800 1284 989" style="text-align: center;">  </div> <p>The string-literal method then returns a string having the specified value.</p>
	<p>The substring method returns the substring of the first argument (<i>string1</i>) starting at the position specified in the second argument (<i>number2</i>) with length specified in the third argument (<i>number3?</i>). If the third argument is not specified, it returns the substring starting at the position specified in the second argument and continuing to the end of the string.</p>
	<p>The substring-after method returns the substring of the first argument (<i>string1</i>) that follows the first occurrence of the second argument (<i>string2</i>) in the first argument string. Returns an empty string if the first argument string does not contain the second argument string.</p>
	<p>The substring-before method returns the substring of the first argument (<i>string1</i>) that precedes the first occurrence of the second argument (<i>string2</i>) in the first argument string. Returns an empty string if the first argument string does not contain the second argument string.</p>

Table 42 XSLT String Methods

Method Box	Description/Usage
 <p>The diagram shows a method box titled 'system-property'. It has a single input parameter 'string1' and a return value 'return object'.</p>	<p>The system-property method returns an object representing the value of the system property identified by <i>string1</i>. If there is no such system property, the empty string should be returned.</p>
 <p>The diagram shows a method box titled 'translate'. It has three input parameters: 'string1', 'string2', and 'string3'. The return value is 'return string'.</p>	<p>The translate method returns the first argument (<i>string1</i>) with occurrences of characters in the second argument (<i>string2</i>) replaced by the character at the corresponding position in the third argument (<i>string3</i>). If a character occurs more than once in the second argument (<i>string2</i>), then the first occurrence determines the replacement character. If the third argument (<i>string3</i>) is longer than the second argument (<i>string2</i>), then excess characters are ignored. Refer to the W3C <i>XML Path Language</i> documentation for additional conditions.</p>
 <p>The diagram shows a method box titled 'unparsed-entity-uri'. It has a single input parameter 'string1' and a return value 'return string'.</p>	<p>The unparsed-entity-uri method returns the URI of the unparsed entity with the specified name (<i>string1</i>) in the same document as the context node. It returns an empty string if there is no such entity.</p>

7.4.3 Number Methods

Figure 162 Method Palette: Number Methods

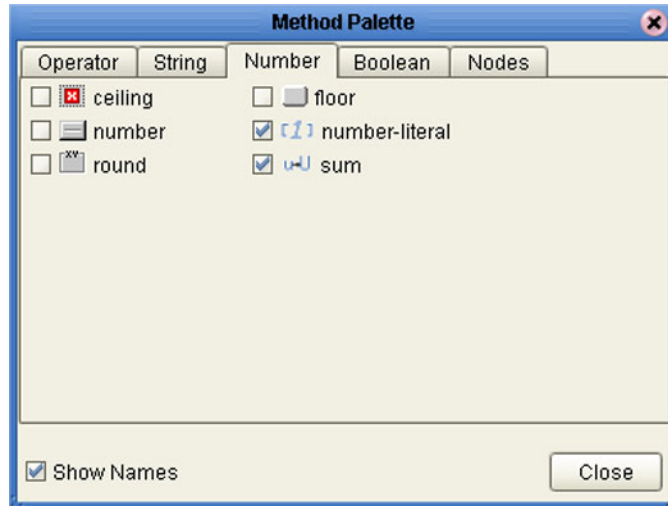
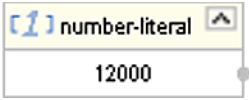
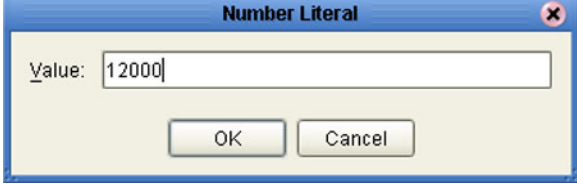

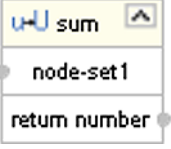


Table 43 XSLT Number Methods

Method Box	Description/Usage
	The ceiling method returns the smallest (closest to negative infinity) number that is not less than the argument and that is an integer.
	The floor method returns the largest (closest to positive infinity) number that is not greater than the argument and that is an integer.
	The number method converts its argument (<i>object1</i>) to a number. An object of a type other than the four basic types is converted to a number in a way that is dependent on that type.

Table 43 XSLT Number Methods

Method Box	Description/Usage
 <p>The method box for <code>number-literal</code> shows a blue icon with the number 1, the text "number-literal", and a value of "12000" in a text field.</p>	<p>Dragging the icon first displays the <i>Number Literal</i> dialog box, where you enter the literal value, such as "12000":</p>  <p>The number-literal method then returns a number having the specified value.</p>
 <p>The method box for <code>round</code> shows an icon with "xv", the text "round", an input field for "number1", and a return value of "return number".</p>	<p>The round method returns the number that is closest to the argument <i>number1</i> and that is an integer.</p> <ul style="list-style-type: none"> ▪ If there are two such numbers, then the one that is closest to positive infinity is returned. ▪ If the argument is not a number (NaN), then NaN is returned. ▪ If the argument is positive infinity, then positive infinity is returned. ▪ If the argument is negative infinity, then negative infinity is returned. ▪ If the argument is positive zero, then positive zero is returned. ▪ If the argument is negative zero, then negative zero is returned. ▪ If the argument is less than zero, but greater than or equal to -0.5, then negative zero is returned.
 <p>The method box for <code>sum</code> shows a blue icon with a summation symbol, the text "sum", an input field for "node-set1", and a return value of "return number".</p>	<p>The sum method returns the sum, for each node in the argument <i>node-set1</i>, of the result of converting the string-values of the node to a number.</p>

7.4.4 Boolean Methods

Figure 163 Method Palette: Boolean Methods

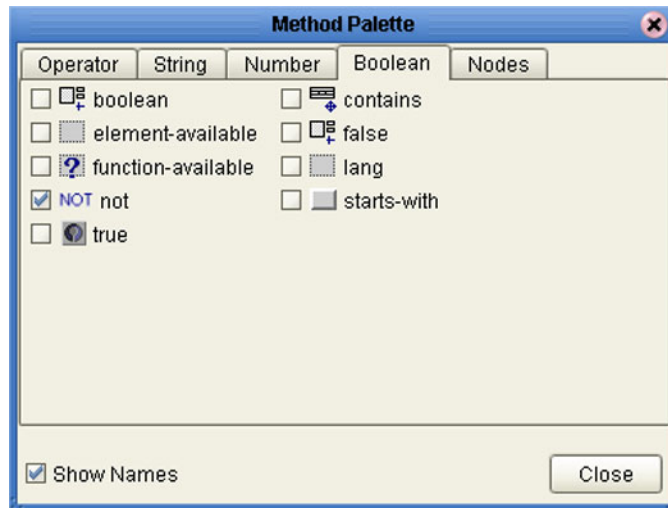
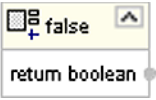
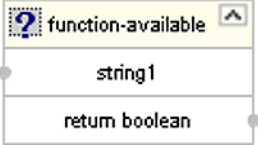
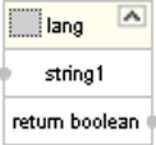
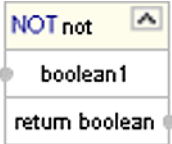
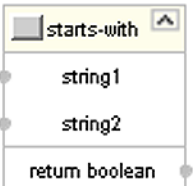
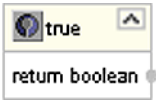


Table 44 XSLT Boolean Methods

Method Box	Description/Usage
	<p>The boolean method converts the argument <i>object1</i> to a Boolean true or false as follows:</p> <ul style="list-style-type: none"> ▪ A number is true if and only if it is neither \pmzero nor NaN (not a number). ▪ A node-set is true if and only if it is non-empty. ▪ A string is true if and only if it is non-zero. ▪ An object of a type other than the four basic types is converted to a Boolean in a way that is dependent on that type.
	<p>The contains method returns Boolean true if the first argument (<i>string1</i>) contains the second argument (<i>string2</i>); if not, returns Boolean false.</p>
	<p>The argument <i>string1</i> must be a QName, which is expanded into an expanded-name using the namespace declarations in scope for the expression. The element-available method returns true if and only if the expanded-name is the name of an instruction. If the expanded-name has a namespace URI equal to the XSLT namespace URI, then it refers to an element defined by XSLT. Otherwise, it refers to an extension element. If the expanded-name has a null namespace URI, the element-available function will return false.</p>

Table 44 XSLT Boolean Methods

Method Box	Description/Usage
 <p>The method box for 'false' contains a title bar with a plus icon and the text 'false', and a single input field labeled 'return boolean'.</p>	<p>The false method returns Boolean false.</p>
 <p>The method box for 'function-available' contains a title bar with a question mark icon and the text 'function-available', an input field labeled 'string1', and an output field labeled 'return boolean'.</p>	<p>The argument <i>string1</i> must be a QName, which is expanded into an expanded-name using the namespace declarations in scope for the expression. The function-available method returns true if and only if the expanded-name is the name of a function in the function library. If the expanded-name has a non-null namespace URI, then it refers to an extension function; otherwise, it refers to a function defined by XPath or XSLT.</p>
 <p>The method box for 'lang' contains a title bar with the text 'lang', an input field labeled 'string1', and an output field labeled 'return boolean'.</p>	<p>The language method returns Boolean true or false depending upon whether the language of the context node as specified by <i>xml:lang</i> attributes is the same as, or is a sub-language of, the language specified by the argument string (<i>string1</i>). Returns Boolean false if the attribute <i>xml:lang</i> does not exist.</p>
 <p>The method box for 'NOT not' contains a title bar with a 'NOT' icon and the text 'not', an input field labeled 'boolean1', and an output field labeled 'return boolean'.</p>	<p>The NOT method returns the inverse of <i>boolean1</i>.</p>
 <p>The method box for 'starts-with' contains a title bar with the text 'starts-with', two input fields labeled 'string1' and 'string2', and an output field labeled 'return boolean'.</p>	<p>The starts-with method returns Boolean true if the first argument (<i>string1</i>) starts with the second argument (<i>string2</i>); if not, returns Boolean false.</p>
 <p>The method box for 'true' contains a title bar with a globe icon and the text 'true', and an output field labeled 'return boolean'.</p>	<p>The true method returns Boolean true.</p>

7.4.5 Nodes Methods

Figure 164 Method Palette: Nodes Methods

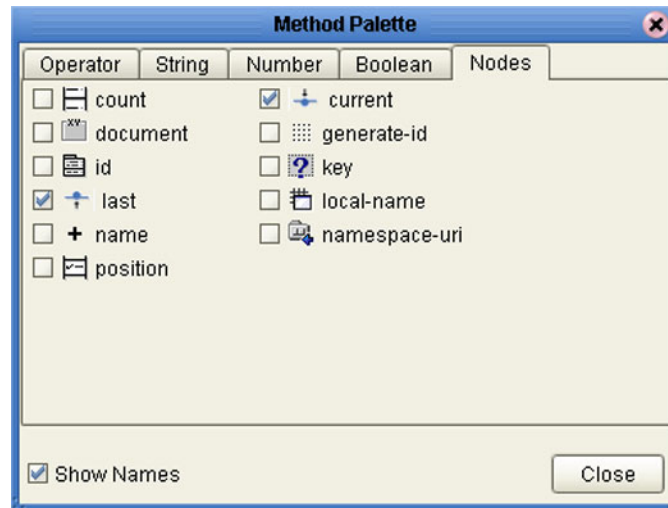


Table 45 XSLT Nodes Methods

Method Box	Description/Usage
	<p>The count method returns the number of nodes in the argument <i>node-set1</i>.</p>
	<p>The current method returns a node-set that has the current node as its only member.</p>
	<p>The document method allows access to XML documents other than the main source document.</p> <ul style="list-style-type: none"> When the method has exactly one argument and the argument is a node-set, then for each node in the argument node-set, the result is the union of the result of calling the document function with the first argument being the string-value of the node, and the second argument being a node-set with the node as its only member. When the method has two arguments and the first argument is a node-set, then for each node in the argument node-set, the result is the union of the result of calling the document function with the first argument being the string-value of the node, and with the second argument being the second argument passed to the document function. When the first argument is not a node-set, the first argument is converted to a string as if by a call to the string function. This string is treated as a URI reference; the resource identified by the URI is retrieved.

Table 45 XSLT Nodes Methods

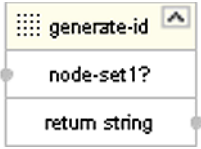
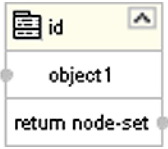
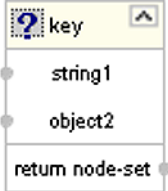
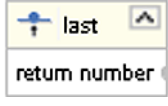
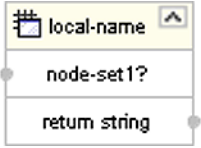
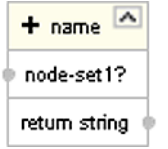
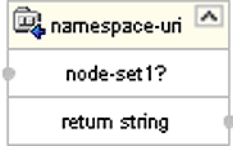
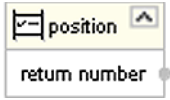
Method Box	Description/Usage
	<p>The generate-id method returns a string that uniquely identifies the node in the argument <i>node-set1?</i> that is first in document order. The unique identifier must consist of ASCII alphanumeric characters and must start with an alphabetic character. Thus, the string is syntactically an XML name.</p>
	<p>When the argument <i>object1</i> is not of type node-set, the id method converts the argument to a string as if by a call to the string function; the string is split into a whitespace-separated list of tokens; the result is a node-set containing the elements in the same document as the context node that have a unique ID equal to any of the tokens in the list.</p>
	<p>The key method does for keys what the id function does for IDs. The value of the first argument (<i>string1</i>), which specifies the name of the key, must be a QName—which is expanded into an expanded-name using the namespace declarations in scope for the expression.</p> <ul style="list-style-type: none"> ▪ When the second argument is of type node-set, then the result is the union of the result of applying the key function to the string value of each of the nodes in the argument node-set. ▪ When the second argument is of any other type, the argument is converted to a string as if by a call to the string function; it returns a node-set containing the nodes in the same document as the context node that have a value for the named key equal to this string.
	<p>The last method returns a number equal to the context size from the expression evaluation context.</p>
	<p>The local-name method returns the local part of the expanded-name of the node in the argument <i>node-set1?</i> that is first in document order. If the argument is empty or the first node has no expanded-name, an empty string is returned. If the argument is omitted, it defaults to a node-set with the context node as its only member.</p>
	<p>The name method returns a string containing a QName representing the expanded-name of the node in the argument <i>node-set1?</i> that is first in document order. If the argument is empty or the first node has no expanded-name, an empty string is returned. If the argument it omitted, it defaults to a node-set with the context node as its only member.</p>
	<p>The namespace-uri method returns the namespace URI of the expanded-name of the node in the argument <i>node-set1?</i> that is first in document order. If the argument is empty, the first node has no expanded-name, or the namespace URI of the expanded-name is null, an empty string is returned. If the argument is omitted, it defaults to a node-set with the context node as its only member.</p>

Table 45 XSLT Nodes Methods

Method Box	Description/Usage
	The position method returns a number equal to the context position from the expression evaluation context.

7.5 XSLT Encapsulation

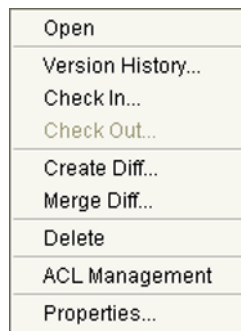
XSLT Collaboration Definitions can be saved to a Diff (.sdf) file. This feature allows two sites working with the same XSLT Collaboration Definition to seamlessly merge changes.

7.5.1 Creating a Modified XSLT Collaboration Definition

This section explains how to create a *different* version of a XSLT Collaboration Definition file.

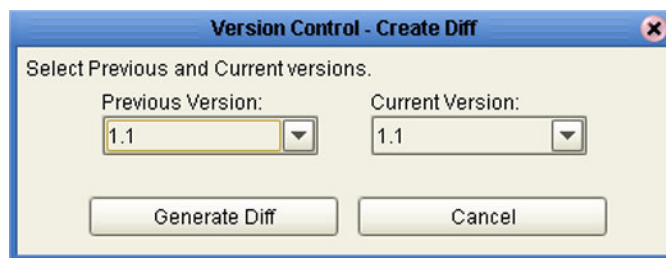
- 1 From the Project Explorer tab, select a **XSLT Collaboration Definition** icon.
- 2 Right-click to display the context menu shown in Figure 165.

Figure 165 XSLT Collaboration Definition Context Menu



- 3 Select **Create Diff** to display the dialog box shown in Figure 166.

Figure 166 Version Control - Create Diff Dialog Box



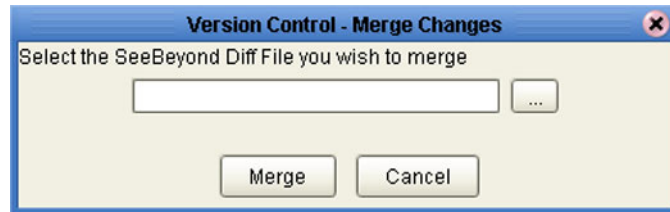
- 4 Click **Generate Diff** to display the *Specify Name ...* dialog box.
- 5 Enter a new name for the Diff file in the File Name box if you do not want to use the default file name.
- 6 Click **Save** to save the Diff file.

7.5.2 Merging Two Versions of a XSLT Collaboration Definition

This section explains how to merge two versions of a XSLT Collaboration Definition.

- 1 From the Project Explorer tab, select a **XSLT Collaboration Definition** icon.
- 2 Right-click to display the XSLT Collaboration Definition context menu.
- 3 Select **Merge Diff...** to display the dialog box shown in Figure 167.

Figure 167 Version Control - Merge Changes Dialog Box



- 4 Click the **Ellipsis (...)** button to display the *Specify Name ...* dialog box.
- 5 Locate and select the Diff file.
- 6 Click **Open** add the Diff file to the dialog box shown in Figure 167.
- 7 Click **Merge** to merge the Diff file with the corresponding XSLT Collaboration Definition in your Project.

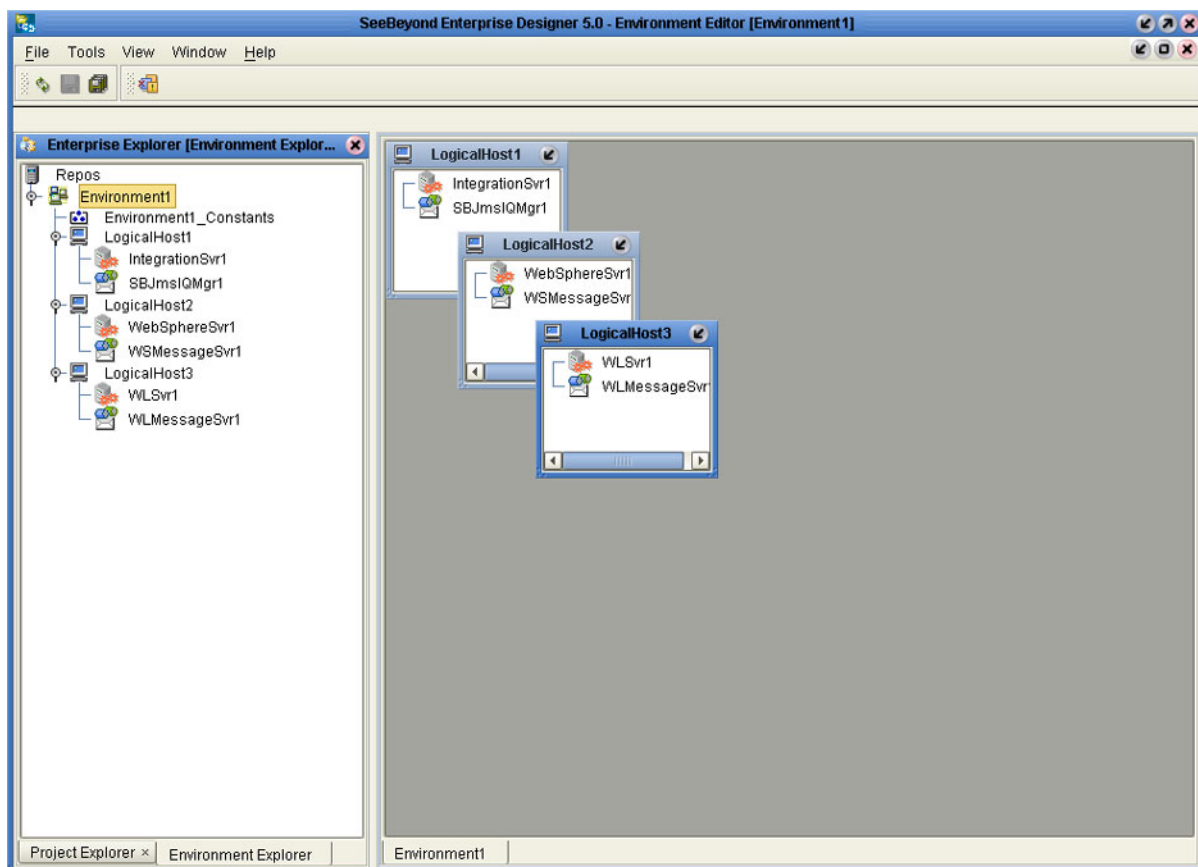
Environments and Project Deployment

This chapter describes the process of defining environments, creating deployment profiles, and activating the deployed projects.

8.1 Overview

Clicking on an Environment icon in the Environment Explorer invokes the Environment Editor, which provides a canvas in which you can create and customize an Environment (see Figure 168).

Figure 168 Environment Editor



Here you can see the various components (Logical Hosts, servers, and external systems) included in the selected Environment. New Environments are added through the use of the Repository context menu (see Figure 169). Components are added to the Environment by selecting options in the Environment and Logical Host context menus (see Figure 170 and Figure 171. See [Context Menus](#) on page 44 for additional information.

Figure 169 Repository Context Menu



Figure 170 Environment Context Menu

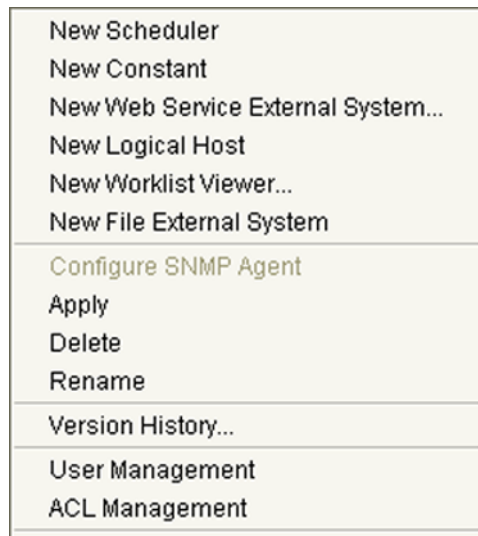
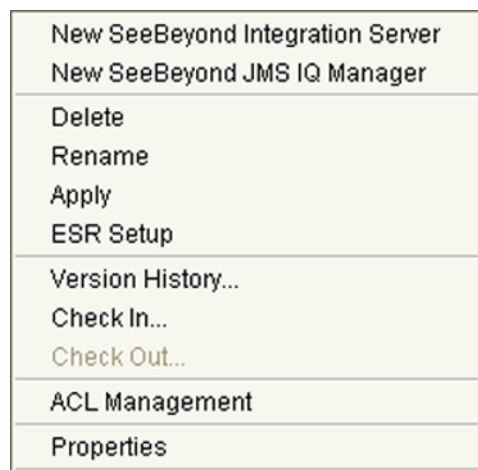


Figure 171 Logical Host Context Menu

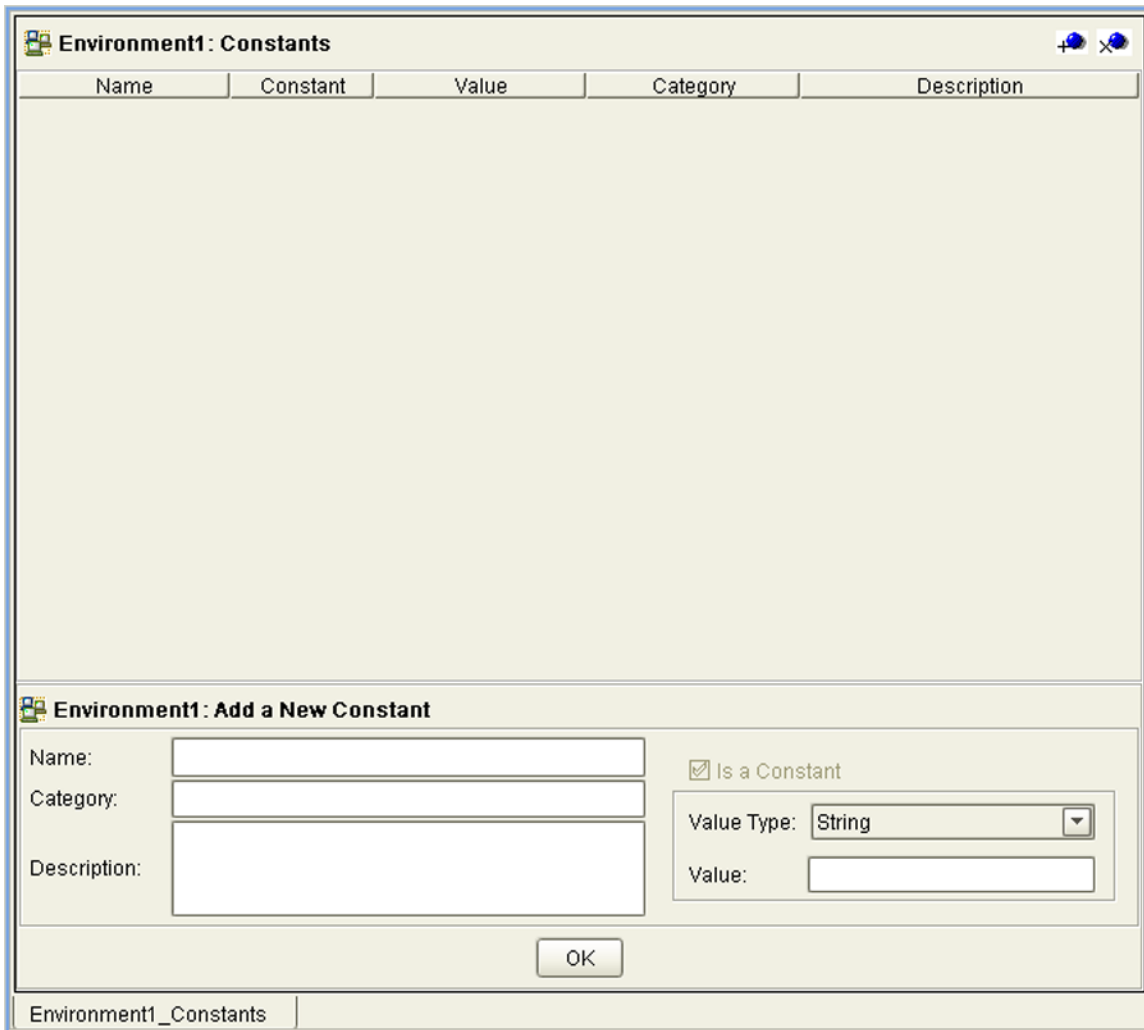


8.2 Environmental Constants

You can define constants for a specific Environment. Environmental constants are name/value pairs that are visible across the Environment. When you create a constant you assign a permanent value to it, which cannot be overridden.



Selecting the **New Constant** option from the Environment context menu displays the Constants panel in the Environment Editor (see Figure 172).

Figure 172 Environmental Constants Editor



New constants are added using the bottom panel in the Constants editor. All constants defined for the specific Environment are listed in the top panel, along with their various properties.

Table 46 Environmental Constants Editor Icons

Icon	Name	Function
	Add a New Constant	Adds a new constant to the list.
	Delete a Highlighted Constant	Deletes the selected constant from the list.

8.3 Deployment Profiles

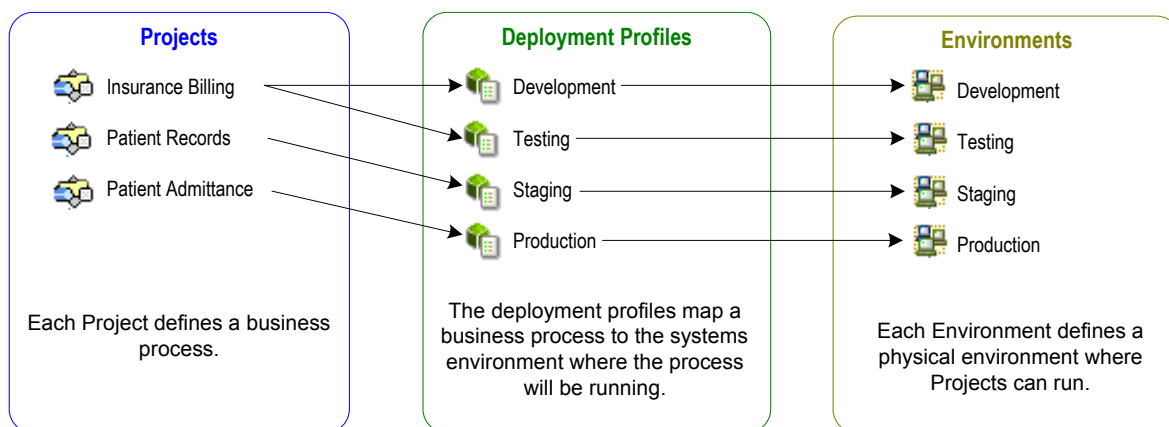
Deployment Profiles are specific instances of a Project in a particular Environment. A deployment profile contains information about the assignment of Services and Message Destinations to integration and message servers (JMS IQ Managers). It also contains version information for all relevant objects in the Project. The Enterprise Designer includes a Deployment Editor, which you can use to create and customize deployment profiles.

Note that:

- Each Project can have zero or more Deployment Profiles, but each of a Project’s active Deployment Profiles must be in a separate Environment.
- Each Environment can have zero or more Deployment Profiles assigned to it, but any given Environment can have only one Deployment Profile from a given Project.

Repeating Figure 3 from the [System Overview](#) on page 24:

Figure 173 eGate Integrator Implementation Model



8.4 Using the Deployment Editor

The Deployment Editor (see Figure 174) appears when you create a new Deployment Profile or click on an existing Deployment Profile icon in the Project Explorer tab.

Figure 174 Deployment Editor Window

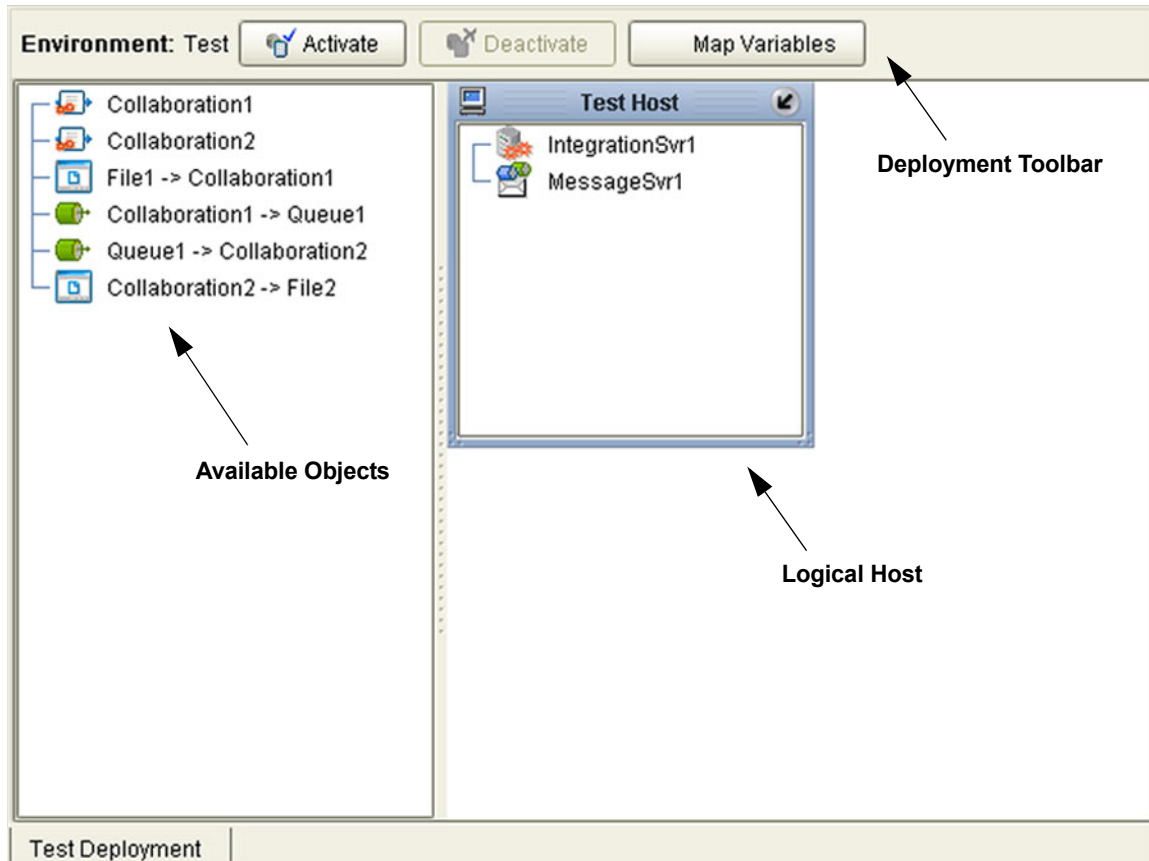




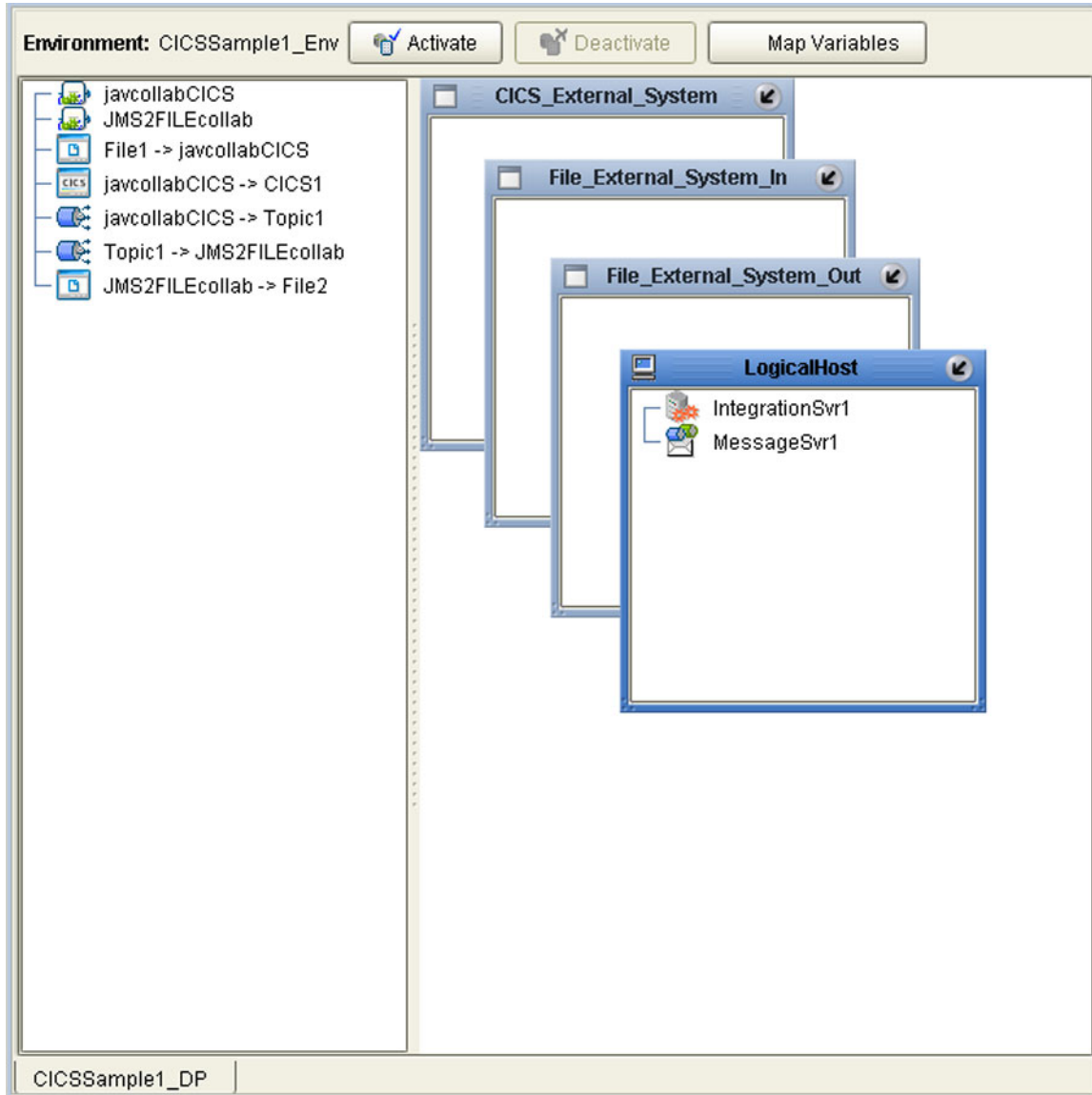
Table 47 Deployment Toolbar Buttons

Button	Function
 Activate	Starts the Project by creating an enterprise archive (EAR) file based on the Connectivity Map and linking this file with the SeeBeyond Integration Server. See Activating and Deactivating Deployments on page 208.
 Deactivate	Stops the Project by terminating the link between the EAR file and the SeeBeyond Integration Server, sets the Deployment Profile to <i>inactive</i> , and saves to the Repository.
Map Variables	Allows you to assign names and values to Project variables for the specific Deployment Profile. See Mapping Variables on page 210.

8.5 Creating a Deployment Profile

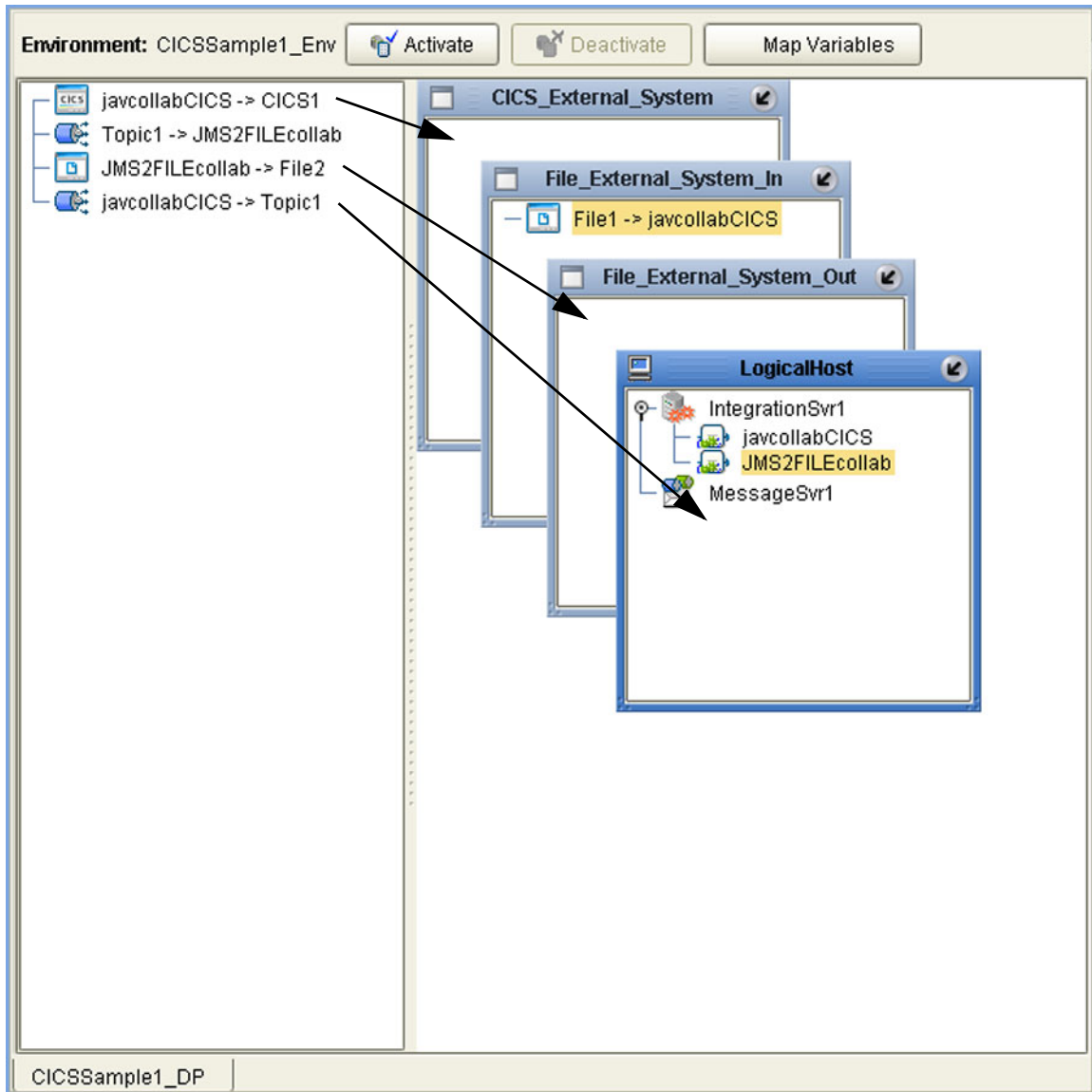
In the Environment Explorer, create an Environment and right-click on the Environment to display its context menu. From the menu, select the Environment components you need and name them appropriately. They will appear on the Deployment Editor canvas, as shown in Figure 175.

Figure 175 Example Deployment Profile (1)



Drag the Project components from the left panel and drop them into the appropriate Environment components in the right panel, as illustrated in Figure 176. As you do so, they will disappear from the left panel.

Figure 176 Example Deployment Profile (2)

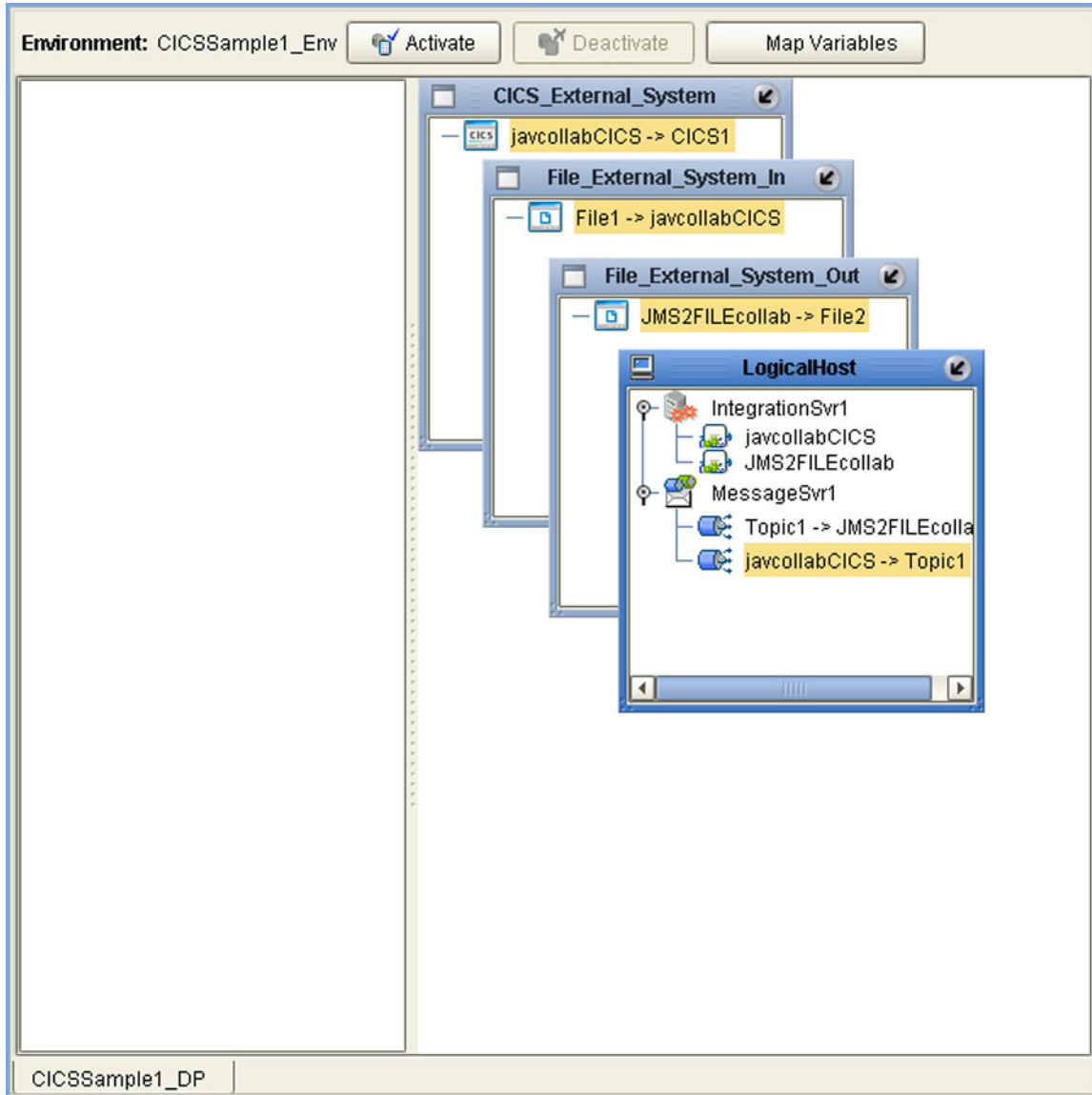


Note that:

- The eWay objects are placed into their appropriate External Systems.
- Collaboration objects are placed into the appropriate Integration Server on the appropriate Logical Host.
- Topic and queue objects are placed into the appropriate Message Server (JMS IQ Manager) on the appropriate Logical Host.

When the Environment components are fully populated, the left panel will be blank, as shown in Figure 177. The Deployment Profile is now ready to be saved and Activated.

Figure 177 Example Deployment Profile (3)



8.6 Activating and Deactivating Deployments

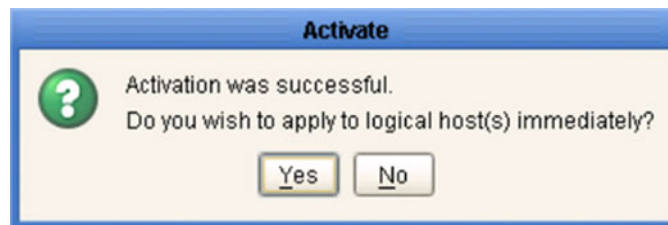
Using the Activate and Deactivate toolbar buttons, you have the option of immediately applying the changes to the Logical Host or deferring the changes to a later time. Activating the Deployment Profile without applying the changes checks the validity of the entire Deployment Profile.

Another advantage to activating the Deployment Profile without applying the changes comes into play when you have multiple Deployment Profiles to deploy at once. To save time, you can activate each of the Deployment Profiles without applying the changes. Then when you do apply all of the changes to the Logical Host in one batch.

To activate a Deployment Profile

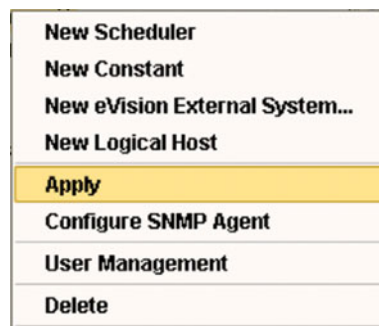
- 1 In the Deployment Profile, select the Deployment you wish to activate.
- 2 Click the **Activate** button. The following message appears:

Figure 178 Activate Dialog Box



- 3 Answer the question following these criteria:
 - If the Logical Host is running, and you wish to apply the changes immediately, click **Yes**.
 - If the Logical Host has not yet been bootstrapped, or you wish to apply the changes at a later time, click **No**. To apply the changes later, right-click the Logical Host and select **Apply** from the menu (see Figure 179). This will apply all of the changes for that Logical Host.

Figure 179 Logical Host Context Menu - Apply

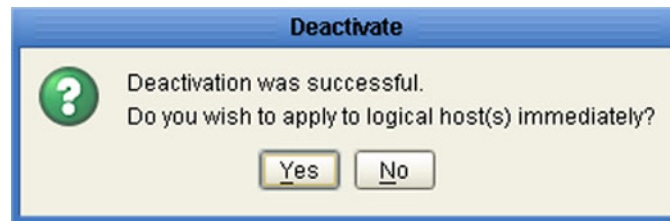


Note: The **Apply** action assumes that the Logic Host is running, since it invokes a trigger to the Logical Host causing it to download the latest settings from the Repository and deploy those settings to all components on the Logical Host.

To deactivate a Deployment Profile

- 1 In the Deployment Profile, select the Deployment you wish to deactivate.
- 2 Click the **Deactivate** button. The following message appears:

Figure 180 Activate Dialog Box



- 3 Answer the question following these criteria:
 - If the Logical Host is running, and you wish to apply the changes immediately, click **Yes**.
- 4 If the Logical Host has not yet been bootstrapped, or you wish to apply the changes at a later time, click **No**. To apply the changes later, right-click the Logical Host and select **Apply** from the menu (see Figure 179). This will apply all of the changes for that Logical Host. See the **Note** following the activation procedure.

Note: In Windows and NFS, application working directories cannot be deleted during deactivation. This is because .jar files in these directories have been added to a ClassLoader and the JVM maintains **locks/handles?** on any such files. At the subsequent startup of the Integration Server, leftover work directories in the repository/application directory are deleted.

8.7 Mapping Variables

Project variables function as placeholders, having values that are determined when you create a specific Deployment Profile. These values can be literals or Environmental constants. Clicking the **Map Variable** button displays the Deployment Profile Mappings panel, where you can assign names (see Figure 181) and values (see Figure 182).

Figure 181 Deployment Profile Mappings

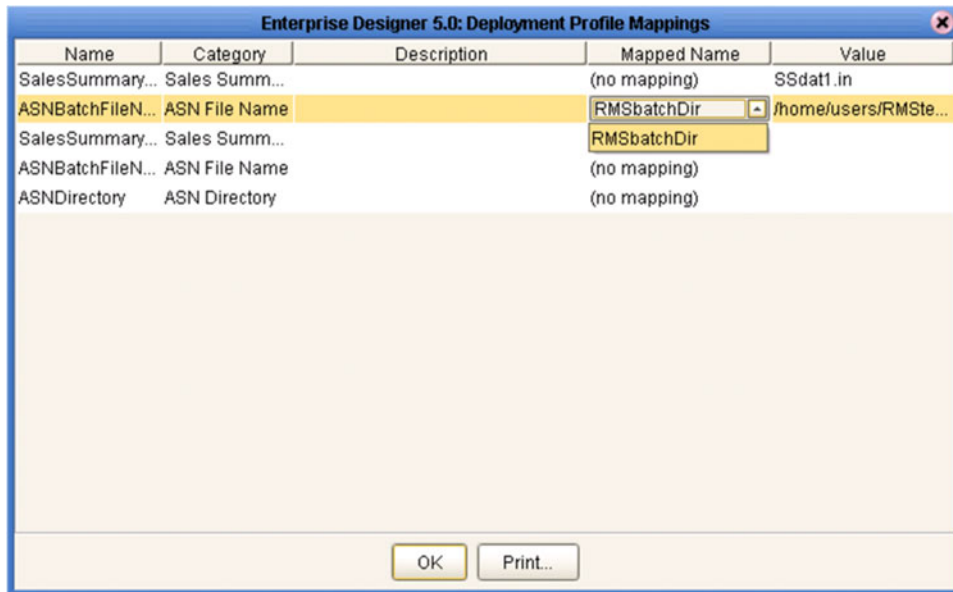
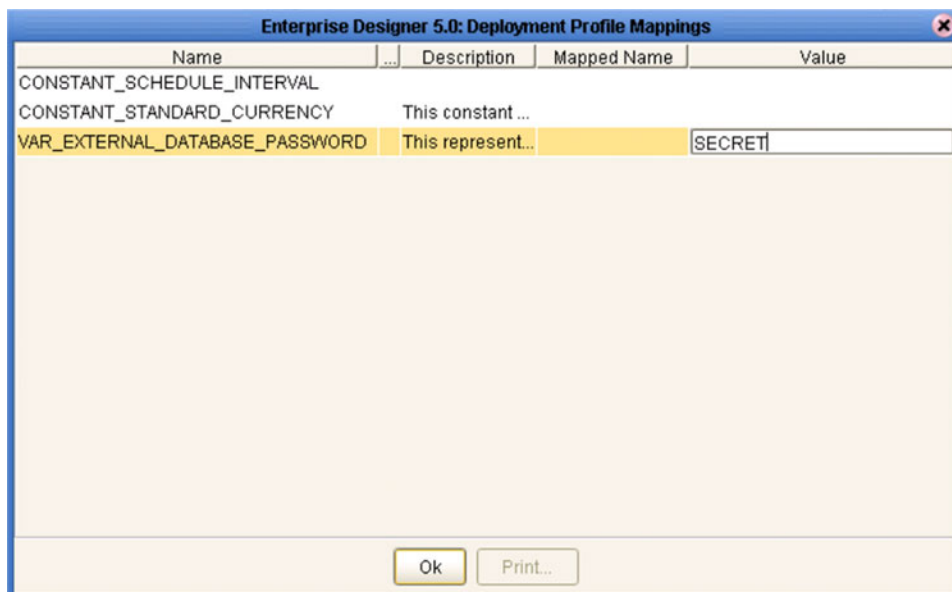


Figure 182 Project Variable Value Entry



8.8 Deploying Projects to Third-Party Servers

SeeBeyond's eGate Integrator allows you to develop Projects using Enterprise Designer and deploy them to a BEA WebLogic or IBM WebSphere environment. The SAR files for these third-party products must be installed prior to deployment, as described in the *eGate Integrator JMS Reference Guide*.

Important: The file `log4j.jar` must be added to the classpath for both WebLogic and WebSphere. This file can be obtained from the following URL:

<http://jakarta.apache.org/log4j/docs/>

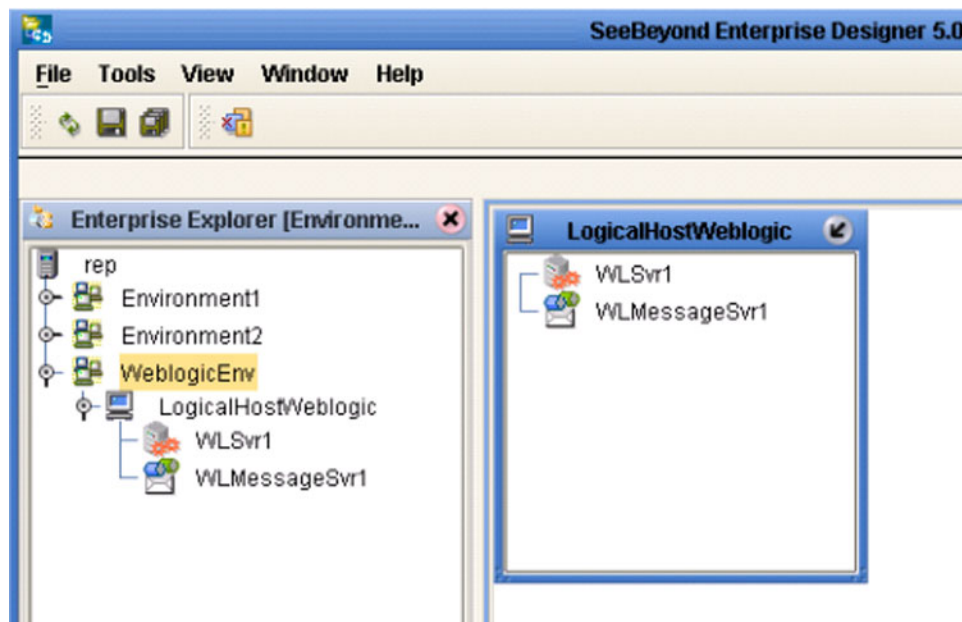
Note: The initial capability provided by eGate Integrator 5.0 is restricted to topic-to-topic Collaborations and JMS connections.

8.8.1 BEA WebLogic

To deploy an eGate Project to a BEA WebLogic environment

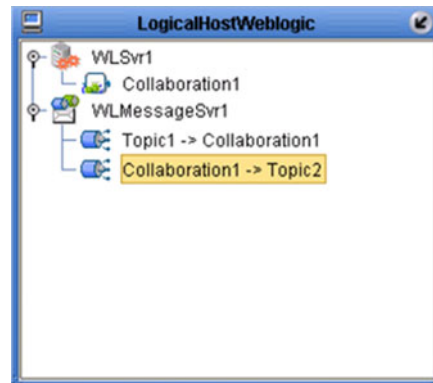
- 1 Create the following components in Enterprise Designer (see Figure 183):
 - A A new environment
 - B A Logical Host
 - C A WebLogic Integration Server
 - D A WebLogic JMS Message Server

Figure 183 WebLogic Deployment (1)



- 2 Create a new Deployment Profile to bind the Connectivity Map to the new WebLogic environment (see Figure 184).
 - A Drag the two topics and drop onto the WebLogic message server.
 - B Drag the Collaboration and drop onto the WebLogic integration server.

Figure 184 WebLogic Deployment (2)



- 3 Activate the Deployment Profile.

Activating the Deployment Profile creates an Environment Archive (EAR) file, which contains all files necessary to create and run an application in WebLogic. This file can be found in the following location (where **eGate50** is your eGate root directory):

```
eGate50\repository\data\files\WLEnvironmentName\  
ProjectName_DeploymentProfileName.ear
```

Note: The remainder of this procedure is performed in the WebLogic user interface, and is only outlined here. Please refer to your BEA WebLogic documentation for current information regarding interface layout and deployment details.

- 4 Start the BEA WebLogic server.
- 5 Navigate to **Server Administration Console > Deployments > Applications**.
- 6 Perform the following steps:
 - A Add a new JMS Connection Factory.
 - B Enter a JNDI name for the JMS Connection Factory:

```
jms/connectionfactory/xa-topic/  
LogicalHostName_MessageServerName
```

For example, the default name would be:

```
jms/connectionfactory/xa-topic/LogicalHost1_WLMessageSvr1
```

- C Verify that the WebLogic JMS Server Destination names for topics match those in eGate.
- D Select **Deploy a new Application**.
- E Upload and install the EAR file described in step 3.

- F Select the EAR file you just installed as the archive for the new application.
- G Enter a name for the new application.
- H Click **Deploy**.
- I Verify the success of the deployment (see Figure 185 , which shows a WebLogic 8.1 example).

Figure 185 WebLogic Deployment Verification

The screenshot shows the WebLogic Administration Console interface. At the top, the breadcrumb navigation is 'mydomain > Applications > weblogic-demo'. Below this, there is a status bar indicating 'Connected to : localhost :7001 | You are logged in as : rrr | Logout'. The main content area has tabs for 'Configuration', 'Targets', 'Deploy', and 'Notes', with 'Deploy' selected. A text box explains the page's purpose: 'This page allows you to view the deployment status of each module in the application, and to stop or redeploy individual modules. You may also choose to stop and redeploy all modules within the application using the buttons at the bottom of the page. (To configure additional deployment targets for this application, click the Targets tab.)'. Below this is a section titled 'Deployment status for EJB Modules' containing a table with the following data:

Module	Module Status	Target	Target Type	Status of Last Action
Collaboration1.jar	Active	myserver	Server	Success
Topic1_C290971529.jar	Active	myserver	Server	Success
Collabor_u002D_1977709066.jar	Active	myserver	Server	Success

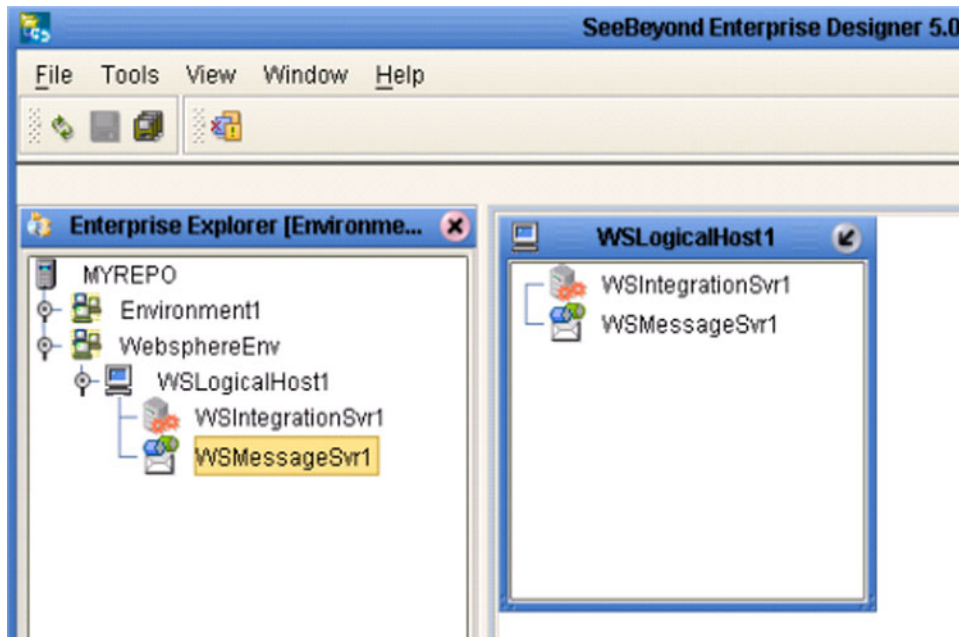
At the bottom of the table, there are two buttons: 'Stop Application' and 'Redeploy Application'. A red circle is drawn around the 'Status of Last Action' column in the table, highlighting the 'Success' status for all three modules.

8.8.2 IBM WebSphere

To deploy an eGate Project to an IBM WebSphere environment

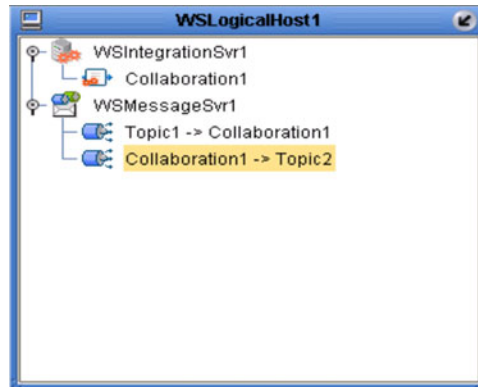
- 1 Create the following components in Enterprise Designer (see Figure 186):
 - A A new environment
 - B A Logical Host
 - C A WebSphere Integration Server
 - D A WebSphere JMS Message Server

Figure 186 WebSphere Deployment (1)



- 2 Create a new Deployment Profile to bind the Connectivity Map to the new WebSphere environment (see Figure 187).
 - A Drag the two topics and drop onto the WebSphere message server.
 - B Drag the Collaboration and drop onto the WebSphere integration server.

Figure 187 WebSphere Deployment (2)



- 3 Activate the Deployment Profile.

The activated Deployment Profile creates an Environment Archive (EAR) file, which contains all files necessary to create and run an application in WebSphere. This file can be found in the following location (where **eGate50** is your eGate root directory):

```
eGate50\repository\data\files\WSEnvironmentName\  
ProjectName_DeploymentProfileName.ear
```

Note: *The remainder of this procedure is performed in the WebSphere user interface, and is only outlined here. Please refer to your IBM WebSphere documentation for current information regarding interface layout and deployment details.*

- 4 Start the IBM WebSphere server.
- 5 From the Administrative Console, navigate to **Servers > Application Servers > server_name > Message Listener Service > Listener Ports**.
- 6 Add a new Listener port.
- 7 Enter a Connection Factory JNDI name for the new port:

```
jms/connectionfactory/xa-topic/  
LogicalHostName_MessageServerName
```

For example, the default name would be:

```
jms/connectionfactory/xa-topic/LogicalHost1_WSMMessageSvr1
```

This binds the JNDI name with the WebSphere Message Server Listener port.

- 8 From the Administrative Console, navigate to **Applications > Enterprise Applications > Install New Application**.
- 9 In *Preparing for the application installation*:
 - A Enter the path for the EAR file described in step 3 and click **Next**.
 - B Select **Generate Default Bindings** and click **Next**.

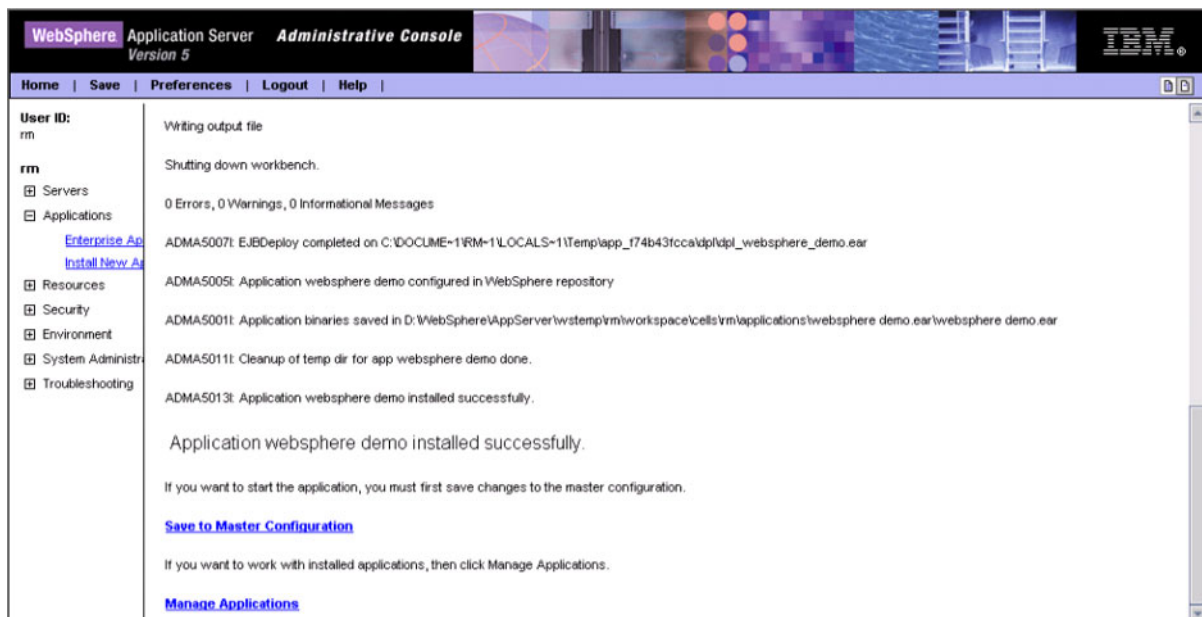
- 10 In *Step 1, Provide options ...*:
 - A Check **Deploy EJBs**.
 - B Enter the application name.
 - C Click **Next**.
- 11 In *Step 2, Provide options ...*, click **Next**.
- 12 In *Step 3, Provide Listener Ports ...*, accept the default value and click **Next**.

Note: The Listener port number should match the port number entered in step 6.

Note:

- 13 In *Step 4, Provide JNDI Names ...*, accept the default value and click **Next**.
- 14 In *Step 5, Provide EJB references ...*, accept the default value and click **Next**.
- 15 In *Step 6, Map resource references ...*, enter the JNDI name from step 7, and click **Next**.
- 16 In *Step 7, Map modules ...*, check all modules and click **Next**.
- 17 In *Step 8, (protection levels)*, check all modules and click **Next**.
- 18 In *Step 9, Summary*, click **Finish**.
- 19 Verify the success of the deployment (see Figure 188 , which shows a WebSphere 5 example).

Figure 188 WebSphere Deployment Verification



Logical Hosts

This chapter describes the procedures for configuring and running a Logical Host.

9.1 Overview

A Logical Host is an instance of the eGate runtime environment that is installed on a host hardware platform. A Logical Host can be a member of only one Environment, but each Environment can contain multiple Logical Hosts. When multiple Logical Hosts reside on a single hardware platform, you must configure the base port numbers (see [Configuring a Logical Host](#) on page 219).

9.1.1 Integration Servers

The Logical Host contains one or more Integration Servers, which are the engines that run eGate Collaborations and eWays. See [Integration Servers](#) on page 218.

9.1.2 Message Servers

The Logical Host contains one or more Message Servers, which manage JMS topics (publish-and-subscribe messaging) and queues (point-to-point messaging). eGate Integrator includes the SeeBeyond JMS IQ Manager as its Java Messaging Service (JMS) implementation. The JMS IQ Manager conforms to the Java Message specification 1.0.2b, and supports both topic (publish-and-subscribe) and queue (point-to-point) messaging styles.

eGate also includes support for HP NonStop JMS for eGate implementations on HP NonStop Server platforms. Third-party integration servers such as BEA WebLogic and IBM WebSphere incorporate their own Message Servers. For more information on the JMS IQ Manager, and deploying Project components to third-part message servers, see the *eGate Integrator JMS Reference Guide*.

9.1.3 Management Agent

The Management Agent is the master service of the Logical host. The service starts the other services on the Logical Host as part of the bootstrap process. The Management Agent also communicates with the Enterprise Manager via JMX (Java Management Extensions) to report the status of the JMS IQ Managers and Integration Servers.

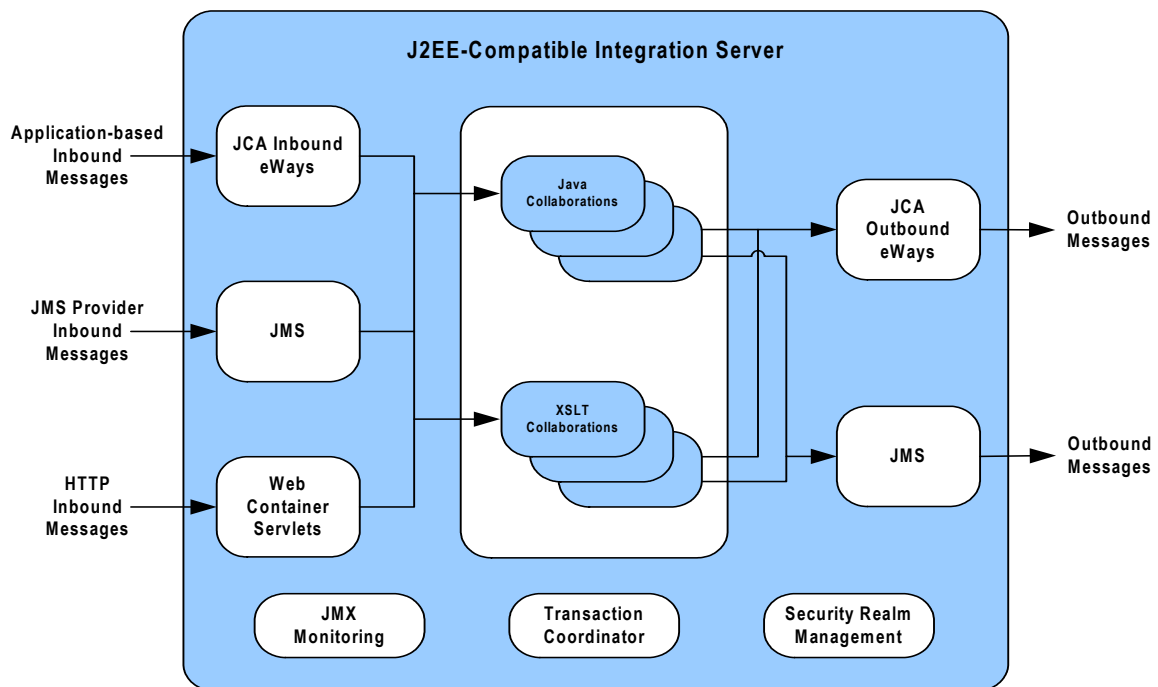
9.1.4 Bootstrap

The bootstrap is a process that launches the Management Agent and subsequent services. Each Logical Host has a separate bootstrap process. The process is started from a batch file (bootstrap/bin/bootstrap.bat). It finds the Repository via command-line parameters or from the configuration file (bootstrap/config/logical-host.properties). See [Starting the Logical Host](#) on page 224.

9.2 Integration Servers

The Integration Server is the engine that runs eGate Collaborations for processing business logic and eWays that communicate with external applications. It provides services for security, transactions, business rules execution, and connectivity management. The SeeBeyond Integration Server is based on Java 2 Enterprise Edition (J2EE). eGate Integrator also runs on third-party integration servers such as BEA WebLogic and IBM WebSphere (see [Deploying Projects to Third-Party Servers](#) on page 211).

Figure 189 Integration Server (J2EE Compatible)



The Integration Server runs the JCA eWays and Collaborations. For HTTP/HTTPS, rather than using a JCA adapter, the Integration Server uses servlets running in the Web container to provide J2EE HTTP facilities.

9.3 Configuring a Logical Host

9.3.1 Logical Host Startup Parameters

The startup command syntax is as follows:

```
bootstrap arguments
```

Table 48 Command Arguments for bootstrap

Parameter	Description	Req/Opt
-d <i>debug</i>	Overrides bootstrap sequence. Displays all cached (default) argument values	Optional
-e <i>environment name</i>	The name of the Environment to which this Logical Host belongs.	Required (first time only)
-h <i>help</i>	Overrides bootstrap sequence. Displays the usage report.	Optional
-i <i>id</i>	The user ID used for accessing the Repository. Note that the user ID is the same as the username, and that the Administrator can set up more than one user ID.	Required (first time only)
-l <i>logicalhost name</i>	The name of this Logical Host.	Required (first time only)
-n <i>physical host name</i>	The name of this Physical Host.	Required (first time only)
-p <i>password</i>	The password used for accessing the Repository.	Required (first time only)
-r <i>repository URL</i>	The root URL for the Repository containing the Logical Host data.	Required (first time only)

Note: *Required (first time only)* indicates that the argument is required the first time you start the Repository. You do not need to use it again unless you change the name of the Logical Host.

9.3.2 Modifying the Logical Host Startup Configuration File

Instead of setting the arguments when you start the Logical Host at the command prompt, you can preset them in the **logical-host.properties** file. Although set, the arguments can be overridden by starting the Logical Host as described in [To start the Logical Host](#) on page 226.

To modify the startup configuration file

- 1 Navigate to `c:\eGate50\logicalhost\bootstrap\config\logical-host.properties`.
- 2 Use a text editor (such as Windows WordPad) to open the file.

Figure 190 Example logical-host.properties File

```
#####
#
#                               Logical Host Properties
#
#####

# These properties are automatically persisted by the bootstrap sequence.
# They are used by default if none are provided at the command line.

#####
# repository.url: (USER MODIFIABLE)
#   Specifies the remote URL for connecting to the repository.
#   Takes the form:
#       http://<repository-server-hostname>:<port>/
#       <repository-name>
#   For example:
#       http://localhost:10000/myRep
#####
repository.url=

#####
# repository.username: (USER MODIFIABLE)
#   Username for connecting to the repository.
#####
repository.username=

#####
# repository.password: (USER MODIFIABLE)
#   Plain text form of password used for connecting to the
#   repository. Any value provided here will be cleared out
#   by the system and written in encrypted form to the
#   repository.password.encrypted field.
#####
repository.password=

#####
# repository.password.encrypted:
#   Encrypted form of the repository password. NOTE: This value
#   is generated by the system, so it is improper to edit this
#   field manually.
#####
repository.password.encrypted=

#####
# physical.host.name: (USER CONFIGURABLE)
#   Specifies the physical host on which this logical host is
#   running. The host name should include the domain name.
#   Example: host.company.com
#####
physical.host.name=

#####
# logical.host.environment.name: (USER MODIFIABLE)
#   Specifies the name of the environment containing the
#   current logical host.
#####
logical.host.environment.name=

#####
# logical.host.name: (USER MODIFIABLE)
#   Specifies the name of the current logical host.
#####
logical.host.name=
```

```
#####
# logical.host.root.dir:
#       Specifies the root directory of a logical host
#       installation.
#####
logical.host.root.dir=

#####
# os.type:
#       Specifies the OS type of the machine on which logical host
#       is going to run
#####
os.type=

# *** THE FOLLOWING PROPERTIES ARE PRIVATE; MODIFY AT YOUR OWN RISK ***

# bootstrap
managementagent.jar.path=lib/managementagent.jar

# deployment manager
deployment.manager.jar.relative.path=deploymentmanager.jar

# management agent
managementagent.config.file=./config/ManagementAgent-config.xml
managementagent.command.line.windows=cmd.exe /c ..\jre\bin\java.exe -Xrs -
Dlogical.host.properties.file=.\bootstrap\config\logical-host.properties -classpath
.\config;.\lib\managementagent.jar;.\lib\com.stc.hnsems.jar;.\lib\jta.jar;.\lib
\jms.jar;.\lib\jta.jar;.\lib\concurrent.jar;.\lib\log4j.jar;.\lib\jargs.jar;.\
lib\mx4j.jar;.\lib\mx4j-
tools.jar;.\lib\stcjs.jar;.\lib\stcrepository.jar;.\lib\stccore.jar;.\lib\stcr
epositoryclient.jar;.\lib\stcrepositoryserver.jar;.\lib\velocity.jar;.\lib\ejb.ja
r;.\lib\commons-
collections.jar;.\stcis\lib\com.stc.isbootstrap.jar;.\stcis\lib\gnu-regexp-
1.1.4.jar;.\lib\com.stc.isapi.jar;.\stcis\lib\com.stc.isimpl.jar;.\lib\com.stc.
compatimpl.jar;.\lib\com.stc.eventmanagementapi.jar;.\lib\xmlrpc.jar;.\lib\smscli
ent.jar;.\lib\com.stc.stcutil4Jimpl.jar;.\lib\xercesImpl.jar;.\lib\xalan.jar;.\l
ib\xml-apis.jar;.\lib\stcqueueviewer.jar;.\lib\stcjournaller.jar;.\lib\jakarta-
regexp-1.2.jar;.\lib\commons-codec-1.1.1-dev.jar;
com.stc.processmanager.ma.ManagementAgent

managementagent.command.line.unix=sh ../bootstrap/bin/magent.sh

managementagent.shutdown.windows=cmd.exe /c ..\jre\bin\java.exe -
Djava.ext.dirs=.\lib com.stc.sms.mbeans.ManagementAgentControl -f
.\config\ManagementAgent-config.xml -c shutdown

managementagent.shutdown.unix=sh ../jre/bin/java -Djava.ext.dirs=./lib
com.stc.sms.mbeans.ManagementAgentControl -f ./config/ManagementAgent-config.xml -c
shutdown

# misc
repository.relative.url=data/repository/SBYN00.properties
log4j.config.path=bootstrap/config/log4j.xml
jre.zip.name=jre.zip
```

- 3 In the top part of the **logical-host.properties** file, locate the following properties (they may appear in a different order than that shown):

```
logical.host.environment.name=
logical.host.name=
logical.host.root.dir=
os.type=
physical.host.name=
repository.url=
repository.username=
repository.password=
repository.password.encrypted=
```

- 4 Set the properties in your **logical-host.properties** file. See the following table for descriptions of the properties' values:

Note: Do not enter spaces before or after the equal (=) sign and the property values. Spaces are allowed only in the value itself.

Table 49 Logical Host Properties file

Property	Description
logical.host.environment.name	Specifies the name of the Logical Host environment deployment, for example, EnvironmentDeployment1 .
logical.host.name	Specifies the name for the Logical Host, for example, logicalhost1 .
logical.host.root.dir	Specifies the full path of the Logical Host directory, for example, c:\eGate50\logicalhost .
os.type	Specifies the operating system type under which logical host is going to run.
physical.host.name	Specifies the physical host on which this logical host is running. The host name should include the domain name, for example: host.company.com .
repository.url	Specifies the name of the Repository URL, for example http://hostname:port/repositoryname where: <ul style="list-style-type: none"> ▪ hostname is the physical name of the computer on which the Repository resides. For example, localhost. ▪ port is the port number that the Repository uses to receive requests. For example, 9999. ▪ repositoryname is the name you specified for the Repository. For example, Repository1.
repository.username	Specifies the user name you are using to access the Repository, for example, Administrator .
repository.password	Specifies the password you are using to access the Repository, for example, myPassword . When you launch the bootstrap process, this password is encrypted and written to the repository.password.encrypted property. After the encrypted password has been written, this repository.password value is removed from the logical-host.properties file.
repository.password.encrypted	This property is automatically updated based on changes made to the repository.password property. <i>Do not enter anything for this property or modify its contents.</i>

- 5 Save the **logical-host.properties** file.

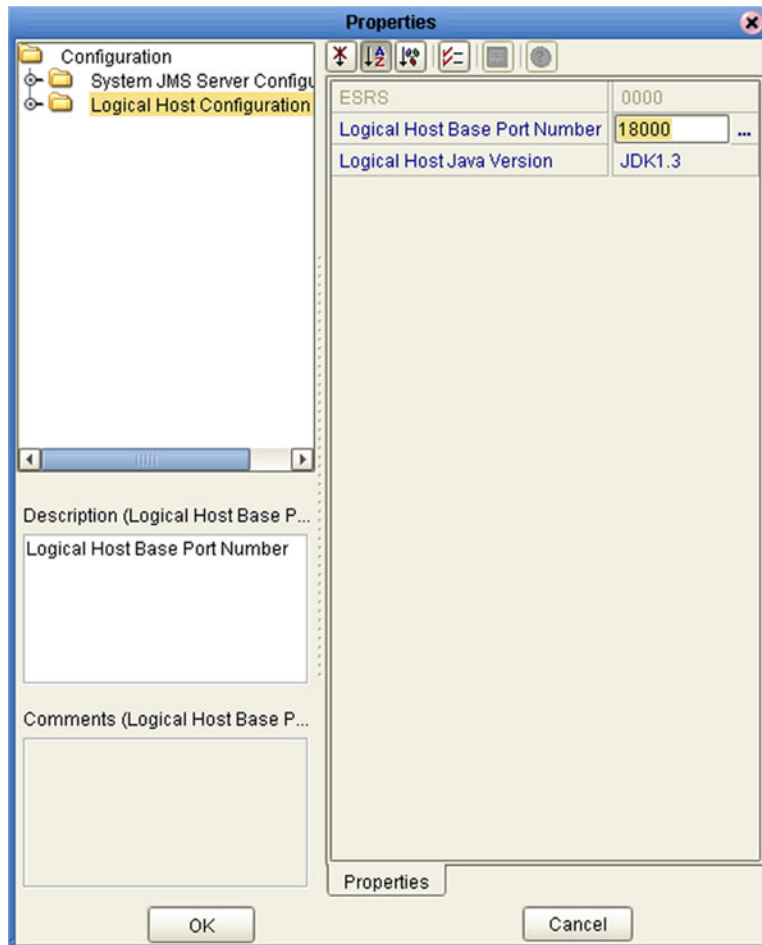
To start the Logical Host after setting the configuration file arguments

- 1 Navigate to **c:\eGate50\logicalhost\bin**.
- 2 Double-click **bootstrap.bat**.

9.3.3 Configuring the Base Port Number

Port numbers assigned to the Logical Host and its components (Integration Server, JMS IQ Manager, etc.) can be viewed and modified in the Enterprise Explorer by selecting **Properties** from the component's context menu (see Figure 191).

Figure 191 Logical Host Properties Dialog Box



If multiple Logical Hosts reside in the same Environment, you must ensure that each Logical Host has a different base port number to avoid conflicts. This base port number is propagated throughout the Logical Host, so that the various components are automatically given successive port numbers following that assigned to the Logical Host itself.

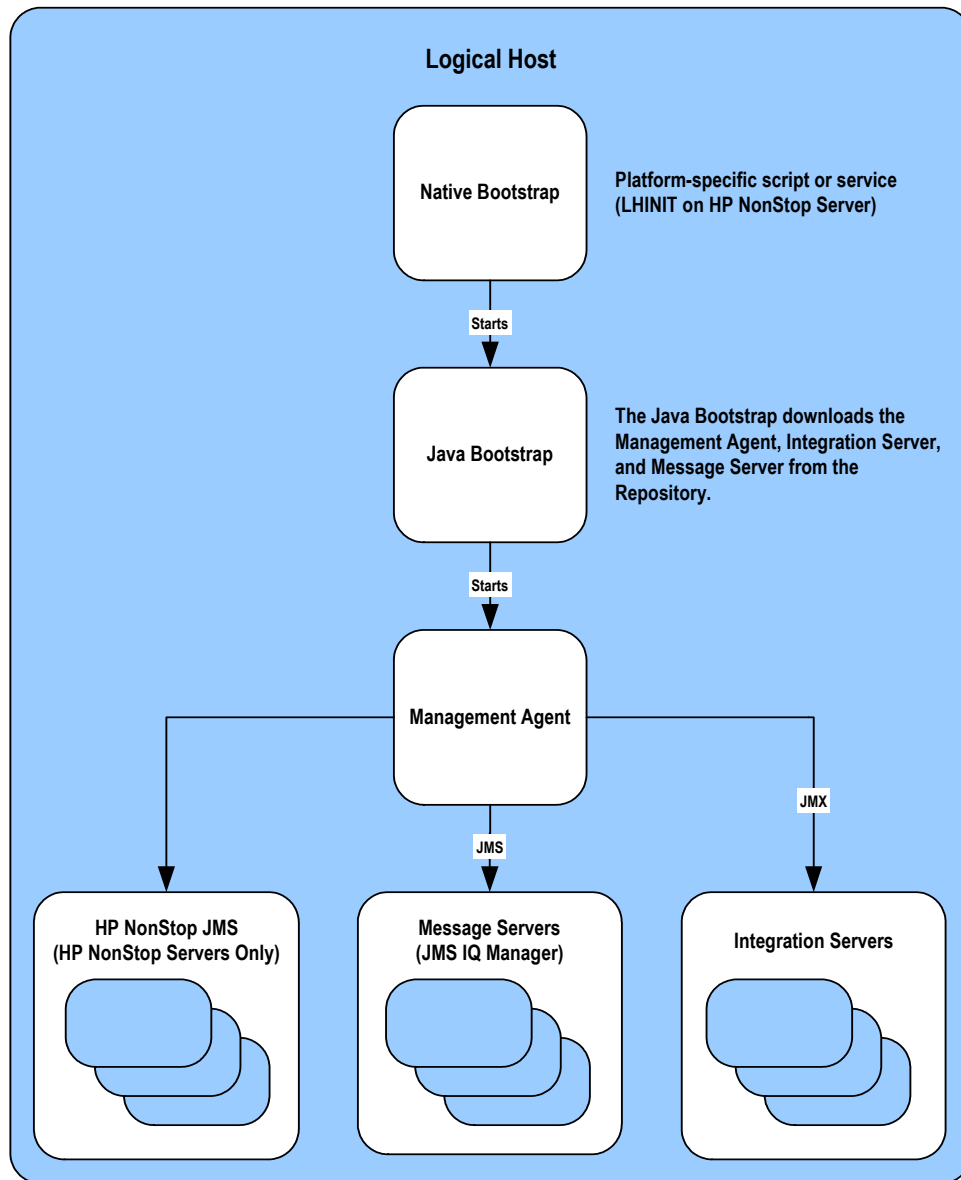
The number of port numbers used in a Logical Host varies according to the specific implementation, so when assigning new base port numbers you need to skip successive numbers by an adequate amount. The default base port number is 18000, so base port numbers of 19000, 20000, and so on are recommended.

If you need to assign a specific port number to a particular Logical Host component, you are allowed to do so. In this case, the automatic numbering process will skip the component port number you have assigned manually. Just be sure this port number is not used elsewhere.

9.4 Starting the Logical Host

The Logical Host can be started manually by using the bootstrap batch script specific to the platform where it is running (see Figure 192). Additionally, the Logical Host can be started as a service on Windows platforms only.

Figure 192 Logical Hosts



At run time, the bootstrap script starts the bootstrap Java program that downloads the Management Agent, the Message Server, and the Integration Server from the Repository. The Management Agent is then started, which in turn starts the Message Server(s) and Integration Server(s).

9.4.1 Starting the Logical Host as a Windows Service

Installing the Logical Host as a Windows service configures the Logical Host to automatically start up at system startup. This makes it possible for the Logical Host to automatically restart after an abnormal system shutdown.

Starting the Logical Host as a service requires you to have configured the **logical-host.properties** file. For instructions on configuring the **logical-host.properties** file, see [“Modifying the Logical Host Startup Configuration File” on page 220](#).

You must have Administrator rights to the local Windows machine in order to configure the Logical Host to start as a service. The installation script writes to the Windows Registry; this cannot be done without Administrator rights.

To start the Logical Host as a Windows Service

- 1 Use Windows Explorer to navigate to the Logical Host’s **bootstrap\bin** directory. For example, `c:\eGate50\LogicalHost\bootstrap\bin`.
- 2 Double-click **install-as-nt-service.bat**.

This runs the script that configures the Windows service to automatically run the Logical Host at system startup. The only parameter is the name of your Logical Host Service; the default is “ICAN 5.0 Logical Host” (see Figure 193).

Figure 193 Install as Service Script

```
C:\ican50\logicalhost\bootstrap\bin>install-as-nt-service.bat

-----
--- eGate 5.0 Management Agent Install Bootstrap NT Service ---
--- Copyright (c) 2003, SeeBeyond Technology Corporation, ---
--- All Rights Reserved ---
-----

To start the service use: net start "ICAN 5.0 LogicalHost"
To stop the service use: net stop "ICAN 5.0 LogicalHost"
C:\ican50\logicalhost\bootstrap\bin\..\..
C:\ican50\logicalhost\bootstrap\bin\..\..
C:\ican50\logicalhost\bootstrap\bin\..\..\bootstrap\lib
C:\ican50\logicalhost\bootstrap\bin\..\..\bootstrap\config
The service was successfully installed.
```

- 3 Verify the installation by opening the Windows Services facility and searching for the Logical Host name (see Figure 194). Note that by default, the service is listed as *Automatic*—it will not be running, however, until you click **Start** or reboot the computer.

Figure 194 Windows Services List

Hummingbird Ineco	Ineco Hummi...		Manual	LocalSystem
Hummingbird Jconfi...	Jconfig Da...		Manual	LocalSystem
ICAN 5.0 LogicalHost			Automatic	LocalSystem
IIS Admin Service	Allows adm...	Started	Automatic	LocalSystem
Indexing Service		Started	Automatic	LocalSystem
Informatica			Manual	STC\ivanh

To remove the Logical Host Windows Service

- 1 Use Windows Explorer to navigate to the Logical Host's **bootstrap\bin** directory. For example, **c:\eGate50\LogicalHost\bootstrap\bin**.
- 2 Double-click **uninstall-nt-service.bat**.

This runs the script that removes the Windows service that automatically starts the Logical Host at system startup (see Figure 195).

Figure 195 Uninstall as Service Script

```
C:\ican50\logicalhost\bootstrap\bin>install-as-nt-service.bat

-----
eGate 5.0 Management Agent Install Bootstrap NT Service
Copyright (c) 2003, SeeBeyond Technology Corporation,
All Rights Reserved
-----

To start the service use: net start "ICAN 5.0 LogicalHost"
To stop the service use: net stop "ICAN 5.0 LogicalHost"
C:\ican50\logicalhost\bootstrap\bin\..\..
C:\ican50\logicalhost\bootstrap\bin\..\..
C:\ican50\logicalhost\bootstrap\bin\..\..\bootstrap\lib
C:\ican50\logicalhost\bootstrap\bin\..\..\bootstrap\config
The service was successfully installed.
C:\ican50\logicalhost\bootstrap\bin>uninstall-nt-service.bat

-----
eGate 5.0 Management Agent Uninstall Bootstrap NT Service
Copyright (c) 2003, SeeBeyond Technology Corporation,
All Rights Reserved
-----

The service was successfully uninstalled.
```

9.4.2 Starting the Logical Host Manually on a Windows System

Before you can manually start a Logical Host you must first create a Project in the Enterprise Designer (see [Enterprise Designer](#) on page 32). After completing a Project, you start the Logical Host by opening a **command prompt** and typing a command on the command line which includes a string of required, optional, and initial (first-time only) command line arguments appended to the startup command. The startup command string uses the following syntax:

```
<path>\logicalhost\bootstrap\bin>bootstrap <arguments>
```

For example, the command

```
c:\eGate50\logicalhost\bootstrap\bin>bootstrap -h
```

displays Help text that explains the command line arguments in detail.

To start the Logical Host

- 1 On your desktop, open a **command prompt**.
- 2 Navigate to **c:\eGate50\logicalhost\bootstrap\bin**.
- 3 To start the Logical Host, on the command line, do one of the following:
 - If you are starting the Logical Host for the first time, run the bootstrap command and include all of the required parameters shown in Table 48.

- If you are starting the Logical Host after having started it at least once previously, do the following:
 - A Navigate to `c:\eGate50\logicalhost\bin`.
 - B Double-click `bootstrap.bat`.

9.4.3 Starting the Logical Host on a UNIX System

To start the Logical Host

- 1 Navigate to the `bootstrap/bin` directory in the location where you installed the Logical Host, for example:

```
cd /home/my_user_name/egate50/logicalhost/bootstrap/bin
```

- 2 Run the bootstrap script using the following command:

```
./bootstrap.sh arguments
```

See [Logical Host Startup Parameters](#) on page 219 for a complete list of the command-line arguments. Additionally, see [Modifying the Logical Host Startup Configuration File](#) on page 220 for instructions on customizing the `logical-host.properties` file.

Note: Remember that the Logical Host service will continue to run until you manually shut it down.

9.4.4 Starting the Logical Host on a Red Hat Linux System

To start the Logical Host

- 1 Navigate to the `bootstrap/bin` directory in the location where you installed the Logical Host, for example:

```
cd /home/my_user_name/egate50/logicalhost/bootstrap/bin
```

- 2 Run the bootstrap script using the following command:

```
./bootstrap.sh arguments
```

- 3 After the bootstrap command is executed, the script prompts you for the RedHat server release number.

See [Logical Host Startup Parameters](#) on page 219 for a complete list of the command-line arguments. Additionally, see [Modifying the Logical Host Startup Configuration File](#) on page 220 for instructions on customizing the `logical-host.properties` file.

9.4.5 Starting the Logical Host on an HP NonStop Server

On HP NonStop Servers, a platform-specific script called LHINIT provides the bootstrap service. This script works with the pathway process to define the primary and backup CPUs for the Logical Host.

To start the Logical Host

- 1 Open a **command prompt** and type the following startup command string:

```
>sh ./admin.sh start <CPU number>
```

For example:

```
>sh ./admin.sh start 2
```

- 2 To check the status of the bootstrap, type the following string:

```
>sh ./admin.sh status <CPU number>
```

Repository Tools

The Repository stores and manages the setup, component, and configuration information for all of your Projects. To take advantage of this design, the Enterprise Designer includes several analysis and archiving tools. This chapter describes how to use each of these tools with your Projects.

Note: Refer to [Repository](#) on page 28 for more details about the Repository's role in eGate Projects.

10.1 Overview

Each of the Repository-related tools covered in this chapter is briefly discussed below:

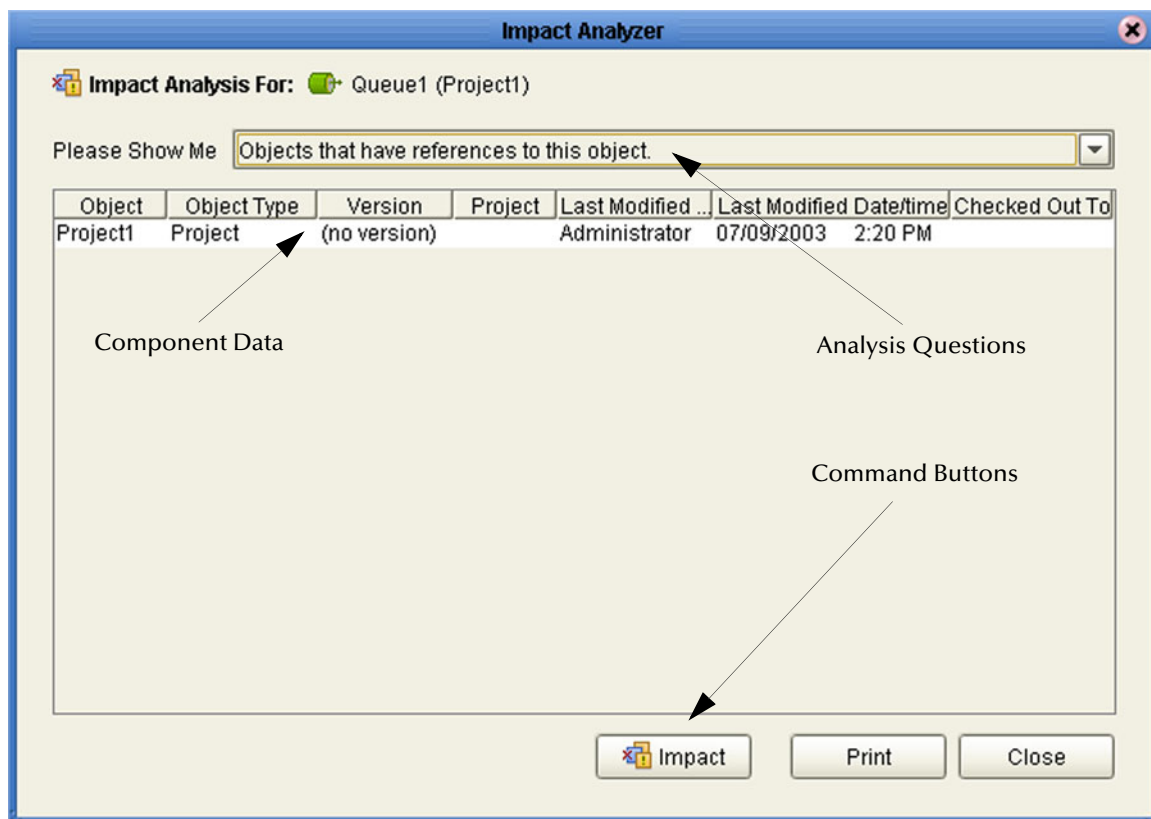
- **Impact Analyzer**
Helps you visualize how a change to one part of a Project would affect the rest of the Project. See [Impact Analysis](#) on page 230.
- **Version Control**
Allows you to maintain multiple versions of Project components. See [Version Control](#) on page 233.
- **Repository Backup/Restore**
Allows you to back up the Repository to an external file and restore a Repository from an external file. See [Repository Backup and Restoration](#) on page 236.
- **Project Export/Import**
Allows you to export a Project from Enterprise Designer to an external file, and import a Project into Enterprise Designer from an external file. See [Project Export and Import](#) on page 238.

10.2 Impact Analysis

The Impact Analyzer tool helps you to determine how changing one part of a Project (or Environment) will affect the rest of the Project. You accomplish this by selecting a Project component and then picking a question to ask about the Project.

Select a Project component, then select **Tools > Impact Analyzer** or click the **Impact Analyzer** toolbar button to display the Impact Analyzer dialog box shown in Figure 196.

Figure 196 Impact Analyzer Dialog Box






The Impact Analyzer is organized into three sections represented by tabs. Each of these tabs is briefly described below:

- **Analysis Questions**
Includes a a drop-down list that contain questions to ask about the displayed component.
- **Component Data**
Lists information about the affected components, including the name of the component, associated Project, who last modified the component, and the date/ time of the last modification.
- **Command Buttons**
Includes four buttons used to execute commands in the dialog box. Refer to [Command Buttons](#) on page 231 for more details.

10.2.1 Command Buttons

Each command button on the Impact Analyzer is briefly described in Table 50.

Table 50 Navigation Buttons

Button	Function
	Select an item from the Component Data list, then click Impact to perform an impact analysis for the that item. See “ Using the Impact Analyzer ” for more information.
	Displays the Print dialog box, which you can use to print the list of component data. Refer to Printing Impact Analysis Results on page 232 for more details.
	Closes the Impact Analyzer dialog box.

10.2.2 Using the Impact Analyzer

This section describes how to use the Impact Analyzer to figure out how changing a Project component impacts other Project components.

To use the Impact Analyzer

- 1 In the Enterprise Explorer, click the **Project Explorer** tab.
- 2 Select a Project component.
- 3 Use one of the following methods to display the **Impact Analyzer** dialog box.
 - ♦ Select **Tools > Impact Analyzer**.
 - ♦ Click the **Impact Analyzer** toolbar button.
- 4 In the *Please show me* drop-down list, select a question to view the Project components that meet the criteria of the question.

Note: Select a Project component on the Component Data list and click **Impact** to see how that component is affected.

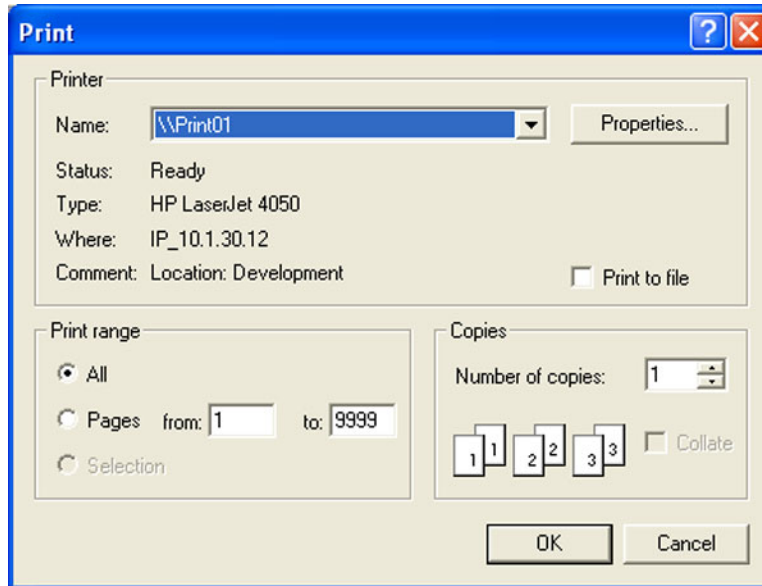
10.2.3 Printing Impact Analysis Results

This section explains how to print the results of component analysis performed with the Impact Analyzer.

To Print Component Data

- 1 From the **Impact Analyzer** dialog box, click **Print** to display the Print dialog box shown in Figure 197.

Figure 197 Print Dialog Box



- 2 Click **OK** to print the component data to your default printer. Your printout will look like Figure 198 shown below.

Figure 198 Printed Component Data

Object	Version	Project	Last Modified By	Last Modified Date/time	Checked Out To
Topic1	(no version)	Test Project	Ben Lee	05/29/2003 11:20 AM	
Queue1	(no version)	Test Project	John Smith	05/29/2003 11:12 AM	
eGate ProjectManager API	(no version)	Unknown	John Smith	05/29/2003 11:12 AM	
Test CM	(no version)	Test Project	Ben Lee	05/29/2003 11:20 AM	
Test Deployment	(no version)	Test Project	John Smith	05/29/2003 11:12 AM	

10.3 Version Control

Version control allows you to maintain several versions and a version history of Project components. Once you have created and configured a Project component, you are able to check that object in. The initial check in occurs when you right-click on a component and select **Check In** from the Component context menu. You use the same method to check out Project components.

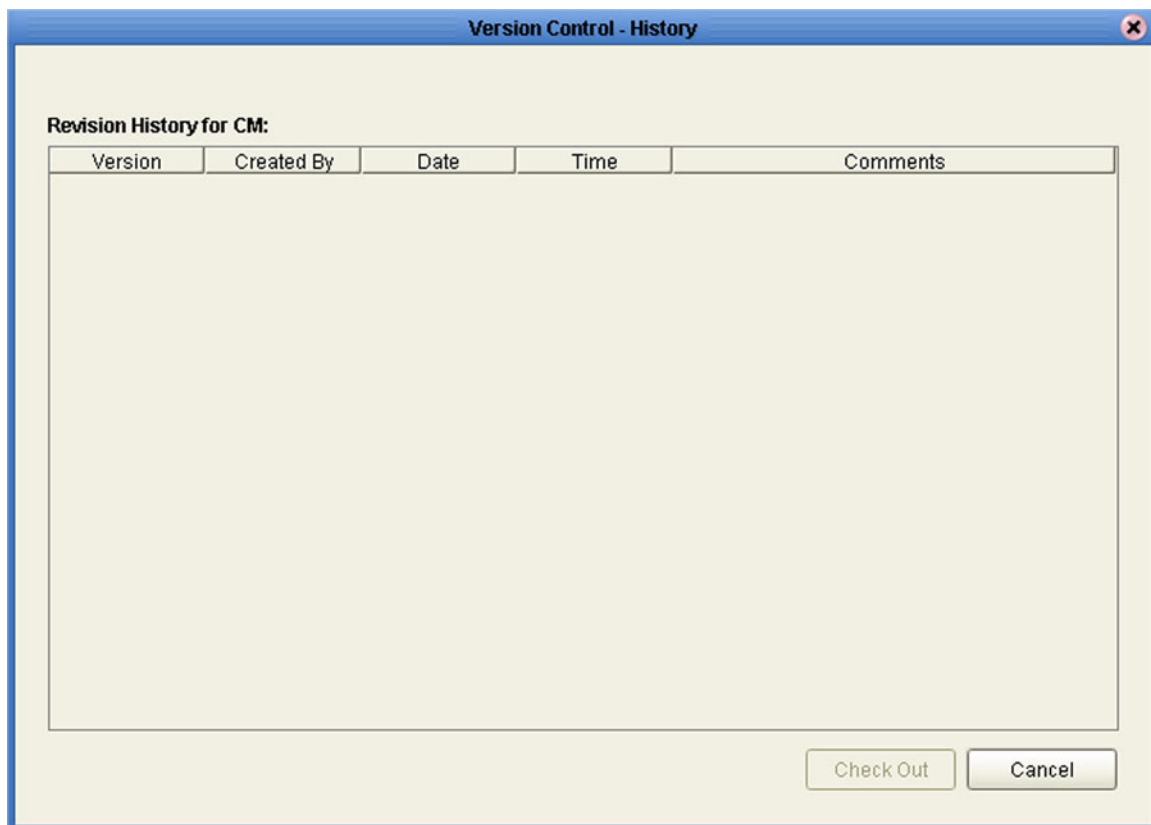
10.3.1 Viewing History of a Project Component

This section describes how to view the version history of a Project component.

To view version history for a Project component

- 1 Click the Project Explorer tab from the Enterprise Explorer.
- 2 Right-click on an icon to display a context menu for that Project component.
- 3 Select **Version History** to display the *Version Control - History* dialog box shown in Figure 199.

Figure 199 Version Control - History Dialog Box



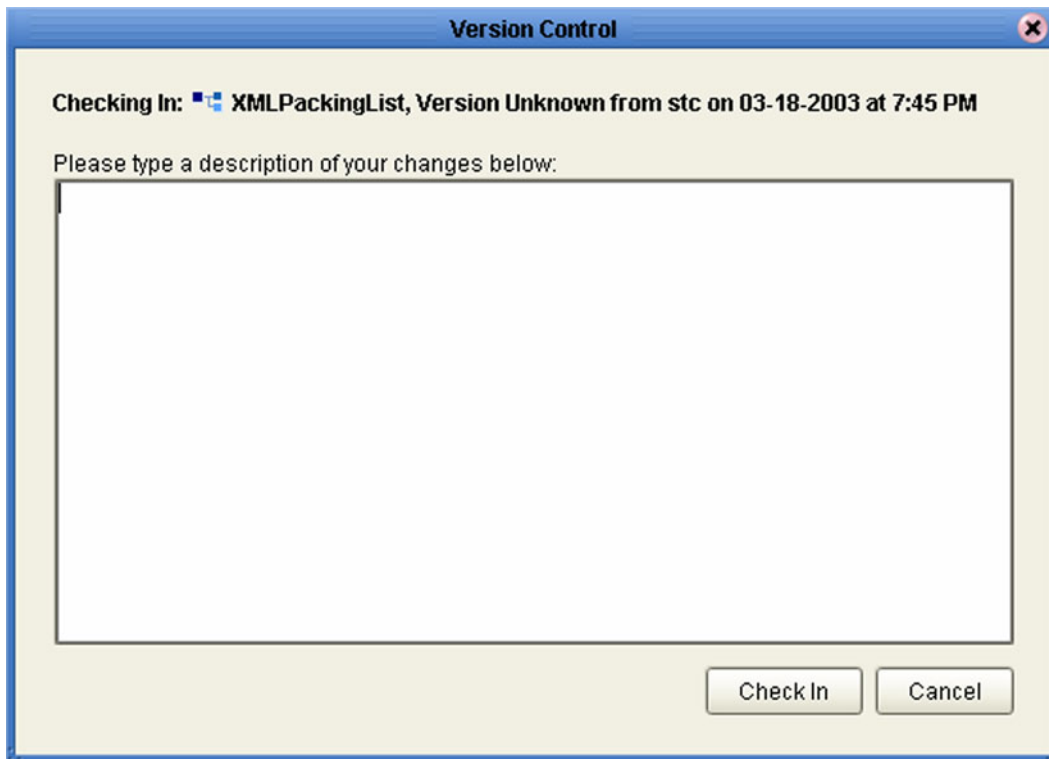
10.3.2 Checking In a Project or Environment Component

This section describes how to check in a version of a Project/Environment component.

To check in a version of a Project/Environment component

- 1 Click the Project/Environment Explorer tab from the Enterprise Explorer.
- 2 Right-click on an icon to display to display the component's context menu.
- 3 Select **Check In** to display the *Version Control* dialog box shown in Figure 200.

Figure 200 Version Control Dialog Box



- 4 Click **Check In** to save your changes to a new version.

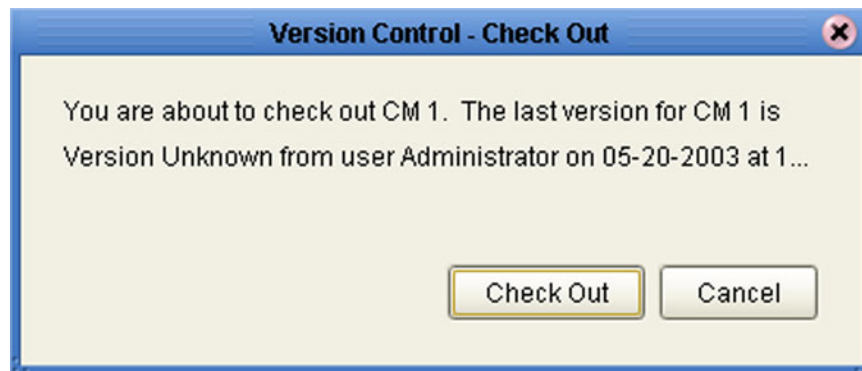
10.3.3 Checking Out a Project or Environment Component

This section describes how to check out a version of a Project/Environment component.

To check out a version of a Project/Environment component

- 1 Click the Project/Environment Explorer tab from the Enterprise Explorer.
- 2 Right-click on an icon to display to display the component's context menu.
- 3 Select **Check Out** to display the *Version Control - Check Out* dialog box shown in Figure 201.

Figure 201 Version Control - Check Out Dialog Box



- 4 Click **Check Out** to open the component.

Note: *Only one user can have a file checked out for editing at a time. If another user attempts to check out the same file, they will receive a message indicating that the file is currently checked out.*

10.4 Repository Backup and Restoration

10.4.1 Backing Up a Repository

The backup function allows you to back up an entire eGate Repository using a command-line script. The backup script creates a backup of all the repository objects and files in the **repository\data** directory including .jar, .nbn, and other binaries, workspaces, users and locks.

During the process of backup the server is locked, so that users are not able to change objects while a backup is in progress. The backup files are .zip files and can be viewed using a decompression utility such as WinZip.

Location of script file:

```
repository/util/backup.bat (or .sh)
```

Command Syntax:

```
backup username password filename
```

Important: *The Repository backup produces a complete snapshot of the Repository, including all installed products—the resulting file, even though compressed, is very large.*

To back up a Repository using the backup script

- 1 Open a command prompt and change directory to *source_repository_root/util*.
- 2 Type (for example): **backup Administrator stc c:/mybackup.zip**.

10.4.2 Restoring a Repository

The restore function allows you to restore an entire eGate Repository using a command-line script. The restore script restores from a backup file. It will wipe out any existing objects/files in the repository and overwrite them with the values from the backup file.

In effect, it will restore the complete snapshot of the repository contained in the backup file including the workspaces, users and locks (checkouts). You can restore the backup to the same repository or a different repository.

Before restoring, the repository server must be running. During the restore process the repository is locked. You **must** restart the repository server after restoring.

Location of script file:

```
repository/util/restore.bat (or .sh)
```

Command Syntax:

```
restore username password filename
```

When restoring a Repository, note that:

- Restoring overwrites the contents of the target Repository.
- After restoring a Repository, you must:
 - A Restart the Repository.
 - B Reactivate all deployments.

To restore a Repository using the restore script

- 1 Open a command prompt and change directory to *target_repository_root/Util*.
- 2 Type (for example): **restore Administrator stc c:/mybackup.zip**.
- 3 Restart the Repository.
- 4 Stop Enterprise Designer.
- 5 Restart Enterprise Designer.

10.5 Project Export and Import

Logs of these processes can be found in *repository_root/logs/repository.log*, where *repository_root* represents the path to your repository root directory.

10.5.1 Exporting a Project

The export function allows you to export an eGate Project to an external file using either the Enterprise Designer or a command-line script.

When exporting a Project, note that:

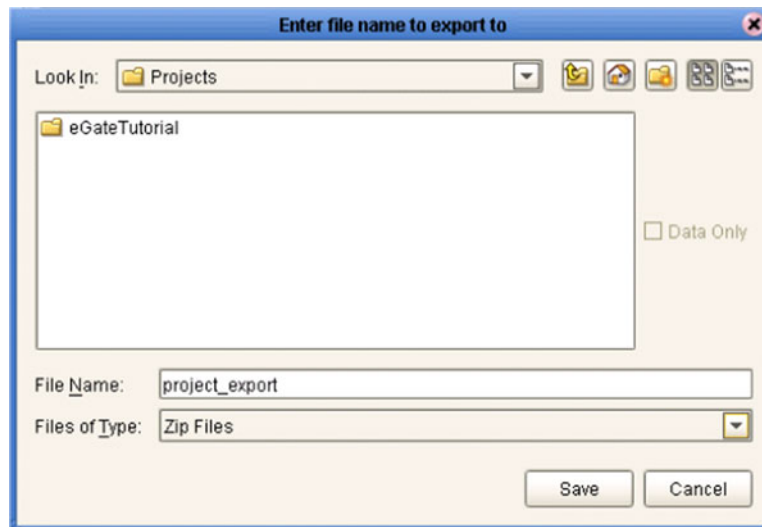
- The exported Project may have references to elements that are in other Projects. A list of such references is generated during the export process.
- Project deployment objects are not exported, because they have references to both Project and Environment elements that are not required at the Project level.

Enterprise Designer Method

To export a Project using Enterprise Designer

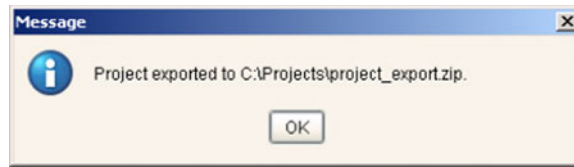
- 1 From the Project context menu, select **Export Project ...** to display the *Enter file name to export to* dialog box shown in Figure 202.

Figure 202 Enter File Name Dialog Box



- 2 Select a location to save the Project file to using the **Look In** drop-down list.
- 3 Enter a name for the Project file in the **File Name** text box.
- 4 Click **Save** to save the Project file. The Message dialog box shown in Figure 203 is displayed when the Project file has been created. (This process takes a few minutes.)

Figure 203 Message Dialog Box



- 5 Click **OK** to close the Message dialog box.

Command-Line Method

You can also export a Project using the following command-line script.

Location of script file:

```
repository_root/util/exportProject.bat (or .sh)
```

Command Syntax:

```
exportProject repositoryURL username password filename projectname
```

To export a Project using the export script

- 1 Open a command prompt and change directory to *export_repository_root/util*.
- 2 Type (for example): **exportProject http://localhost:12345/myrepository Administrator stc c:/myprojectExport.zip myProject1.**

10.5.2 Importing a Project

The import function allows you to import an eGate Project file using the Enterprise Designer.

When importing a Project, note that:

- Existing Projects are not affected by the imported Project.
- During import, if another Project having the same name exists in the target Repository, you will receive an error message and the existing file will not be overwritten.
- You can specify a new Project name and location (in Project Explorer) during import.
- References are validated during import.

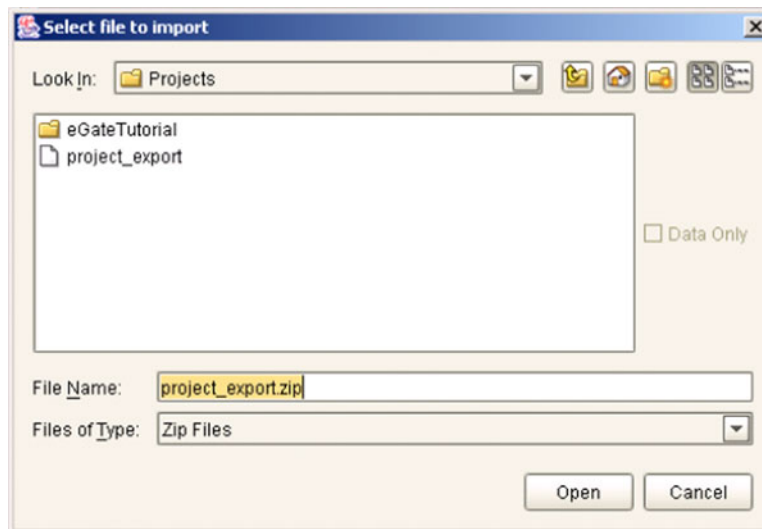
Important: APIs installed in the source Repository must be installed in the Repository into which the Project is imported.

Enterprise Designer Method

To import a Project using Enterprise Designer

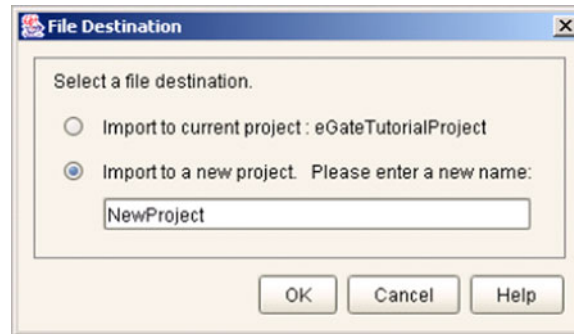
- 1 Create a new Project folder in Project Explorer.
- 2 From the Project context menu, select **Import Project ...** to display the *Select file to import* dialog box shown in Figure 204.

Figure 204 Select File Dialog Box



- 3 Locate and select the Project file that you want to import.
- 4 Click **Open** to import the file.
- 5 A File Destination dialog box will appear as shown in Figure 205, in which you choose whether to import to the same Project name or to a different name.

Figure 205 File Destination Dialog Box



- 6 If you choose to import to a new Project, enter the name for the Project.
- 7 Click **OK** to import the file.
- 8 The Message dialog box shown in Figure 206 displays when the Project file has been imported. (This process takes a few minutes.)

Figure 206 Message Dialog Box



- 9 Click **OK** to close the Message dialog box.

Command-Line Method

You can also import a Project using the following command-line script.

Location of script file:

```
repository_root/util/importProject.bat (or .sh)
```

Command Syntax:

```
"importProject repositoryURL username password filename  
projectname ParentProjectName"
```

Note: If root is the parent, replace **ParentProjectName** with "".

To import a Project using the import script

- 1 Open a command prompt and change directory to `import_repository_root/util`.
- 2 Type (for example): `"importProject http://localhost:12345/myrepository Administrator stc c:/myprojectExport.zip myNewProject1 myExistingProject1"`.

This will take the Project that exists in the export file, rename it to `myNewProject1` and attach it as a subproject of `myExistingProject1`. If the Project is to be attached to the root, then leave the last parameter as an empty string. Make sure you include the quotes ("") around the entire command.

Security

This chapter contains information on the various security features provided in the ICAN Suite.

11.1 Overview

Security features described in this chapter are as follows:

- **Configuration User Management** on page 243 describes the management of users in the ICAN Suite.
- **Environment User Management** on page 246 describes the management of users who would access the applications deployed in an enterprise, using the ICAN Suite.
- **ACL Management** on page 248 describes the management of access control to various components and features in the ICAN Suite.
- **SSL/HTTPS** on page 250 describes the use of Secure Sockets Layer protocols in Web communications.
- **LDAP** on page 254 describes how to configure the ICAN Suite to use an LDAP server for authentication.

11.2 Configuration User Management

This section describes the management of users for the ICAN Suite. These users are able to use the ICAN suite of tools for design, administration, and management of applications in an enterprise.

11.2.1 Roles

The Enterprise Designer allows you to manage user access, based on *roles* and user ID. There are three predefined roles in the ICAN Suite:

- **all**
Users assigned this role can log in to ICAN 5.0. Once logged in, they can inherently connect to the Repository, perform downloads, and access documentation in Enterprise Manager.

Note: The **all** role is assigned to **all users** registered on the system, and provides the most basic, and restrictive, access privileges.

- **administration**
Users assigned this role can log in and connect to the Repository, perform downloads *and uploads*, and access documentation in Enterprise Manager.
- **management**
Users assigned this role can log in and connect to the Repository, perform downloads *but not uploads*, and access documentation in Enterprise Manager. They also can start and stop components using the eGate Monitor.

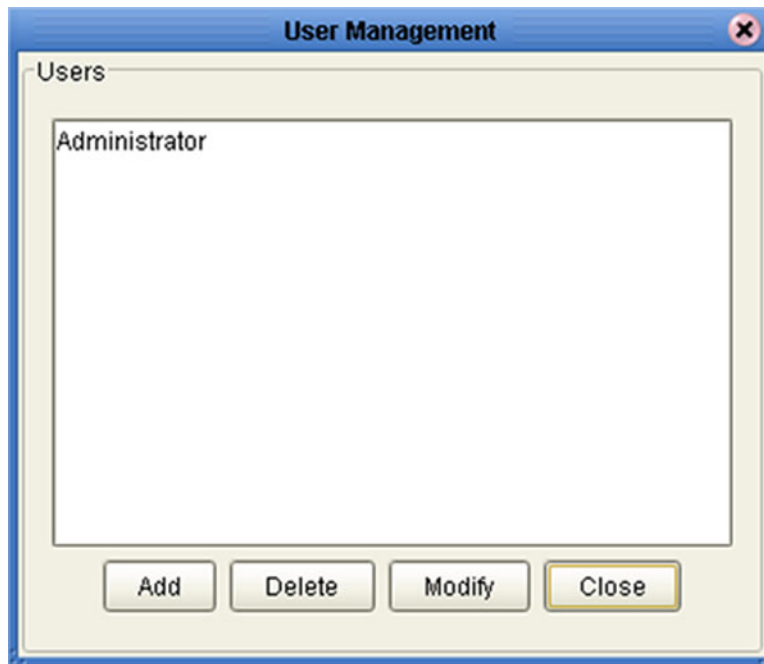
Note: A user can have more than one role, in which case his privileges will be the combined privileges from both roles.

11.2.2 User Management Interface

To open the User Management dialog box

- 1 In the Project Explorer, select the **Repository** icon.
- 2 Right-click to display the Repository context menu.
- 3 Select **User Management** to display the User Management dialog box shown in Figure 207.

Figure 207 User Management Dialog Box

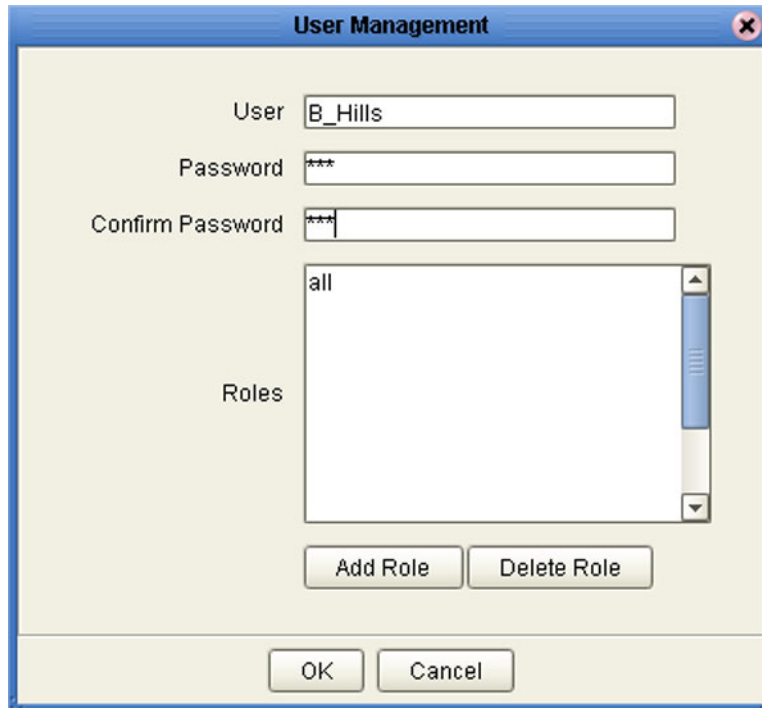


From this dialog box, you can manage user access to the Repository with options for adding, modifying, and deleting users.

To Add a New User

- 1 From the User Management dialog box, click **Add** to display a second User Management dialog box shown in Figure 208.

Figure 208 User Management - Add User



The screenshot shows a dialog box titled "User Management" with a close button in the top right corner. The dialog contains the following fields and controls:

- User:** A text input field containing "B_Hills".
- Password:** A text input field containing three asterisks "***".
- Confirm Password:** A text input field containing three asterisks "***".
- Roles:** A list box containing the role "all".
- Buttons:** "Add Role" and "Delete Role" buttons are located below the Roles list box. "OK" and "Cancel" buttons are located at the bottom of the dialog.

- 2 Enter a name for the user in the **User** box. This is the name the user will enter as their login ID during system login.
- 3 Enter a code for the user in the **Password** box. This is the code the user will enter as their password during system login.
- 4 Click the **Add Role** button to add another role for the user.

Note: Every user is automatically assigned to the **all** role, which is required to connect to the Repository; therefore, the **all** role cannot be deleted from the user.

- 5 Click **OK** to create the new user. The newly created user is listed on the User Management dialog box. This user can now access the Enterprise Designer and Repository with the assigned login ID and password.

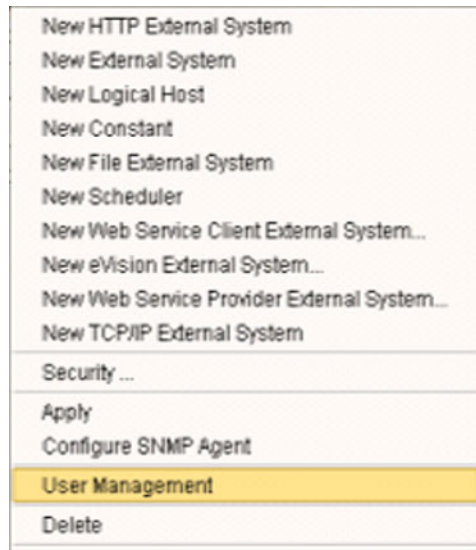
11.3 Environment User Management

This section describes the management of users who would access the applications deployed in an enterprise, using the ICAN suite.

To Open the User Management Dialog Box

- 1 In the Environment Explorer, right-click on an **Environment** to display its context menu (see Figure 209).

Figure 209 Environment Context Menu



- 2 Select **User Management** to display the series of User Management dialog boxes for the Environment, which are identical to those shown in [User Management Interface](#) on page 244. Follow the same procedure described in that section.

11.3.1 Message Server Security

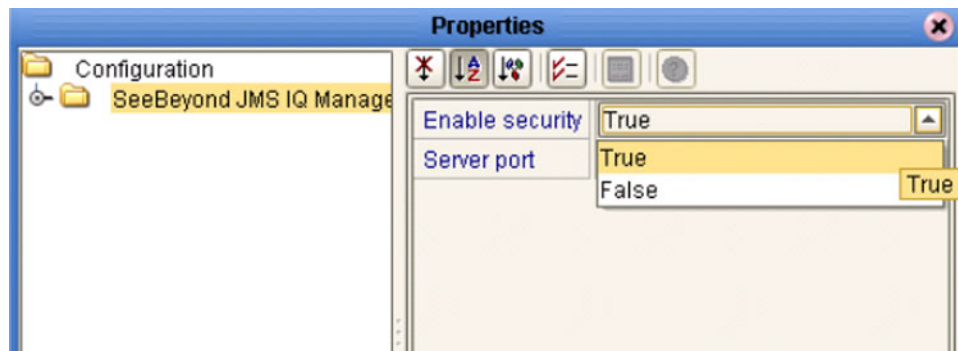
This section describes procedures for enabling security on the message server and clients.

Message Server Settings

To enable message server security

- 1 In the Environment Explorer, right-click on a message server to display its context menu.
- 2 Select **Properties** to display the message server Properties Dialog Box.
- 3 Select **JMS IQ Manager** in the left panel of the Properties Dialog Box to display its properties (see Figure 210).

Figure 210 JMS IQ Manager Properties

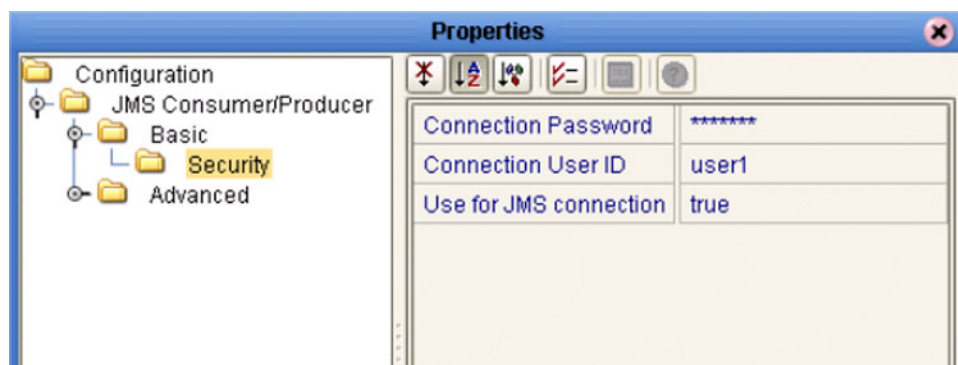


- 4 Select **True** from the drop-down menu to enable security.

JMS Client Security

To enable authentication of Collaborations in JMS Client connections, set the properties as shown in Figure 211.

Figure 211 JMS Client Security Properties



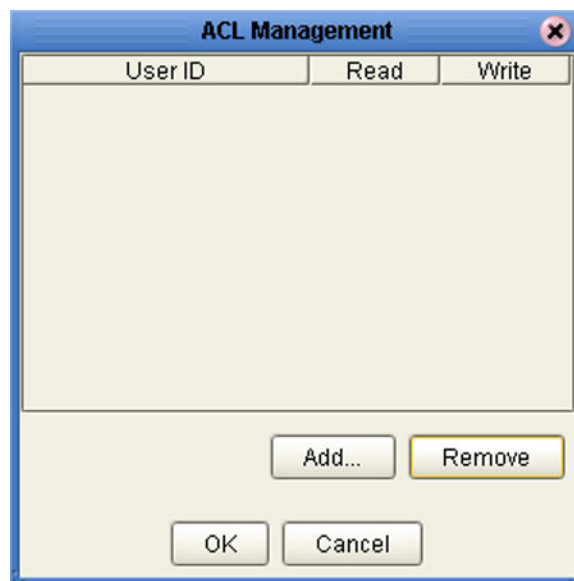
11.4 ACL Management

Access Control List (ACL) Properties allow you to assign read and write access to registered users for a selected object within a Repository. This section describes how to set permissions using the ACL Management dialog box.

To open the ACL Properties dialog box

- 1 From the Project Explorer, select an object icon.
- 2 Right-click to display the object's context menu.
- 3 Select **ACL Management** to display the dialog box shown in Figure 212.

Figure 212 ACL Management Dialog Box

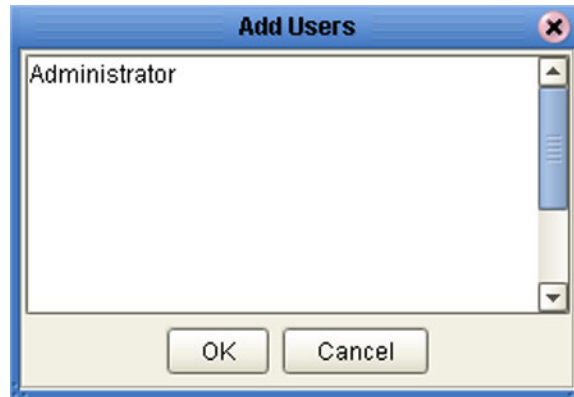


From this dialog box, you can manage user access to the object with options for adding and deleting users and granting them read and write access rights.

To add and assign access rights to a user

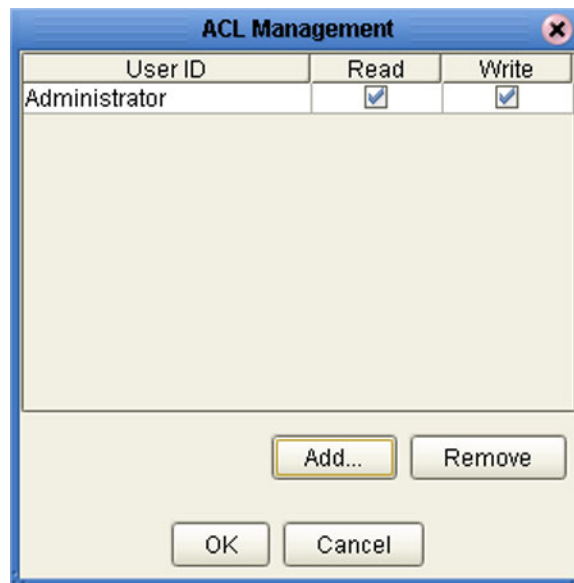
- 1 From the ACL Management dialog box, click **Add** to display the dialog box shown in Figure 213.

Figure 213 ACL Add Users Dialog Box



- 2 Enter the name of the *existing Repository user* to whom you want to grant access to the object into the text box.
- 3 Click **OK** to add the user to the ACL Management list. The user is automatically assigned read access to the object.

Figure 214 ACL Management Dialog Box - Read and Write Access



- 4 Select the **Write** check box for the user if you want them to be able to edit the Project.
- 5 Click **OK** to save your changes to the ACL Properties dialog box.

11.5 SSL/HTTPS

11.5.1 Overview

Secure Socket Layer (SSL) is a technology that allows Web browsers and Web servers to communicate over a secured connection. In this secure connection, the data that is being sent is encrypted before being sent, then decrypted upon receipt and prior to processing. Both the browser and the server encrypt all traffic before sending any data.

Another important aspect of the SSL protocol is *authentication*. During your initial attempt to communicate with a Web server over a secure connection, that server will present your Web browser with a set of credentials in the form of a server certificate. The purpose of the certificate is to verify that the site is who and what it claims to be. In some cases, the server may request a certificate that the client is also who and what it claims to be (which is known as client authentication).

Certificates and Keys

In order to implement SSL, a Web server must have an associated certificate for each external interface, or IP address, that accepts secure connections. The theory behind this design is that a server should provide some kind of reasonable assurance that its owner is who you think it is, particularly before receiving any sensitive information. It may be useful to think of a certificate as a “digital driver's license” for an Internet address. It states with which company the site is associated, along with some basic contact information about the site owner or administrator.

A certificate is a digitally-signed statement from one entity (person, company, etc.), saying that the public key (and some other information) of some other entity has a particular value. When data is digitally signed, the signature can be verified to check the data integrity and authenticity. *Integrity* means that the data has not been modified or tampered with, and *authenticity* means the data indeed comes from whoever claims to have created and signed it.

The certificate is cryptographically signed by its owner and is difficult for anyone else to forge. For sites involved in e-commerce, or any other business transaction in which authentication of identity is important, a certificate can be purchased from a well-known Certificate Authority (CA) such as **Verisign** or **Thawte**.

Certificates are used with the HTTPS protocol to authenticate Web clients. The HTTPS service of the ICAN Repository server will not run unless a server certificate has been installed. Use the procedure outlined below to set up a server certificate that can be used by the ICAN repository server to enable SSL.

The Keytool Utility

One tool that can be used to set up a server certificate is **keytool**, a key and certificate management utility that ships with the J2SE SDK. It enables users to administer their own public/private key pairs and associated certificates for use in self-authentication (where the user authenticates himself/herself to other users/services) or data integrity

and authentication services, using digital signatures. It also allows users to cache the public keys (in the form of certificates) of their communicating peers.

Keytool stores the keys and certificates in a so-called *keystore*. The default keystore implementation implements the keystore as a file. It protects private keys with a password.

The keytool utility enables you to create the certificate. The version that ships with the J2SE SDK programmatically adds a Java Cryptographic Extension provider that has implementations of RSA algorithms. This provider enables you to import RSA-signed certificates.

11.5.2 Installation and Configuration

To install and configure SSL support, you need to perform the following steps:

- 1 Generate a key pair and a self-signed signature.
- 2 Obtain a digitally-signed certificate from a Certificate Authority.
- 3 Import/install the certificate.
- 4 Configure the SERVER.XML file.
- 5 Test the new SSL connection.

The following procedures employ the **keytool** utility.

To generate a key pair and a self-signed signature

- 1 From the command prompt, enter:

```
JAVA_HOME\bin\keytool -genkey -keyalg RSA -alias ICAN -keystore  
keystore_filename
```

Where, for example:

```
keystore_filename = ICAN_HOME\repository\server\conf\ssl\mykeystore
```

- 2 Enter your keystore password: (for example, **seebeyond**)
- 3 The keytool program will ask some questions, such as the following, which you need to answer appropriately:
 - A What is your first and last name?
 - B What is the name of your organizational unit?
 - C What is the name of your organization?
 - D What is the name of your City or Locality?
 - E What is the name of your State or Province?
 - F What is the two-letter country code for this unit?
 - G Is CN=*your_first_and_last_name*, OU=*your_organizational_unit*, O=*your_organization_name*, L=*your_city*, ST=*your_state_or_province*, C=*your_two_letter_country_code* correct?
- 4 Enter key password for ICAN: (RETURN, if same as keystore password)

Note: The example used the following name for the keystore file to be generated: `ICAN_HOME\repository\server\conf\ssl\mykeystore`. You can use this name or another, as long as you use the same name throughout the configuration process.

To obtain a digitally-signed certificate

- 1 From the command prompt, enter the following to generate a Certificate Signing Request (CSR):

```
JAVA_HOME\bin\keytool -certreq -alias ICAN -keyalg RSA -file  
  csr_filename -keystore keystore_filename
```

- 2 Send the CSR for signing.
 - ♦ For example, if you are using Verisign CA, go to <http://digitalid.verisign.com/>. Verisign will send the signed certificate in E-mail.
- 3 Store the signed certificate in a file.

Note: You may skip the following step if you are using only the self-signed certificate. If you are using a self-signed certificate or a certificate signed by a CA that your browser does not recognize, a dialog will be triggered the first time you try to access the server. You can then choose to trust the certificate for this session only, or permanently.

To import the certificate

From the command prompt, enter the following to install the CA certificate:

```
JAVA_HOME\bin\keytool -import -trustcacerts -alias ICAN -file ca-  
  cert_filename -keystore keystore_filename
```

Note: You must have the required permissions to modify the `JAVA_HOME\jre\lib\security\cacerts` file.

An example Connector element for an SSL connector is included in the default `server.xml` file. This Connector element is commented out by default. To enable the SSL Connector for the ICAN repository, remove the comment tags around the SSL Connector element. Changes to the `server.xml` file are read by the ICAN repository server when it is started.

To configure SERVER.XML

- 1 Shut down the ICAN repository server if it is running.
- 2 Open the file `ICAN_HOME/repository/conf/server.xml` using a text editor.
- 3 Find the following section of code in the file (try searching for *Connector*). The following snippet can be put before the first instance of *Connector* in the `server.xml` file.

```
<Connector  
  className="org.apache.catalina.connector.http.HttpConnector"  
  port="8443"  
  minProcessors="5"  
  maxProcessors="75"  
  enableLookups="true"
```

```
    acceptCount="10"  
    debug="0"  
    scheme="https"  
    secure="true"  
  >  
    <Factory  
      className="org.apache.catalina.net.SSLServerSocketFactory"  
      clientAuth="false"  
      protocol="TLS"  
      keystoreFile="ICAN_HOME\repository\server\conf\ssl"  
      keystorePass="seebeyond"  
    />  
  </Connector>
```

- 4 Save and close the file.
- 5 Start the ICAN repository server.

Step 5: Test the new SSL connection

- 1 For testing purposes, and to verify that SSL support has been correctly installed on the ICAN repository server, load the default ICAN repository server introduction page with the following URL:

```
https://localhost:8443/
```

The *https* in this URL indicates that the browser should be using the SSL protocol. The port of 8443 is where the SSL Connector was created in the previous step.

- 2 The first time you load this application, the New Site Certificate dialog displays. Select **Next** to move through the series of New Site Certificate dialogs, and select **Finish** when you reach the last dialog.

Note: You should still have the option to use HTTP to connect to the Enterprise Designer (system administrators should **not** block the HTTP port!).

11.6 LDAP

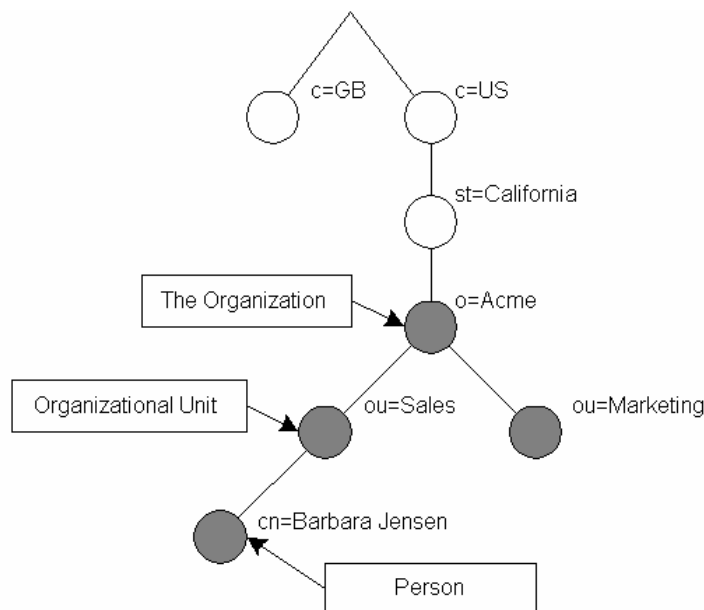
11.6.1 Overview

The Lightweight Directory Access Protocol (LDAP) is a protocol for accessing X.500-based directory services. LDAP runs over TCP/IP or other connection oriented transfer services. LDAP provides a mechanism for a client to authenticate, or prove, its identity to a directory server, paving the way for rich access control to protect the information the server contains. LDAP also supports privacy and integrity security services.

The LDAP information model is based on *entries*. An entry is a collection of attributes that has a globally-unique Distinguished Name (DN). The DN is used to refer to the entry unambiguously. Each of the entry's attributes has a *type* and one or more *values*. The types are typically mnemonic strings, like **cn** for common name, or **mail** for E-mail address. The syntax of values depend on the attribute type. For example, a **cn** attribute might contain the value Babs Jensen. A **mail** attribute might contain the value babs@example.com. A **jpegPhoto** attribute would contain a photograph in the JPEG (binary) format.

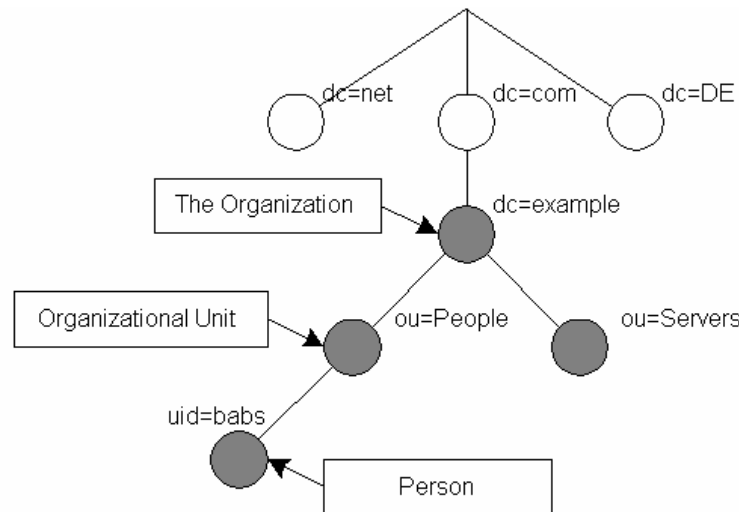
In LDAP, directory entries are arranged in a hierarchical tree-like structure. Traditionally, this structure reflected the geographic and/or organizational boundaries. Entries representing countries appear at the top of the tree. Below them are entries representing states and national organizations. Below them might be entries representing organizational units, people, printers, documents, or just about anything else you can think of. Figure 215 shows an example LDAP directory tree using traditional naming.

Figure 215 LDAP Directory Tree (Traditional Naming)



The tree may also be arranged based upon Internet domain names. This naming approach is becoming increasingly popular, since it allows for directory services to be located using the DNS. Figure 216 shows an example LDAP directory tree using domain-based naming.

Figure 216 LDAP Directory Tree (Internet Naming)



In addition, LDAP allows you to control which attributes are required and allowed in an entry through the use of a special attribute called `objectClass`. The values of the `objectClass` attribute determine the schema rules the entry must obey.

11.6.2 Referencing and Accessing Information

An entry is referenced by its distinguished name (DN), which is constructed by taking the name of the entry itself (called the Relative Distinguished Name or RDN) and concatenating the names of its ancestor entries. For example, the entry for Barbara Jensen in the Internet naming example above has an RDN of `uid=babs` and a DN of `uid=babs,ou=People,dc=example,dc=com`.

LDAP defines operations for interrogating and updating the directory. Operations are provided for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Most of the time, though, LDAP is used to search for information in the directory. The LDAP search operation allows some portion of the directory to be searched for entries that match some criteria specified by a search filter. Information can be requested from each entry that matches the criteria.

For example, you might want to search the entire directory sub-tree at and below `dc=example, dc=com` for people with the name Barbara Jensen, retrieving the E-mail address of each entry found. On the other hand, you might want to search the entries directly below the `st=California, c=US` entry for organizations with the string Acme in their name, and that have a fax number. LDAP allows you to do either, easily.

11.6.3 Implementing LDAP

The LDAP directory service is based on a client-server model. One or more LDAP servers contain the data making up the directory information tree (DIT). The client connects to servers and asks it a question. The server responds with an answer and/or with a pointer to where the client can get additional information (typically, another LDAP server). No matter which LDAP server a client connects to, it sees the same view of the directory; a name presented to one LDAP server references the same entry it would at another LDAP server. This is an important feature of a global directory service, like LDAP.

11.6.4 Configuring the ICAN Suite to Use LDAP

LDAP servers can be used for both Configuration User Management and Environment User Management.

Configuration User Management

To use an LDAP server with the ICAN system, a *Realm* element needs to be added to the ICAN repository server's `server.xml` file. The Realm element attributes are shown in Table 51.

To add the Realm element to the `server.xml` file

- 1 Open `ICAN_HOME\repository\server\conf\server.xml`.
- 2 Add the following code snippet inside the **ENGINE** tag:

```
<Realm className="org.apache.catalina.realm.JNDIRealm"
        connectionURL="ldap://LDAP_DSN:LDAP_port"
        userPattern="uid={0}, ou=users, o=company.com"
        userRoleName="memberOf"
/>
```

- 3 Save your changes and close the file.
- 4 Start your LDAP server.
- 5 Shut down and restart your ICAN repository server.

Table 51 Realm Element Attributes

Attribute	Parameter	Description/Notes
className		Always use the default className: "org.apache.catalina.realm.JNDIRealm"
connectionURL		Identifies the location of the LDAP server.
	LDAP_port	The port your LDAP server listens for requests; for example, 389.
	LDAP_DSN	The necessary DSN to connect to your LDAP server; for example, 'localhost'.
userPattern		Identifies the Distinguished Name (DN) that represents a user.

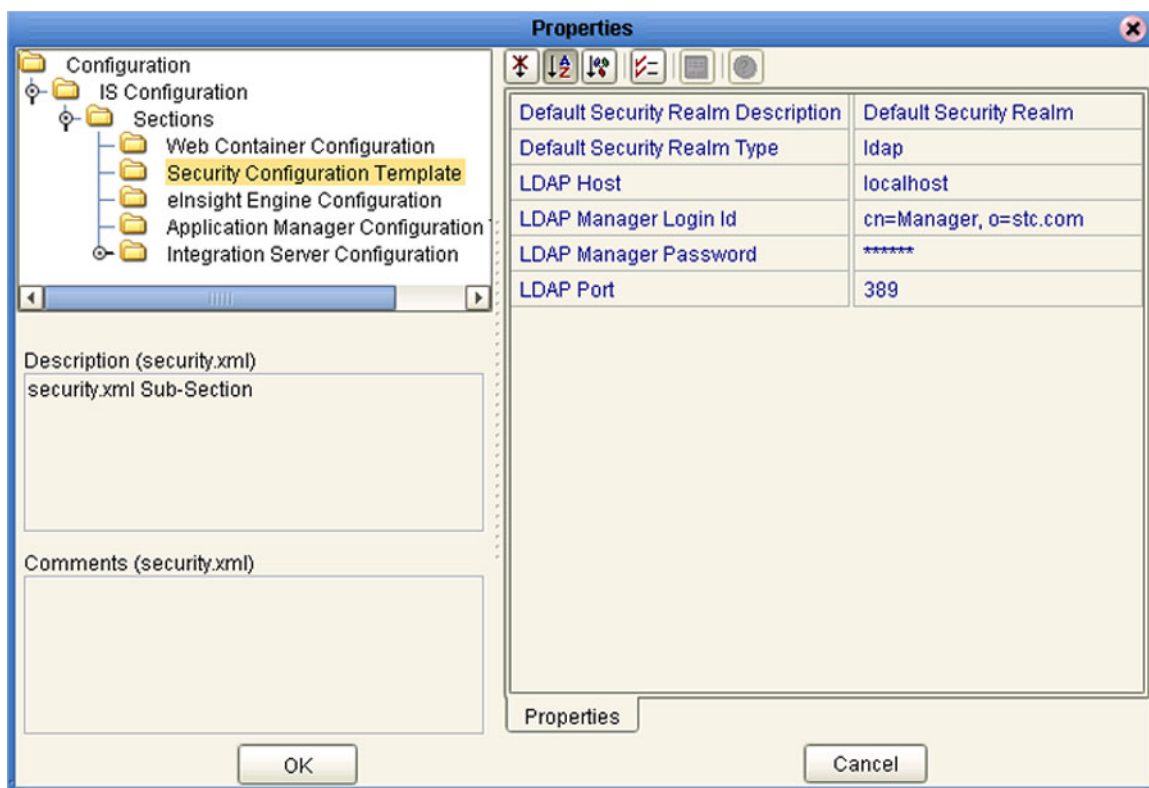
Attribute	Parameter	Description/Notes
userRoleName		The user's globally-unique Distinguished Name (DN) must have an additional attribute that identifies the role(s) to which this user belongs. For example, if the name of the attribute is <i>memberOf</i> , then the ICAN repository server will be configured as userRoleName="memberOf".

Runtime User Management

Integration servers running on the Logical Hosts can be configured to use the LDAP server for their authentication. Run-time authentication for an Integration Server is configured in the Properties Dialog Box for the Security Configuration Template (see Figure 217), where:

- **LDAP Host** is the machine on which the LDAP server is running, for example: *localhost*.
- **LDAP Manager Login Id** is the root distinguished name, for example: *cn=Manager, o=stc.com*.
- **LDAP Manager Password** is the root password, for example: *secret*.
- **LDAP Port** is the port on which the LDAP server is running, for example: 389.

Figure 217 Security Configuration Template Properties Dialog Box



Enterprise Manager

This chapter provides an introduction to the eGate Integrator Enterprise Manager.

12.1 Overview

Enterprise Manager is a specialized integration development environment (IDE) for monitoring and managing eGate Projects and installing product updates. Additional features provided by Enterprise Manager include impact analysis and version control, which are described in [Impact Analysis](#) on page 230 and [Version Control](#) on page 233, respectively.

Important: *Enterprise Manager works only with Microsoft Internet Explorer.*

Note: *Installing Enterprise Manager automatically installs JDK on the client system, whether or not another installation of JDK already exists. The JDK installed with Enterprise Designer is local to the application, and does not interfere with other JDK installations.*

Monitoring and Managing eGate

The Enterprise Manager allows you to monitor and manage eGate components in real-time.

- *Component monitoring* refers to observing the attribute value of a component at specific intervals. Whenever the value crosses a preset threshold, an alert is sent to the Monitoring Server. These alerts are accessible through the Enterprise Manager.
- *Component management* refers to controlling a component's behavior through an external agent—Enterprise Manager.

Product Installation

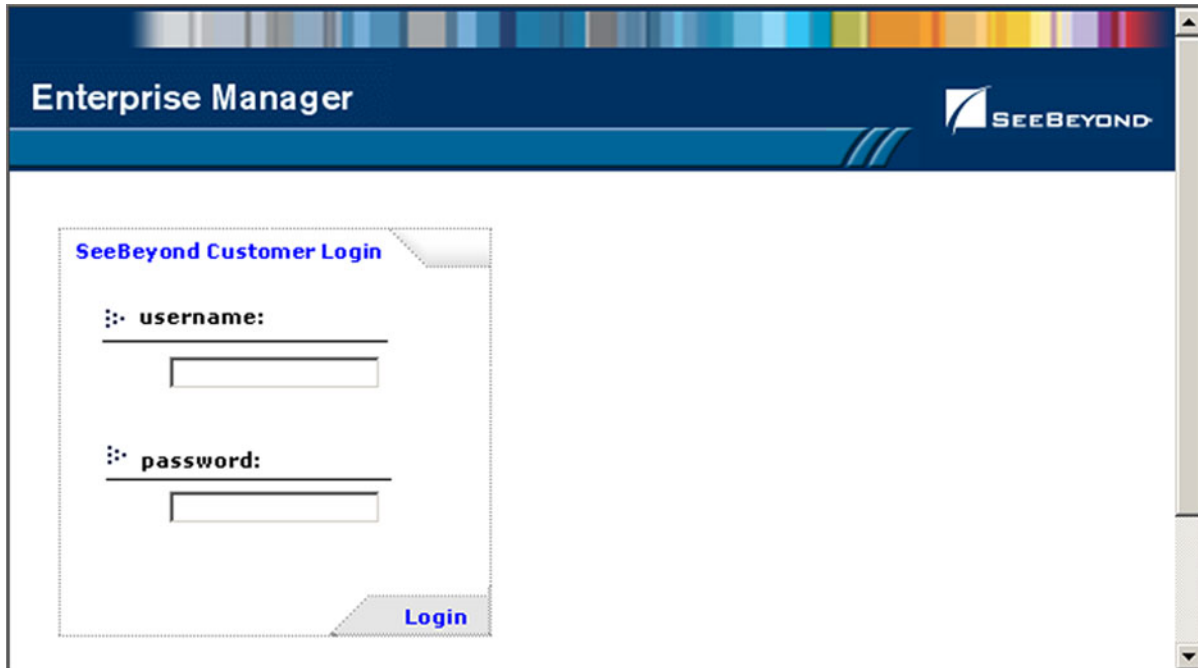
ICAN Suite products are uploaded from the installation media (CD-ROMs) to the Repository Server via the Enterprise Manager. These products are then available to be downloaded and installed from the Repository Server. Uploading and installing products is performed via the *Admin* and *Download* pages in the Enterprise Manager.

12.2 Starting Enterprise Manager

To Start the Enterprise Manager

- 1 Launch Internet Explorer.
- 2 Enter `http://hostname:portnumber` in the **Address** box to display the SeeBeyond Customer Login window shown in Figure 218.

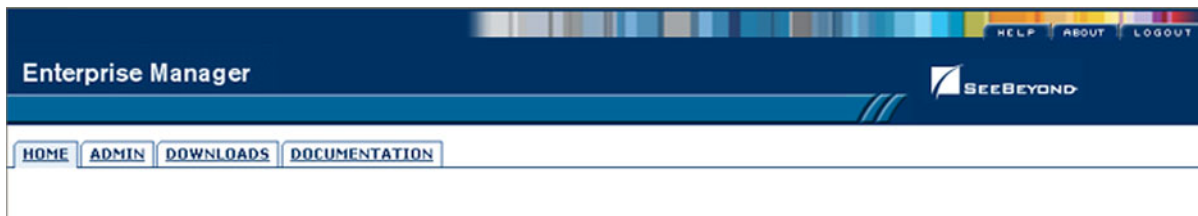
Figure 218 Enterprise Manager Login



Note: The *hostname* is the TCP/IP host name of the server where you installed the Repository. The *portnumber* is the number of the port you entered during installation of the Repository.

- 3 Enter your login ID and password in the **Username** and **Password** boxes and click **Login**.

Figure 219 Enterprise Manager GUI



The Enterprise Manager is organized into three sections represented by tabs (see Figure 219), which are described in the following sections:

- **Home** on page 261
- **Administration** on page 265
- **Downloads** on page 268
- **Documentation** on page 269

To Stop the Enterprise Manager

Click the *Logout* tab (in the upper-right corner of the window) to log out of the Enterprise Manager and return the *SeeBeyond Customer Login window*.

12.3 Home

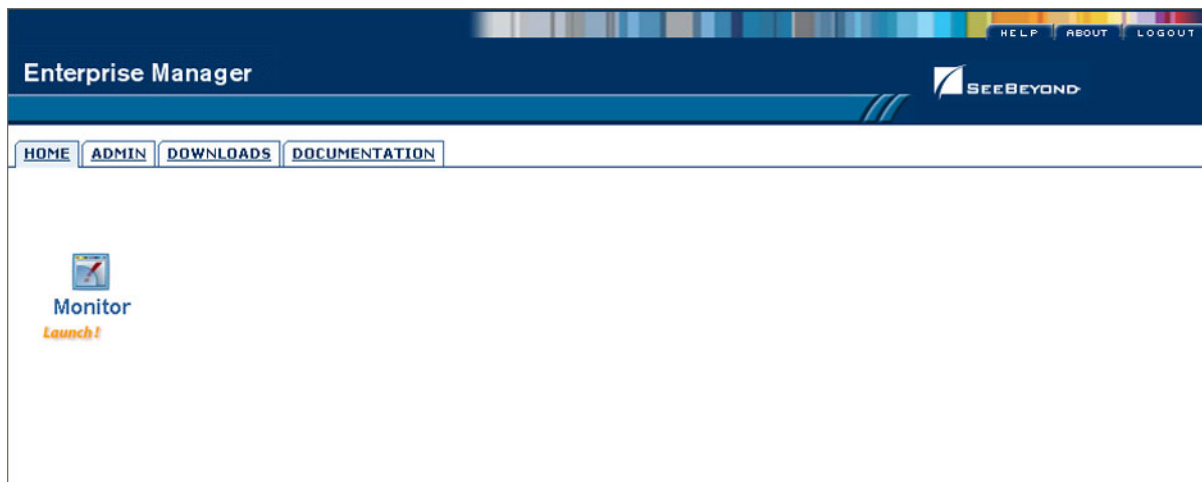
The **Home** tab contains a link to the Monitor Server. This server controls and monitors all the eGate Projects in the Enterprise Designer. Statistics on the status, health, and performance of Project components are available through the Monitor Server.

12.3.1 Monitor Server

To Run the Monitor Server

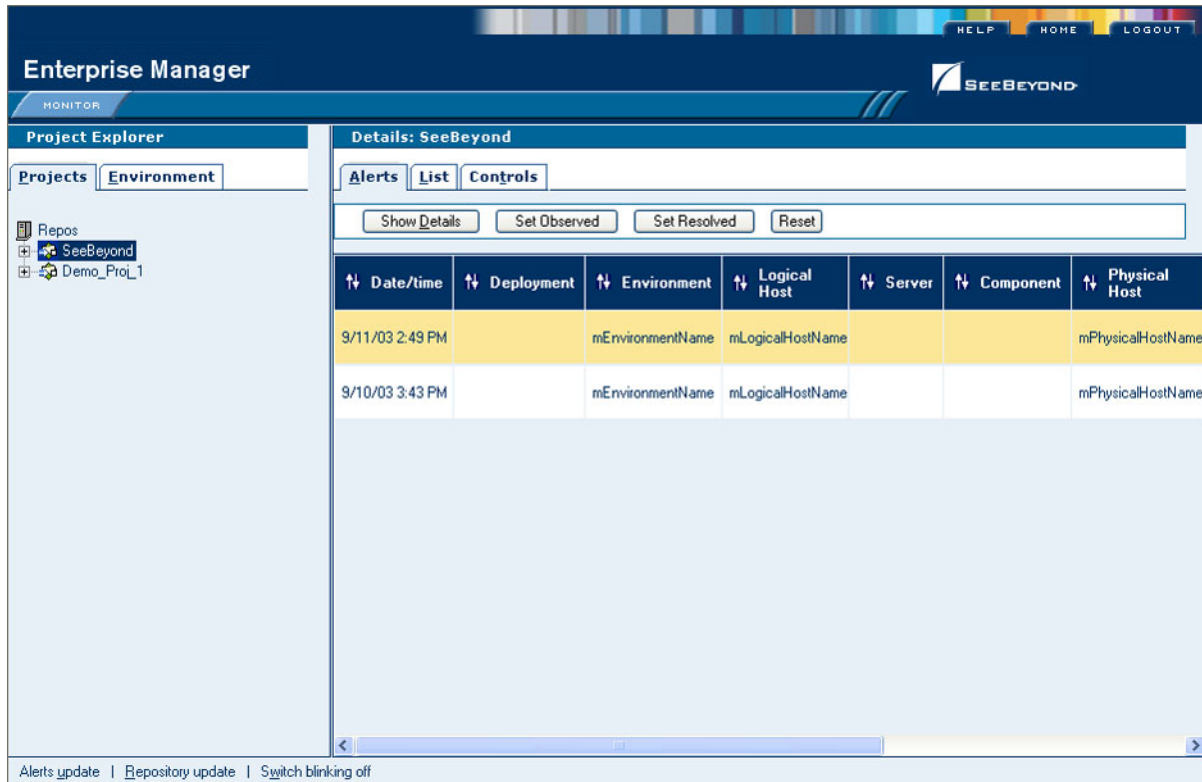
- 1 From the Enterprise Manager GUI, click the **Home** tab to display the Monitor Launch page shown in Figure 220.

Figure 220 Monitor Launch Window



- 2 Click the **Monitor** icon to launch the Monitor page. As with the Enterprise Designer, the Environment Manager contains a Project Explorer and an Environment Explorer.
- 3 Click the **Projects** tab (if not already selected).
- 4 Click any Project icon to display the **Details** area, as shown in Figure 221.

Figure 221 Monitor Page - Projects



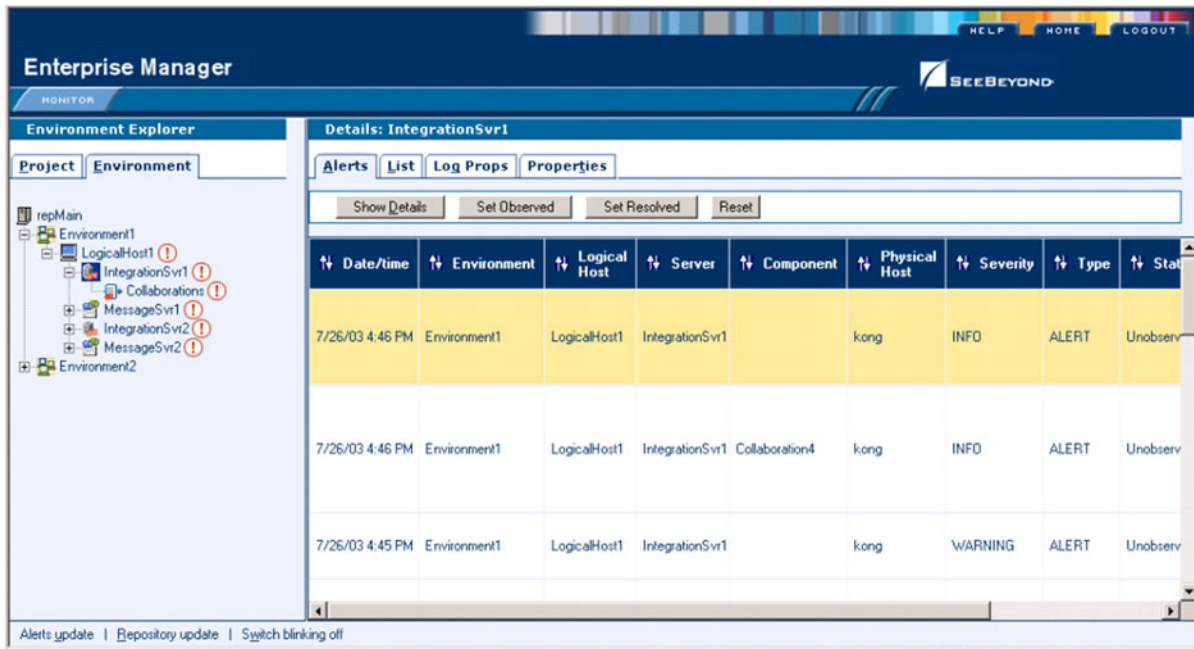
Like the Enterprise Manager, the **Details** area is organized into three sections represented by tabs. Each of these tabs is briefly described below:

- **Alerts**
Displays functionality-related information about the selected Project component.
- **List**
Displays Project component information as a list.
- **Logging**
Displays ...
- **Controls**
Displays ...

Note: Click the **Home** tab (at the top of the window) to exit from the Monitor window and return to the **Admin** tab on the Enterprise Manager.

Switching to the Environment Explorer and selecting the Alerts tab in the Details area displays a list of alerts, which can be sorted by different criteria or marked as observed or resolved by means of the command buttons.

Figure 222 Monitor Page - Environment - Alerts



Selecting an Integration Server and clicking *Collaborations* in the Environment Explorer, then selecting the *List* tab in the Details panel, displays all Collaborations on that Host.

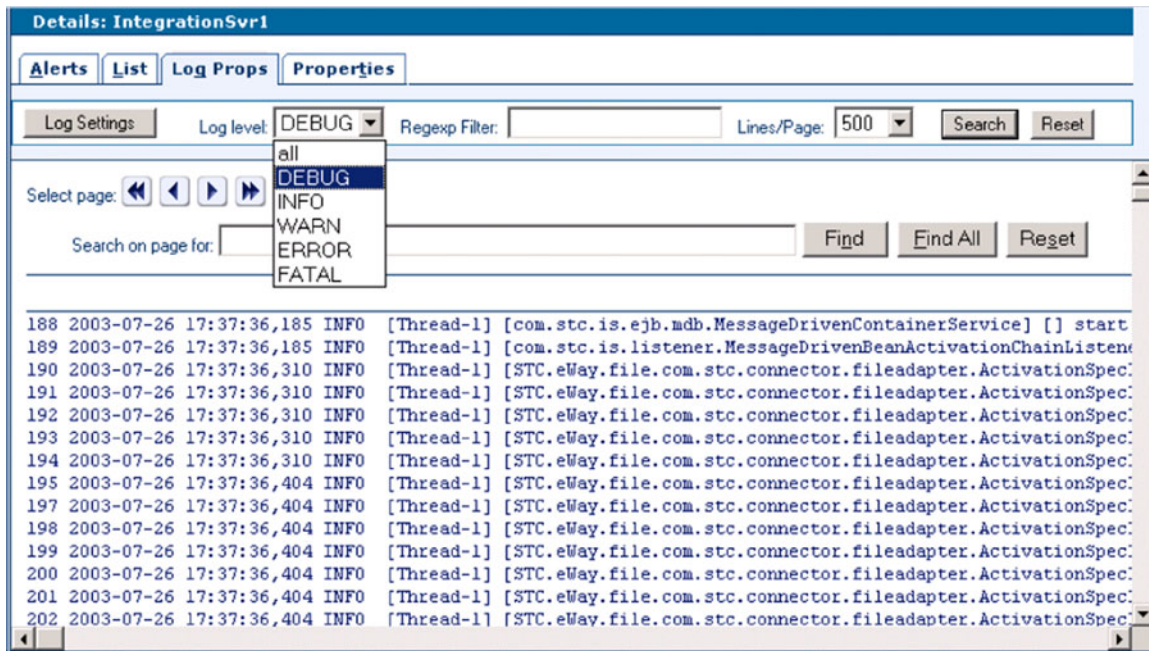
Figure 223 Monitor Page - Environment - List



The Collaborations can be started or stopped from here, and the number of messages waiting or processed can be displayed.

Selecting a Logical Host and clicking the *Log Props* tab in the Details panel displays the log files on the server, which can be sorted and filtered, as well as viewed.

Figure 224 Monitor Page - Environment - Log Props



Additional information on monitoring Projects and viewing logs using the Enterprise Manager Monitor can be found in [Managing Projects](#) on page 270.

12.4 Administration

The **Admin** tab contains tools for accessing the latest eGate 5.0 manifest file and corresponding program files from the SeeBeyond Web site.

To Access the Upload System Component Manifest


- 1 From the Enterprise Manager GUI, click the **Admin** tab. The first time you do this, during installation, the page for selecting the License file is displayed (see Figure 225). **Browse** for the file, then **Submit** for upload.

Figure 225 Upload Product License



- 2 Once you have uploaded the License file, the Enterprise Manager allows you to select the eGate archive file, as shown in Figure 226.

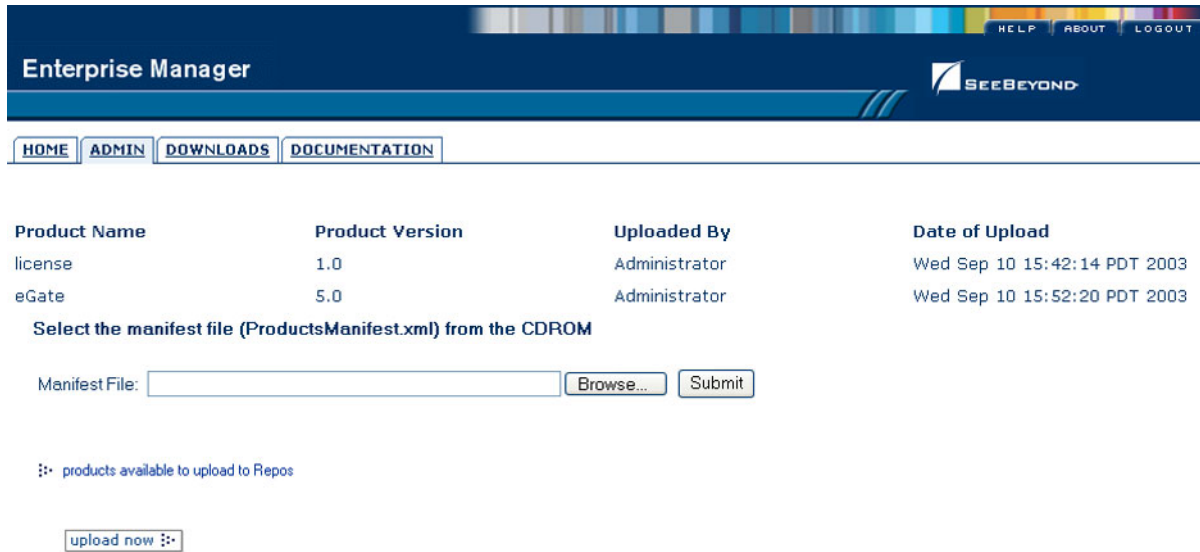
Figure 226 Upload eGate SAR File



Product Name	Product Version	Uploaded By	Date of Upload
license	1.0	Administrator	Wed Sep 10 15:42:14 PDT 2003

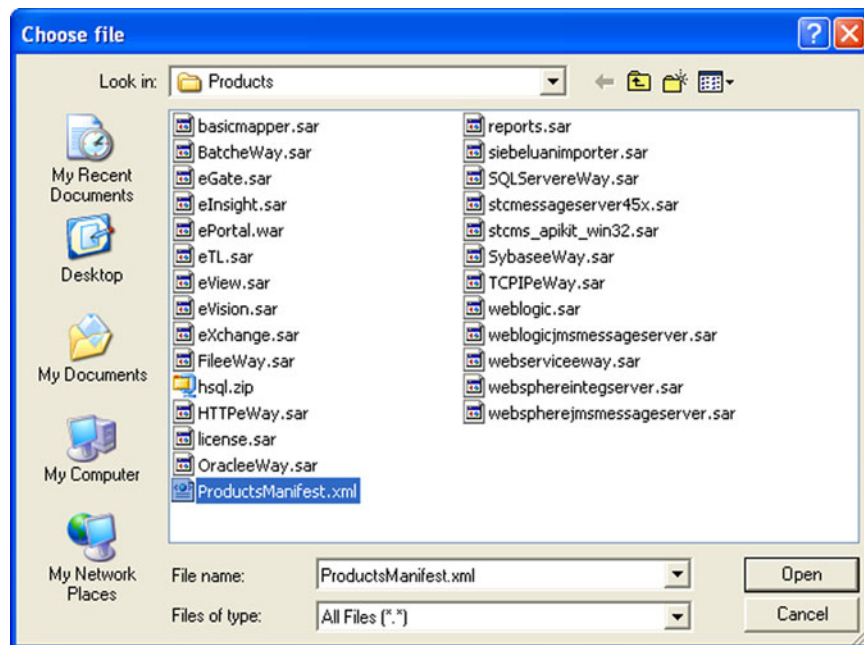
- 3 Once you have uploaded the eGate SAR file, the Enterprise Manager allows you to select the Product Manifest file containing the products included in your license, as shown in Figure 227. If your license and eGate files are already uploaded, this will be the first page you will see.

Figure 227 Upload Product Manifest



- 4 Click **Browse** to display the **Choose File** dialog box shown in Figure 228.

Figure 228 Choose File Dialog Box



- 5 Locate and **Open** the **ProductsManifest.xml** file.

- 6 Click **Submit** to select the manifest file and display the Products Available to Upload list shown in Figure 229.

Figure 229 Products Available to Upload Page

Select the manifest file (ProductsManifest.xml) from the CDROM

Manifest File:

products available to upload to Repos

SeeBeyond Product Suite

Products	Version	
eGate	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eVision	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eInsight	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eView	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eIndex	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eTL	5.0	<input type="text"/> <input type="button" value="Browse..."/>
eXchange	5.0	<input type="text"/> <input type="button" value="Browse..."/>
license	5.0	<input type="text"/> <input type="button" value="Browse..."/>

- 7 Select the products you want to install and click **Upload Now**.

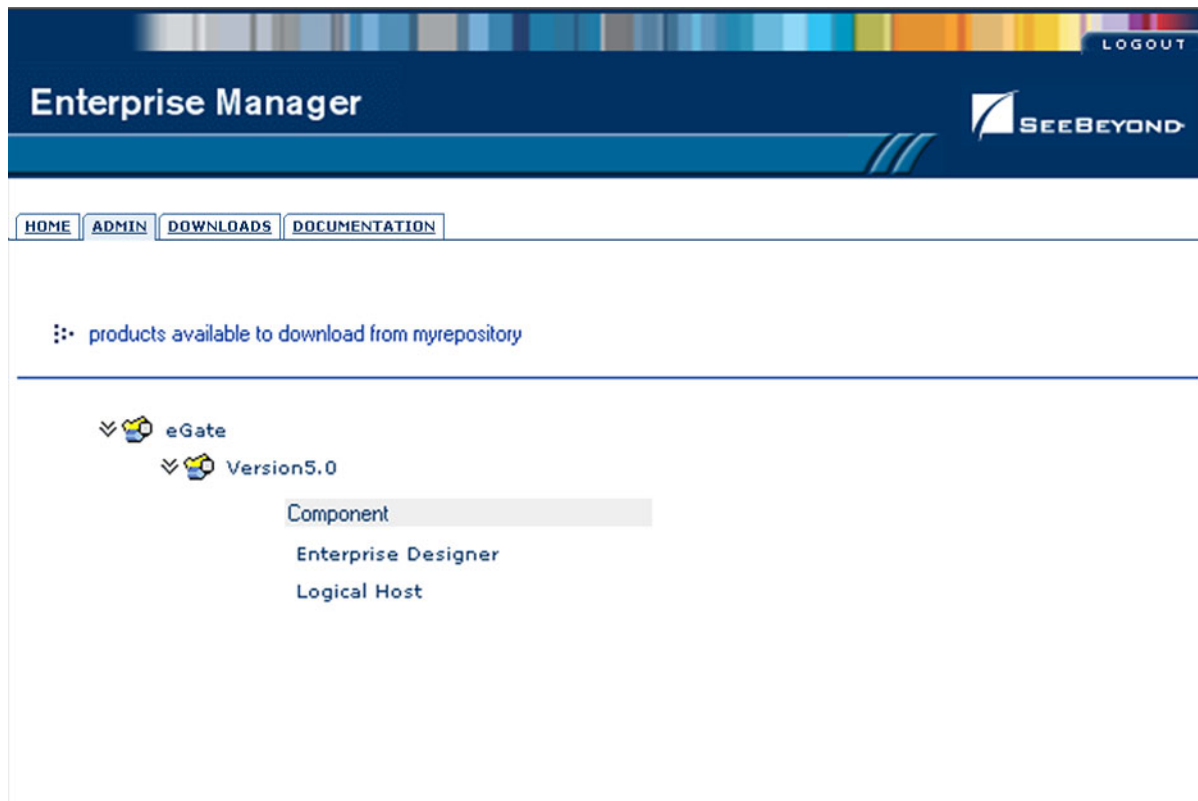
12.5 Downloads

The **Downloads** tab contains links to the latest versions of the Enterprise Designer and Logical Host.

To View the Products Available to Download

- 1 From the Enterprise Manager GUI, click the **Downloads** tab to display the Products Available to Download window shown in Figure 230.

Figure 230 Products Available to Download Page



The Products Available to Download window includes the following products:

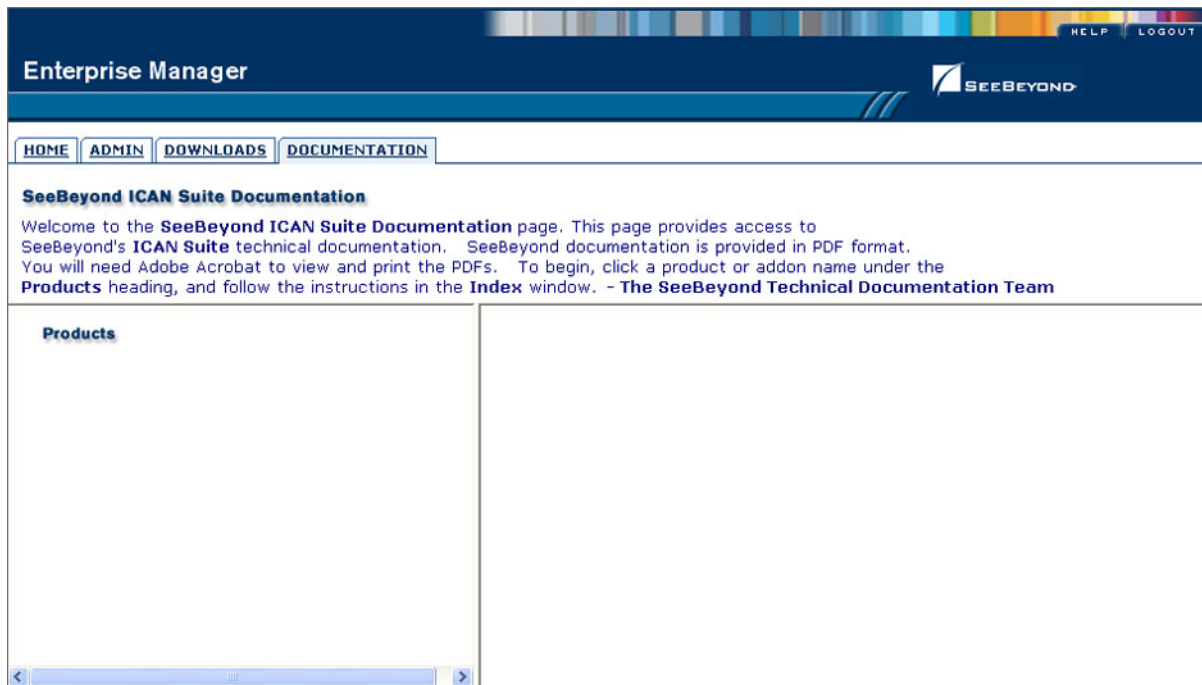
- **Enterprise Designer**
Displays the **File Download** dialog box for the **edesigner.zip** file, which contains the Enterprise Designer program files.
- **Logical Host**
Displays the **File Download** dialog box for the **logicalhost.zip** file, which contains the Logical Host program files.

For comprehensive information on product installation, see the *eGate Integrator Installation Guide*.

12.6 Documentation

The **Documentation** tab contains links to the latest versions of the SeeBeyond ICAN documentation in PDF format. You need to have Adobe Acrobat Reader installed to view or print these documents. Figure 231 shows the initial Documentation page.

Figure 231 Documentation Tab



Note: You must download the documentation SAR files from the installation disk before you can access any documents from this page.

Managing Projects

This chapter describes how to use the Enterprise Manager's facilities for monitoring Projects, and viewing and editing logs and Alerts.

13.1 Overview

Enterprise Manager contains extensive facilities for monitoring and managing your Projects, including the viewing and editing of logs. This chapter includes the following topics:

- **Accessing the Monitor Server** on page 271 shows the initial Enterprise Monitor page, with the Launch Monitor icon.
- **Monitoring Collaborations** on page 272 describes the various ways you can monitor the Collaborations and control various components in your Project.
- **Viewing Alerts** on page 278 describes how to view and set the status of Alerts.
- **Viewing Logs** on page 280 briefly describes how to view, sort, search, and filter messages in the log files.
- **Setting Log Levels** on page 282 describes how to set logging levels in Enterprise Manager.

13.2 Accessing the Monitor Server

The Enterprise Manager's **Home** tab contains a link to the Monitor Server. Click the **Monitor** icon to launch the Monitor page.

Figure 232 Monitor Launch Window



The Monitor Server interface offers the following viewing controls:

- ALT and drag the cursor to scroll.
- CTRL and click to zoom out.
- CTRL-SHIFT and click to zoom in.

13.3 Monitoring Collaborations

Enterprise Manager has extensive monitoring capabilities for Projects, one of the most useful being for Collaborations. To demonstrate these capabilities, we shall take an example Project whose Connectivity Map is shown in Figure 233.

Figure 233 Example Project Connectivity Map



When you launch the Monitor Server, the Environment Explorer is displayed by default. If you select an Integration Server and click **Collaborations**, the Details: List tab will display all Collaborations on that Logical Host (see Figure 234).

Figure 234 Monitor - Details - List Collaborations

The screenshot shows the Enterprise Manager interface. On the left is the Environment Explorer with a tree view showing Repository, Environment1, LogicalHost1, IntegrationSvr1, Collaborations, MessageSvr1, Topics, and Queues. The main pane displays the 'Details: Repository|Environment1|LogicalHost1|IntegrationSvr1' view with a 'List' tab selected. Below the tabs is a table with the following data:

Name	Project	Deployment	Status
Collaboration1	Test1	Test1DP	Running
Collaboration2	Test1	Test1DP	Running
Collaboration3	Test1	Test1DP	Running

Note: When a Project is first started, or freshly activated after de-activation, the status will appear as **Unknown** until the first message is sent. The status will then change to **Running**.

By selecting a Collaboration in the list, you can start or stop the Collaboration by means of the buttons above the tab (see Figure 235) or by right-clicking on the component in Environment Explorer and using the resulting menu (see Figure 236). By default, a Collaboration summary is displayed in the Collaboration detail panel below the list.

Figure 235 Starting and Stopping Collaborations - Details Panel

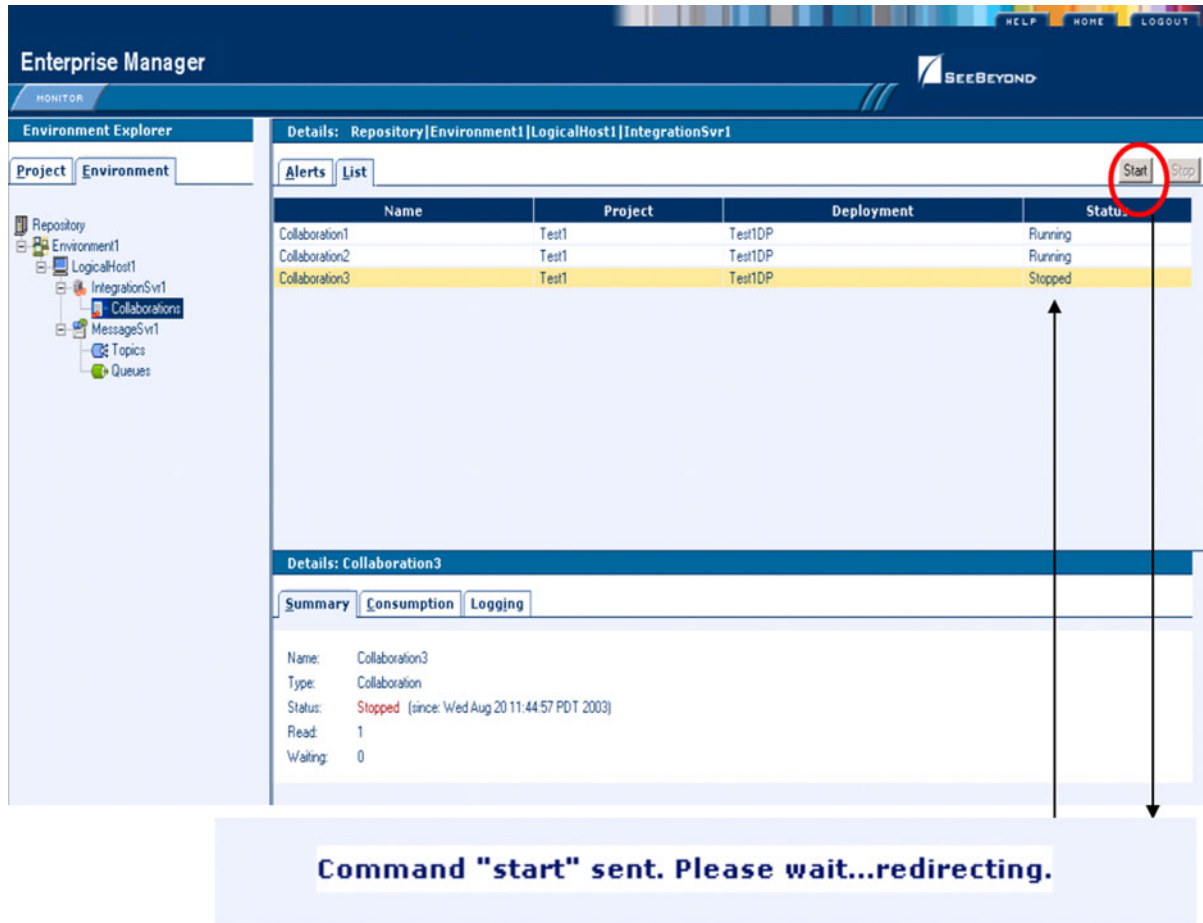
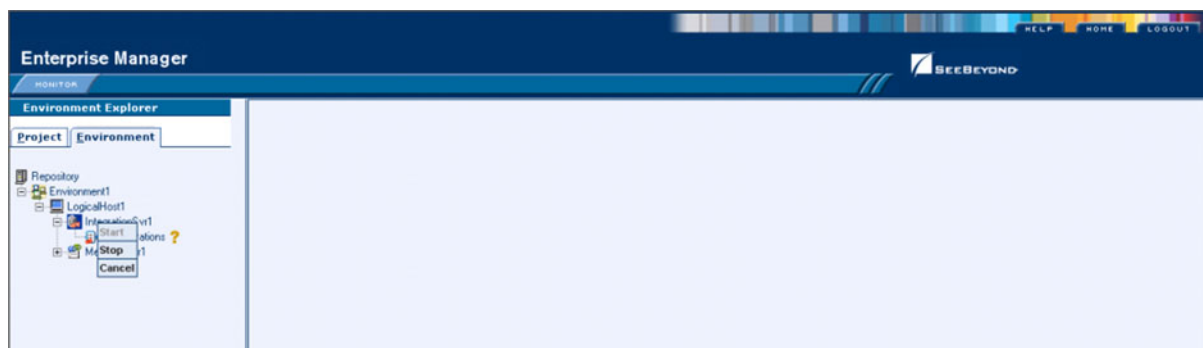
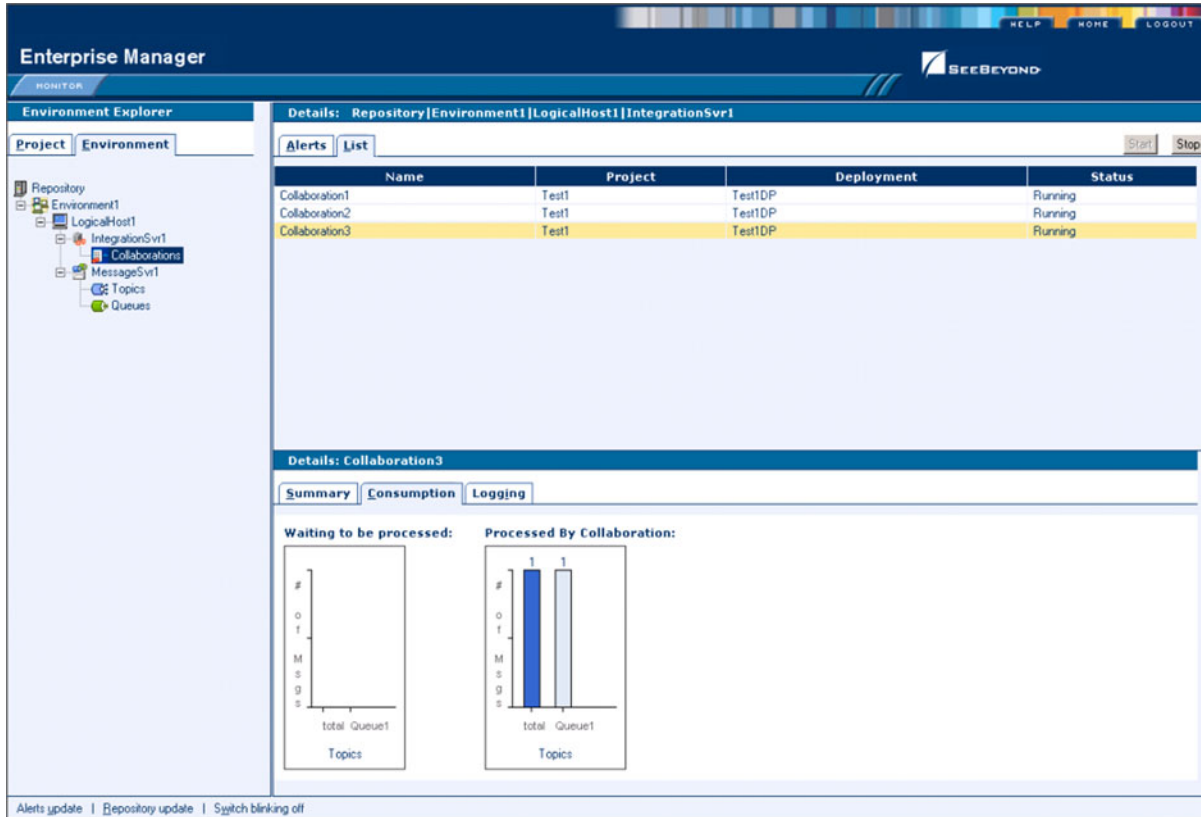


Figure 236 Starting and Stopping Collaborations - Explorer



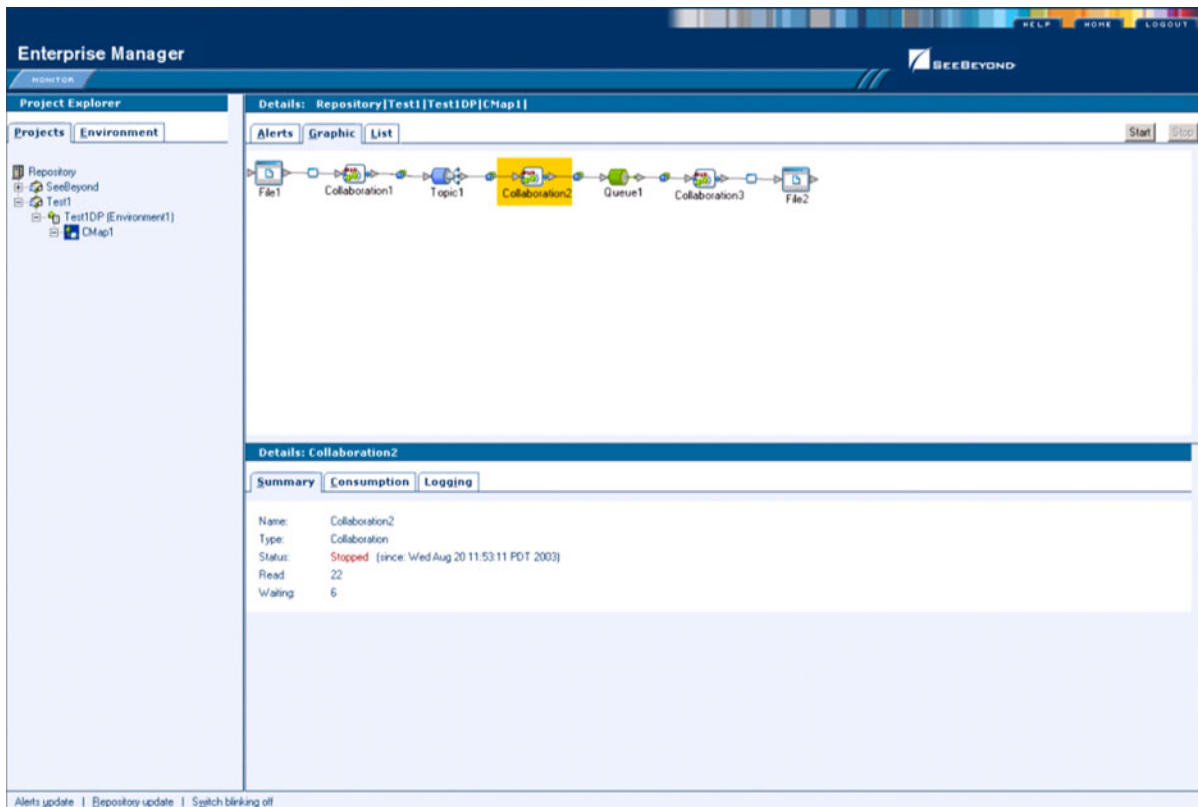
If you select the **Consumption** tab, you will see the number of pending and processed messages in graphical form (see Figure 237).

Figure 237 Pending and Processed Messages



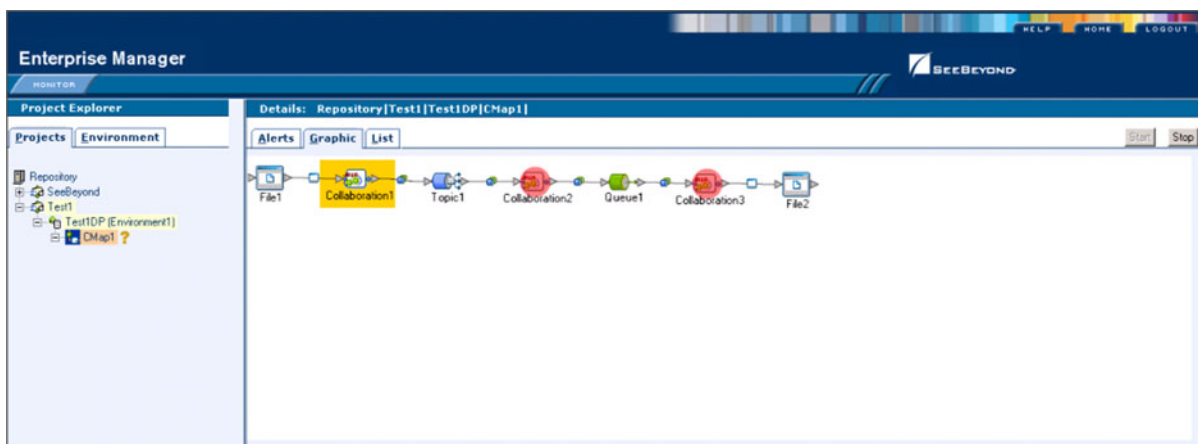
An alternative way to monitor Collaborations is to use the Projects Explorer, which displays all existing Deployment Profiles and Connectivity Maps. You will see the same Connectivity Map as in the Enterprise Designer, with the currently selected Collaboration highlighted in yellow (see Figure 238).

Figure 238 Monitor - Project Explorer



Additionally, non-active Collaborations are highlighted in red, making them immediately visible (see Figure 239).

Figure 239 Active/Non-active Collaborations



As with the monitoring option using the Environment Explorer (shown in [Figure 237 on page 274](#)), selecting a specific Collaboration in the Connectivity Map will display details in the lower panel (see [Figure 240](#) and [Figure 241](#)).

Figure 240 Pending and Processed Messages

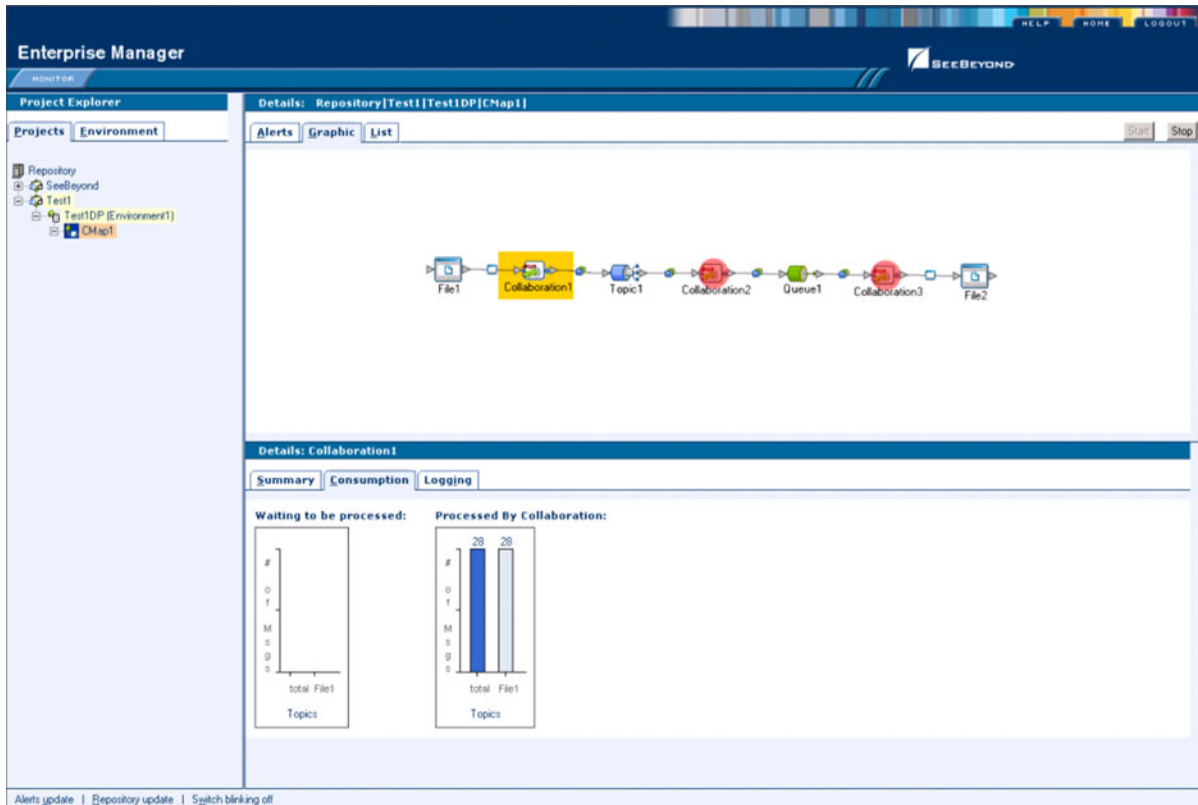
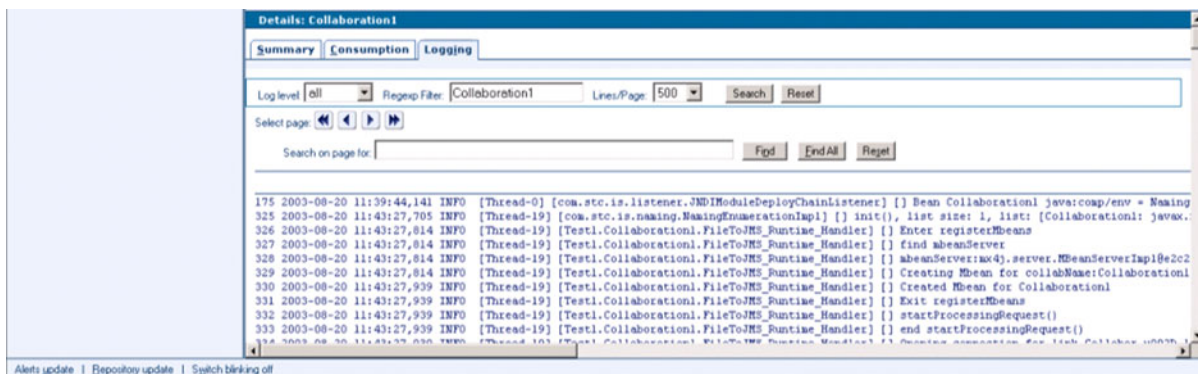


Figure 241 Logging Details



If you select a Connectivity Map in Project Explorer, you can see the status of all components of that Connectivity Map (see Figure 242).

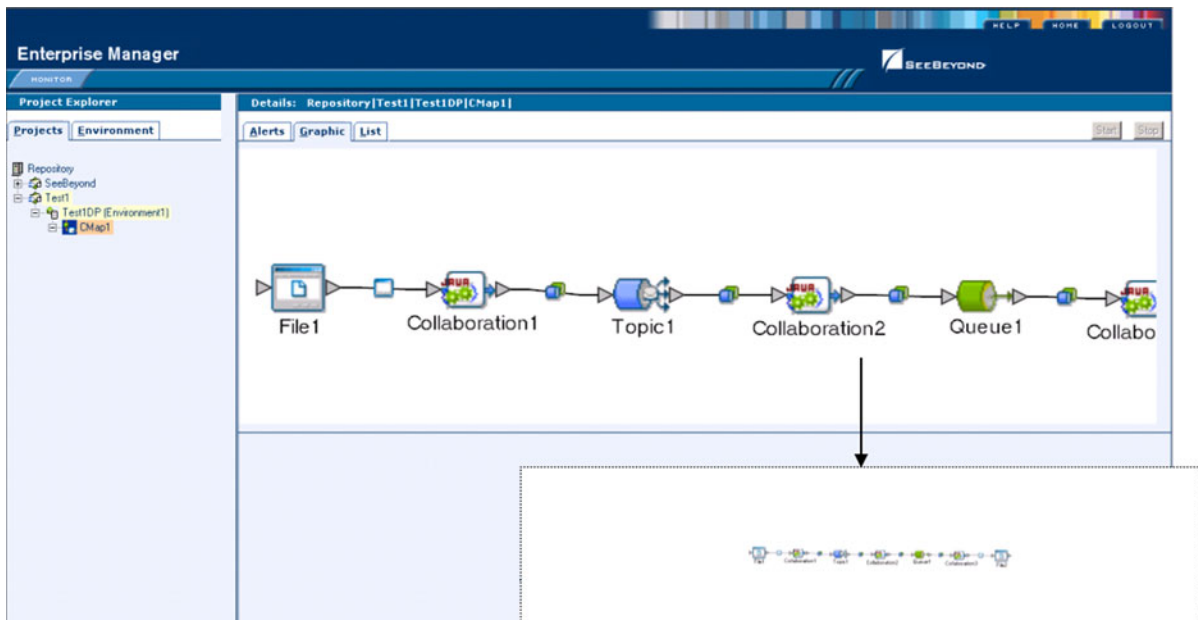
Figure 242 Connectivity Map Details: Components



You can scroll and zoom in or out on a Connectivity Map display using the following commands (see Figure 243):

- ALT and drag the cursor to scroll.
- CTRL and click on the Connectivity Map to zoom out.
- CTRL-SHIFT and click on the Connectivity Map to zoom in.

Figure 243 Connectivity Map View - Zoom In/Out



13.4 Viewing Alerts

Selecting a Collaboration and clicking the **Alerts** tab in the upper Details panel displays a list of all alerts for the selected Collaboration (see Figure 244). These can be sorted by different criteria, or marked as observed or resolved.

Figure 244 Collaboration Alerts

Date/Time	Environment	Logical Host	Server	Component	Physical Host	Severity	Type	Status	State	Description
8/20/03 11:45 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.
8/20/03 11:44 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Unobserved	Stopped	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is stopped.
8/20/03 11:43 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.
8/20/03 11:43 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration2	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration2 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.

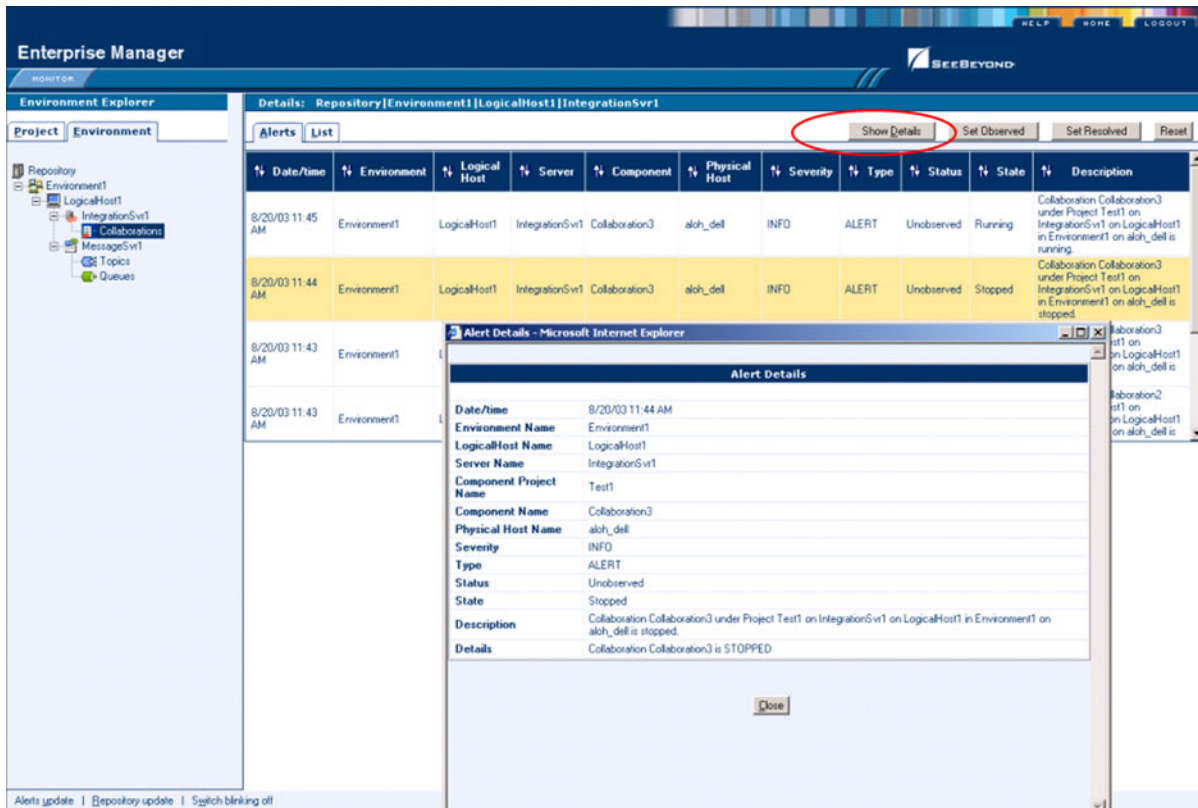
By clicking the **Set Resolved** button, you change the status of the selected Alert to *Resolved* (see Figure 245).

Figure 245 Alert Status

Date/Time	Environment	Logical Host	Server	Component	Physical Host	Severity	Type	Status	State	Description
8/20/03 11:45 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.
8/20/03 11:44 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Resolved	Stopped	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is stopped.
8/20/03 11:43 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration3	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration3 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.
8/20/03 11:43 AM	Environment1	LogicalHost1	IntegrationSrv1	Collaboration2	aloh_dell	INFO	ALERT	Unobserved	Running	Collaboration Collaboration2 under Project Test1 on IntegrationSrv1 on LogicalHost1 in Environment1 on aloh_dell is running.

By clicking the **Show Details** button, you can see all details for the selected Alert (see Figure 246).

Figure 246 Alert Details



13.5 Viewing Logs

13.5.1 Integration Server Level

In the Environment view, select an Integration Server (or Logical Host) and click the **Logging** tab in the Details panel to display all log messages for the selected component (see Figure 247). You can also filter the list for a specific log level (see Figure 248).

Figure 247 Integration Server Log Messages

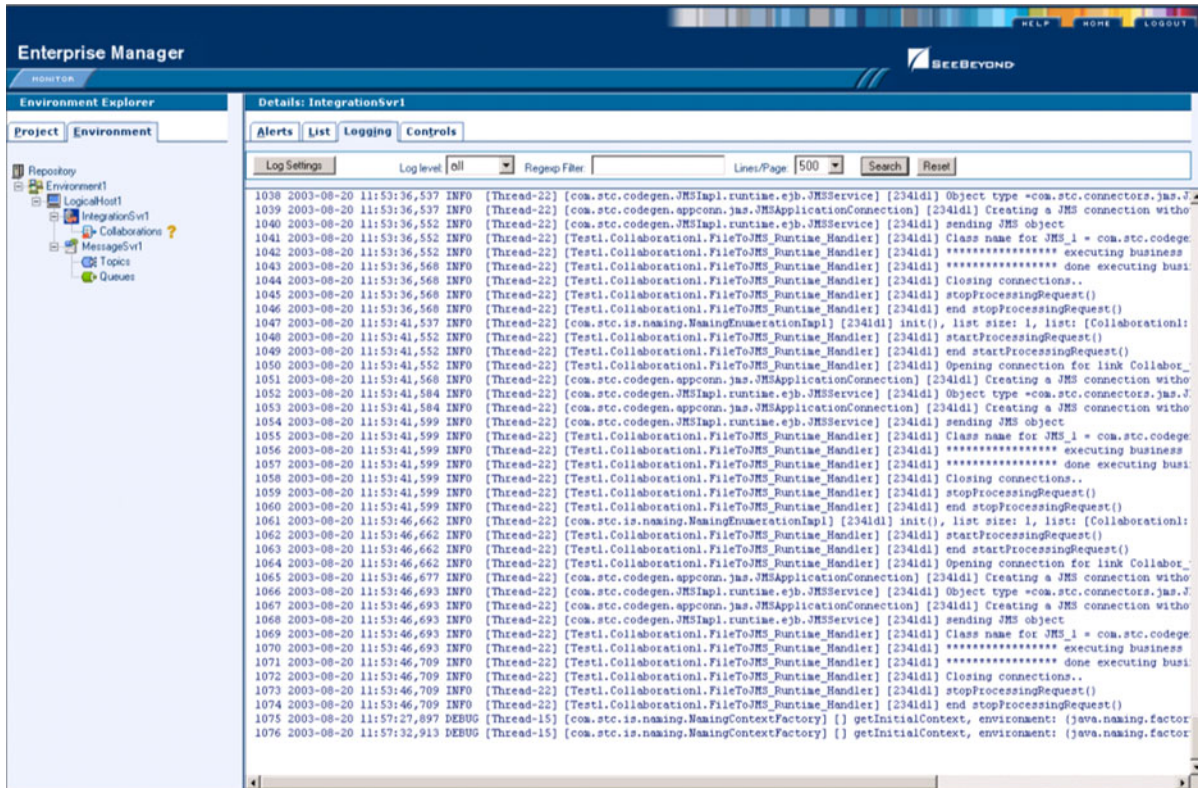
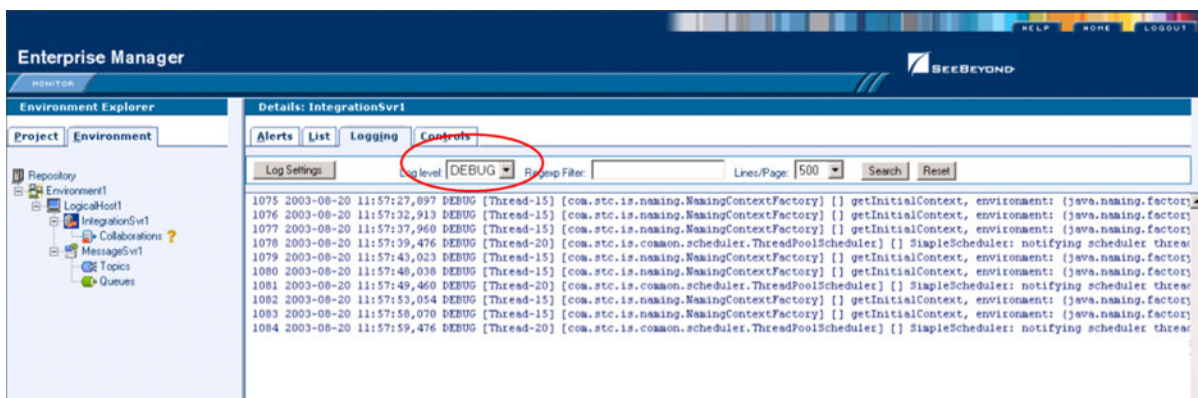


Figure 248 Integration Server Log Messages - Filtered



13.5.2 Collaboration Level

Select a Collaboration, click the **Log** tab in the upper Details panel, and click the **Logging** tab in the lower Details panel to display all log messages for the Collaboration (see Figure 249). This tab contains controls for viewing, sorting, and filtering of the log files on the server. By entering a keyword, you can search for a particular word in the log file (see Figure 250).

Figure 249 Collaboration Log File

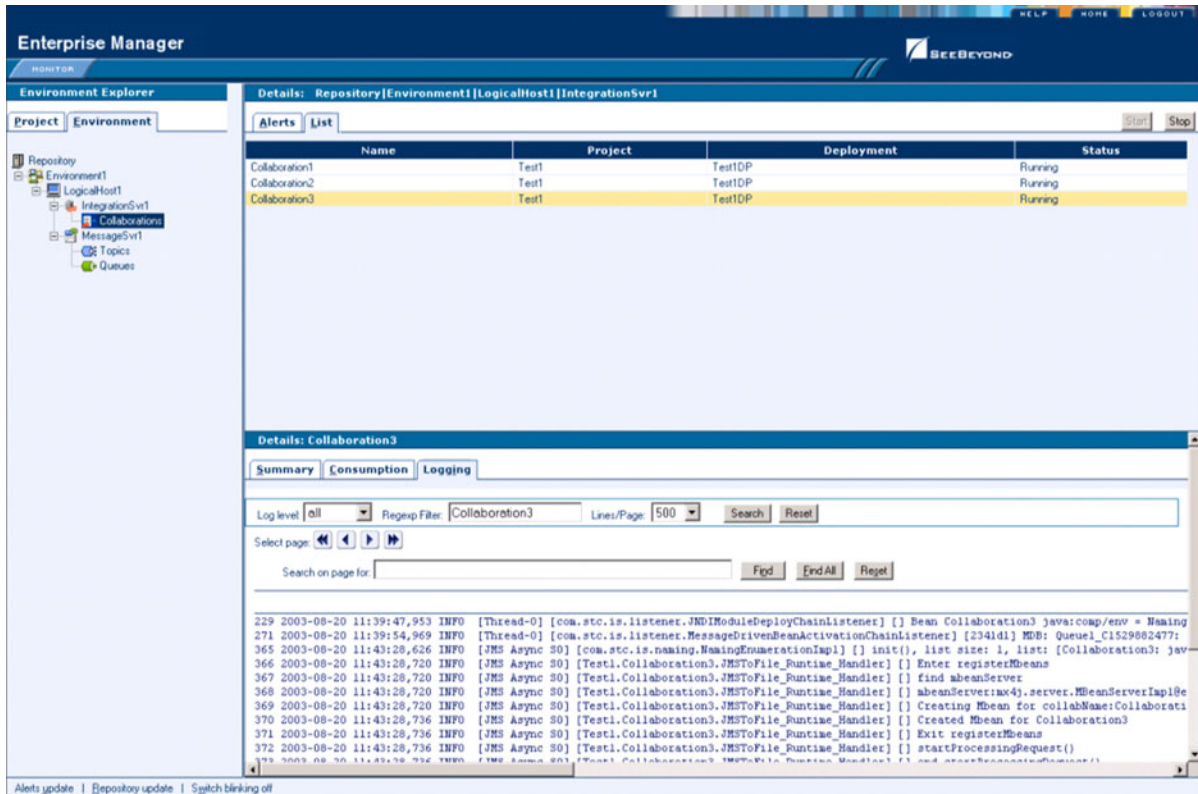
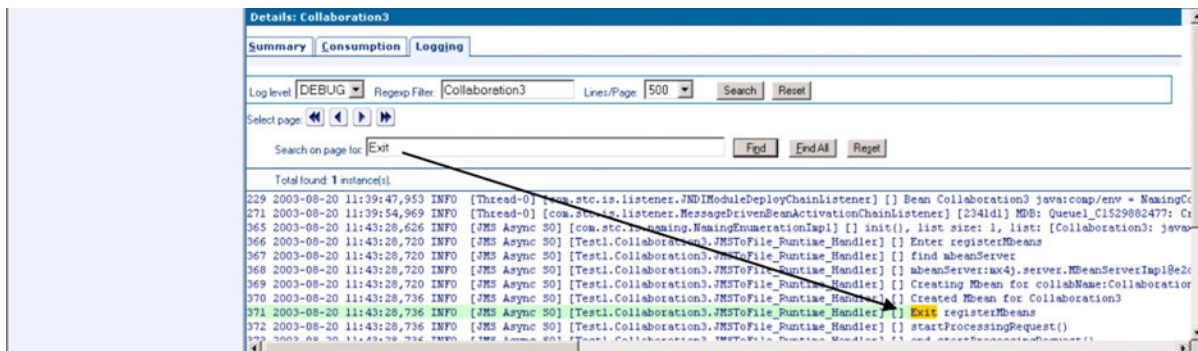


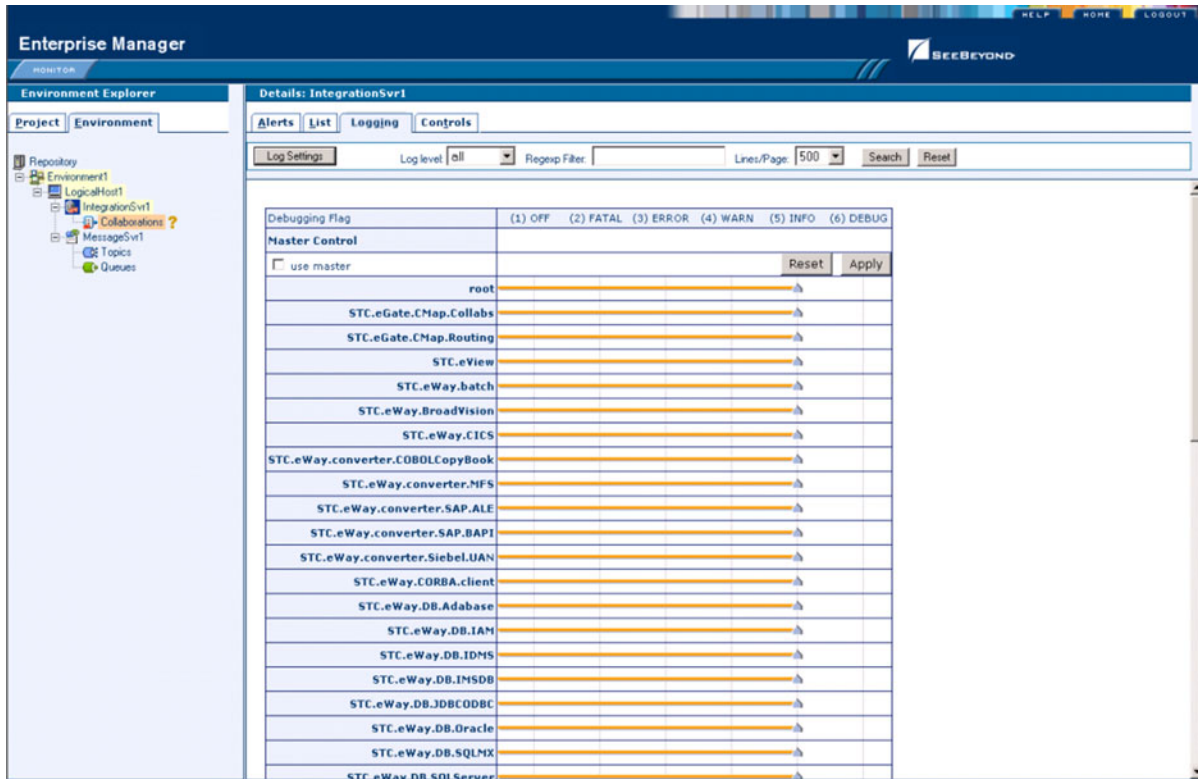
Figure 250 Search on Keyword



13.6 Setting Log Levels

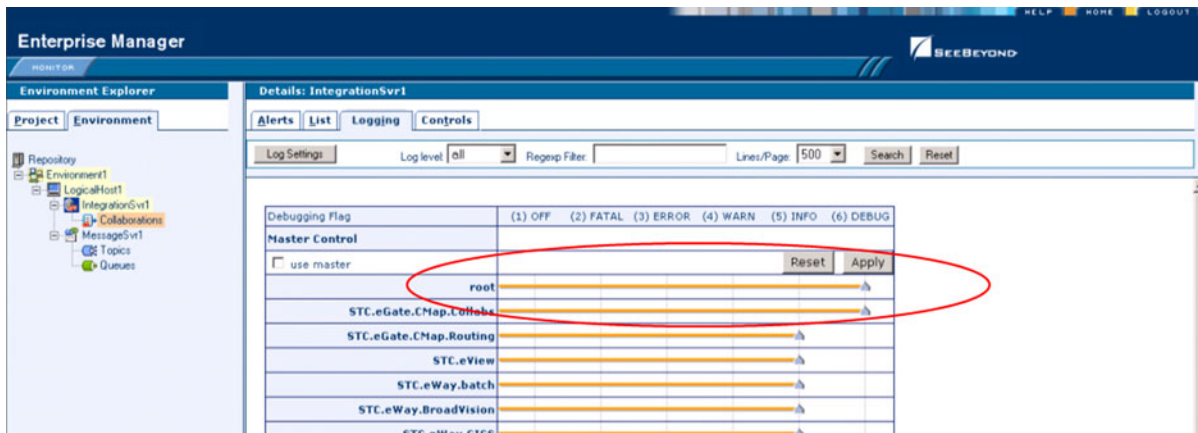
Select an Integration Server or Logical Host and click the **Log Settings** button to display the Logging Properties page (see Figure 251).

Figure 251 Logging Properties Page



To set a log level for a specific property, drag the line to the desired log level and click the **Apply** button. Figure 252 shows the first two properties reset to the **debug** level.

Figure 252 Resetting Log Levels



13.7 Indoubt Transaction Editing

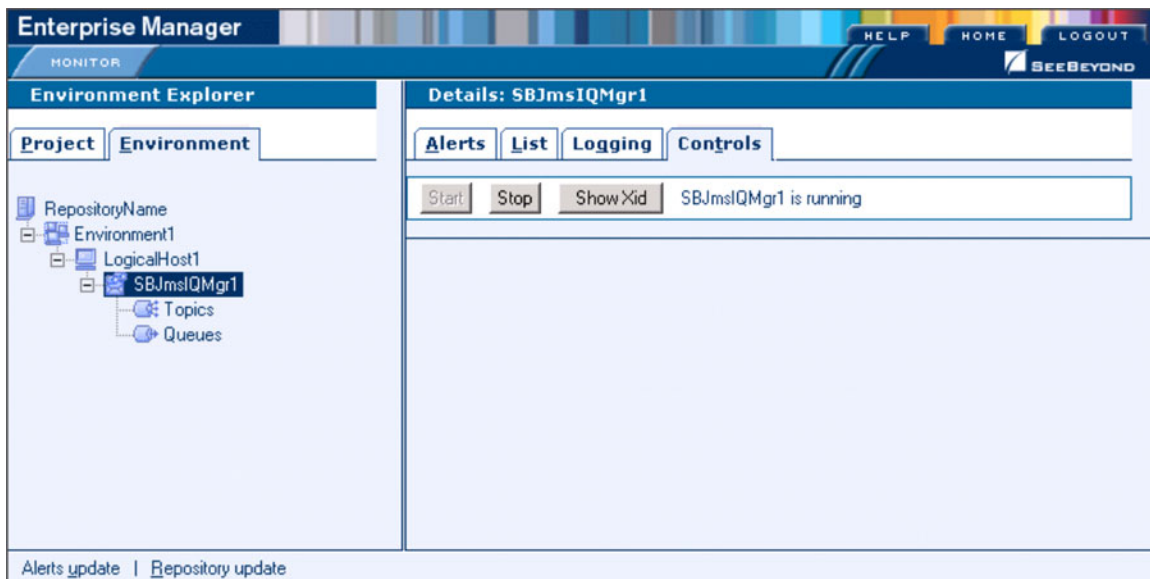
Occasionally, one of the Resource Managers (such as a database server or an external program) involved in a transaction will fail to commit. In that case, the transaction stays open until either the Resource Manager commits or rolls back, or the user intervenes.

The following feature is provided so that an Administrator can force those “indoubt” transactions to roll forward or backward. Typically, an external user will advise the Administrator of the problem, specifying the XID. The Administrator can then search for the indoubt transaction using this XID.

To force an Indoubt Transaction

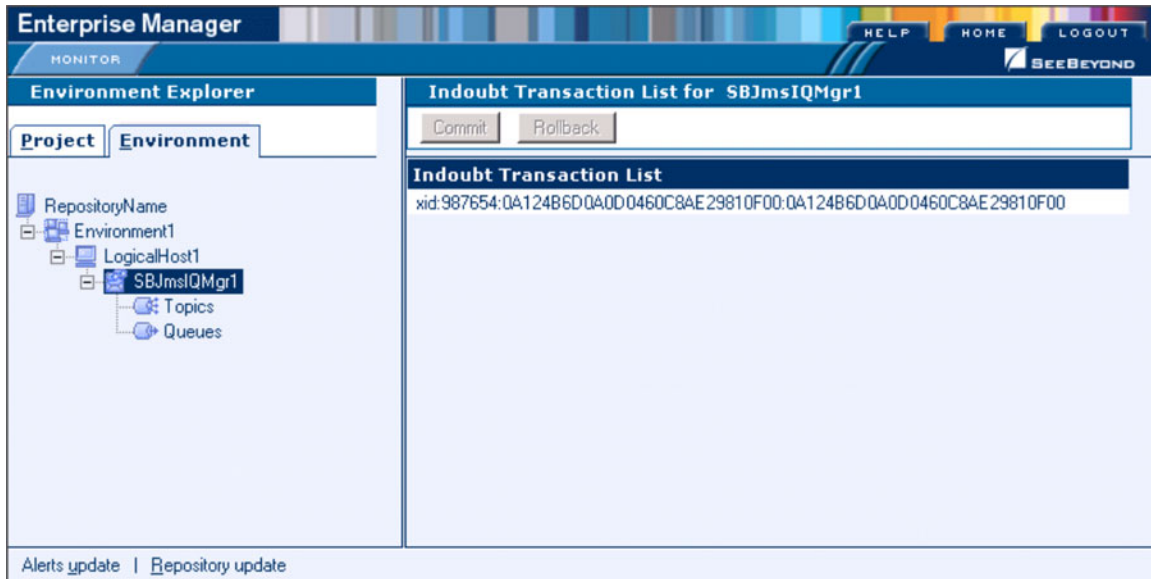
- 1 Click the **Controls** tab in the **Details** panel for the appropriate message server to display the interface shown in Figure 253.

Figure 253 Message Server Details - Controls Tab



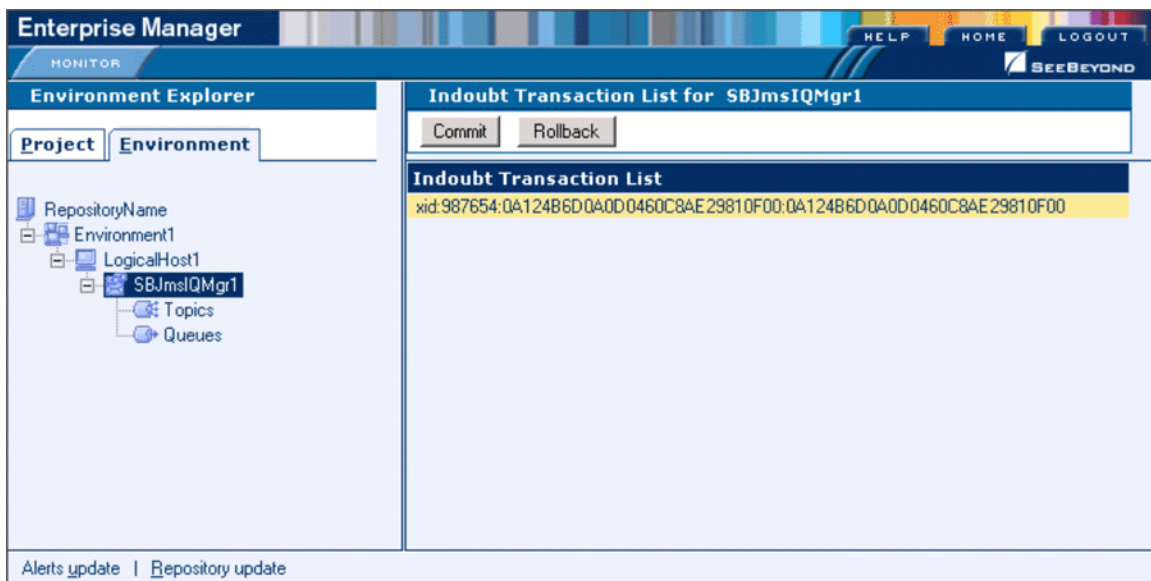
- 2 Click the **ShowXid** button to display the Indoubt Transaction List shown in Figure 254.

Figure 254 Indoubt Transaction List



- 3 Select the transaction having the specified XID, as shown in Figure 255, and click either **Commit** or **Rollback**.

Figure 255 Indoubt Transaction List - Transaction Selected



Debugging and Logging

This chapter provides information about eGate Integrator's debugging and logging features.

14.1 Overview

14.1.1 Debugging

The Java Debugger enables you to debug Java Collaborations as deployed within an integration server on a Logical Host. It features commonly known IDE debug functionalities that facilitate the eGate development process. The debugger offers an alternative to creating logs and warnings in an Java Collaboration and subsequently inspecting them via the Enterprise Manager. A similar facility, the Business Process Debugger, is included in the eInsight Business Process Manager.

14.1.2 Logging

While a deployment profile is active and running, eGate Integrator automatically generates log messages for the Repository, Logical Host, and Enterprise Designer. The logging level determines what type of information is recorded, as the logs will only contain messages of the configured severity level and higher.

The log message files that are generated for the Repository, Logical Host, and Enterprise Designer, and their locations, are described in [Log Files and Locations](#) on page 301.

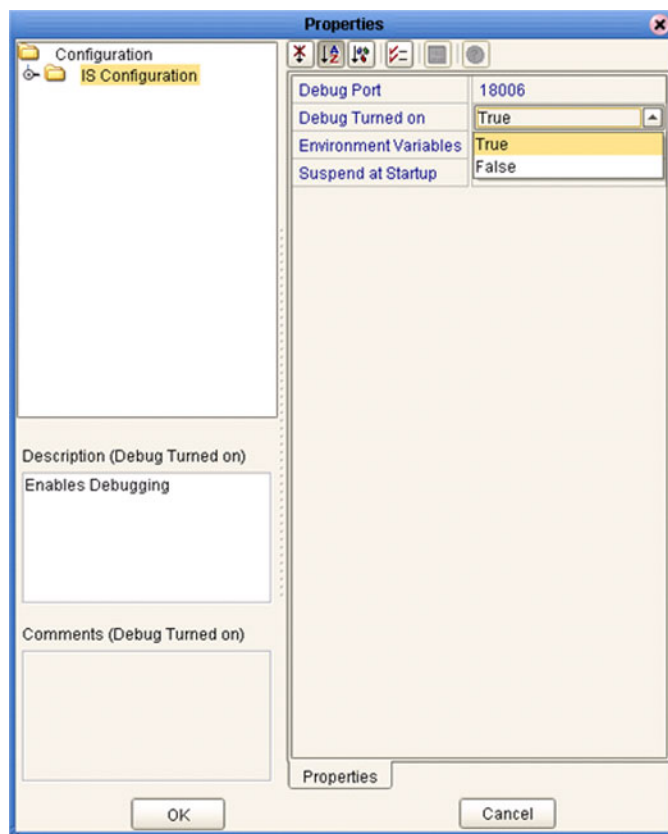
Viewing logs and setting log levels are accomplished in the Enterprise Manager, and are described in [Viewing Logs](#) on page 280 and [Setting Log Levels](#) on page 282.

14.2 The Java Debugger

14.2.1 Enabling the Debugger

- 1 Configure the integration server.
 - A In Enterprise Explorer, display the context menu for the desired integration server.
 - B Click **Properties** to display the integration server's Properties Dialog Box (see Figure 256).

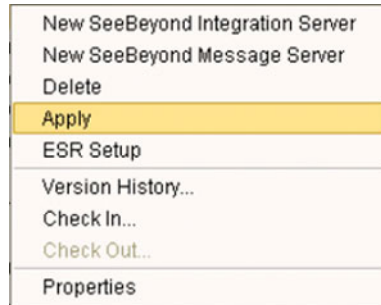
Figure 256 Integration Server Properties Dialog Box



- C Select **True** for the *Debug Turned On* property.

- 2 Apply the changes to the Local Host.
 - A In Enterprise Explorer, display the context menu for the Logical Host and click **Apply** (see Figure 257).

Figure 257 Logical Host Context Menu



- B The **Java Debugger** option now is enabled in the Integration Server context menu.

14.2.2 Invoking the Java Debugger

To invoke the Java Debugger

- 1 In Enterprise Explorer, display the context menu for the desired integration server.
- 2 Click **Java Debugger** (see Figure 258), and the Java Debugger appears in the Enterprise Designer Editor panel (see Figure 259).

Figure 258 Integration Server Context Menu

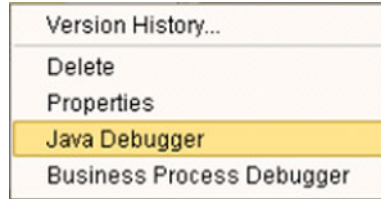
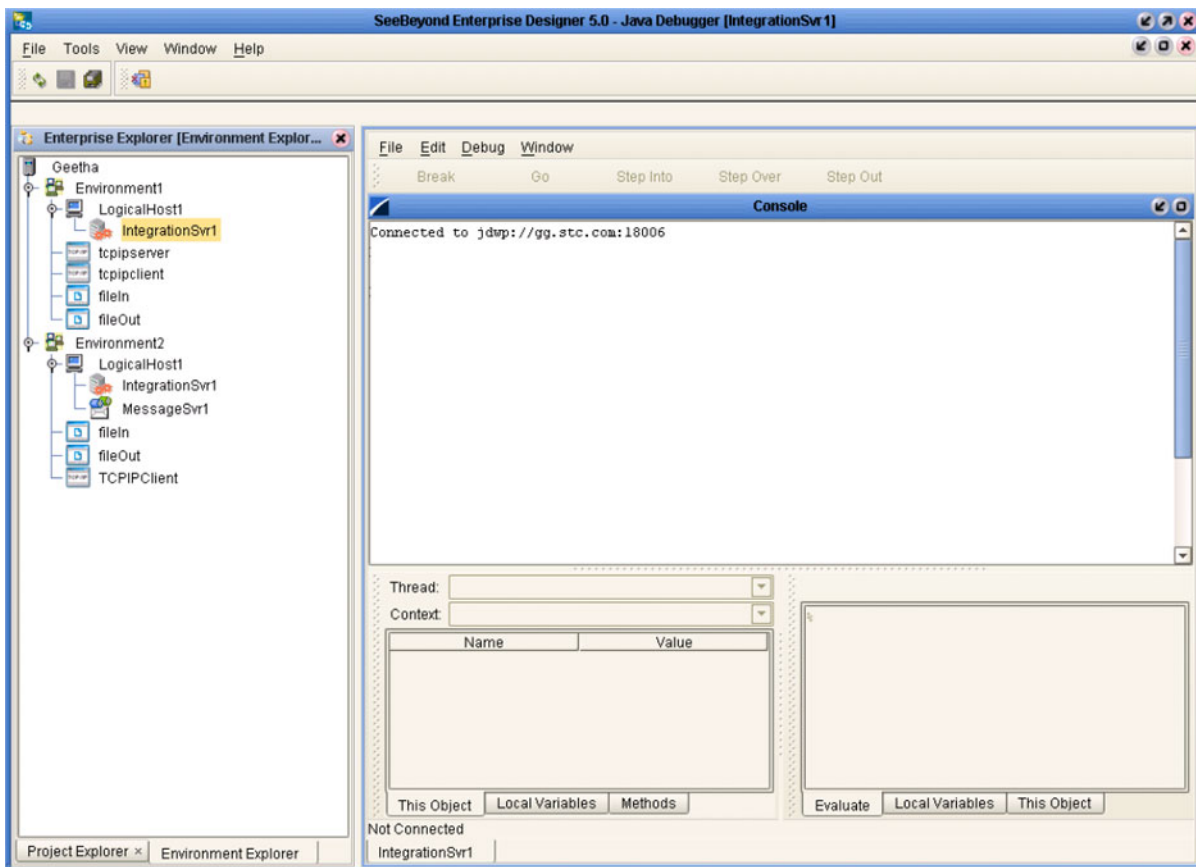


Figure 259 Java Debugger



- 3 The Java Debugger appears whether or not the connection was successful. If there is no *Connected to ...* message, do the following:
 - A Select **Attach to JVM ...** from the File menu (see Figure 260), which displays the Attach to JVM dialog box (see Figure 261).

Figure 260 File Menu

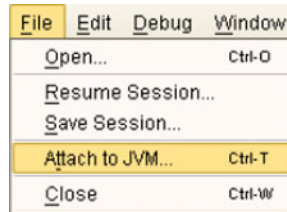
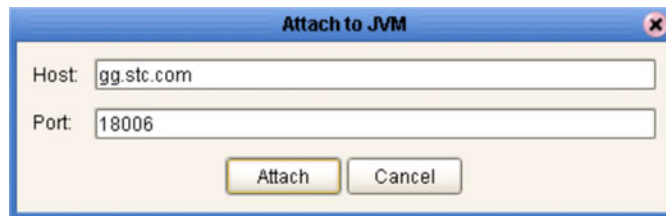
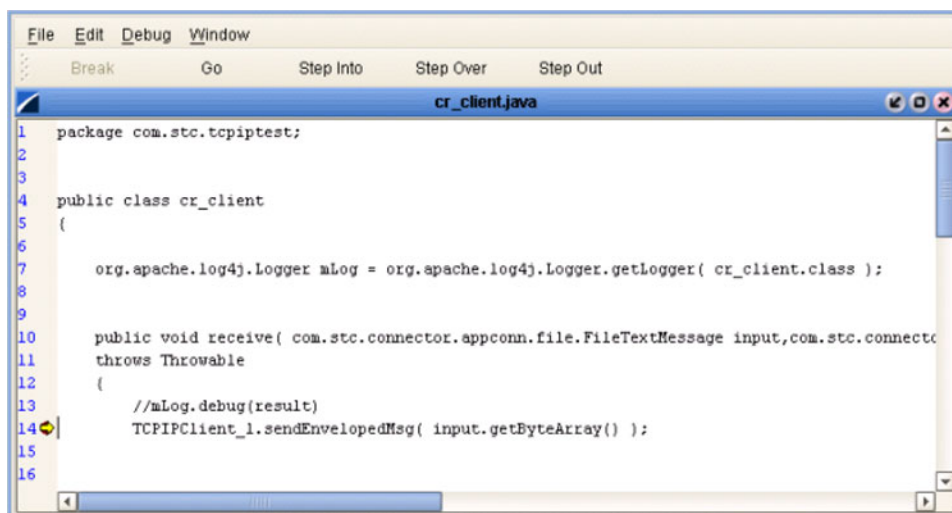


Figure 261 Attach to JVM Dialog Box



- B Enter the integration server's host name and port number into the text boxes and click **Attach**. The debugger then re-attempts to connect to the integration server.
- 4 Once the Java Debugger is running, Java source code is displayed as soon as a Java Collaboration executes (see Figure 262).

Figure 262 Collaboration Source Code Display



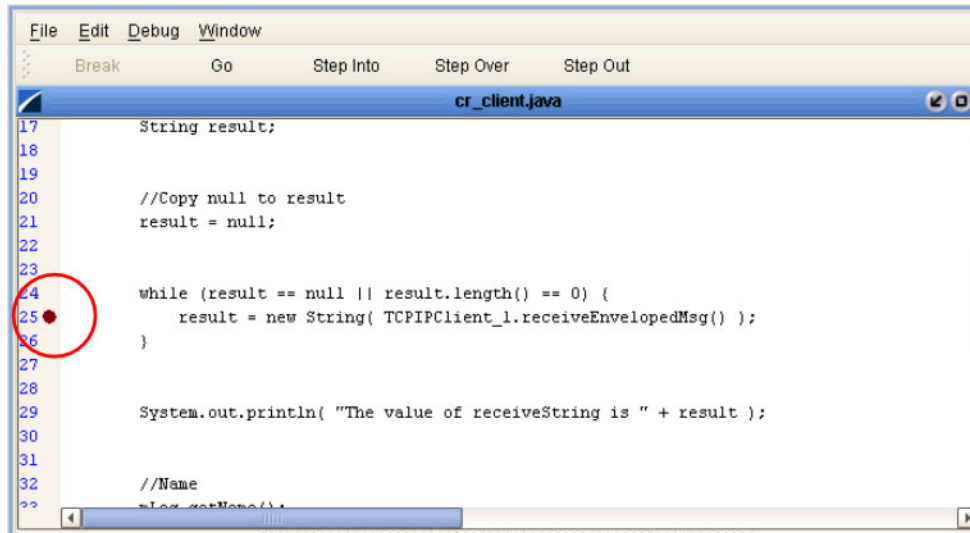
- 5 You can now set breakpoints to assist in examining and debugging the code.

14.2.3 Setting Breakpoints

To set a breakpoint

- 1 Click next to a line number in the executed source code. A red dot is displayed as a marker (see Figure 263).

Figure 263 Breakpoint Example



- 2 Alternatively, you can set stops in a specific class or method, or have the debugger break on an exception. These options are available from the **Debug** menu.
 - A To set a stop in a method, for example, select **Stop in Method ...** from the Debug menu (see Figure 264). A *Stop in Method* dialog box is displayed, in which you can select the desired method (see Figure 265).

Figure 264 Debug Menu

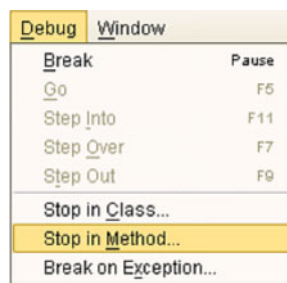
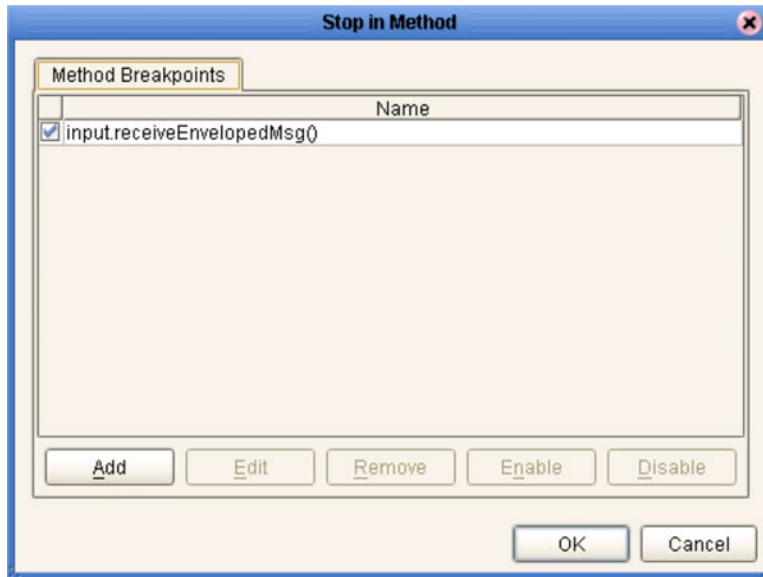


Figure 265 Stop in Method Dialog Box



- B To break on an exception, select **Break on Exception ...** from the Debug menu, which displays a *Choose Exception* dialog box (see Figure 266). All occurrences of the specified exception are then trapped and reported (see Figure 267).

Figure 266 Choose Exception Dialog Box

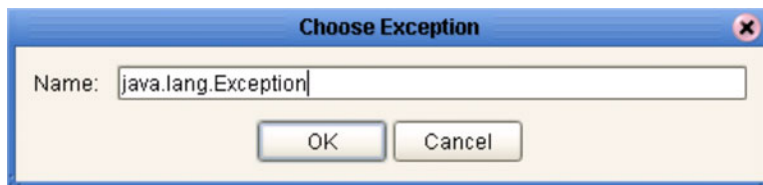


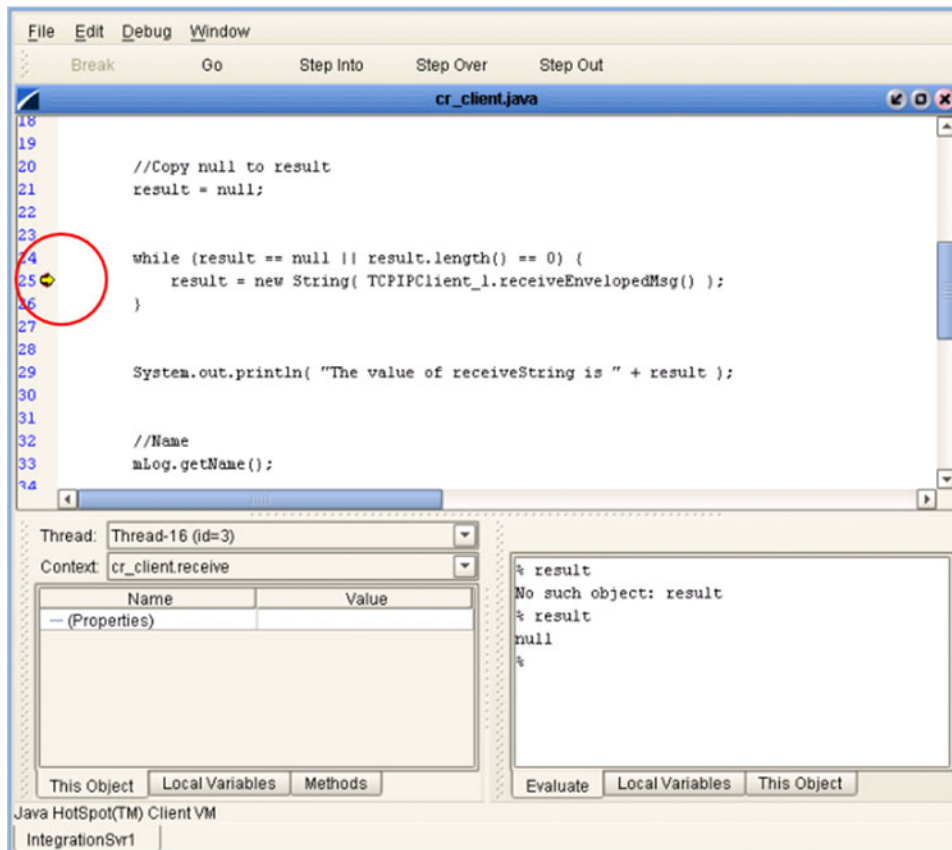
Figure 267 Break on Exception Dialog Box



14.2.4 Inspecting and Editing the Source Code

As soon as the execution of the Java Collaboration arrives at a set breakpoint, it stops executing and displays an right arrow indicator next to the line number in the source code (see Figure 268).

Figure 268 Breakpoint Indicator

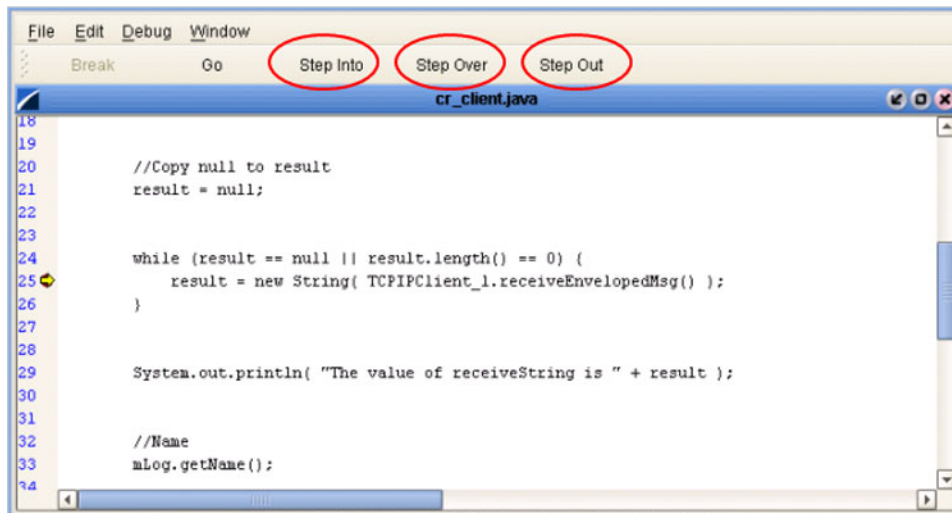


At this point, you can continue by (for example):

- ◆ Stepping Into
- ◆ Stepping Over
- ◆ Stepping Out
- ◆ Inspecting a local variable
- ◆ Setting a local variable

Stepping Into, Over, or Out

Figure 269 Stepping Into, Over, and Out Commands

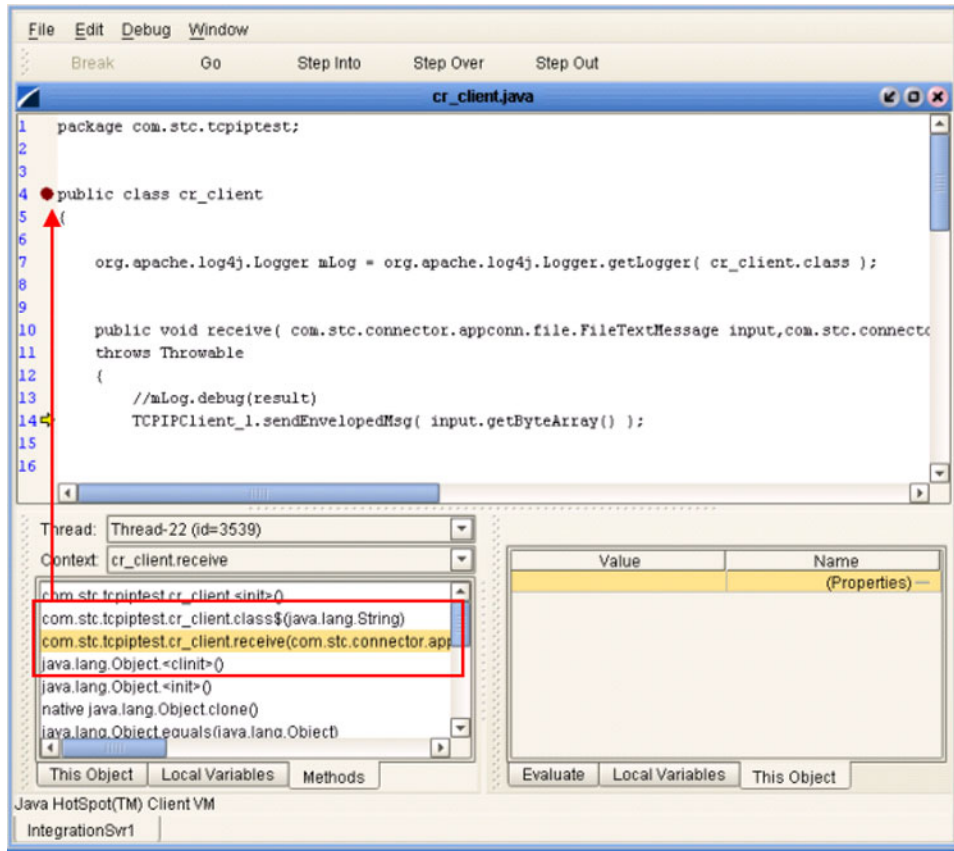


- By selecting the **Step Into** option (see Figure 269), the breakpoint is lifted and execution of the Collaboration will continue, *including* the line of code at the breakpoint.
- By selecting the **Step Over** option, the breakpoint is lifted and execution of the Collaboration will continue, *ignoring* the line of code at the breakpoint.
- By selecting the **Step Out** option, execution of the Collaboration is terminated.

Inspecting Java Threads and Methods

Selecting the **Methods** tab in the left bottom panel of the debugger displays the currently executed Java thread and method (see Figure 270).

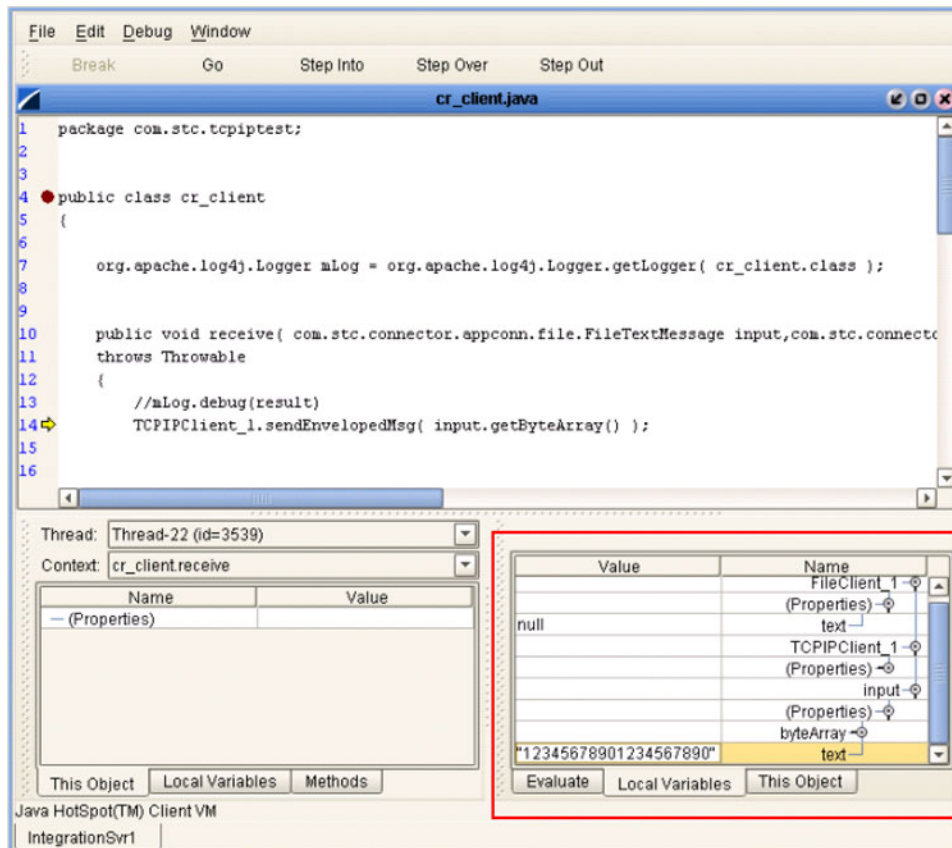
Figure 270 Java Thread and Method Display



Inspecting a Local Variable or Method

You can inspect a local variable by selecting the **Local Variables** tab in the right bottom panel of the debugger (see Figure 271). All nodes of the currently executed Java class are displayed here, and you can expand or collapse certain nodes to search for the value of the desired variable.

Figure 271 Local Variables Tab

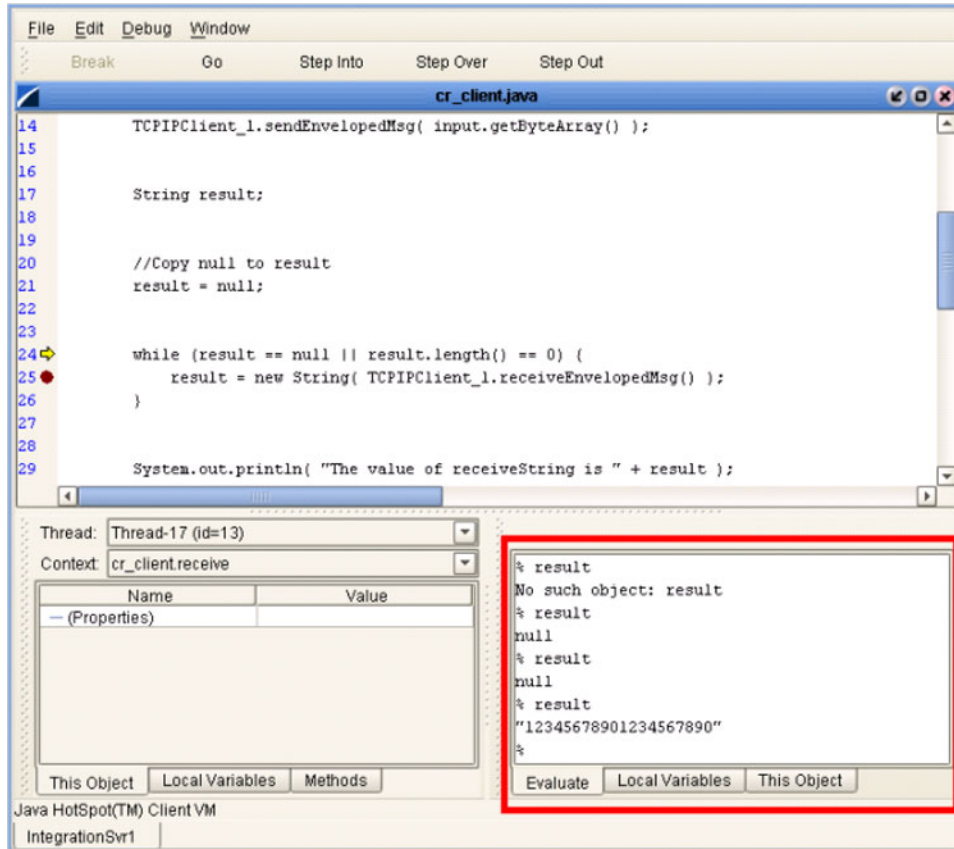


You also can inspect a local variable by selecting the **Evaluate** tab and entering the variable name in the panel using the following syntax:

```
% <variable_name>
```

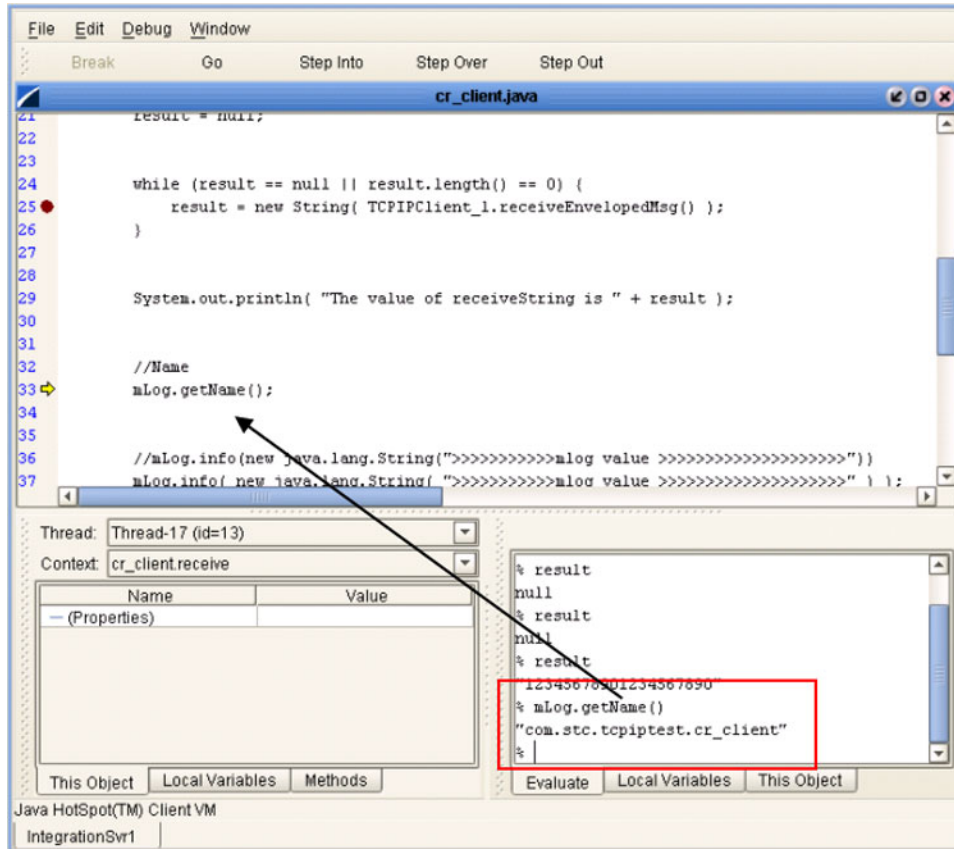
If the code has not initialized a variable, *No such object* is displayed; otherwise, the current value of the variable is displayed (see Figure 272).

Figure 272 Evaluate Local Variable



You can also inspect the result of a method by entering the method name, following the same syntax format (see Figure 273).

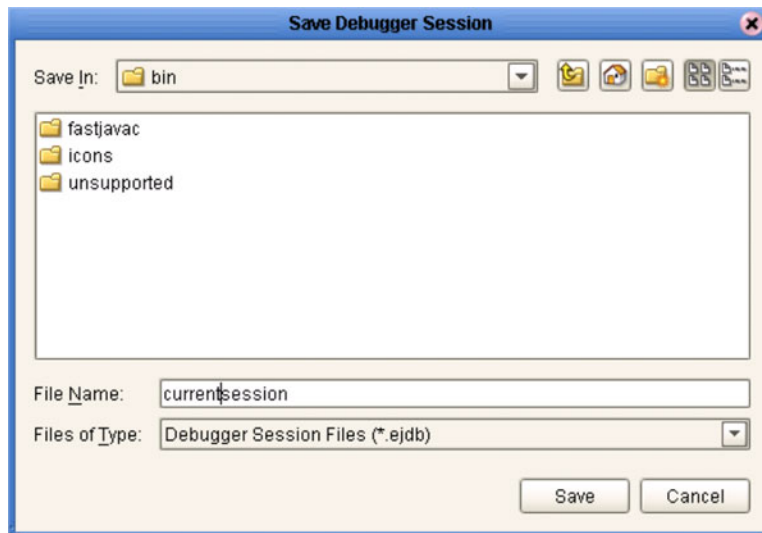
Figure 273 Evaluate Method



Saving and Resuming Debug Sessions

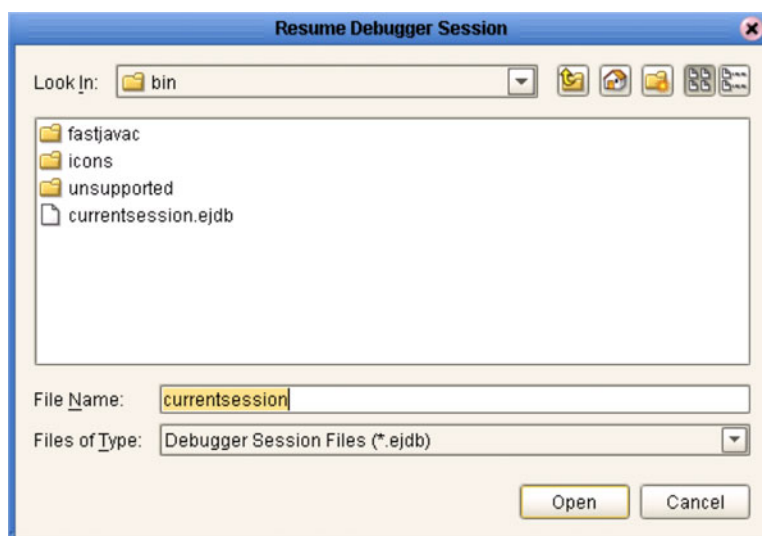
You can pause the debugging process by saving the session to a file. Selecting **Save Session** from the debugger File menu displays the *Save Debugger Session* dialog box (see Figure 274), in which you specify the file name and location.

Figure 274 Save Debugger Session Dialog Box



You can continue a debugging process that was paused by saving to a file. Selecting **Resume Session** from the debugger File menu displays the *Resume Debugger Session* dialog box (see Figure 275), in which you specify the file name and location.

Figure 275 Resume Debugger Session Dialog Box

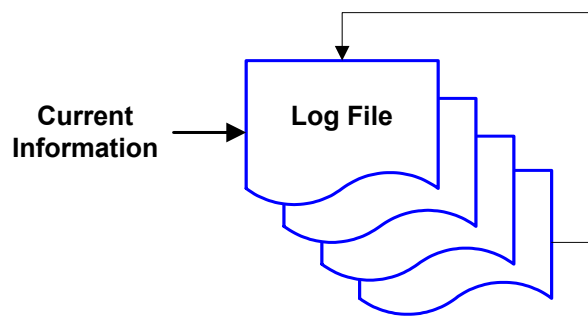


14.3 eGate Logs

14.3.1 Log File System

While a deployment profile is active and running, eGate Integrator automatically generates log messages for the Repository, Logical Host, and Enterprise Designer. The log files constitute a recirculating stack (see Figure 276). You can specify both the file size and number of files in the stack for each Logical Host and Integration Server instance (the respective property names are `MaxFileSize` and `MaxBackupIndex`).

Figure 276 Recirculating Log File Stack



As soon as the maximum file size is reached in the currently active log file, a new log file is created. When the number of files in the stack reaches the specified limit, the oldest one is deleted when the new one is created. The effect is that the oldest one is emptied and moved to the top of the stack. A separate stack is maintained for each log file type.

Log files are initialized during the installation of a new logical host, so if you reinstall a logical host, all existing log files are deleted. If you would like to preserve log files (for example, on a weekly basis), you can copy the log files to a backup storage location.

14.3.2 Log File Management

Log files are managed from the Enterprise Manager; see:

- [Viewing Logs](#) on page 280
- [Setting Log Levels](#) on page 282

14.3.3 Log File Properties

Logical Host

Log file properties are set in the following properties file.

`ican50/logicalhost/logconfigs/LH/log4j.properties`

Table 52 Log Properties for Logical Hosts

Property	Default Value
log4j.appender.FILE	org.apache.log4j.RollingFileAppender
log4j.appender.FILE.File	C:/ican50/logicalhost/logs/stc_lh.log
log4j.appender.FILE.MaxFileSize	10MB
log4j.appender.FILE.MaxBackupIndex	10
log4j.appender.FILE.layout	org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern	%d{ISO8601} %-5p [%t] [%c] [%x] %m%n
log4j.rootCategory	INFO, FILE

***Note:** If it becomes necessary to increase space for Logical Host log files, you must shut down the Logical Host, change the **MaxFileSize** and/or **MaxBackupIndex** properties, and re-bootstrap the Logical Host.*

Integration Server

Log file properties are set in the following properties file.

`ican50/logicalhost/logconfigs/IS_integservername/log4j.properties`

Table 53 Log Properties for Integration Servers

Property	Default Value
log4j.appender.FILE	org.apache.log4j.RollingFileAppender
log4j.appender.FILE.File	C:/ican50/logicalhost/logs/stc_is_IntegrationSvr1.log
log4j.appender.FILE.MaxFileSize	10MB
log4j.appender.FILE.MaxBackupIndex	10
log4j.appender.FILE.layout	org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern	%d{ISO8601} %-5p [%t] [%c] [%x] %m%n
log4j.rootCategory	INFO, FILE

14.4 Log Files and Locations

Note: *The Repository Deployment Application log is related to all deployment actions spawned by invoking the **Apply** menu item from Enterprise Designer or by invoking the bootstrap script. If any errors occur during these invocations and the Enterprise Designer/bootstrap logs do not contain enough information to locate the problem, this log will contain the root cause of the problem—if it originated from the deployment application residing on the repository server.*

14.4.1 Repository

- Administration servlet log:

```
repository-root/repository/server/logs/hostname_admin_log.date.txt
```

- Examples servlet log:

```
repository-root/repository/server/logs/  
hostname_examples_log.<date>.txt
```

- Default repository and manifest servlet log:

```
repository-root/repository/server/logs/hostname_log.date.txt
```

- ♦ Init file:

```
repository-root/server/conf/log4j.properties.
```

- Repository installation log:

```
repository-root/repository/logs/install.log
```

- Master repository log:

```
repository-root/repository/logs/repository.log
```

- ♦ Init file:

```
repository-root/server/webapps/repositoryconfig.properties
```

- Deployment Application log:

```
repository-root/repository/logs/deployment-servlet.log
```

- ♦ Log4J init file:

```
repository-root/server/work/Standalone/localhost/deployment /WEB-  
INF/classes/deployment-servlet-log4j.properties.
```

- Enterprise monitor log.:

```
repository-root/repository/logs/install.log
```

- SeeBeyond Security Service (SSS) installation log:

```
repository-root/repository/logs/sss_install.log
```

- ♦ Log4J init file:

```
repository-root/server/SSS/logs/log4j-init.properties
```

- Velocity templates log (related to SSS).:

```
repository-root/repository/logs/velocity.log
```

14.4.2 Logical Host

- Master log file:

logical-host-root/stcma/stc_lh.log

- Init file:

logical-host-root/bootstrap/config/log4j.xml

14.4.3 Management Agent

- Master log file:

logical-host-root/stcma/stc_lh.log

- Deployment event log file:

logical-host-root/bootstrap/bin/deploy.log

- Deployment log file:

logical-host-root/stcma/deploy.log

- Velocity template log file:

logical-host-root/stcma/Velocity.log

- Init file:

logical-host-root/bootstrap/config/log4j.xml

14.4.4 Integration Server

- Master log file:

logical-host-root/stcis/logs/integ_server_instance_name/stc_is.log

- Init file:

*logical-host-root/test/stcis/integ-server-name/log4j-
config.properties*

14.4.5 Message Server

- Master log file:

logical-host-root/stcms/message-server-instance-name/stc_ms.log

- Init file:

*logical-host-root/stcms/message-server-instance-name/
stcms.default.Properties*

14.4.6 Enterprise Designer

- Master log file:

enterprise-designer-root/bin/stc.log

- Init file:

enterprise-designer-root/bin/log4j.properties

Glossary

Collaboration

(See Service and Collaboration Definition.)

Collaboration Definition

The encoding of business rules, in Java or XSLT format. Typically, the encoding consists of operations on OTDs (see **“OTD” on page 304**). Several Collaborations can have the same Collaboration Definition.

Connection

Consists of the configuration information that enables an eWay to connect to an external system.

Connectivity Map

Contains business logic and routing information about the data transmission. A Connectivity Map usually includes one or more Collaborations, Passthrough Collaborations, topics, queues, and eWays. A Connectivity Map is created under a Project. A Project may have multiple Connectivity Maps.

Constants

A name or value pair that is visible across a Project.

Deployment Profile

Contains the information about how the Project components will be deployed in an Environment. A Project can have multiple Deployment Profiles, but only one Deployment Profile can be activated for a Project in any one Environment.

Derived Collaboration

Collaboration that inherits operations from another, according to standard object-oriented practice.

eGate System

See **“Project”**.

Environment

A collection of physical resources and their configurations that are used to host Project components. An Environment contains logical hosts and external systems.

eWay

A link between a Collaboration and an external connection including the message server connection (topic or queue) or external application.

External Application

A logical representation of an external application.

External System

A representation of an external application system.

ICAN Suite

The SeeBeyond Integrated Composite Application Network Suite, which is based on eGate Integrator.

Integration Server

Software platform that houses the business logic container used to run Collaborations. Provides transaction services, persistence, and external connectivity.

Link

The JMS Connection between a Collaboration and a topic or queue in a JMS-compliant message server.

Linked Message Destination

A reference to a Message Destination defined in another Connectivity Map.

Logical Host

An instance of the eGate runtime Environment that is installed on a machine. A Logical Host contains the software and other installed components that are required at runtime, such as application and message servers.

Management Agent

Uses J2EE technology to manage and monitor an eGate 5.0 deployment that may contain other application servers in addition to the SeeBeyond Integration Server. Defines management interfaces and services designed for distributed environments, focusing on providing functionality for managing networks, systems, and applications.

Message Destination

A general term for a topic or queue. Two or more Projects can share a message destination that has the same name and is deployed on the same message server. A single Project may also have a single message destination referenced in multiple Connectivity Maps.

Message Server

JMS-compliant, guaranteed delivery store, forwarding, and queueing service.

OTD

An acronym for Object Type Definition. OTDs contain the data structure and rules that define an object. An OTD is used in Java Collaboration Definitions for creating data transformations and interfacing with external systems.

Project

Contains a collection of logical components, configurations, and files that are used to solve business problems. A Project organizes the files and packages and maintains the settings that comprise an eGate system in SeeBeyond's Enterprise Designer.

Queue

A JMS queue is a shareable object that conforms to the *point-to-point* (p2p, or PTP) messaging domain, where one sender delivers a message to exactly one receiver. When the SeeBeyond Message Server sends a message to a queue, it ensures it is received once and only once, even though there may be many receivers “listening” to the queue. This is equivalent to the subscriber pooling in other queue implementations. You can reference a queue that exists in another Connectivity Map or Project.

Repository

Stores and manages the setup, component, and configuration information for eGate Projects. The Repository also provides monitoring services for Projects, which include version control and impact analysis.

Schema Runtime Environment

An add-on in eGate 5.0 that provides the upgrade path for e*Gate 4.x users to upgrade to eGate 5.0. Also known as the SRE.

Security Server

A standalone server that is the connection point to underlying eGate security environments.

Service

Contains the information about executing a set of business rules. These business rules can be defined in a Java Collaboration Definition, XSLT Collaboration Definition, Business Process, eTL Definition, or other service. A Service also contains binding information for connecting to JMS Topics, Queues, eWays, and other services.

Subproject

An independent Project that is included as part of another Project and listed on the Enterprise Explorer tree beneath the main Project icon.

Topic

A JMS topic is a shareable object that conforms to the *publish-and-subscribe* (pub/sub) messaging domain, where one publisher broadcasts messages to potentially many subscribers. When the SeeBeyond Message Server publishes a message on a topic, it ensures that all subscribers receive the message.

XSLT

An acronym for Extensible Stylesheet Language Transformations. A file format used in eGate to generate Collaboration Definitions.

e*Gate 4.x Terms in eGate 5.0

Table 54 provides definitions for the terms that are new with eGate release 5.0, as well as equivalent terms from eGate release 4.x.

Table 54 eGate 5.0 Terms

5.0 Term	4.x Equivalent Term
Collaboration	Collaboration
Collaboration Definition	Collaboration Definition
Connection	e*Way Connection
Connectivity Map	Closest: Network View of an entire Schema
Deploy	Run the Control Broker
Deployment	<none>
Deployment Profile	Closest: Schema
Enterprise Designer	Enterprise Manager
Enterprise Manager	Enterprise Monitor
Environment	Schema (except only includes physical information, not business logic)
eWay	e*Way Connection e*Way
eWay Configuration	e*Way Connection Configuration
External Application	e*Way Connection
External System	e*Way Connection
JMS Connection	e*Way Connection
Integration Server	<none>
Link	JMS e*Way Connection
Linked Message Destination	<none>
Logical Host	Participating Host
Message Destination	Topic or queue
Message Server	JMS IQ Manager
Object Type Definition (OTD)	Event Type Definition (ETD)
Process Manager	Control Broker
Project	Schema (except not including physical layer)
Queue	JMS queue
Repository	Registry
Subproject	Schema

Table 54 eGate 5.0 Terms (Continued)

5.0 Term	4.x Equivalent Term
Topic	JMS topic
XSLT	<none>

Index

A

- ACL properties 40–41, 46–47
- activate deployment profile 204
- addition method 139, 185
- admin 265
- Advanced Mode command 125
- analysis questions 230
- AND method 136, 185
- arrayAccess method 147
- arrayAssign method 147
- arrayLength method 147
- Auto Layout command 131

B

- bitNot method 149
- boolean method 193
- Break command 128
- business rules
 - tree 129
- Business Rules on Left command 125
- Business Rules on Top command 125

C

- Call Java Method command 131
- Call New Constructor command 131
- cast method 141
- ceiling method 191
- charAt method 144
- Collaboration 303, 306
 - derived 303
- Collaboration definition 303, 306
- Collaboration Definitions
 - Java 115
 - XSLT 176
- Collapse All Methods command 131
- commands
 - Open File 109
 - Refresh OTD 109
 - Run Tester 109
 - Save as New Name 109
 - Save File 109
 - Sort by Name 109

- Tester 109
- Toggle Reference Tab Panel 109
- Commit Changes command 125
- Commit Code Changes command 183
- component data 230
- concat method 144
- concatenate method 188
- connection 303, 306
- Connectivity Map 303, 306
 - toolbar 61
- Connectivity Map Editor 60
- constants 303
- contains method 193
- Continue command 128
- Control Broker 306
- conventions, writing in document 22
- count method 195
- create diff 151, 198
- Create Literal command 131
- current method 195
- customizer 66

D

- deactivate deployment profile 204
- deploy 306
- Deployment 306
- Deployment Profile 203, 303, 306
 - components 204
- deployment profile
 - activate 204
 - deactivate 204
 - map variables 204
- derived Collaboration 303
- division method 139, 185
- document method 195

E

- e*Way 306
- e*Way Connection 306
- e*Way Connection Configuration 306
- Editor
 - Connectivity Map 60
- eGate system 303
- element-available method 193
- endsWith method 144
- enter file name to export 238
- Enterprise Designer 306
 - enterprise explorer 37
 - menu bar 34
 - object type definition 77
- enterprise explorer
 - environments view 42

- project explorer view 37
- Enterprise Manager 306
 - Administration 265
 - Documentation 269
 - Downloads 268
 - Home 261
 - Home tab 261
 - login 259
 - monitor 261
- Enterprise Monitor 306
- Environment 303, 306
 - components 42
- Environments 28
- equal method 137, 186
- equals method 141
- ETD 306
- Event Type Definition 306
- eWay 303, 306
- eWay Configuration 306
- Expand All Methods command 131
- Export Java Rule command 125
- external
 - application 304, 306
 - system 304, 306

F

- false method 194
- Field command 128
- Find command 126
- Find Next command 126
- Find Previous command 126
- floor method 191
- For command 128
- format_number method 188
- function_available method 194

G

- generate-id method 196
- greater_or_equal method 137, 186
- greater_than method 137
- greater_then method 186

I

- ICAN Suite 304
- Icons
 - JCE Business Rules Designer toolbar 131
 - JCE Business Rules toolbar 128
 - JCE toolbar 125
 - XCE toolbar 183
- id method 196

- If-Then command 128
- impact analyzer
 - command buttons 231
 - overview 230
 - printing 232
 - toolbar button 36
 - using 231
- Import a Local File command 125
- Import JAR File command 126
- Import Static Field command 131
- Import XSL from a Local File command 183
- indexOf method 145
- indoubt transaction 283
- instanceof method 141
- Integration Server 304, 306

J

- Java Collaboration Definitions 115
- Java methods
 - addition 139
 - AND 136
 - arrayAccess 147
 - arrayAssign 147
 - arrayLength 147
 - bitNot 149
 - cast 141
 - charAt 144
 - concat 144
 - division 139
 - endsWith 144
 - equal 137
 - equals 141
 - greater_or_equal 137
 - greater_than 137
 - indexOf 145
 - instanceof 141
 - length 145
 - lesser_or_equal 138
 - lesser_than 138
 - multiplication 139
 - negative 149
 - NOT 136
 - not_equal 138
 - OR 136
 - positive 149
 - postDecrement 149
 - postIncrement 150
 - preDecrement 150
 - preIncrement 150
 - remainder 140
 - replace 145
 - substring 145
 - subtraction 140

- toLowerCase 145
- toString 141
- toUpperCase 145
- trim 146

JCE commands

- Advanced Mode 125
- Auto Layout 131
- Break 128
- Business Rules on Left 125
- Business Rules on Top 125
- Call Java Method 131
- Call New Constructor 131
- Collapse All Methods 131
- Commit Changes 125
- Continue 128
- Create Literal 131
- Expand All Methods 131
- Export Java Rule 125
- Field 128
- Find 126
- Find Next 126
- Find Previous 126
- For 128
- If-Then 128
- Import a Local File 125
- Import JAR File 126
- Import Static Field 131
- Local Variable 128
- Method 128
- New Array 131
- Redo 126
- Refresh Collaboration 126
- Replace 126
- Return 128
- Roll Back Changes 125
- Rule 128
- Source Code Mode 125
- Standard Mode 125
- Throw 128
- Try 128
- Undo 126
- Validate 125
- While 128

JMS

- connection 306
- e*Way Connection 306
- IQ Manager 306
- queue 306
- topic 307

K

- key method 196

L

- language method 194
- last method 196
- length method 145
- lesser_or_equal method 138, 186
- lesser_than method 138, 186
- link 304, 306
- linked message destination 304, 306
- Local Variable command 128
- local-name method 196
- Logical Host 304, 306
 - Starting as a Windows Service 225
 - Starting Manually on Windows 226
 - Starting on an HP NonStop Server 228
 - Starting on Linux 227
 - Starting on UNIX 227
 - Startup Configuration File 220
 - Startup Parameters (Windows) 219

M

- Management Agent 304
- map variables 204
- menu bar 34
- merge diff 152, 199
- message
 - destination 304, 306
 - server 304, 306
- Method command 128
- method palette
 - XSLT 184
- mod method 186
- Monitor Server 261, 271
- multiplication method 139, 187

N

- name method 196
- namespace-uri method 196
- negative method 149, 187
- network view 306
- New Array command 131
- normalize_space method 188
- NOT method 136, 194
- not_equal method 138, 187
- number method 191
- number-literal method 192

O

- Object Type Definition 304, 306
 - wizard 82, 101, 105
- Open File command 109

OR method 136, 187
OTD 304, 306

P

Participating Host 306
position method 197
positive method 149
postDecrement method 149
postIncrement method 150
preDecrement method 150
preIncrement method 150
Process Manager 306
Project 304, 306
 project explorer view 37

Q

queue 305–306

R

Redo command 126
Refresh Collaboration command 126
Refresh OTD command 109
Registry 306
remainder method 140
Replace command 126
replace method 145
Repository 28, 305–306
Return command 128
roles 243
Roll Back Changes command 125
Roll Back Code Changes command 183
round method 192
Rule command 128
Run Tester command 109

S

Save as New Name command 109
Save File command 109
Save XSL to a Local File command 183
Schema 306
Schema Runtime Environment 305
security 218
Security Server 305
select file to import 240
Show Mapping Only command 183
Show Maps and Code command 183
Show XSLT Code Only command 183
Sort by Name command 109
Source Code Mode command 125

SRE 305
Standard Mode command 125
starts_with method 194
string method 189
string_length method 189
string_literal method 189
subproject 305–306
substring method 145, 189
substring-after method 189
substring-before method 189
subtraction method 140, 187
sum method 192
supporting documents 23
system-property method 190

T

Tester command 109
Throw command 128
Toggle Reference Tab Panel command 109
toLowerCase method 145
topic 305–307
toString method 141
toUpperCase method 145
translate method 190
trim method 146
true method 194
Try command 128

U

Undo command 126
unparsed-entity-uri method 190
user management 244

V

Validate command 125
version control 233–235

W

While command 128

X

XCE commands
 Commit Code Changes 183
 Import XSL from a Local File 183
 Roll Back Code Changes 183
 Save XSL to a Local File 183
 Show Mapping Only 183
 Show Maps and Code 183

- Show XSLT Code Only 183
- XSLT 305, 307
- XSLT Collaboration Definitions 176
- XSLT Collaboration Editor
 - method palette 184
- XSLT methods
 - addition 185
 - AND 185
 - boolean 193
 - ceiling 191
 - concatenate 188
 - contains 193
 - count 195
 - current 195
 - division 185
 - document 195
 - element-available 193
 - equal 186
 - false 194
 - floor 191
 - format_number 188
 - function_available 194
 - generate-id 196
 - greater_or_equal 186
 - greater_than 186
 - id 196
 - key 196
 - language 194
 - last 196
 - lesser_or_equal 186
 - lesser_than 186
 - local-name 196
 - mod 186
 - multiplication 187
 - name 196
 - namespace-uri 196
 - negative 187
 - normalize_space 188
 - NOT 194
 - not_equal 187
 - number 191
 - number-literal 192
 - OR 187
 - position 197
 - round 192
 - starts_with 194
 - string 189
 - string_length 189
 - string_literal 189
 - substring 189
 - substring-after 189
 - substring-before 189
 - subtraction 187
 - sum 192
 - system-property 190
 - translate 190
 - true 194
 - unparsed-entity-uri 190