

SeeBeyond ICAN Suite

eGate Integrator User's Guide

Release 5.0.3



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.

© 2003-2004 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 This work is protected as an unpublished work under the copyright laws.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 20040303143600.

Contents

Figures	10
Tables	18
<hr/>	
Chapter 1	
Introduction	20
Purpose and Scope	20
Intended Audience	20
Organization of Information	21
Writing Conventions	22
Additional Conventions	22
Supporting Documents	22
The SeeBeyond Web Site	23
<hr/>	
Chapter 2	
System Overview	24
Introduction	24
Integration Model	25
System Architecture	27
Repository	28
Environments	28
User Interfaces	29
Enterprise Manager	29
Enterprise Designer	29
<hr/>	
Chapter 3	
Enterprise Manager	32
Overview	32
Installing and Updating eGate	32

Monitoring and Managing eGate	32
Starting Enterprise Manager	33
The Enterprise Manager Interface	34
Home	35
Documentation	36
The ICAN Monitor	37
The SRE Monitor	38
Starting and Using the SRE Monitor	39

Chapter 4

Enterprise Designer	43
Overview	43
Editors	44
Analysis and Archiving tools	44
User Interface	44
Starting Enterprise Designer	45
Interface Features	46
Menus	46
File Menu	46
Tools Menu	46
View Menu	47
Window Menu	47
Help Menu	47
Toolbar	48
Browser Buttons	48
Enterprise Explorer	49
Project Explorer	49
Environment Explorer	50
Enterprise Designer Editors	50
Connectivity Map Editor	51
OTD Editor	52
Collaboration Editor (Java)	53
Collaboration Editor (XSLT)	54
Environment Editor	55
Deployment Editor	56
Additional Tools and Features	57
Project/Environment Import	57
Importing a Project Using Enterprise Designer	57
Importing a Project Using the Command Line	61
Project/Environment Export	62
Exporting a Project Using Enterprise Designer	62
Exporting a Project Using the Command Line	66
Impact Analyzer	67
Version Control	69
Checking a Component In	69

Checking a Component Out	70
Viewing a Component’s Version History	71

Chapter 5

eGate Projects	72
Overview	72
Project Components	72
The Project Explorer	73
Project Explorer Icons	74
Context Menus	75
Repository Menu	75
Project Menu	76
Connectivity Map Menu	77
Using the Connectivity Map Editor	79
Services	81
Collaborations	81
Message Destinations	82
Topics	82
Queues	82
External Applications	82
Schedulers	82
Component Connections	84
Configuring a Connection	85
Defining Constants and Variables	87

Chapter 6

Object Type Definitions	89
Overview	89
The <i>Bean-like</i> Interface	89
OTD Types	90
Externally-Defined OTDs	90
User-Defined OTDs	90
Building OTDs	90
Using the OTD Wizard	91
Externally-Defined OTDs	93
Using the DTD Wizard	93
Using the WSDL Wizard	98
WSDL OTD Structure	101
WSDL Operation Elements	102
Using the XSD Wizard	103
User-Defined OTDs	108
Using the User-Defined OTD Wizard	108

Editing the OTD Properties	110
Node Properties	110
Element Properties	111
Field Properties	112
Specifying the Node Type	113
Specifying Delimiters	114
Delimiter Properties	115
Multiple Delimiters	119
Escape Delimiters	119
Precedence	120
Creating a Delimiter List	121
Using the OTD Editor	125
Node Management	127
Using the OTD Tester	128

Chapter 7

Collaboration Definitions (Java)	132
Overview	132
Using the Collaboration Definition Wizard (Java)	133
Creating a Java-based Collaboration Definition	134
New Web Service	135
Existing Web Service	139
Collaboration Definition Properties	141
Using the Collaboration Editor (Java)	143
Java Toolbar Icons	144
Business Rules Editor	146
Commands	146
Business Rules Toolbar Icons	147
Business Rules Tree	148
Business Rules Designer	149
Method-Access Nodes	150
Business Rules Designer Toolbar Icons	151
Collaboration Method Palette (Java)	155
Collaboration Method Boxes (Java)	155
Java Source Editor	156
Collaboration Methods (Java)	157
Boolean Methods	157
Comparison Methods	158
Math Methods	160
Object Methods	162
Object Method Dialog Boxes	163
String Methods	165
Array Operation Methods	168
Operators	170
Version Control	172
Creating a Modified Collaboration Definition (Java)	172
Merging Two Versions of a Collaboration Definition (Java)	173

Setting Up a Collaboration Definition Variable (Java)	174
Creating a Variable	174
Invoking Variable Constructor	177
Displaying Method Classes	179
Using <i>Try-Catch</i>	180
Adding and Using Third-Party Java Classes	183
Adding Class Instances to a Collaboration	187
Validating Java-based Collaborations	189
Debugging Java-based Collaboration Definitions	190
Enabling the Debugger	190
Invoking the Java Debugger	192
Setting Breakpoints	194
Inspecting and Editing the Source Code	196
Stepping Into, Over, or Out	197
Inspecting Java Threads and Methods	198
Inspecting a Local Variable or Method	199
Saving and Resuming Debug Sessions	202
Creating Alerts	203
Creating Log Entries	206

Chapter 8

Collaboration Definitions (XSLT)	208
Overview	208
Using the Collaboration Definition Wizard (XSLT)	209
Creating a Collaboration Definition (XSLT)	210
New Web Service	211
Existing Web Service	214
Using the Collaboration Editor (XSLT)	215
XSLT Toolbar Icons	216
Collaboration Method Palette (XSLT)	217
Collaboration Method Boxes (XSLT)	217
Collaboration Methods (XSLT)	218
Operator Methods	218
String Methods	221
Number Methods	224
Boolean Methods	226
Nodes Methods	228
UAN Extension Methods	231
Version Control	238
Creating a Modified Collaboration Definition (XSLT)	238
Merging Two Versions of a Collaboration Definition (XSLT)	238

Chapter 9

Environments	240
Overview	240
Environment Explorer	240
Environment Explorer Icons	241
Context Menus	242
Repository Menu	242
Environment Menu	243
Logical Host Menu	244
Environment Editor	246
Defining Environmental Constants	246
Logical Hosts	248
Overview	248
Configuring a Logical Host	249
Configuring the Base Port Number	250
Integration Servers	251
Configuring an Integration Server	251
Web Container	253
Default Web Server	253
Performance Monitoring (Profiling)	255
Security Realm	256
eInsight Engine	257
Application Manager	258
Integration Server	259
Oracle JDBC Connection Pool	261
Deploying User-Defined Stateless Session Beans	263
Message Servers	266

Chapter 10

Project Deployment	267
Deployment Profiles	267
The Deployment Editor	268
Creating a Deployment Profile	269
Activating and Deactivating Deployment Profiles	272
Command-line Activation and Deactivation	273
Mapping Variables	274
Deploying Projects to Third-Party Servers	275
BEA WebLogic	275
IBM WebSphere	278

Chapter 11

Web Services	281
Overview	281
SeeBeyond Web Services	282
UDDI Repository	283
Building a Web Client	285
Object Type Definition	286
eInsight Business Process	290
eGate Project	292
Building a Web Server	294
eInsight Business Process	298
eGate Project	298

Appendix A

OTD Interfaces	300
The <i>Bean-like</i> Interface	300
The <i>Generic</i> Interface	301
Glossary	302
Index	309

Figures

Figure 1	eGate Integrator	24
Figure 2	eGate Integrator Implementation Model	26
Figure 3	Typical eGate Integrator System	27
Figure 4	SeeBeyond Enterprise Manager	29
Figure 5	Enterprise Designer	30
Figure 6	Connectivity Map Editor	31
Figure 7	Enterprise Manager Login	33
Figure 8	Enterprise Manager GUI	34
Figure 9	ICAN Monitor Launch Icon	35
Figure 10	ICAN Monitor/SRE Monitor Launch Icons	35
Figure 11	Documentation Tab	36
Figure 12	ICAN Monitor Interface - Initial	37
Figure 13	ICAN Monitor Interface - Environment	37
Figure 14	SRE Monitor	38
Figure 15	Enterprise Manager Home Page	39
Figure 16	SRE Monitor Initial Page	39
Figure 17	Add Registry/Repository Dialog Box	40
Figure 18	Viewing SRE Components	40
Figure 19	FromExternal e*Way Status (Not Running)	41
Figure 20	FromExternal e*Way Status (Running)	41
Figure 21	Refreshing the Explorer Tree	42
Figure 22	SeeBeyond Enterprise Designer	43
Figure 23	Login Dialog Box	45
Figure 24	Enterprise Explorer: Project Explorer View	49
Figure 25	Enterprise Explorer: Environment Explorer View	50
Figure 26	Connectivity Map Editor	51
Figure 27	OTD Editor	52
Figure 28	Collaboration Editor (Java)	53
Figure 29	Collaboration Editor (XSLT)	54
Figure 30	Environment Editor	55
Figure 31	Deployment Editor	56
Figure 32	Import Message Box	57

Figures

Figure 33	Import Manager Dialog Box (1)	58
Figure 34	Open File Dialog Box	59
Figure 35	Import Manager Dialog Box	60
Figure 36	Import Status Message Box	60
Figure 37	Export Manager Dialog Box (1a)	62
Figure 38	Export Manager Dialog Box (1b)	63
Figure 39	Export Manager Dialog Box (2)	64
Figure 40	Save As Dialog Box	64
Figure 41	Enter File Name Dialog Box (2)	65
Figure 42	Export Status Message Box	65
Figure 43	Impact Analyzer Dialog Box	67
Figure 44	Version Control - Check In Dialog Box	69
Figure 45	Version Control - Check Out Dialog Box	70
Figure 46	Version Control - History Dialog Box	71
Figure 47	Project Explorer	73
Figure 48	Repository Menu	75
Figure 49	Project Menu	76
Figure 50	Connectivity Map Menu	77
Figure 51	Connectivity Map Window	79
Figure 52	Linking JMS Topics	80
Figure 53	Service Component	81
Figure 54	External Application Drop-Down Menu	82
Figure 55	Scheduler Properties Dialog Box	83
Figure 56	Connection Icons in a Connectivity Map	84
Figure 57	Default Configuration Dialog Box	85
Figure 58	Project Variable Creation	87
Figure 59	Project Constant Creation	88
Figure 60	Variables and Constants Object Group	88
Figure 61	OTD Wizard Selection Dialog	91
Figure 62	OTD Wizard Selection: DTD Wizard	93
Figure 63	Select DTD File(s) Dialog Box	94
Figure 64	Select Document Elements Dialog Box	95
Figure 65	Select OTD Options Dialog Box	96
Figure 66	OTD Wizard Selection: WSDL Wizard	98
Figure 67	WSDL Wizard: Select WSDL Location	99
Figure 68	WSDL Wizard: Select WSDL File	100
Figure 69	WSDL Wizard: Select OTD Options	101
Figure 70	OTD Wizard Selection: XSD Wizard	103

Figures

Figure 71	XSD Wizard: Select XSD File(s)	104
Figure 72	Select Document Elements Dialog Box	105
Figure 73	Select OTD Options Dialog Box	106
Figure 74	OTD Wizard Selection: User-Defined OTD	108
Figure 75	Enter OTD Name	109
Figure 76	User-Defined OTD Node Properties	110
Figure 77	User-Defined OTD Element Properties	111
Figure 78	User-Defined OTD Field Properties	112
Figure 79	Node Type Property Options	113
Figure 80	Delimiter List Editor	115
Figure 81	Escape Type Delim	116
Figure 82	Terminal Type Property Example	117
Figure 83	Optional Property (Example 1)	118
Figure 84	Optional Property (Example 2)	118
Figure 85	Multiple Delimiter Example	119
Figure 86	Example User-Defined OTD	121
Figure 87	Activate Field	121
Figure 88	Delimiter List Editor (1)	122
Figure 89	Action Menu	122
Figure 90	Delimiter List Editor (2)	122
Figure 91	Delimiter List Editor (3)	123
Figure 92	Delimiter List Editor (4)	123
Figure 93	Delimiter List Editor (5)	124
Figure 94	Delimiter Specified	124
Figure 95	OTD Editor	125
Figure 96	OTD Tester	128
Figure 97	Test Panel Data Display	128
Figure 98	Select Data File	129
Figure 99	Object Elements and Values	129
Figure 100	Data Display: Refresh Icon	130
Figure 101	Status Data Display	130
Figure 102	Verbose Data Display	131
Figure 103	Initial Wizard Dialog	134
Figure 104	New Web Service: Operation Name	135
Figure 105	New Web Service: Input Message	136
Figure 106	New Web Service: Output Message	137
Figure 107	New Web Service: Auxiliary OTD	138
Figure 108	Existing Web Service: Select Operation	139

Figures

Figure 109	Existing Web Service: Select OTD	140
Figure 110	Collaboration Definition (Java) Context Menu	141
Figure 111	Collaboration Definition (Java) Properties	141
Figure 112	Collaboration Definition (Java) Properties	142
Figure 113	Collaboration Editor (Java)	143
Figure 114	Business Rules Editor	146
Figure 115	Business Rules Designer: Addition Method	149
Figure 116	Method-Access Nodes	150
Figure 117	Java Collaboration Method Palette Dialog Box	155
Figure 118	Expanded Method Box	155
Figure 119	Collapsed Method Box	155
Figure 120	Java Source Editor	156
Figure 121	Method Palette: Boolean Methods	157
Figure 122	Method Palette: Comparison Methods	158
Figure 123	Method Palette: Math Methods	160
Figure 124	Method Palette: Object Methods	162
Figure 125	CAST Dialog Box	163
Figure 126	InstanceOf Dialog Box (1)	163
Figure 127	Find Class Dialog Box	164
Figure 128	InstanceOf Dialog Box (2)	164
Figure 129	Method Palette: String Methods	165
Figure 130	Method Palette: Array Operation Methods	168
Figure 131	Method Palette: Operators	170
Figure 132	Collaboration Definition (Java) Context Menu	172
Figure 133	Version Control - Create Diff Dialog Box	172
Figure 134	Version Control - Merge Changes Dialog Box	173
Figure 135	Create a Variable Dialog Box	174
Figure 136	Find Class Dialog Box	175
Figure 137	Collaboration Definition (Java) with Variable	176
Figure 138	Call New Constructor Dialog Box	177
Figure 139	Collaboration Definition (Java) with Constructor	178
Figure 140	Variable Context Menu	179
Figure 141	Method Selection List Box	179
Figure 142	Try Icon	180
Figure 143	Catch Context Menu	180
Figure 144	Create a New Exception Variable Dialog Box	181
Figure 145	Find Class Dialog Box	181
Figure 146	Catch SQLException Rule	182

Figures

Figure 147	Exception Message	182
Figure 148	Project Context Menu: New File	183
Figure 149	File Selection Dialog Box	183
Figure 150	Import JAR File Icon	184
Figure 151	Add/Remove Jar File Dialog Box (1)	184
Figure 152	Add/Remove Jar File Dialog Box (2)	185
Figure 153	Call Java Method Dialog Box	185
Figure 154	Using the Third-Party Java Method	186
Figure 155	Call New Constructor Dialog Box	187
Figure 156	Constructor Example 1	188
Figure 157	Constructor Example 2	188
Figure 158	Validating a Collaboration Definition	189
Figure 159	Application Server Properties Dialog Box	190
Figure 160	Logical Host Context Menu	191
Figure 161	Application Server Context Menu	191
Figure 162	Java Debugger	192
Figure 163	File Menu	193
Figure 164	Attach to JVM Dialog Box	193
Figure 165	Collaboration Source Code Display	193
Figure 166	Breakpoint Example	194
Figure 167	Debug Menu	194
Figure 168	Stop in Method Dialog Box	195
Figure 169	Choose Exception Dialog Box	195
Figure 170	Break on Exception Dialog Box	195
Figure 171	Breakpoint Indicator	196
Figure 172	Stepping Into, Over, and Out Commands	197
Figure 173	Java Thread and Method Display	198
Figure 174	Local Variables Tab	199
Figure 175	Evaluate Local Variable	200
Figure 176	Evaluate Method	201
Figure 177	Save Debugger Session Dialog Box	202
Figure 178	Resume Debugger Session Dialog Box	202
Figure 179	Empty File Test	203
Figure 180	Alert Menu	204
Figure 181	Alert Severity Selection Window	204
Figure 182	Create Literal Dialog Box	204
Figure 183	Pass Alert Message to Object Argument	205
Figure 184	Logging Menu	206

Figure 185	Logging Level/Method Selection Window	206
Figure 186	Create Literal Dialog Box	207
Figure 187	Pass Log Message to Object Argument	207
Figure 188	Collaboration Wizard (XSLT) Dialog Box	210
Figure 189	New Web Service: Operation Name	211
Figure 190	New Web Service: Input Message	212
Figure 191	New Web Service: Output Message	213
Figure 192	Existing Web Service: Select Operation	214
Figure 193	Collaboration Editor (XSLT)	215
Figure 194	XSLT Collaboration Method Palette Dialog Box	217
Figure 195	Expanded Method Box	217
Figure 196	Collapsed Method Box	217
Figure 197	Method Palette: Operator Methods	218
Figure 198	Method Palette: String Methods	221
Figure 199	Method Palette: Number Methods	224
Figure 200	Method Palette: Boolean Methods	226
Figure 201	Method Palette: Nodes Methods	228
Figure 202	Method Palette: UAN Extension Methods	231
Figure 203	Collaboration Definition (XSLT) Context Menu	238
Figure 204	Version Control - Create Diff Dialog Box	238
Figure 205	Version Control - Merge Changes Dialog Box	239
Figure 206	Enterprise Explorer: Environment Explorer View	240
Figure 207	Repository Menu	242
Figure 208	Environment Menu	243
Figure 209	Logical Host Menu	244
Figure 210	Logical Host Menu with Third-Party Servers	245
Figure 211	Environment Editor	246
Figure 212	Environmental Constants Panel	247
Figure 213	Logical Hosts	248
Figure 214	Startup Sequence	249
Figure 215	Logical Host Configuration Properties	250
Figure 216	Top-level IS Configuration Properties	251
Figure 217	Web Container Configuration Properties	253
Figure 218	Default Web Server Properties	253
Figure 219	Profiling Configuration Properties	255
Figure 220	Security Configuration Properties	256
Figure 221	eInsight Engine Configuration Properties	257
Figure 222	Application Manager Configuration Properties	258

Figures

Figure 223	Integration Server Configuration Properties	259
Figure 224	Oracle JDBC Connection Pool Properties	261
Figure 225	eGate Integrator Implementation Model	267
Figure 226	Deployment Editor Window	268
Figure 227	Web Client Example Project	269
Figure 228	Web Client Example Environment	270
Figure 229	Example Deployment Profile (1)	270
Figure 230	Example Deployment Profile (2)	271
Figure 231	Example Deployment Profile (3)	271
Figure 232	Activate Dialog Box	272
Figure 233	Logical Host Context Menu - Apply	272
Figure 234	Activate Dialog Box	273
Figure 235	Deployment Profile Mappings	274
Figure 236	Project Variable Value Entry	274
Figure 237	WebLogic Deployment (1)	276
Figure 238	WebLogic Deployment (2)	276
Figure 239	WebLogic Deployment Verification	277
Figure 240	WebSphere Deployment (1)	278
Figure 241	WebSphere Deployment (2)	279
Figure 242	WebSphere Deployment Verification	280
Figure 243	SeeBeyond UDDI Repository	283
Figure 244	Example Web Service WSDL File	283
Figure 245	Microsoft Visual Studio Example	284
Figure 246	File Destination Dialog Box	285
Figure 247	Select WSDL Wizard	286
Figure 248	Select File Location	287
Figure 249	Select WSDL File	288
Figure 250	Select External Server	289
Figure 251	Web Client Business Process	290
Figure 252	Web Client Business Process <i>Receive</i> Rule	290
Figure 253	Web Client Business Process <i>Write</i> Rule	290
Figure 254	Sample WSDL File	291
Figure 255	Map Business Process	292
Figure 256	Web Client Connectivity Map	292
Figure 257	Web Client Example Project	293
Figure 258	Project Deployment	293
Figure 259	File Destination Dialog Box	294
Figure 260	Select WSDL Wizard	295

Figures

Figure 261	Select File Location	296
Figure 262	Select WSDL File	296
Figure 263	Select External Client	297
Figure 264	Web Server Business Process	298
Figure 265	Connectivity Map	298
Figure 266	Web Server Example Project	298
Figure 267	Project Deployment	299

Tables

Table 1	Writing Conventions	22
Table 2	Enterprise Manager - Pages	34
Table 3	Enterprise Manager - Control Tabs	34
Table 4	Document Categories	36
Table 5	ICAN Monitor Interface - Details Tabs	37
Table 6	SRE Monitor Explorer Icons	38
Table 7	File Menu Options	46
Table 8	Tools Menu Options	46
Table 9	View Menu Options	47
Table 10	Window Menu Options	47
Table 11	Help Menu Options	47
Table 12	Enterprise Designer Toolbar Icons	48
Table 13	Browser Buttons	48
Table 14	Impact Analyzer Command Buttons	68
Table 15	Project Icons	74
Table 16	Repository Menu Options	75
Table 17	Project Menu Options	76
Table 18	Connectivity Map Menu Options	77
Table 19	Connectivity Map Toolbar Icons	80
Table 20	Configuration Dialog Box Toolbar Buttons	86
Table 21	OTD Wizard Navigation Buttons	92
Table 22	DTD OTD Options	96
Table 23	XSD OTD Options	106
Table 24	Node Properties	110
Table 25	Element Properties	111
Table 26	Field Properties	112
Table 27	Node Type Property Options	113
Table 28	Delimiter Properties and Value Options	115
Table 29	Escape Delimiters	119
Table 30	OTD Editor Toolbar Icons	126
Table 31	Wizard Navigation Buttons	133
Table 32	Java Toolbar Icons	144

Tables

Table 33	Business Rules Toolbar Buttons	147
Table 34	Rules for Placement of Subnodes	148
Table 35	Method Access Nodes	150
Table 36	Business Rules Designer Toolbar Icons	151
Table 37	Boolean Collaboration Methods (Java)	157
Table 38	Comparison Collaboration Methods (Java)	158
Table 39	Math Collaboration Methods (Java)	160
Table 40	Object Collaboration Methods (Java)	162
Table 41	String Collaboration Methods (Java)	165
Table 42	Array Operation Collaboration Methods (Java)	168
Table 43	Operators (Java)	170
Table 44	Wizard Navigation Buttons	209
Table 45	XSLT Toolbar Icons	216
Table 46	Operator Collaboration Methods (XSLT)	218
Table 47	String Collaboration Methods (XSLT)	221
Table 48	Number Collaboration Methods (XSLT)	224
Table 49	Boolean Collaboration Methods (XSLT)	226
Table 50	Nodes Collaboration Methods (XSLT)	228
Table 51	UAN Extension Collaboration Methods (XSLT)	232
Table 52	UAN Extension Collaboration Method Parameter Definitions	236
Table 53	Environment Icons	241
Table 54	Repository Menu Options	242
Table 55	Environment Menu Options	243
Table 56	Logical Host Menu Options	244
Table 57	Environmental Constants Panel Icons	247
Table 58	Logical Host Configuration Properties List	250
Table 59	Top-level IS Configuration Properties List	251
Table 60	Web Container Configuration Properties List	253
Table 61	Default Web Server Properties List	254
Table 62	Profiling Configuration Properties List	255
Table 63	Security Realm Configuration Properties List	256
Table 64	Application Manager Configuration Properties List	258
Table 65	Integration Server Configuration Properties List	259
Table 66	Oracle JDBC Connection Pool Properties List	261
Table 67	Deployment Toolbar Buttons	268
Table 68	Terminology Cross-Reference	308

Introduction

This chapter describes the general purpose, scope, and organization of this document, and also provides references to additional sources of relevant information.

1.1 Purpose and Scope

This User's Guide provides general information about the features and operation of SeeBeyond® eGate Integrator in creating and deploying eGate Projects. For information on eGate Integrator system management, see the *eGate Integrator System Administration Guide*.

Note: *Any operational 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 User's Guide is intended for personnel who are involved in integrating software applications using eGate Integrator. To a large extent, these are individuals who will be using the eGate Enterprise Designer to build eGate Projects to accomplish this task. This guide also provides a basic overview of the eGate product for those attempting to gain a general understanding of how eGate Integrator works.

This guide assumes that the reader is an experienced computer user, familiar with Windows-style GUI operations, and also has an in-depth understanding of the operating system(s) on which eGate Integrator will be installed.

Note: *The eGate Integrator graphical user interface (GUI) runs only on Windows. Refer to the SeeBeyond ICAN Suite Installation Guide for a list of operating systems on which eGate Integrator itself can run.*

1.3 Organization of Information

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

- **Chapter 1 “Introduction”** describes the purpose of this User’s Guide, including writing conventions and a list of related documents.
- **Chapter 2 “System Overview”** provides an overview of the general structure, architecture, and operation of eGate Integrator, and its place within the SeeBeyond ICAN Suite.
- **Chapter 3 “Enterprise Manager”** provides a detailed overview of the Enterprise Manager, including its structure and operation.
- **Chapter 4 “Enterprise Designer”** provides a detailed overview of the Enterprise Designer, including its structure and operation.
- **Chapter 5 “eGate Projects”** explains how to create a Connectivity Map and use the Configuration Editor to modify eWay and JMS connections between Connectivity Map components.
- **Chapter 6 “Object Type Definitions”** describes how to create Object Type Definitions (OTDs).
- **Chapter 7 “Collaboration Definitions (Java)”** describes how to build Collaboration Definitions described in Java.
- **Chapter 8 “Collaboration Definitions (XSLT)”** describes how to build Collaboration Definitions described in XSLT.
- **Chapter 9 “Environments”** explains how to create and populate eGate Environments, and how to configure and start Logical Hosts.
- **Chapter 10 “Project Deployment”** explains how to create and activate Deployment Profiles.
- **Chapter 11 “Web Services”** describes how to use eGate Integrator in concert with other ICAN Suite components to create Web services.
- **Appendix A “OTD Interfaces”** describes the two types of OTD frameworks.

In addition, the [Glossary](#) on page 302 lists various terms used in this User’s Guide.

1.4 Writing Conventions

The following writing conventions are observed throughout this document.

Table 1 Writing Conventions

Text	Convention	Example
Button, file, icon, parameter, variable, method, menu, and object names.	Bold text	<ul style="list-style-type: none"> ▪ Click OK to save and close. ▪ From the File menu, select Exit. ▪ Select the logicalhost.exe file. ▪ Enter the timeout value. ▪ Use the getClassName() method. ▪ Configure the Inbound File eWay.
Command line arguments and code samples	Fixed font. Variables are shown in <i>bold italic</i> .	<code>bootstrap -p <i>password</i></code>
Hypertext links	Blue text	http://www.seebeyond.com

Additional Conventions

Windows Systems

For the purposes of this guide, all references to **Windows** apply to Microsoft Windows Server 2003, Windows XP, and Windows 2000.

Path Name Separator

This guide uses a backslash (\) as the separator within path names. If you are working on a UNIX system, please substitute a forward slash (/).

1.5 Supporting Documents

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

- *eGate Integrator JMS Reference Guide*
- *eGate Integrator Release Notes*
- *eGate Integrator System Administration Guide*
- *eGate Integrator Tutorial*
- *SeeBeyond ICAN Suite Deployment Guide*
- *SeeBeyond ICAN Suite Installation 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 SeeBeyond documentation is included in the *SeeBeyond ICAN Suite Primer*.

The documentation for the SeeBeyond ICAN Suite is distributed as a collection of online documents, which can be accessed through the Enterprise Manager (see **Documentation** on page 36). These documents are in Adobe Acrobat format, which requires that Acrobat Reader be installed on your computer. Acrobat Reader can be from Adobe Systems as a free download from the following URL:

<http://www.adobe.com>

1.6 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>

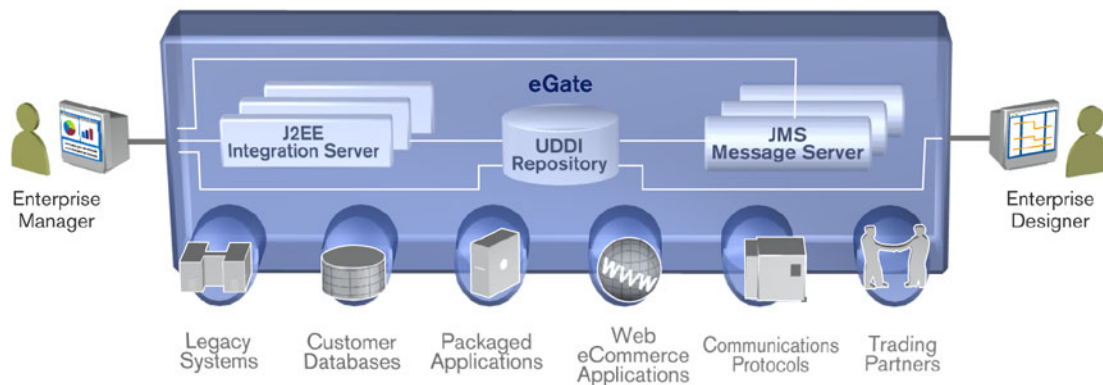
System Overview

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

2.1 Introduction

SeeBeyond® eGate™ Integrator is a fully J2EE certified and Web services-based, distributed integration platform that serves as the foundation of the SeeBeyond Integrated Composite Application Network™ (ICAN™) Suite. eGate Integrator provides the core integration platform, comprehensive systems connectivity, guaranteed messaging and robust transformation capabilities while providing a unified, single sign-on environment for integration development, deployment, monitoring and management. eGate Integrator supports portability of integrations across common J2EE application servers through a completely open, J2EE-certified and Web services-based architecture.

Figure 1 eGate Integrator



As shown in Figure 1, the heart of eGate Integrator is the Repository, which is a comprehensive store of information common to the entire enterprise. An integrated UDDI-compliant server allows publication and discovery of Web services. The run-time environment employs J2EE-compliant integration servers as operational engines and JMS-compliant message servers for the propagation of messages. The flexibility of the

eGate system allows the option of deployment to a SeeBeyond run-time environment or to third-party application servers, across a distributed network of hardware platforms.

Enterprise Manager provides a unified, browser-based framework for managing all aspects of the run-time environment, as well as installing and updating all ICAN Suite components. Enterprise Designer provides a unified, graphical development environment for integrating systems and developing composite applications using Web services.

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.

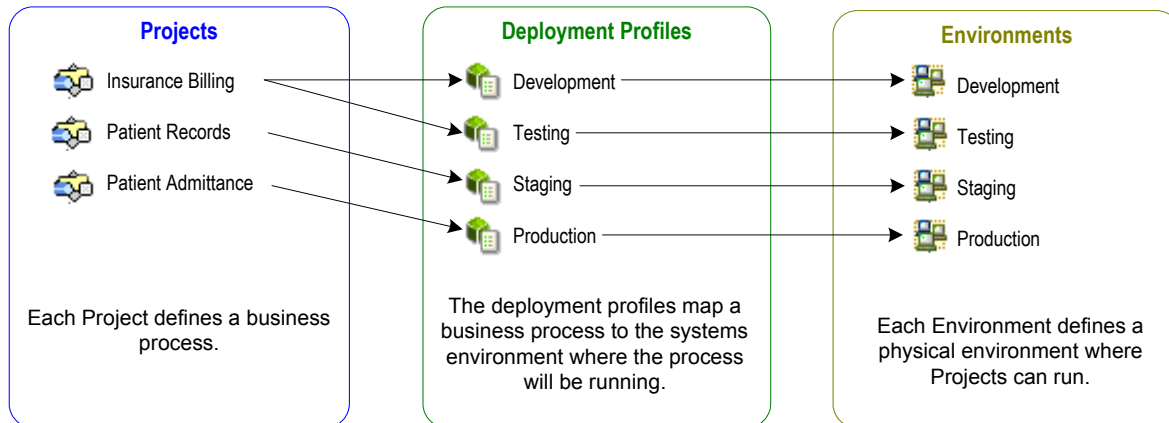
Projects are created using tools contained within Enterprise Designer and, once deployed, can be run and monitored using Enterprise Manager. Projects can also be set up to be run from the business process level using the SeeBeyond eInsight Business Process Manager, if that product is also installed.

Projects are run within individual sets of system definitions, referred to as Logical Hosts. These 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 2 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, implementing an integration Project using eGate Integrator includes the following steps:

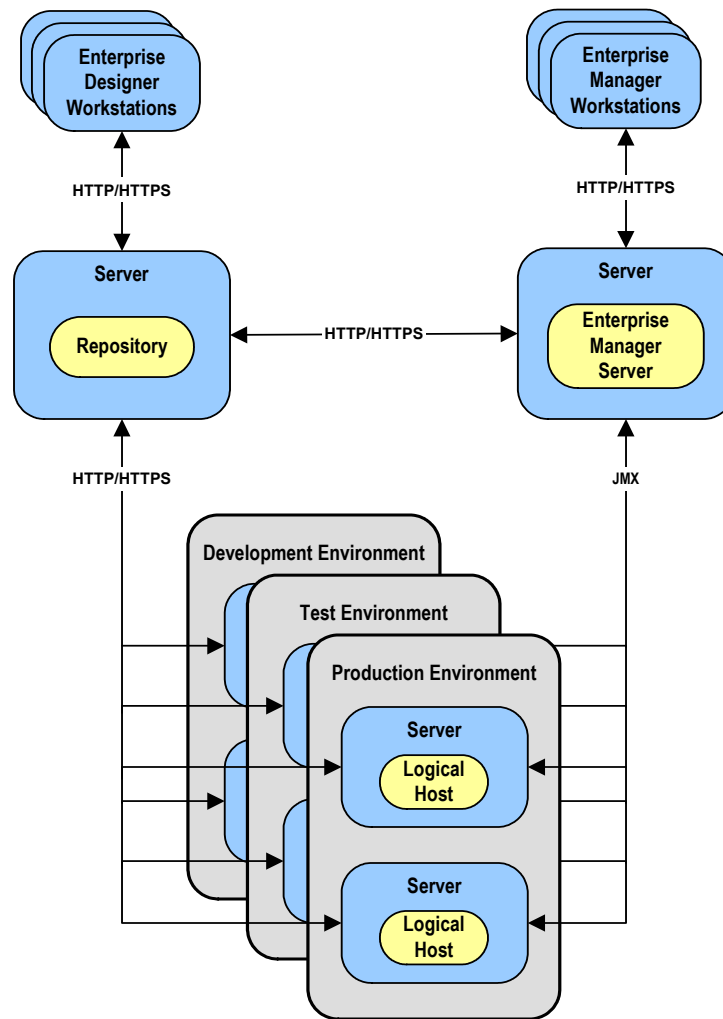
- 1 **Design your Project.**
- 2 **Define your Environments.**
- 3 **Create your Deployment Profiles.**
- 4 **Deploy the eGate Project.**

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

2.3 System Architecture

SeeBeyond's eGate Integrator employs a versatile architecture that is ideally suited to 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 3 shows an example system implementation that is highly distributed.

Figure 3 Typical eGate Integrator System



In Figure 3, the servers shown in blue are hardware platforms; the entities shown in yellow are software.

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 run time 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 3, a single Repository serves the entire enterprise. This common Repository is used for development, testing, and production purposes. Communication between the Repository and other ICAN components can be configured to use either HTTP or HTTPS. The Repository makes Web Services available via a UDDI-compliant server. For more information on Web Services capability, see [Web Services](#) on page 281.

The Enterprise Designer and Enterprise Manager clients can communicate with the Repository through a firewall.

2.3.2 Environments

An eGate Environment represents the total 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 system definitions. Each definition must include one or more **integration servers** such as the SeeBeyond Integration Server, which are the engines that run eGate Collaborations and eWays, and one or more **message servers** such as the SeeBeyond JMS IQ Manager, which manage JMS topics (publish-and-subscribe messaging) and queues (point-to-point messaging). Each collection of integration servers and message servers, plus additional software modules, comprise what is known as a Logical Host.

- **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 3, the production environment is split across two hardware platforms, each supporting 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 3 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

eGate Integrator uses two basic graphical user interfaces (GUIs), each of which addresses a different set of users. Enterprise Manager is an interface used by the entire ICAN Suite, the primary users of which are system administrators. Enterprise Designer is used by personnel who are involved in defining a software system for integrating the various enterprise applications using eGate Integrator and eInsight Business Process Manager.

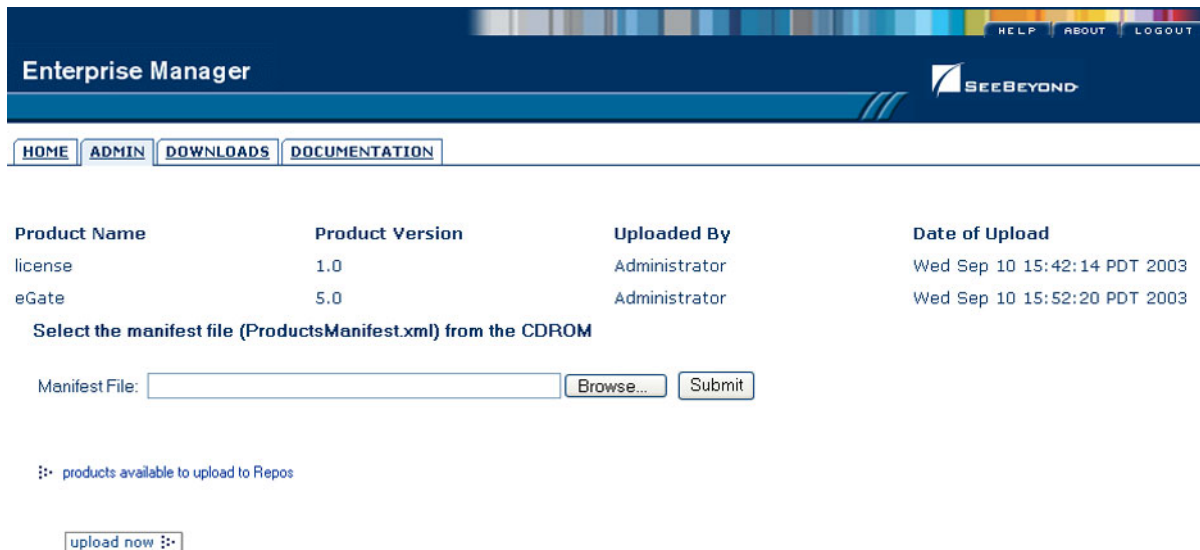
2.4.1 Enterprise Manager

Enterprise Manager is a Web-based application that works within Microsoft Internet Explorer. It is used throughout the SeeBeyond ICAN Suite for:

- Installing and updating ICAN Suite products
- Accessing ICAN Suite product documentation
- Managing and monitoring runtime components

The Enterprise Manager is described in [Enterprise Manager on page 32](#). Figure 4 shows the Enterprise Manager **Admin** page, used in product installation.

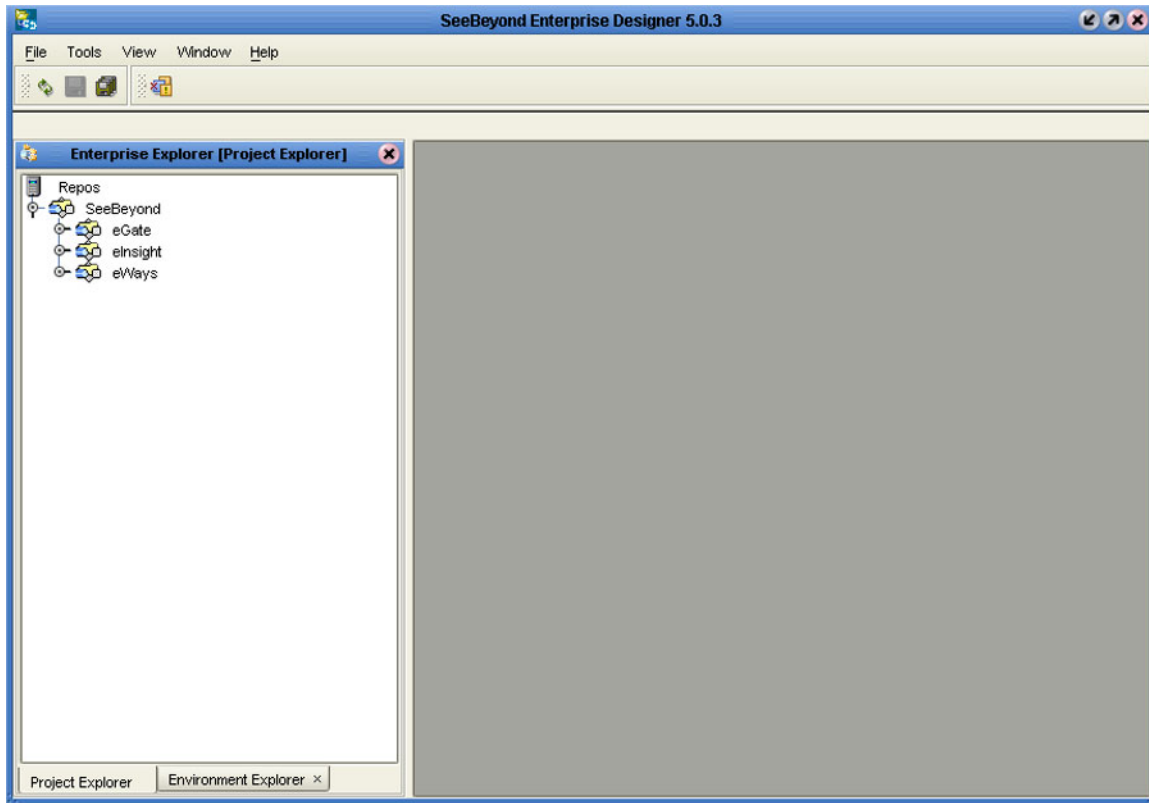
Figure 4 SeeBeyond Enterprise Manager



2.4.2 Enterprise Designer

The SeeBeyond Enterprise Designer 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. Enterprise Designer is also used by other components of the ICAN Suite.

Figure 5 Enterprise Designer

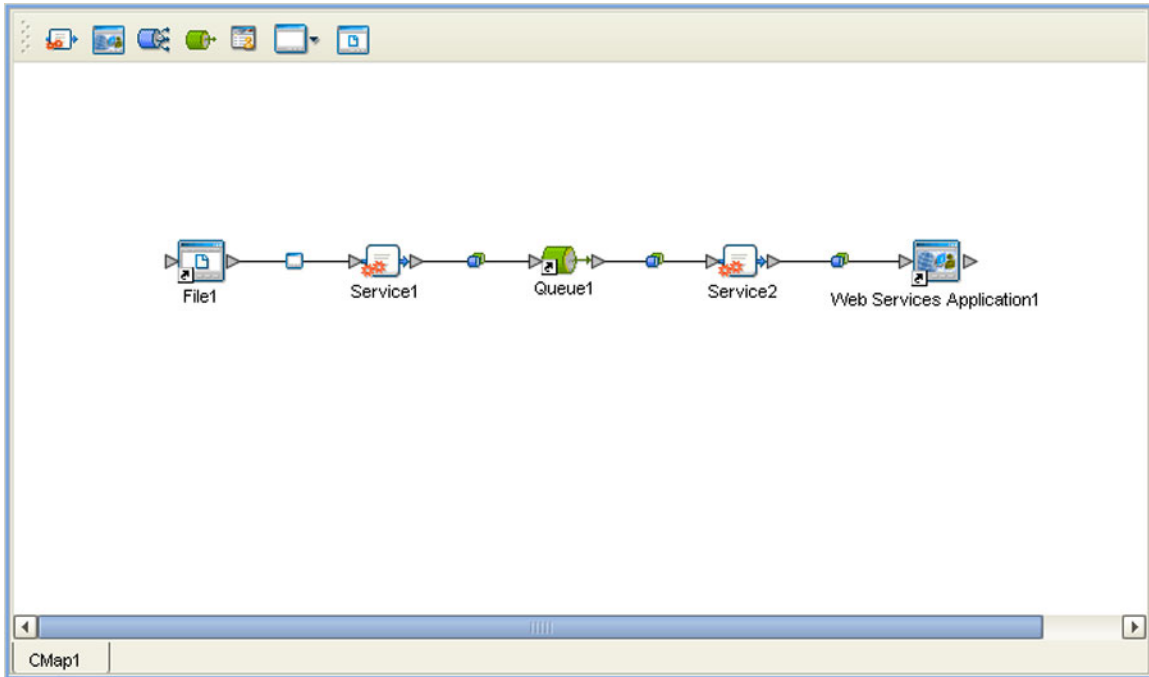


The major features of the Enterprise Designer are the Enterprise Explorer on the left, and an editor panel on the right—which is initially blank. The Enterprise Explorer follows the familiar Windows Explorer format, displaying a tree structure. The editor panel displays a variety of editors, depending upon what component is selected in the Enterprise Explorer. These editors include, for example:

- Connectivity Map Editor
- OTD Editor
- Java Collaboration Editor
- XSLT Collaboration Editor
- Environment Editor
- Deployment Editor

The Connectivity Map Editor (see Figure 6) provides a graphic example of one of these editors, in which logical components of a Project can be created and connected. eGate uses Connectivity Maps to intuitively configure the end-to-end flow of messages within an integration. The integration developer can to drag and drop the various Collaborations, Intelligent Queues and external-system eWay adapters onto the Connectivity Map canvas and link them together to specify message flow. The features and usage of the Connectivity Map Editor are described in [eGate Projects](#) on page 72.

Figure 6 Connectivity Map Editor



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 43](#).

Enterprise Manager

This chapter provides an introduction to the ICAN Suite Enterprise Manager.

3.1 Overview

Enterprise Manager is a Web-based interface with which you can install and update eGate Integrator, and monitor and manage deployed eGate components.

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

3.1.1 Installing and Updating eGate

eGate Integrator components 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. For information on installing and updating eGate components, see the *SeeBeyond ICAN Suite Installation Guide*.

3.1.2 Monitoring and Managing eGate

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

- **The ICAN Monitor** on page 37 describes the basic features of the ICAN Monitor interface. ICAN Monitor usage for specific tasks is described in the *eGate Integrator System Administration Guide*.
- **The SRE Monitor** on page 38 describes an optional facility that allows you to monitor and manage e*Gate 4.x schemas in eGate 5.0, using the Schema Runtime Environment.

3.2 Starting Enterprise Manager

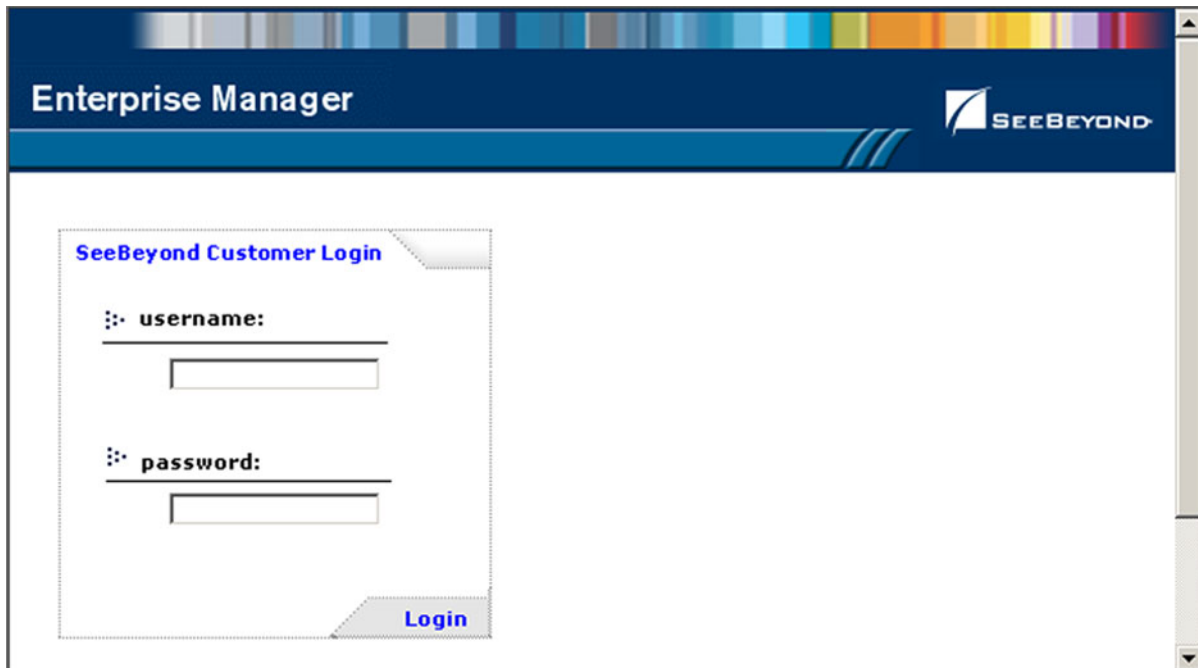
To start the Enterprise Manager

- 1 Launch Internet Explorer.
- 2 Enter **http://hostname:portnumber** in the **Address** box to display the SeeBeyond Customer Login window shown in Figure 7.

Note: The *hostname* is the fully-qualified, network-addressable 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. See the SeeBeyond ICAN Suite Installation Guide.

Important: The TCP/IP host name must be alphanumeric.

Figure 7 Enterprise Manager Login



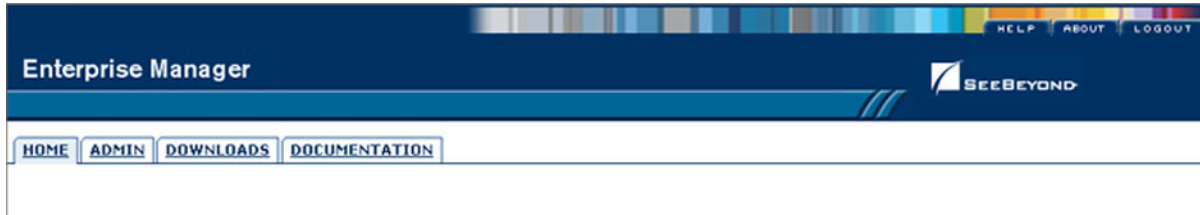
The screenshot shows a web browser window titled "Enterprise Manager" with the SeeBeyond logo in the top right corner. The main content area displays a "SeeBeyond Customer Login" form. The form includes two input fields: one for "username:" and one for "password:". Below the password field is a "Login" button. The form is enclosed in a light gray border with a drop shadow effect.

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

3.3 The Enterprise Manager Interface

Once you have logged in, you see the full Enterprise Manager user interface (see Figure 8).

Figure 8 Enterprise Manager GUI



The Enterprise Manager is organized into four pages, as described in the following table. Each page is accessed by clicking the appropriate tab.

Table 2 Enterprise Manager - Pages

Tab	Function
Home	The Home page is used for accessing the ICAN Monitor, which is the main subject of this chapter. See Home on page 35.
Admin	The Administration page is used for installing and updating ICAN components. See the <i>SeeBeyond ICAN Suite Installation Guide</i> for information.
Downloads	The Downloads page is used in installing and updating ICAN components. See the <i>SeeBeyond ICAN Suite Installation Guide</i> for information.
Documentation	The Documentation page is used for accessing ICAN Suite documentation. See Documentation on page 36, and the following <i>Note</i> .

Note: You must download the documentation SAR files from the installation disk before you can access any documents using the Documentation page (see the *SeeBeyond ICAN Suite Installation Guide*).

There are also three small tabs in the upper-right corner of the Enterprise Manager, which are described in the following table.

Table 3 Enterprise Manager - Control Tabs

Tab	Function
Help	The Help tab provides access to the online help system.
About	The About tab displays the installed version of the product (this tab is not present on the Documentation page).
Logout	The Logout tab logs you out of the Enterprise Manager and returns you to the Login page.

3.3.1 Home

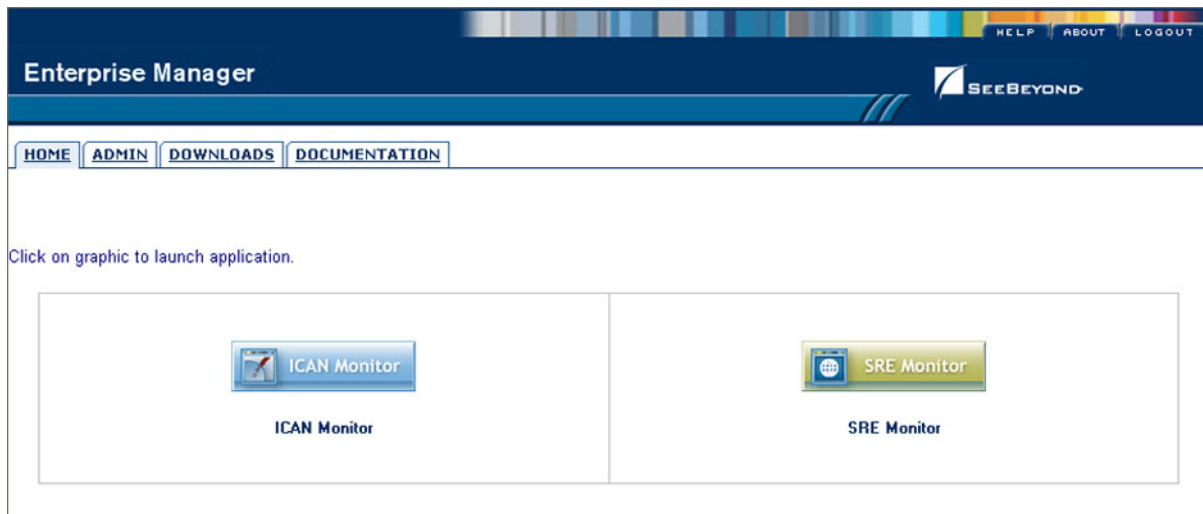
The Enterprise Manager's **Home** tab (see Figure 9) contains a link to the ICAN Monitor. Click the **Monitor** icon to launch the ICAN Monitor (see [The ICAN Monitor](#) on page 37).

Figure 9 ICAN Monitor Launch Icon



If the SRE Monitor (see [The SRE Monitor](#) on page 38) is installed, its icon is also displayed on this page, as shown in Figure 10.

Figure 10 ICAN Monitor/SRE Monitor Launch Icons



Note: *If connection problems are encountered, close all Internet Explorer windows and retry.*

3.3.2 Documentation

The **Documentation** tab (see Figure 11) contains links to the latest versions of the SeeBeyond ICAN documentation in Adobe Acrobat (PDF) format, and also any sample Project files (in ZIP format). Shown is the current set for eGate Integrator.

Figure 11 Documentation Tab



The provided documentation is organized into the major categories listed in Table 4.

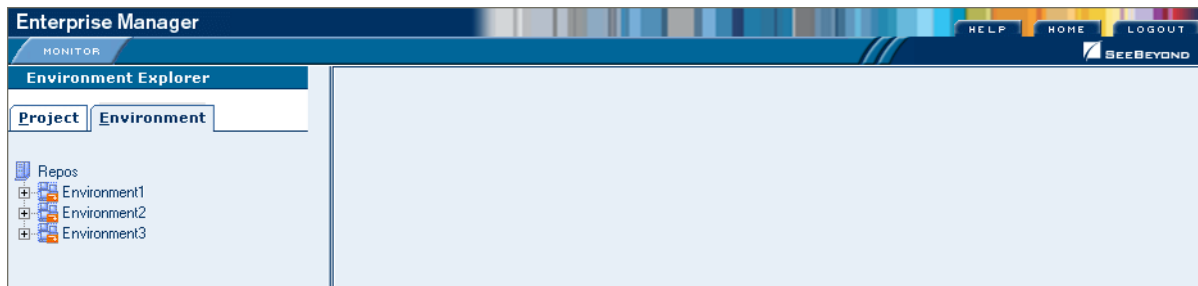
Table 4 Document Categories

Category	Contents
Readme Files	Includes information regarding the latest operating system and hardware requirements, cautions and caveats regarding known issues, and supplementary information arising after the publication of other documentation.
Products	Documentation regarding SeeBeyond core products, such as eGate Integrator and eInsight Business Process Manager. Also includes example Project files, if available.
Add-ons	Documentation regarding optional, ancillary products such as eWays and OTD Libraries.

3.4 The ICAN Monitor

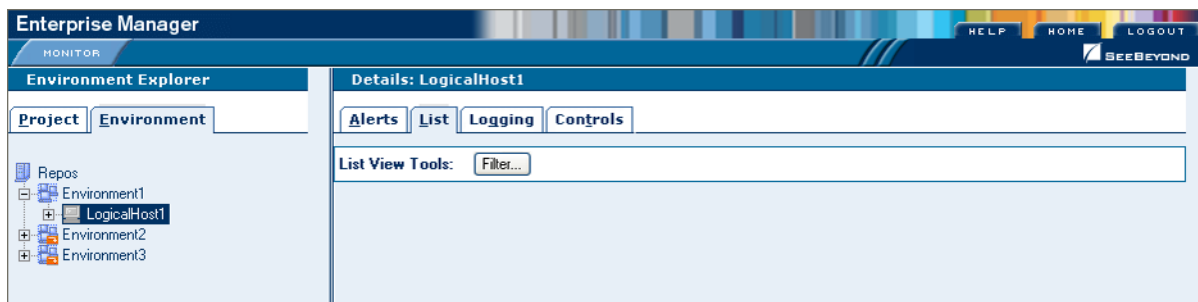
The ICAN Monitor has structure similar to that of the Enterprise Designer, with an Explorer panel on the left and a Details panel on the right. Initially, the Details panel is blank as shown in Figure 12.

Figure 12 ICAN Monitor Interface - Initial



Like the Enterprise Manager itself, the ICAN Monitor's **Details** area is organized into sections represented by tabs (see Table 5). Which tabs are present depends upon the component selected in the Explorer. For example, selecting the Logical Host displays the Monitor page shown in Figure 13.

Figure 13 ICAN Monitor Interface - Environment



At times, the Details panel will have two parts, to display an additional level of information. In this case, different tabs will be displayed in the upper and lower panels.

Table 5 ICAN Monitor Interface - Details Tabs

Tab	Function
Alerts	Displays all alerts for the component selected in the Explorer.
List	Displays a list presenting information about the component selected in the Explorer.
Logging	Displays all log messages for the component selected in the Explorer.
Controls	Displays controls that allow an Administrator to intervene in the run-time process and perform tasks such as starting and stopping components.

Note: See the *eGate Integrator System Administration Guide* for detailed information regarding Monitor usage.

3.5 The SRE Monitor



eGate 5.0 provides a completely different operating environment from earlier versions of the product (e*Gate). The Schema Runtime Environment (SRE) allows you to use schemas developed for e*Gate 4.x with eGate 5.0 by providing the necessary environmental components. Instructions for installing and using the SRE are contained in the SeeBeyond documentation for the SRE.

The SRE Monitor enables you to manage e*Gate 4.x schemas running in the Schema Runtime Environment from within eGate 5.0. The SRE Monitor interface generally resembles the ICAN Monitor, but differs somewhat in detail (see Figure 14). Only the Environment Explorer is displayed, which has two additional icons in the upper left corner; these are described in Table 6.

Figure 14 SRE Monitor



Table 6 SRE Monitor Explorer Icons

Icon	Function
	<p>The Add Registry/Repository icon displays the Add Registry/Repository dialog box, in which you specify the desired Registry or Repository's name and port, and your user name and password.</p> <div data-bbox="500 1278 1263 1627" style="border: 1px solid black; padding: 5px; margin: 10px auto; width: fit-content;"> <p style="text-align: center; margin: 0;">Add Registry/Repository</p> <p>Username: <input style="width: 150px;" type="text"/></p> <p>Password: <input style="width: 150px;" type="password"/></p> <p>Host Name: <input style="width: 300px;" type="text"/></p> <p>Port: <input style="width: 80px;" type="text" value="23001"/></p> <p style="text-align: center;"> <input type="button" value="Add Registry/Repository"/> <input type="button" value="Reset"/> </p> </div>
	<p>The Refresh Registry icon refreshes the SRE Registry and the Explorer tree following changes to component status.</p>

Instructions for installing the SRE Monitor are contained in the *SeeBeyond ICAN Suite Installation Guide*.

3.5.1 Starting and Using the SRE Monitor

To start the SRE Monitor

- 1 Start the SRE Monitor server, as described in the *SeeBeyond ICAN Suite Installation Guide*.
- 2 Launch Internet Explorer and access Enterprise Manager, as described in [Starting Enterprise Manager](#) on page 33.
- 3 On the Enterprise Manager Home page, click the **SRE Monitor** icon shown in Figure 15 to display the initial page of the SRE Monitor (see Figure 16).

Figure 15 Enterprise Manager Home Page

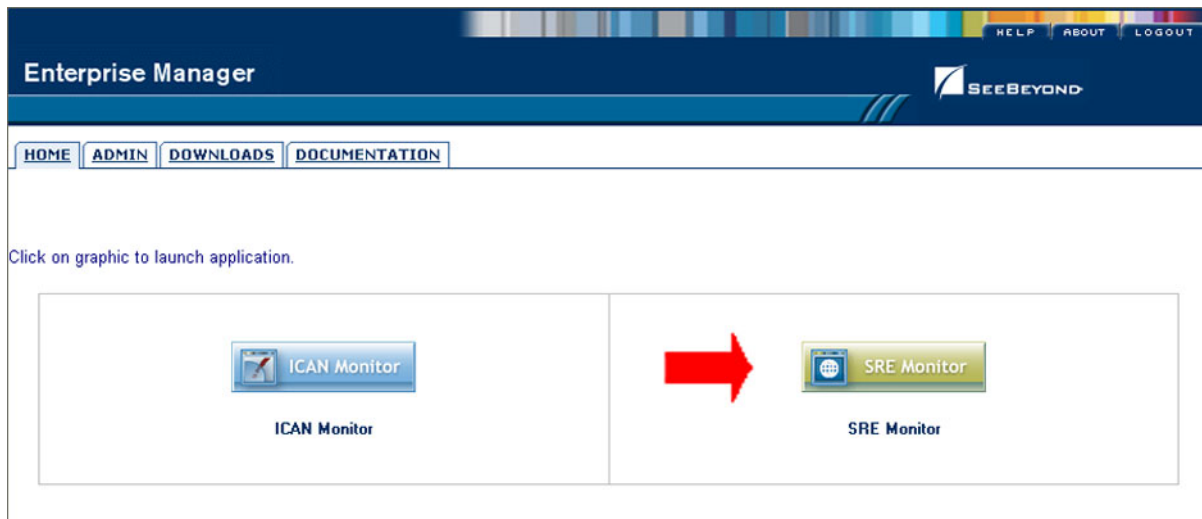
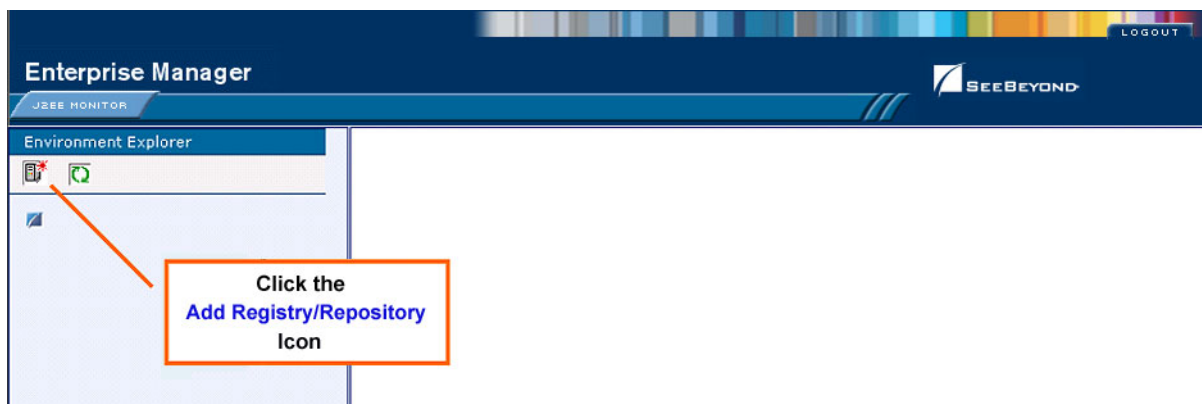
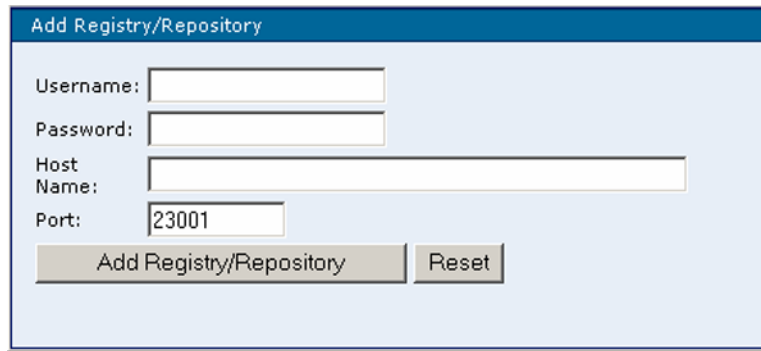


Figure 16 SRE Monitor Initial Page



- 4 Click the **Add Registry/Repository** icon in the upper left corner of the Explorer. This displays the dialog box shown in Figure 17.

Figure 17 Add Registry/Repository Dialog Box



The dialog box titled "Add Registry/Repository" contains the following fields and buttons:

- Username:
- Password:
- Host Name:
- Port:
- Buttons: "Add Registry/Repository" and "Reset"

- 5 Enter your login ID and password, and the Repository Host Name and Port, and click **Add Registry/Repository**.

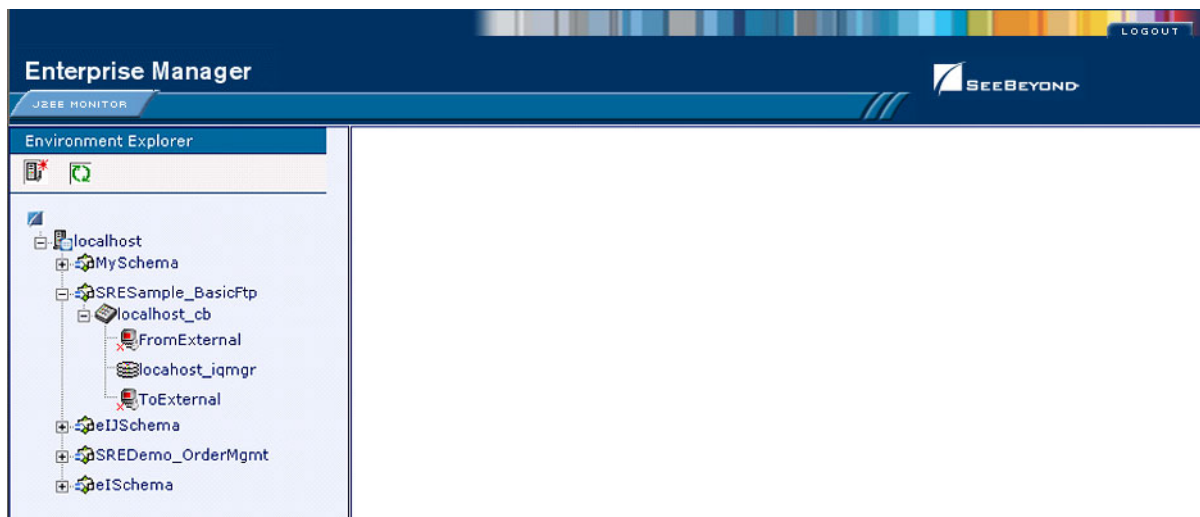
Note: The **Host Name** is the host name of the server where you installed the e*Gate 4.x Registry, and the **Port** is the number of the port you entered during installation of the e*Gate 4.x Registry. See the SeeBeyond ICAN Suite Installation Guide.

Important: The Host Name must be composed of alphanumeric characters only.

To use the SRE Monitor

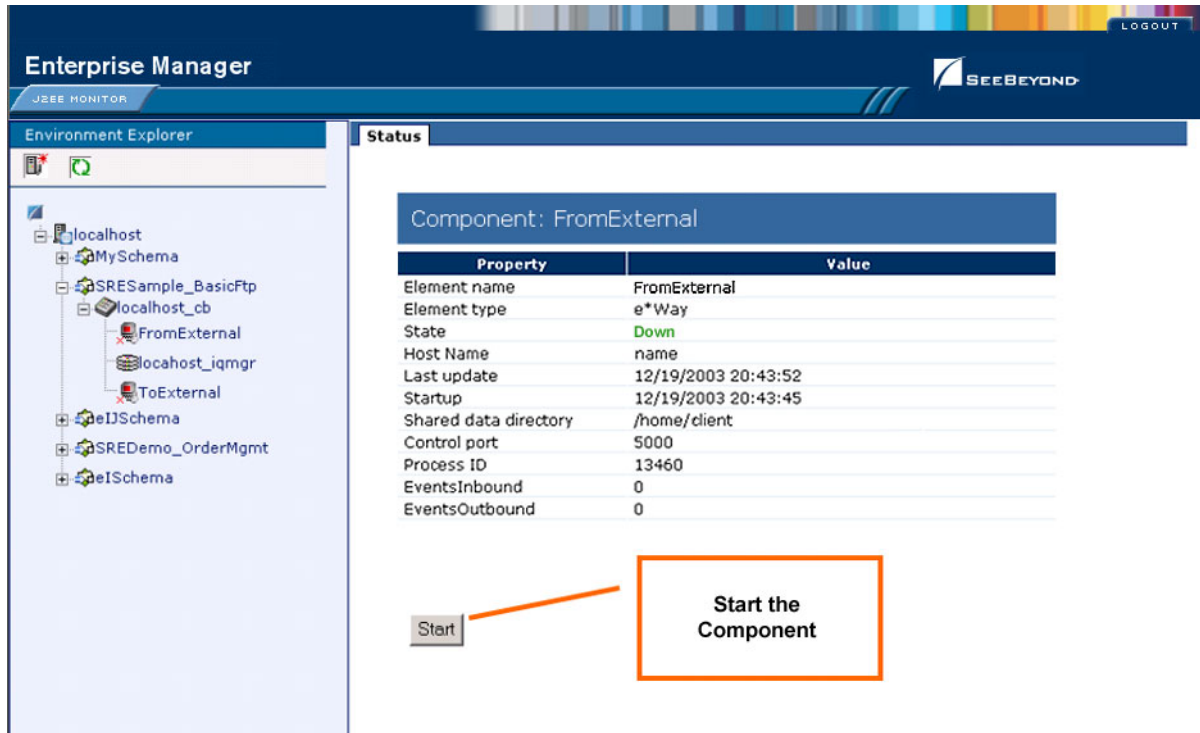
- 1 After adding the Registry, expand the Control Broker in the Explorer tree to view the SRE components, as shown in Figure 18.

Figure 18 Viewing SRE Components



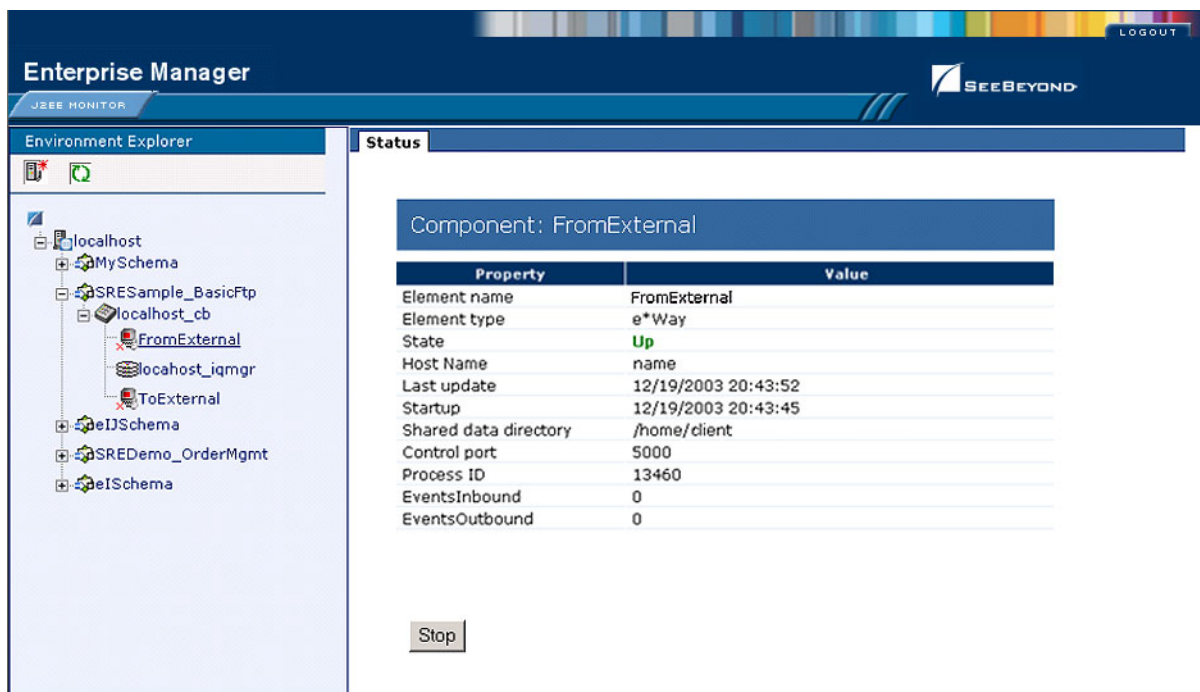
- 2 Click on a component to display its status. As an example, the status of the *FromExternal* e*Way is shown in Figure 19. You can start the component by clicking the **Start** button, which becomes a **Stop** button when the component is running.

Figure 19 FromExternal e*Way Status (Not Running)



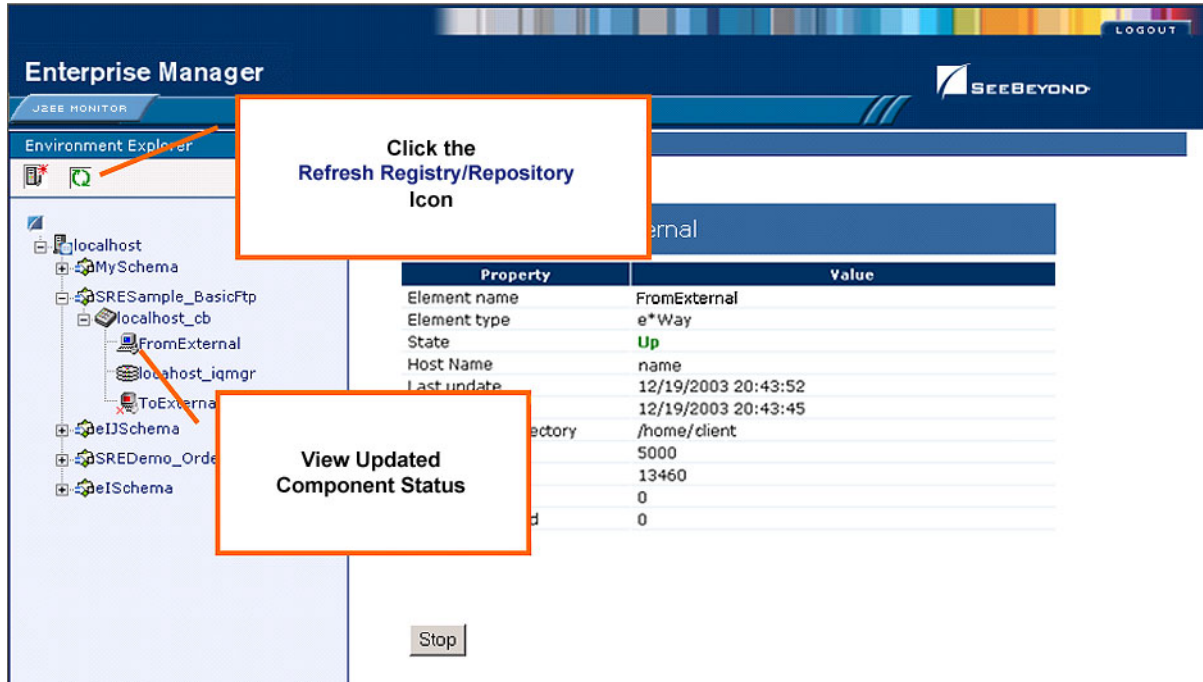
- 3 The status page is updated after you start or stop the component, as shown in Figure 20.

Figure 20 FromExternal e*Way Status (Running)



- 4 To refresh the Explorer tree, click the **Refresh Repository** icon, as shown in Figure 21.

Figure 21 Refreshing the Explorer Tree



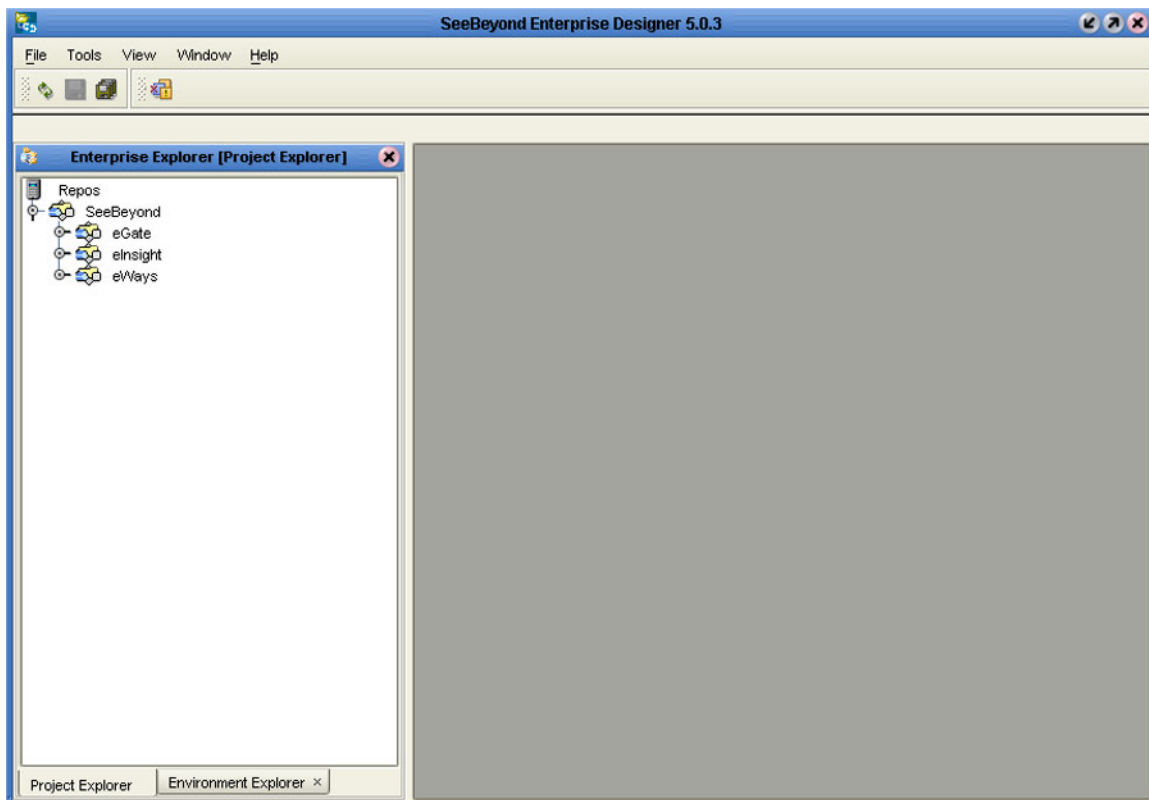
Enterprise Designer

This chapter describes the various features of the Enterprise Designer.

4.1 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 22), you can develop Projects to process and route data through an eGate Integrator system.

Figure 22 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, which are described in the following sections of this chapter:

- [Project Explorer](#) on page 49
- [Environment Explorer on page 50](#)

4.1.1 Editors

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 51
- [OTD Editor](#) on page 52
- [Collaboration Editor \(Java\)](#) on page 53
- [Collaboration Editor \(XSLT\)](#) on page 54
- [Environment Editor](#) on page 55
- [Deployment Editor](#) on page 56

4.1.2 Analysis and Archiving tools

The Enterprise Designer includes several analysis and archiving tools, which are described in the following sections of this chapter:

- [Project/Environment Import](#) on page 57, which allows you to import a Project that has been created elsewhere.
- [Project/Environment Export](#) on page 62, which allows you to export a Project to an external file so that it may be used elsewhere.
- [Impact Analyzer](#) on page 67, which helps you visualize how a change to one part of a Project would affect the rest of the Project.
- [Version Control](#) on page 69, which allows you to maintain multiple versions of Project components.

4.1.3 User Interface

The Enterprise Designer also contains the customary graphical interface features, which are described in the following sections of this chapter:

- [Menus](#) on page 46 describes the options contained in the individual menus.
- [Toolbar](#) on page 48 describes the functionality of the toolbar icons.
- [Browser Buttons](#) on page 48 describes the browser buttons that appear throughout the Enterprise Designer, in various wizards and dialog boxes.

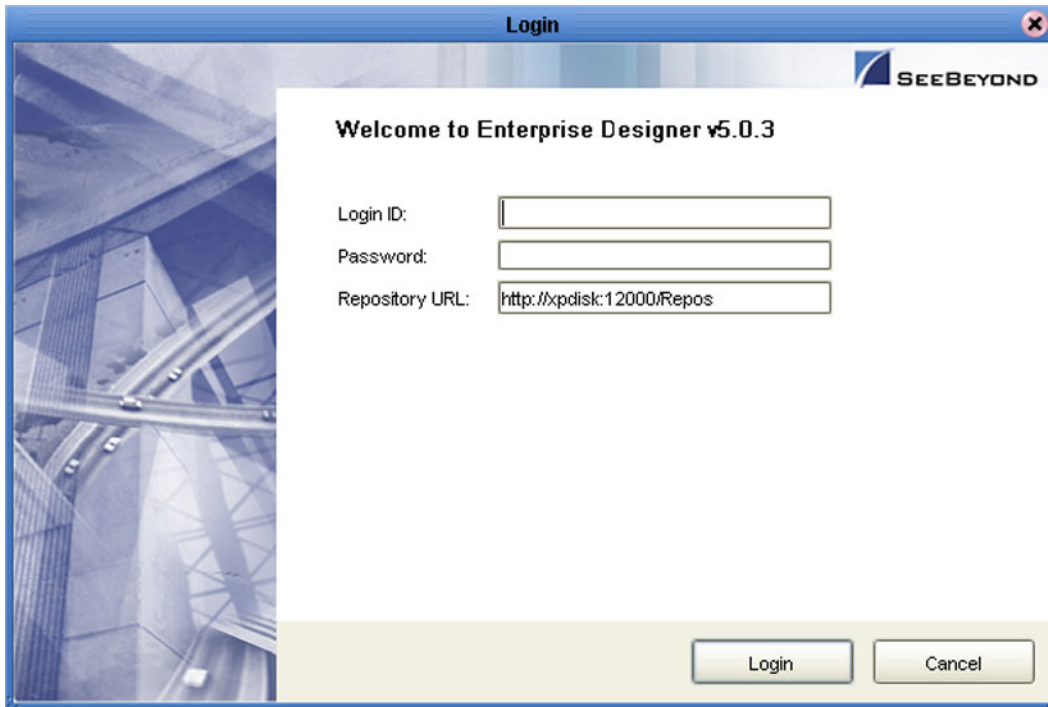
The procedure for invoking the Enterprise Designer is described in [Starting Enterprise Designer](#) on page 45.

4.2 Starting Enterprise Designer

To start the Enterprise Designer on a Windows Platform

- 1 Run the batch file `ICAN-root\edesigner\bin\runed.bat` to display the *Login* dialog box shown in Figure 23 (placing a shortcut on your desktop streamlines this procedure).

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 The URL for the Repository should be displayed in the *Repository URL* text box. If it is incorrect, edit the URL before proceeding. See the *SeeBeyond ICAN Suite Installation Guide* for details.
- 5 Click **Login** to complete the login process and display the Enterprise Designer GUI shown in Figure 22. A progress monitor will appear while the process is running.
- 6 The URL for the Repository should be displayed in the *Repository URL* text box. If it is incorrect, edit the URL before proceeding. See the *SeeBeyond ICAN Suite Installation Guide* for details.
- 7 Click **Login** to complete the login process and display the Enterprise Designer GUI shown in Figure 22.

4.3 Interface Features

4.3.1 Menus

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

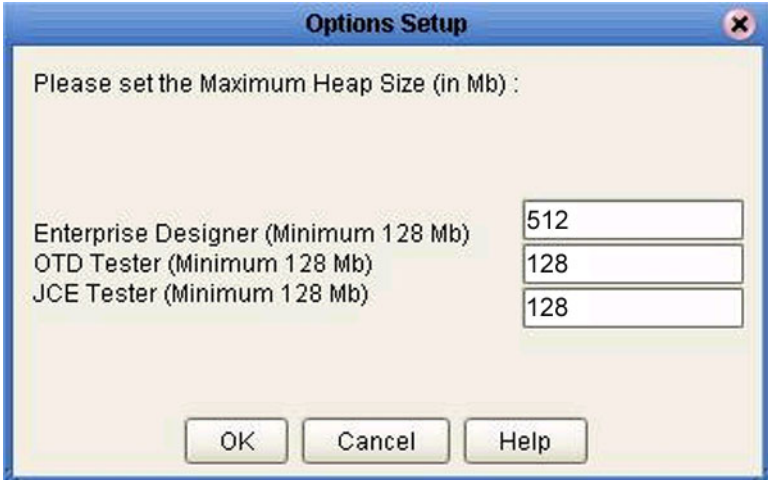
File Menu

Table 7 File Menu Options

Option	Function
Save	Saves changes to the selected objects.
Save All	Saves changes to all objects currently open in the editor.
Exit	Closes the Enterprise Designer.

Tools Menu

Table 8 Tools Menu Options

Option	Function
Impact Analyzer	Displays a dialog box in which you can view how one component of a Project impacts other components. See Impact Analyzer on page 67.
Options	<p>Displays the Options Setup dialog box, in which you can specify the maximum heap size for selected components:</p> 
Update Center	Displays a series of dialog boxes in which you can check for program updates. See the <i>eGate Integrator Installation Guide</i> .

View Menu

Table 9 View Menu Options

Option	Function
Environment Explorer	Activates the Environment Explorer tab on the Enterprise Explorer. See Environment Explorer on page 50 .
Project Explorer	Activates the Project Explorer tab on the Enterprise Explorer. See Project Explorer on page 49 .

Window Menu

Table 10 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.





Help Menu

Table 11 Help Menu Options

Option	Function
Contents	Displays the online help for all installed components of the ICAN Suite, opening to the initial ICAN topic.
Help Sets	Displays the online help for all installed components of the ICAN Suite, opening to the initial topic for the selected component.
About Enterprise Designer	Displays an information box giving the version number, copyright information, and currently-active Repository connection information.

4.3.2 Toolbar






Table 12 Enterprise Designer Toolbar Icons

Icon	Function
	Refresh All from Repository refreshes the Project Explorer and Environment Explorer to display the current contents of the Repository. (You are prompted to save any changes before the refresh occurs.) Open editors are not refreshed.
	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.

4.3.3 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 13 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.).

4.4 Enterprise Explorer

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

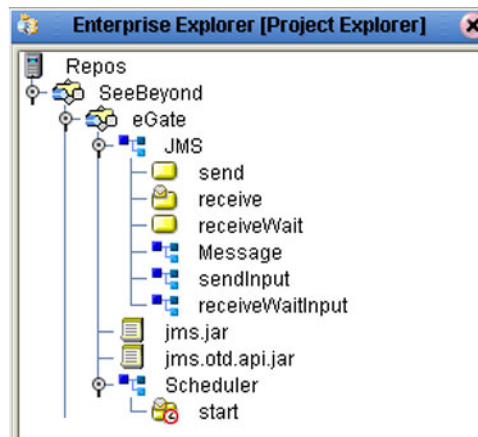
- **Project Explorer** on page 49 deals with logical components.
- **Environment Explorer on page 50** deals with physical resources, including the Logical Host and Integration Server.

Note: *The Project and Environment trees are initially loaded only to the Project or Environment level. The contents of a Project or Environment are loaded when you expand the particular node. This causes a slight delay when you expand the node, but eliminates a potentially-significant delay when you open Enterprise Designer, due to the large size of some OTD libraries.*

4.4.1 Project Explorer

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 24.

Figure 24 Enterprise Explorer: Project Explorer View

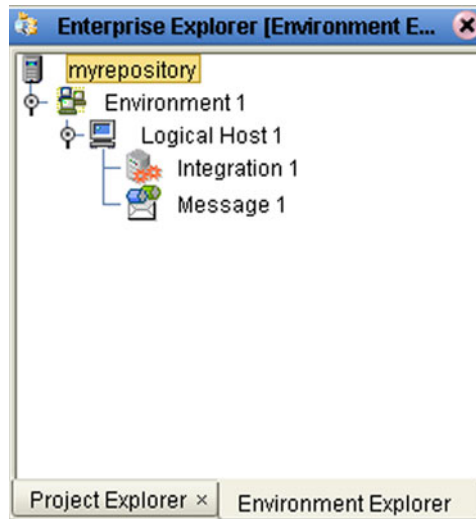


Details of the features and usage of the Project Explorer are found in **eGate Projects** on page 72.

4.4.2 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 25 Enterprise Explorer: Environment Explorer View



Details of the features and usage of the Environment Explorer are found in [Environments](#) on page 240.

4.5 Enterprise Designer Editors

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

- [Connectivity Map Editor](#) on page 51
- [OTD Editor](#) on page 52
- [Collaboration Editor \(Java\)](#) on page 53
- [Collaboration Editor \(XSLT\)](#) on page 54
- [Environment Editor](#) on page 55
- [Deployment Editor](#) on page 56

Additional facilities are also displayed here, such as the Java Debugger (see [Debugging Java-based Collaboration Definitions](#) on page 190).

Note: See the *eGate Integrator Tutorial* for an end-to-end demonstration of the steps involved in setting up a Project.

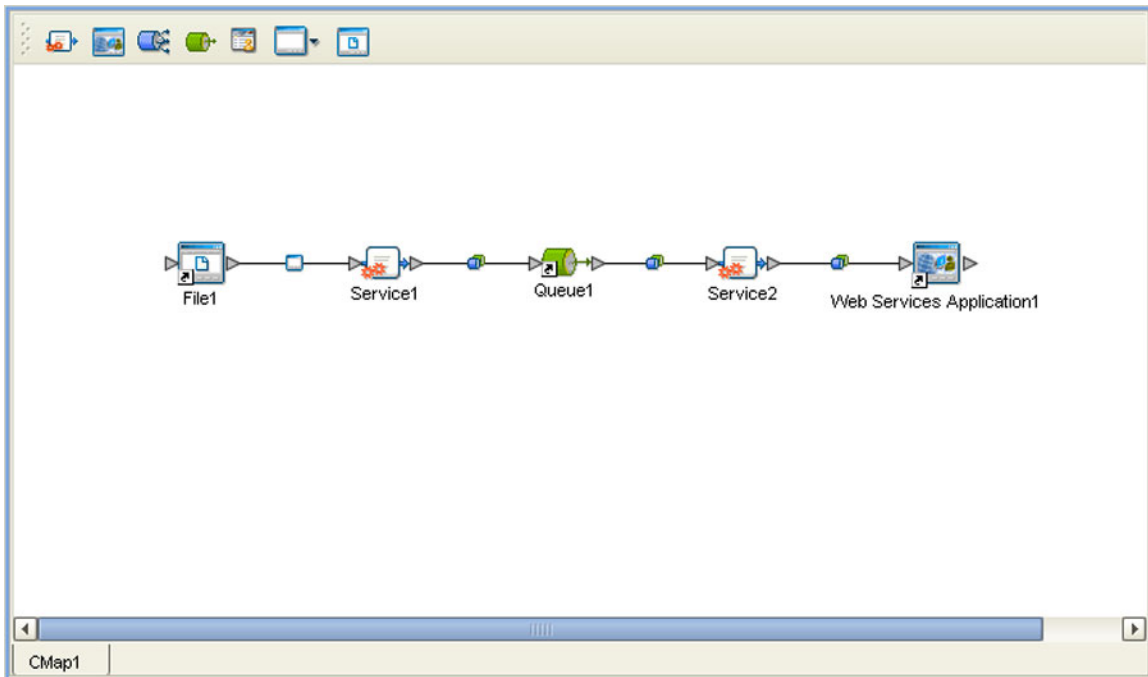
4.5.1 Connectivity Map Editor

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 26, 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: You should create your Collaboration Definitions before using the Connectivity Map to connect components.

See [Using the Connectivity Map Editor](#) on page 79 for detailed information.

Figure 26 Connectivity Map Editor

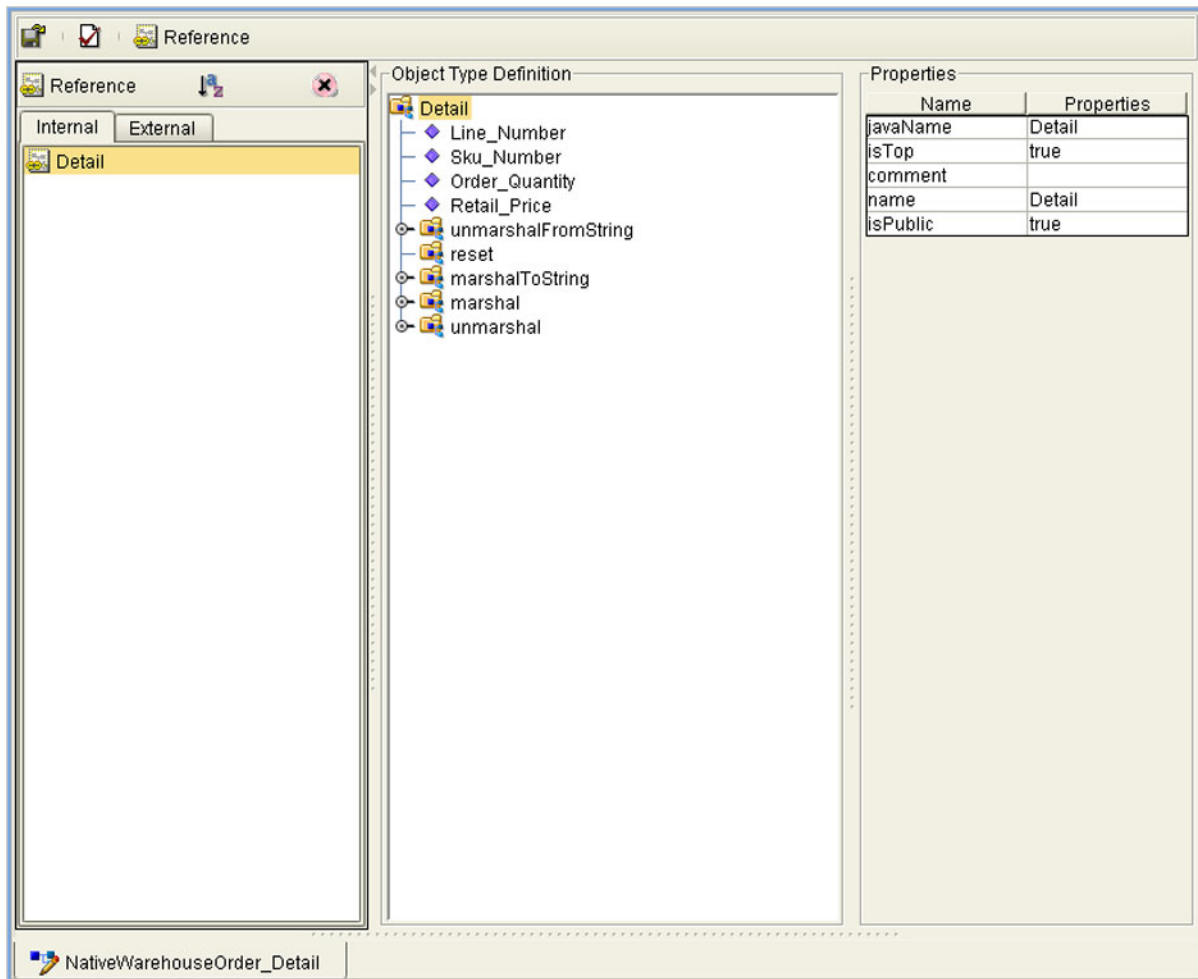


4.5.2 OTD Editor

The OTD Editor window, as shown in Figure 27, displays the source files used to create the Object Type Definitions (OTDs) to use with a Project. You use an OTD wizard tool to create OTD files and add them to the **Project Explorer** tab.

See [Using the OTD Editor](#) on page 125 for detailed information.

Figure 27 OTD Editor

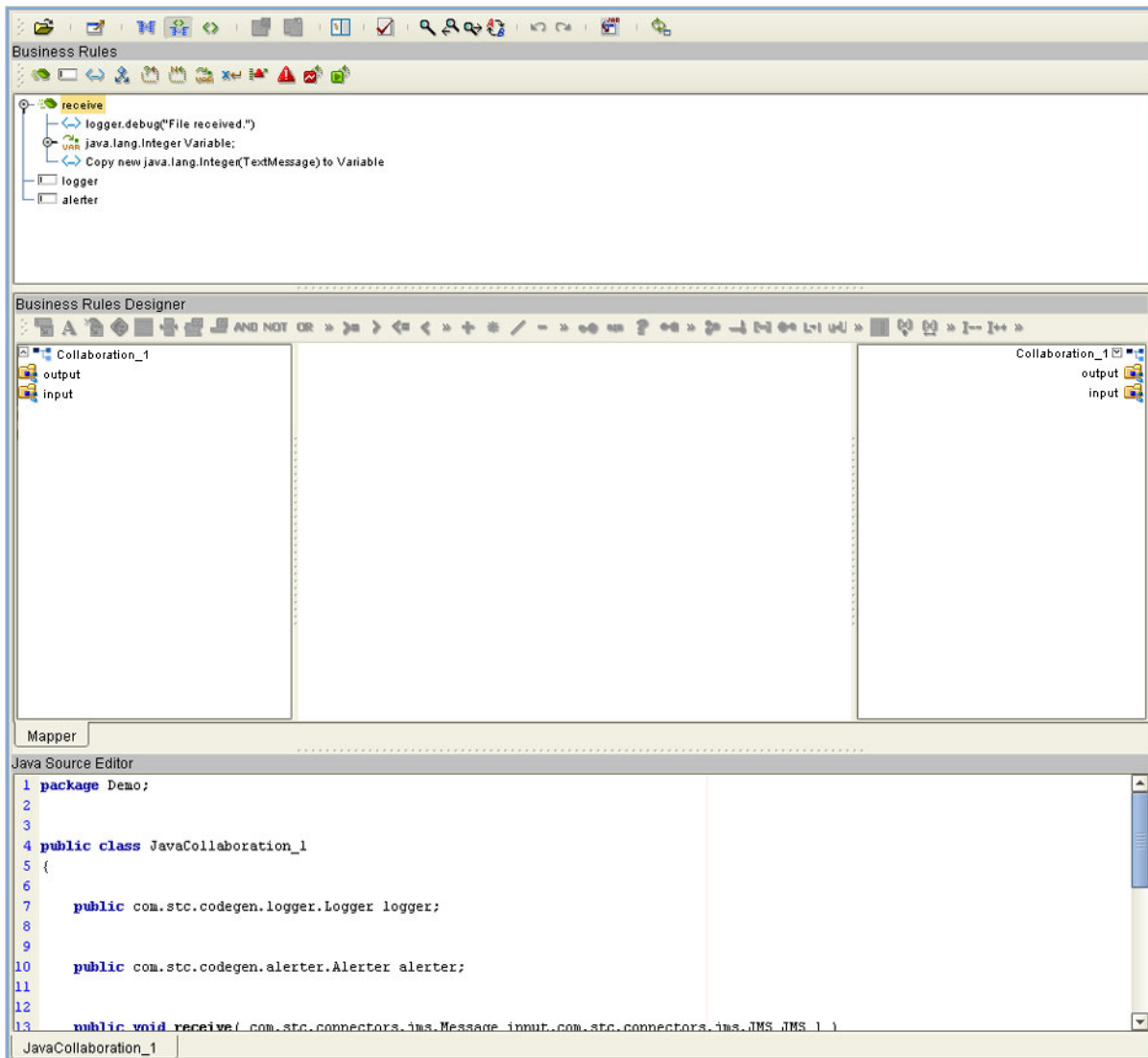


4.5.3 Collaboration Editor (Java)

The Collaboration Editor (Java) window, as shown in Figure 28, displays a Java-based Collaboration Definition that you want to include in a Project. You use a Java wizard tool to create Collaboration Definition files and add them to the **Project Explorer** tab.

See [Using the Collaboration Editor \(Java\)](#) on page 143 for detailed information

Figure 28 Collaboration Editor (Java)

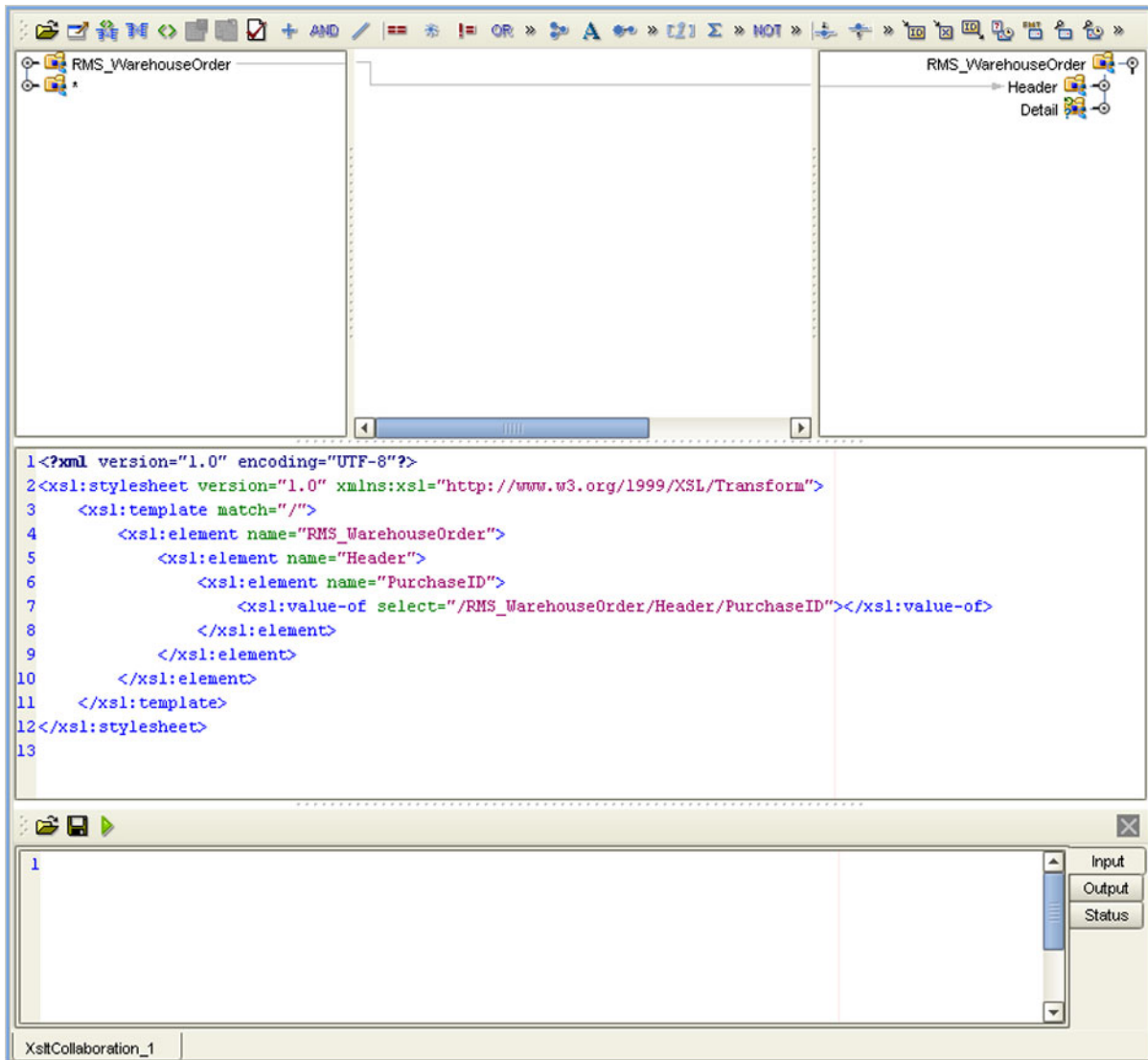


4.5.4 Collaboration Editor (XSLT)

The Collaboration Editor (XSLT) window, as shown in Figure 29, displays the XSLT-based Collaboration Definitions that you need to map together and include in the Project. You use a XSLT wizard tool to create Collaboration Definition files and add them to the **Project Explorer** tab.

See [Using the Collaboration Editor \(XSLT\)](#) on page 215 for detailed information

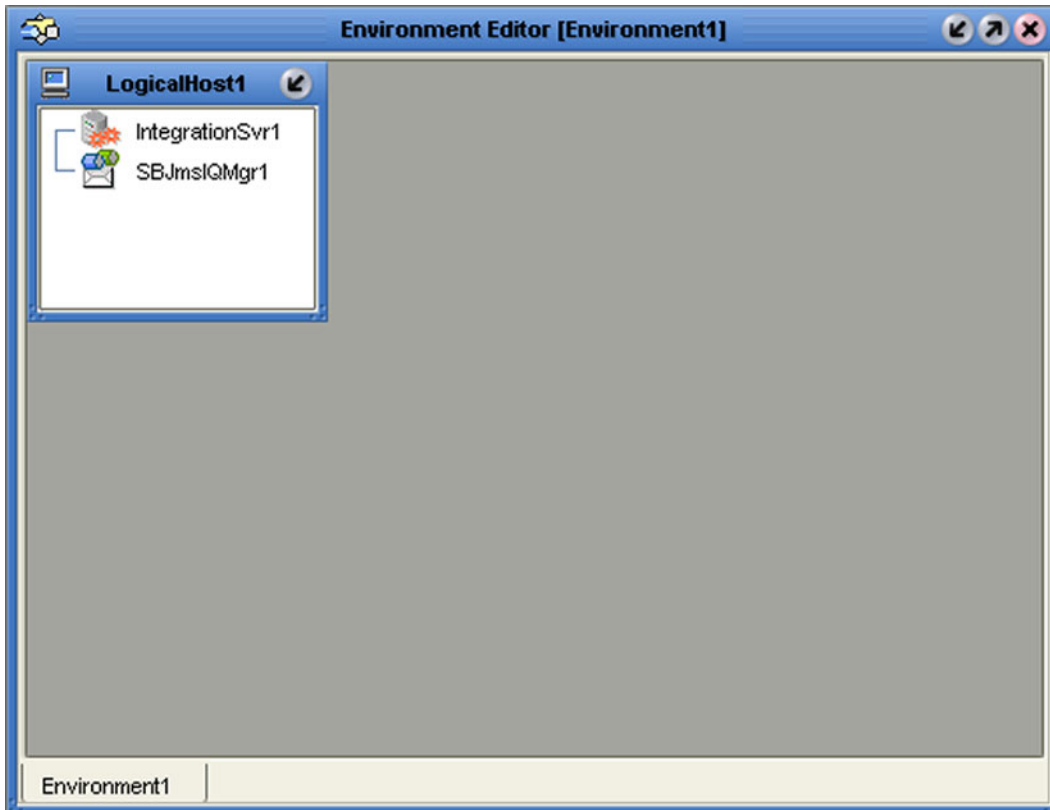
Figure 29 Collaboration Editor (XSLT)



4.5.5 Environment Editor

The Environment Editor 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 30.

Figure 30 Environment Editor

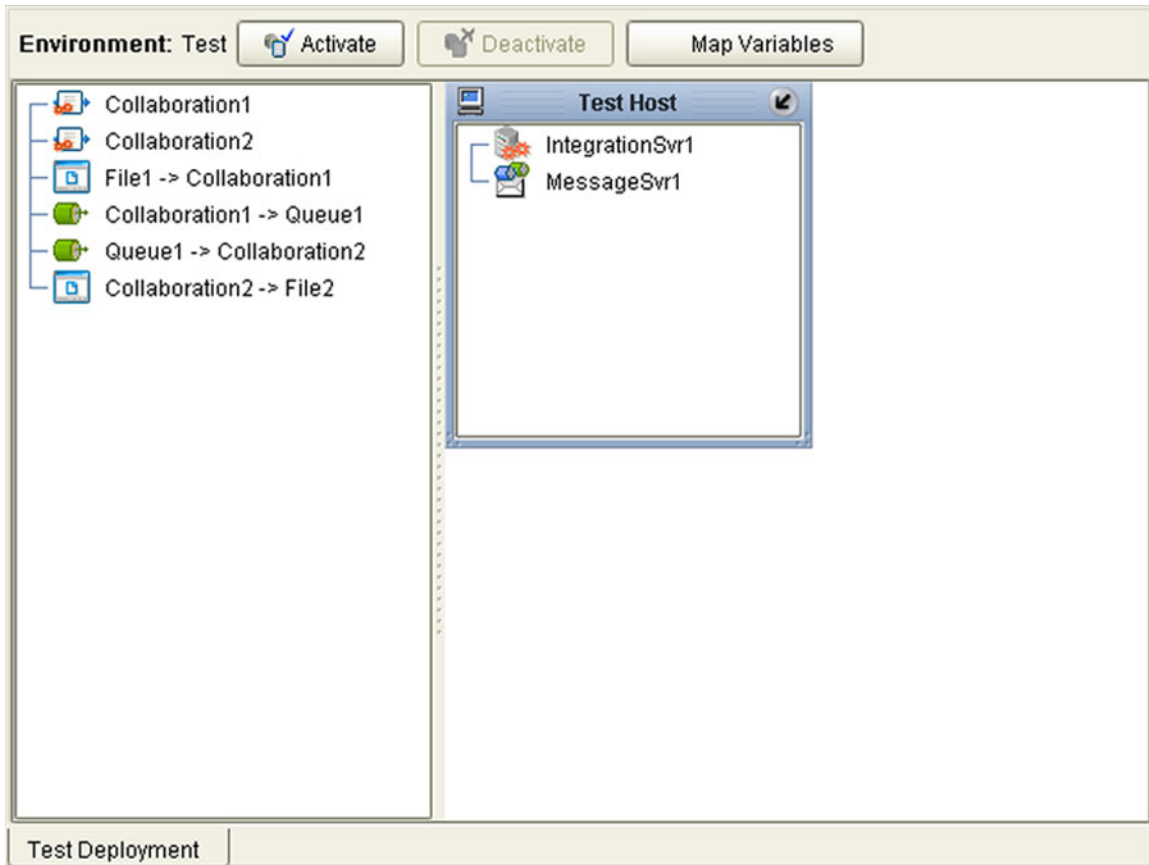


Note: *Unlike changes to Project-related configuration properties, changes to Environment-related properties do not require redeployment, only application.*

4.5.6 Deployment Editor

The Deployment Editor, as shown in Figure 31, contains information about how Project components will be deployed in an Environment. See [The Deployment Editor](#) on page 268 for detailed information

Figure 31 Deployment Editor



4.6 Additional Tools and Features

4.6.1 Project/Environment Import

The import function allows you to import an eGate Project or Environment file using the Enterprise Designer. Both follow essentially the same procedure.

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

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.
- If you have not installed all of the necessary products (such as eWays) that a Project requires, you will not be able to import that Project and will get an error message.
- You can specify a new Project name and location (in Project Explorer) during import.
- References are validated during import.
- Project deployment objects are not imported, because they have references to both Project and Environment elements that are not required at the Project level.

Note: A record of this process can be found in:

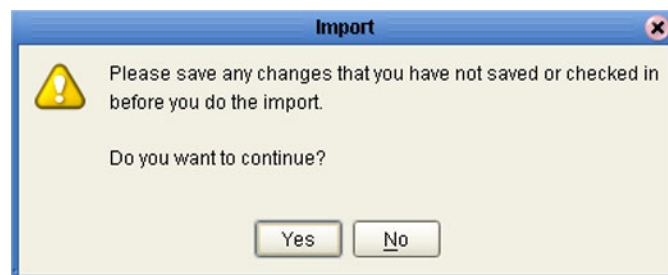
`ICAN-root\repository\logs\repository.log`

Importing a Project Using Enterprise Designer

To import a Project using Enterprise Designer

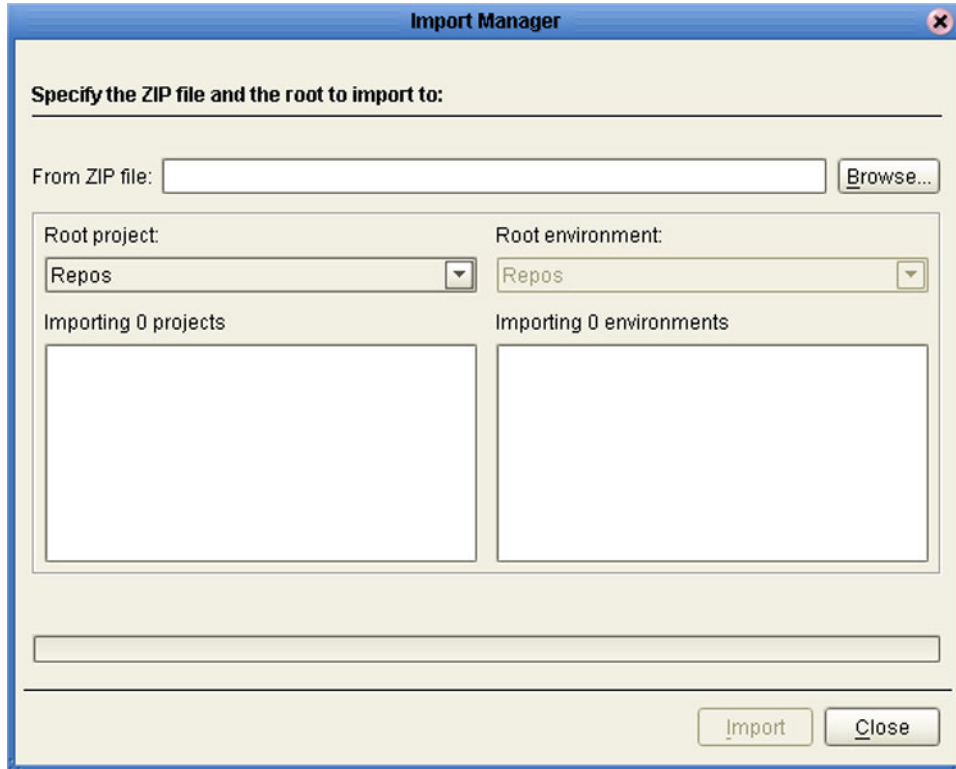
- 1 From the Repository context menu (for Projects) or the Project context menu (for Sub-Projects), select **Import**.
- 2 The message box shown in Figure 32 appears, prompting you to save your changes.

Figure 32 Import Message Box



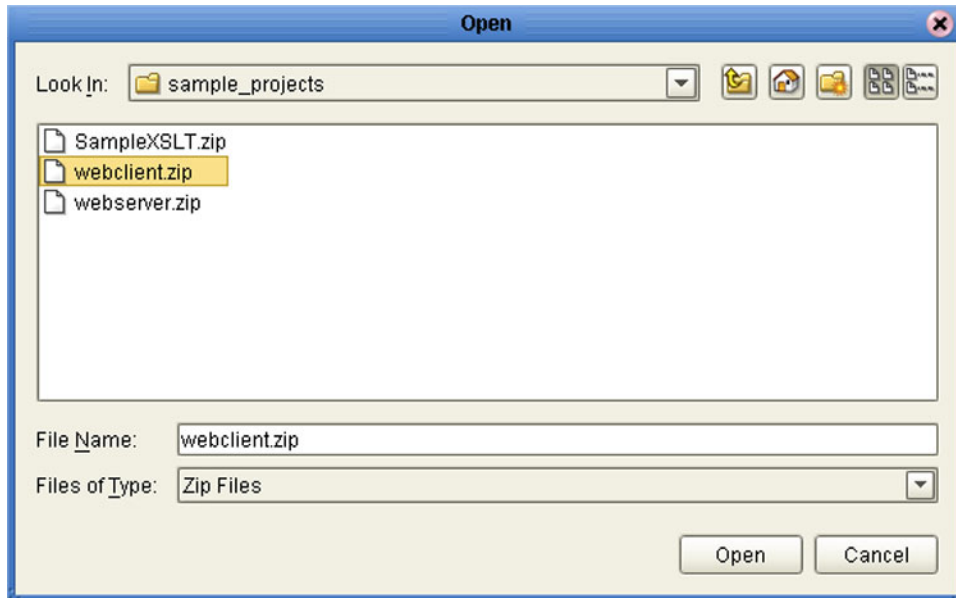
- A If you want to save your changes, but have not already done so, click **No**. Save your changes, and then re-select **Import**, as in step 1.
- B If you have saved any desired changes, click **Yes** to display the dialog box shown in Figure 33.

Figure 33 Import Manager Dialog Box (1)



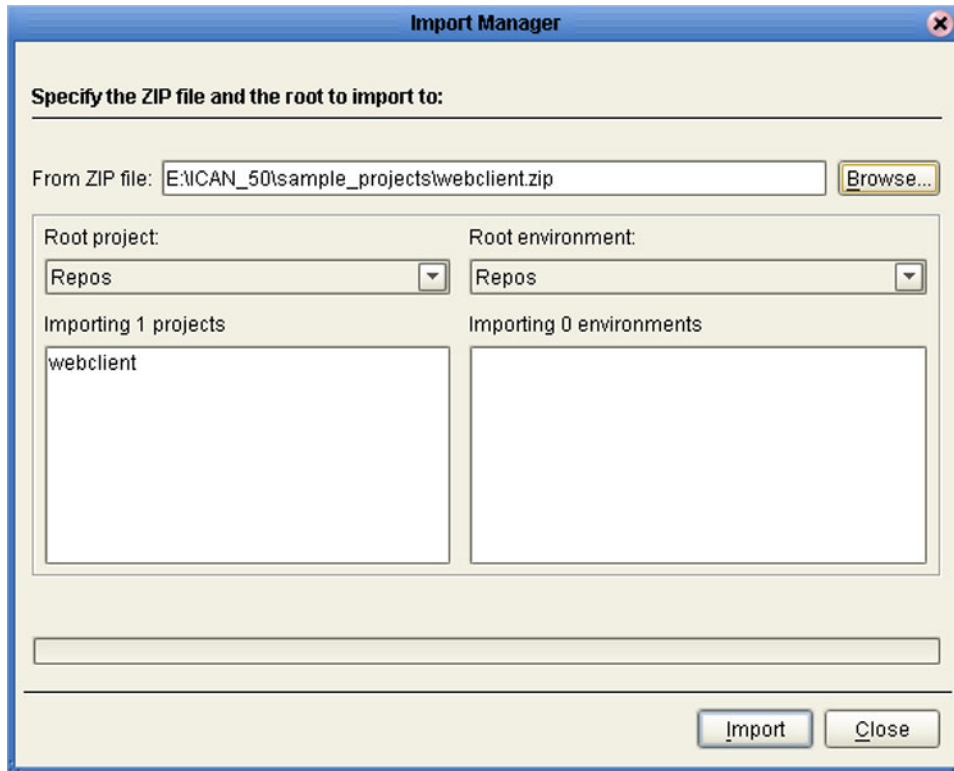
- 3 Click the **Browse** button to display the *Open File* dialog box, as shown in Figure 34. If you browse to an Environment file, the *Root environment* field will be enabled.

Figure 34 Open File Dialog Box



- 4 Locate and select the Project or Environment file that you want to import.
- 5 Click **Open** to import the file.
- 6 The Import Manager dialog box appears as shown in Figure 35, in which you specify the desired destination and file name (if different from the original).

Figure 35 Import Manager Dialog Box



- 7 Click **Import** to import the file.
- 8 The Import Status message box shown in Figure 36 appears after the file has been imported successfully.

Figure 36 Import Status Message Box



- 9 Click **OK** to close the message box. The Project Explorer will automatically be refreshed from the Repository.

Importing a Project Using the Command Line

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

Location of script file:

ICAN-root\repository\util\importProject.bat (or importProject.sh)

Command Syntax:

```
importProject username password importfile rootprojectname
```

where:

- ♦ **importfile** is the fully-qualified archive file name for the Project you are importing.
- ♦ **rootprojectname** is either the name of the parent Project or the ICAN root directory. If the Project is to be attached to *ICAN-root*, then leave this parameter as an empty string.

To import a Project using the import script

- 1 Open a command prompt and change directory to *ICAN-root*\repository\util.
- 2 Type (for example): **importProject Administrator stc c:\project4import.zip myExistingProject**.

This will extract the Project that exists in the file **c:\project4import.zip** and attach it as a Sub-Project of **myExistingProject**.

4.6.2 Project/Environment Export

The export function allows you to export an eGate Project or Environment 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.

Note: A record of this process can be found in:

```
ICAN-root\repository\logs\repository.log
```

Exporting a Project Using Enterprise Designer

To export a Project or Environment using Enterprise Designer

- 1 From the Project context menu, select **Export** to display the Export Manager dialog box. If you do not have any existing Environments in your Repository, you will see the dialog box shown in Figure 37. If you do, you will see the dialog box shown in Figure 38.

Figure 37 Export Manager Dialog Box (1a)

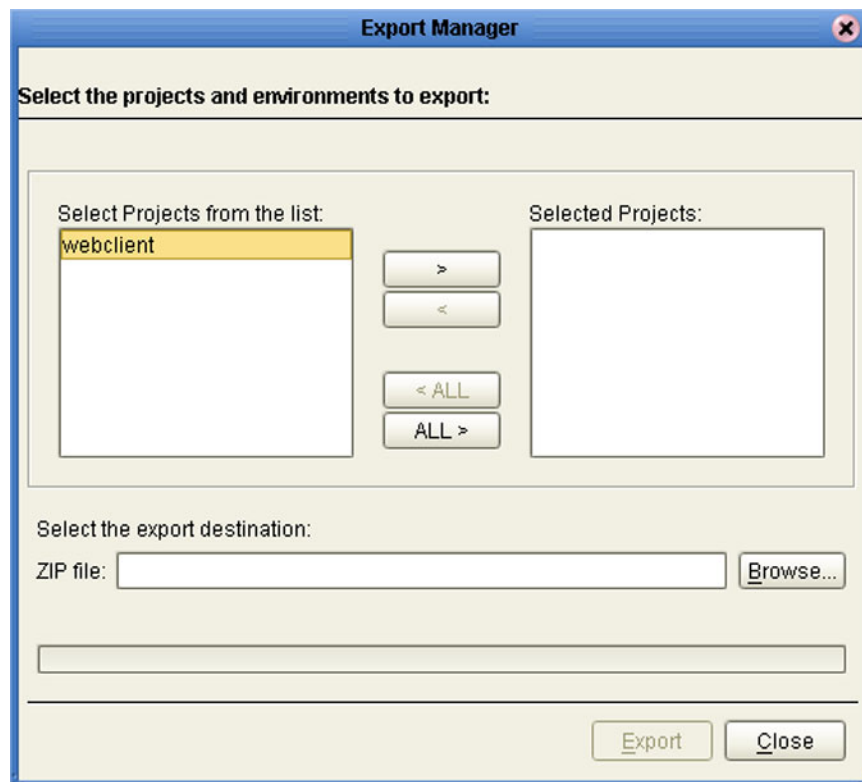
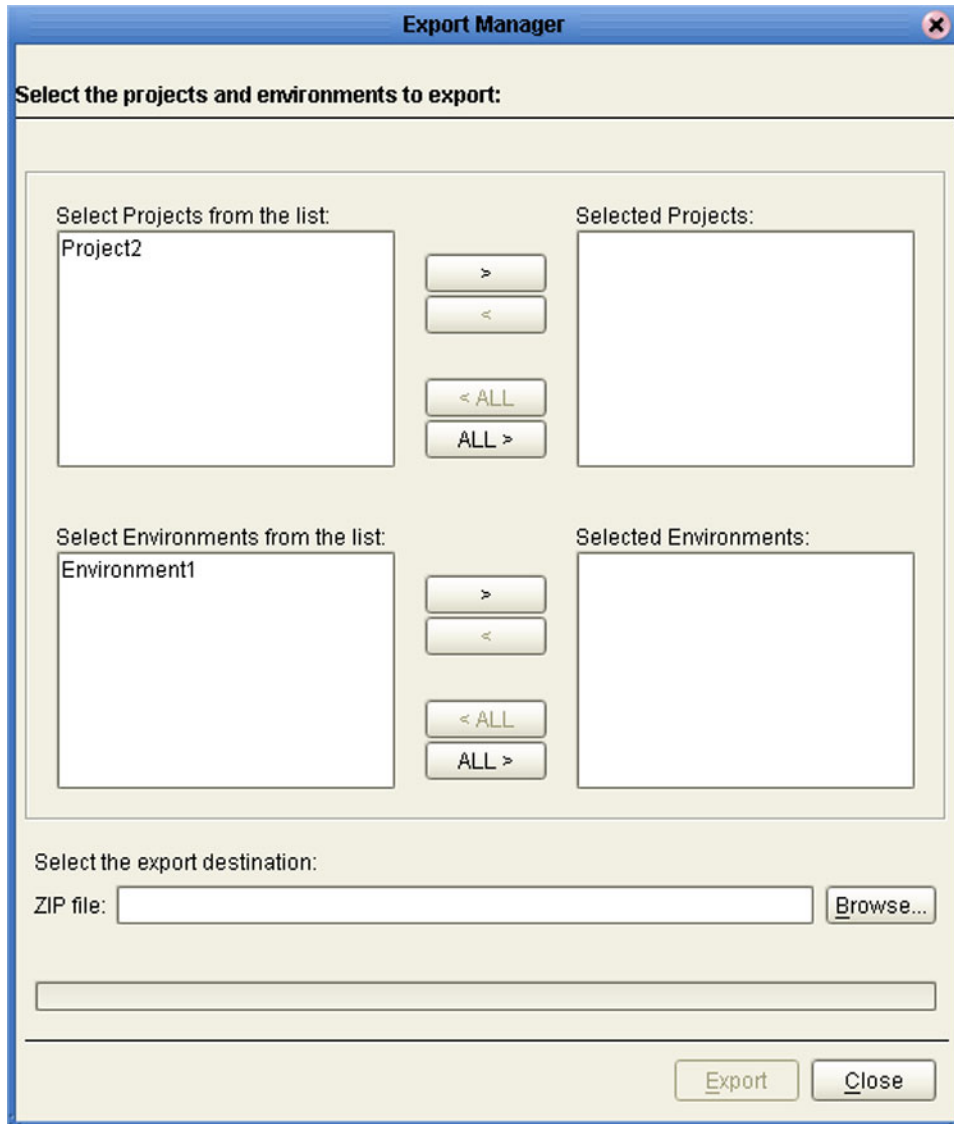
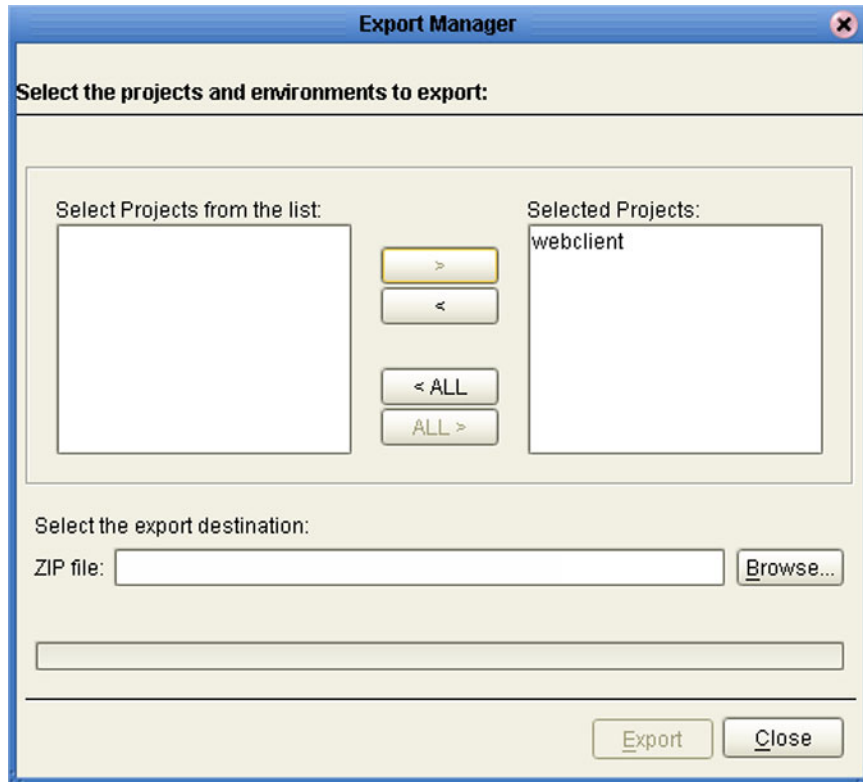


Figure 38 Export Manager Dialog Box (1b)



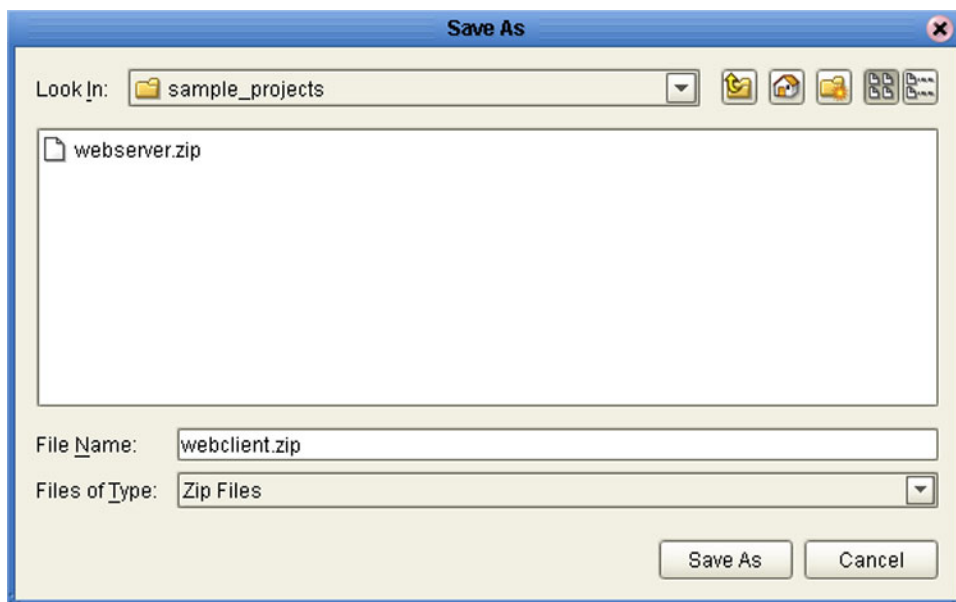
- 2 Highlight the desired Project(s) or Environment(s) in the displayed list, and transfer them to the *Selected Projects* or *Selected Environments* panel using the arrow buttons (see Figure 39).

Figure 39 Export Manager Dialog Box (2)



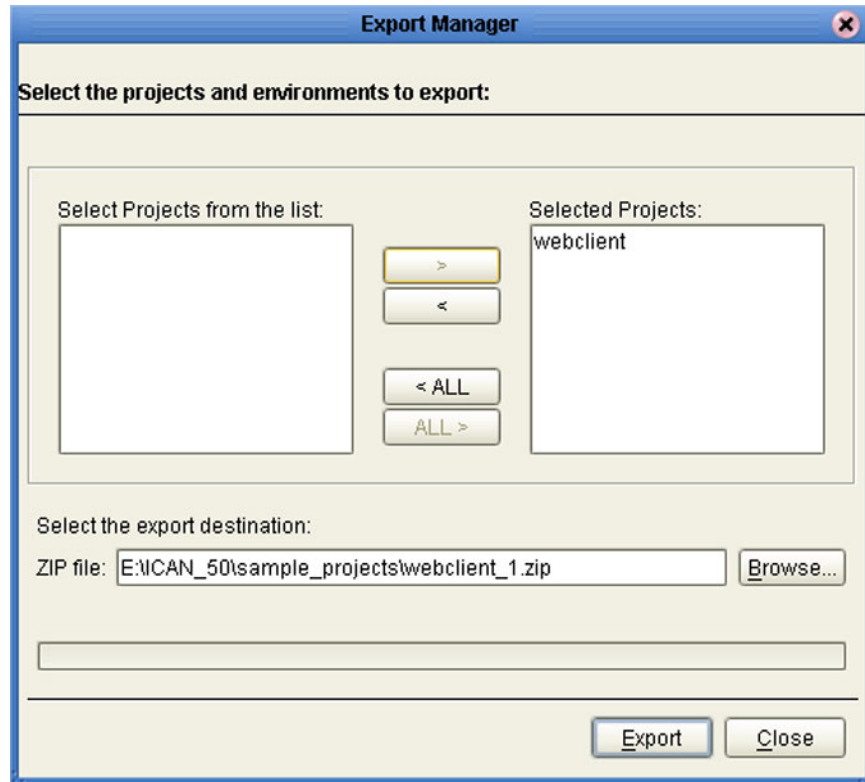
- 3 Click the **Browse** button to display the *Save As* dialog box, as shown in Figure 40.

Figure 40 Save As Dialog Box



- 4 Select the export destination and change the export file name, if desired.
- 5 Click **Save As** to enter the file name.

Figure 41 Enter File Name Dialog Box (2)



- 6 Click **Export** to export the Project file (this process may take a few minutes).
- 7 The Export Status message box shown in Figure 42 appears after the file has been exported successfully.

Figure 42 Export Status Message Box



- 8 Click **OK** to close the message box.

Exporting a Project Using the Command Line

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

Location of script file:

ICAN-root\repository\util\exportProject.bat (or exportProject.sh)

Command Syntax:

```
exportProject username password exportfile projectname  
environmentname
```

where:

- ♦ **exportfile** is the fully-qualified archive file name for the Project you are exporting, indicating where it is to be stored.
- ♦ **projectname** is the name of the Project you are exporting.
- ♦ **environmentname** is the name of the Environment you are exporting.

To export a Project using the export script

- 1 Open a command prompt and change directory to *ICAN-root*\repository\util.
- 2 Type (for example): **exportProject Administrator stc c:\project4export.zip myProject myEnvironment**.

This will save the existing Project **myProject** and Environment **myEnvironment** to the file **c:\project4export.zip**.

4.6.3 Impact Analyzer

The Impact Analyzer helps you determine how a change to one component of a Project or Environment will affect other components in that Project or Environment.

To perform an Impact Analysis

- 1 Select a component in either the Project Explorer or Environment Explorer.
- 2 Click the **Impact Analyzer** button, or select **Impact Analyzer** from the Tools menu, to display the *Impact Analyzer* dialog box shown in Figure 43.
- 3 In the *Please show me* drop-down list, select items you would like to view.
- 4 From the list of objects that appears, select one for which you would like to perform an impact analysis.
 - ♦ You can print the object list by clicking **Print** to display the Windows *Print* dialog box.
- 5 Click **Impact** to see how that object would be affected by a change to the component you selected in step 1.

Figure 43 Impact Analyzer Dialog Box

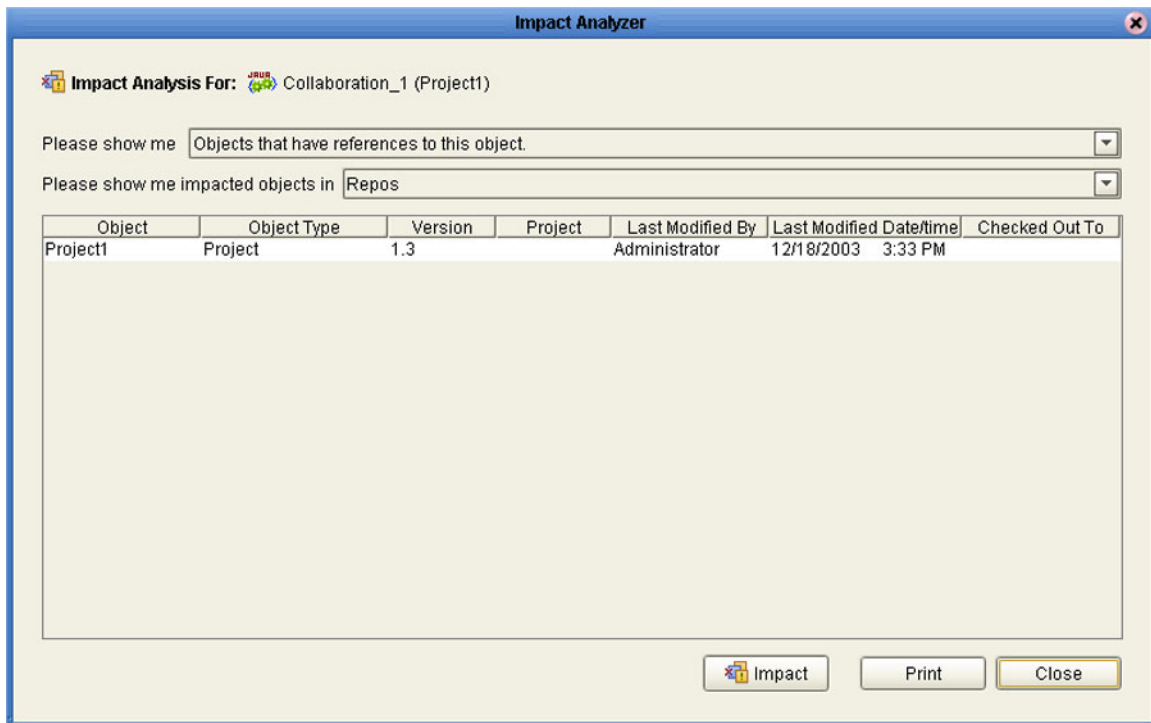



Table 14 Impact Analyzer Command Buttons

Button	Function
 Impact	Performs an impact analysis for the object selected from the object list.
Print	Displays the Windows Print dialog box, which you can use to print the object list.
Close	Closes the Impact Analyzer dialog box.

4.6.4 Version Control

Version control allows you to maintain multiple versions of a Project or Environment component. The version history of each component is recorded to a log file, and can be viewed by means of a menu option.

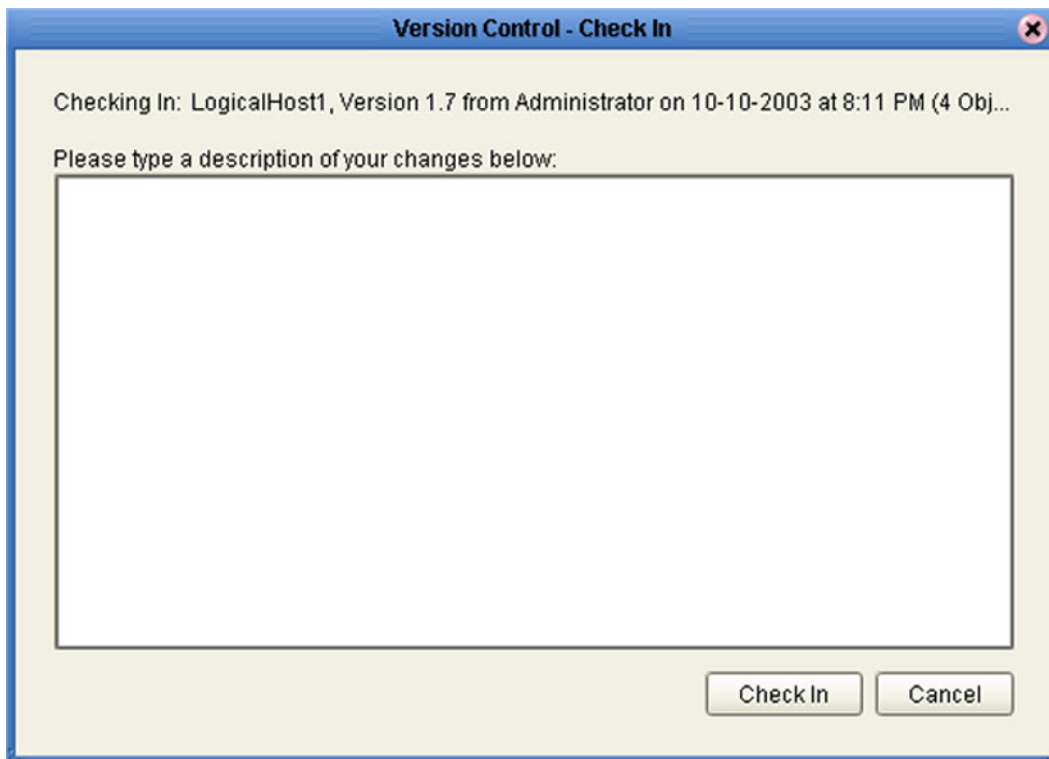
Checking a Component In

Once you have created and configured a Project or Environment component, you can check that object in by using the following procedure.

To check in a version of a Project or Environment component

- 1 Click the Project or Environment Explorer tab in the Enterprise Explorer.
- 2 Right-click on a component to display its context menu.
- 3 Select **Check In** to display the *Version Control - Check In* dialog box shown in Figure 44.

Figure 44 Version Control - Check In Dialog Box



- 4 Type in a description of the changes in the new version.
- 5 Click **Check In** to save your changes to a new version.

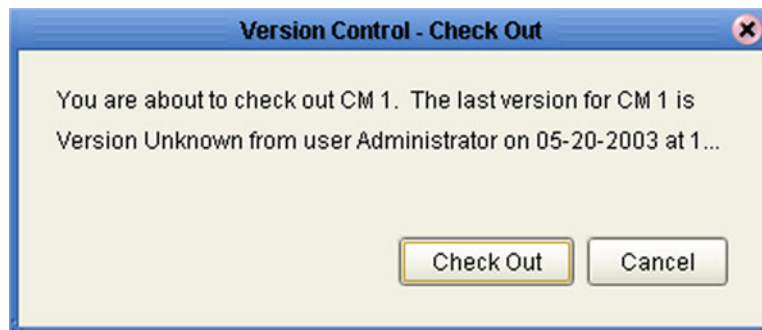
Checking a Component Out

Once an object has been checked in, you can check it out by using the following procedure.

To check out a version of a Project or Environment component

- 1 Click the Project or Environment Explorer tab from the Enterprise Explorer.
- 2 Right-click on a component to display its context menu.
- 3 Select **Check Out** to display the *Version Control - Check Out* dialog box shown in Figure 45.

Figure 45 Version Control - Check Out Dialog Box



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

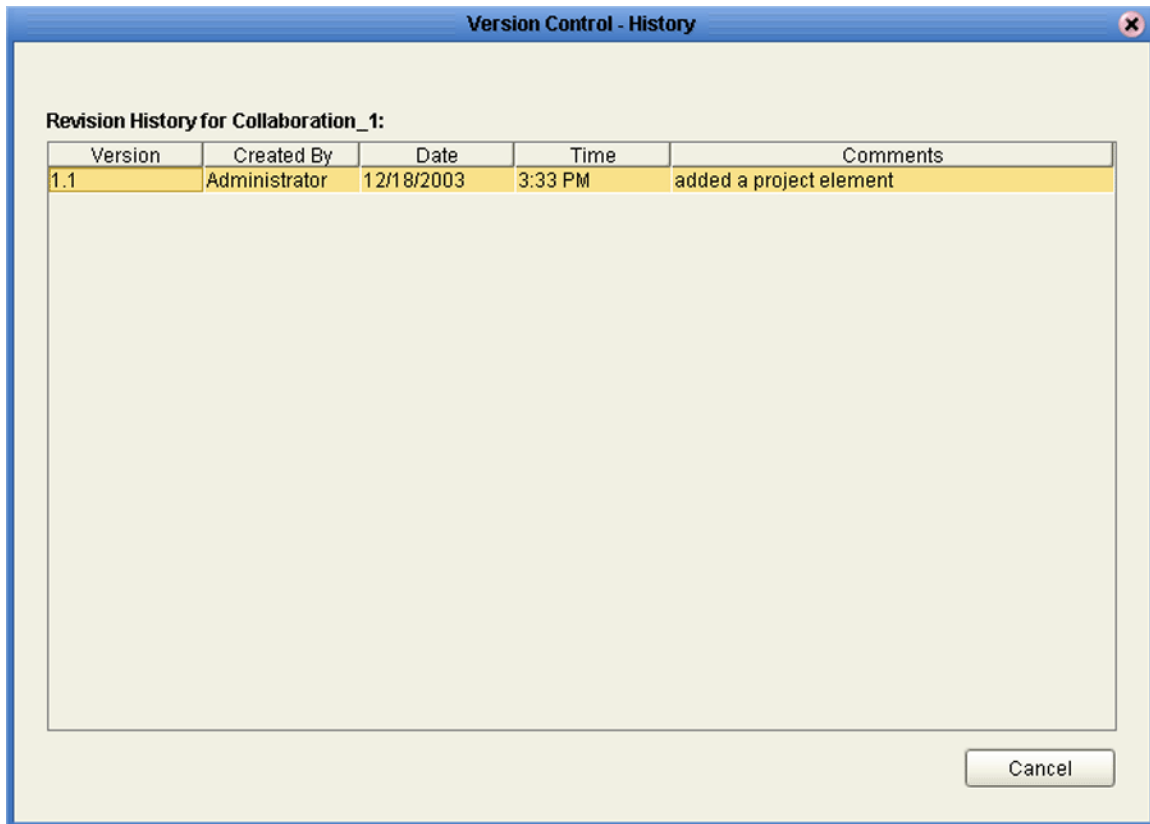
Note: *Only one user can have a file checked out for editing at a time. If another user attempts to check out the same file, they will receive a message indicating that the file is currently checked out.*

Viewing a Component's Version History

To view the version history for a component

- 1 Click the Project or Environment Explorer tab in the Enterprise Explorer.
- 2 Right-click on a component to display its context menu.
- 3 Select **Version History** to display the *Version Control - History* dialog box shown in Figure 46.

Figure 46 Version Control - History Dialog Box



eGate Projects

This chapter describes components of an eGate Project, and the use of the Enterprise Designer in defining your Project.

5.1 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, and are deployed to specific Logical Hosts in specific Environments by means of Deployment Profiles (see [Environments](#) on page 240).

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.

***Note:** See the eGate Integrator Tutorial for an end-to-end demonstration of the steps involved in setting up a Project.*

5.1.1 Project Components

The components found in a typical Project are described in the following sections of this chapter:

- [Services](#) on page 81
- [External Applications](#) on page 82
- [Schedulers](#) on page 82
- [Component Connections](#) on page 84
- [Message Destinations](#) on page 82

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 Collaboration Editor or the XSLT Collaboration Editor, and is based on an Object Type Definition.

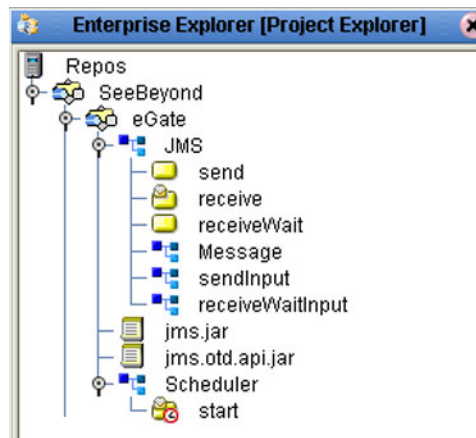
- **Object Type Definitions**

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.

5.2 The Project Explorer

A Project consists of logical constructs and configurations designed to solve some or all of a business problem. The **Project Explorer** displays the contents of the Repository that belong to the selected Project (see Figure 47).

Figure 47 Project Explorer












The Project Explorer is used in conjunction with the Connectivity Map Editor (see [Using the Connectivity Map Editor](#) on page 79) to create and configure a Project.

Note: Select *Refresh All from Repository* before you *Open* any component (such as a Collaboration) to ensure that you open the latest version of that component.

5.2.1 Project Explorer Icons

The icons described in Table 15 appear in the Project Explorer.

Table 15 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 Collaboration Definition (Java) file. A lock is displayed in the lower-left corner for Collaboration Definitions that are currently checked into the version control system.
	Represents an Collaboration Definition (XSLT) 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.

5.2.2 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 75
- **Project Menu** on page 76
- **Connectivity Map Menu** on page 77

Repository Menu

Figure 48 Repository Menu



Table 16 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	Displays a dialog box with which you can import a Project or Environment into the Repository.
Export	Displays a dialog box with which you can export a Project or Environment from the Repository to another location.
Refresh All from Repository	Refreshes the Project Explorer and Environment Explorer to display the current contents of 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 the <i>eGate Integrator System Administration Guide</i> .
Properties	Displays a dialog box the properties of your Repository. See the <i>eGate Integrator System Administration Guide</i> .

Note: Select *Refresh All from Repository* before you **Open** any other component (such as a Collaboration) to ensure that you open the latest version of the component.

Project Menu

Figure 49 Project Menu

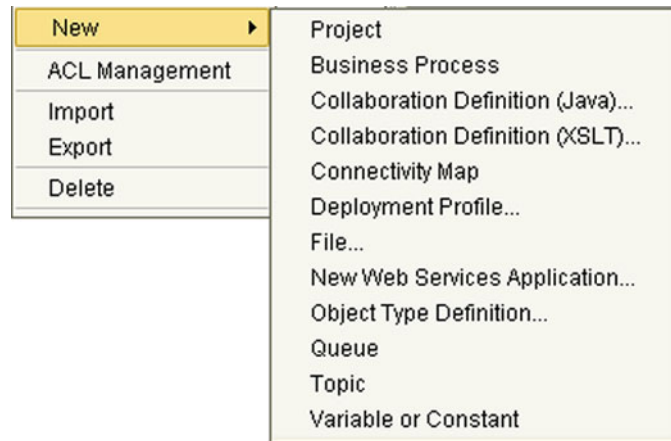


Table 17 Project Menu Options

Option	Option	Function
New	Project	Adds a Subproject folder to the selected Project.
	Business Process	If eInsight Business Process Manager is installed, displays the user interface for creating a new Business Process.
	Collaboration Definition (Java)	Displays the Collaboration Definition Wizard (Java) , with which you can create a Java-based Collaboration Definition. See Using the Collaboration Definition Wizard (Java) on page 133.
	Collaboration Definition (XSLT)	Displays the Collaboration Definition Wizard (XSLT) , with which you can create an XSLT-based Collaboration Definition. See Using the Collaboration Definition Wizard (XSLT) on page 209.
	Connectivity Map	Adds a Connectivity Map to the Project. See Using the Connectivity Map Editor on page 79.
	Deployment Profile	Displays a dialog box with which you can assign a Deployment Profile to the selected Project. See The Deployment Editor on page 268.
	File	Displays a dialog box with which you can create an external file to use with the Project.
	New Web Services ...	Adds a third-party Web service application to the Project Explorer. See SeeBeyond Web Services on page 282.
	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 91 for more information.
	Queue	Adds a queue to your Project.
Topic	Adds a topic to your Project.	

Table 17 Project Menu Options

Option	Option	Function
(New)	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 the <i>eGate Integrator System Administration Guide</i> .
Import		Displays a dialog box with which you can import a Project as a Subproject under the selected Project.
Export		Displays a dialog box with which you can export the selected Project.
Delete		Deletes the selected Project.

Connectivity Map Menu

Figure 50 Connectivity Map Menu

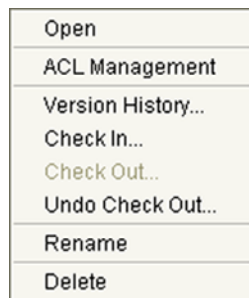


Table 18 Connectivity Map Menu Options

Command	Function
Open	Opens the Connectivity Map Editor for the selected Connectivity Map. See Using the Connectivity Map Editor on page 79.
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 the <i>eGate Integrator System Administration Guide</i> .
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 Viewing a Component's Version History on page 71 for more information.
Check In	Displays a dialog box, with which you can check in the current version of a Project. Refer to Checking a Component In on page 69 for more details.
Check Out	Displays a dialog box with which you can check out a version of a Project. See Checking a Component Out on page 70 for more information.

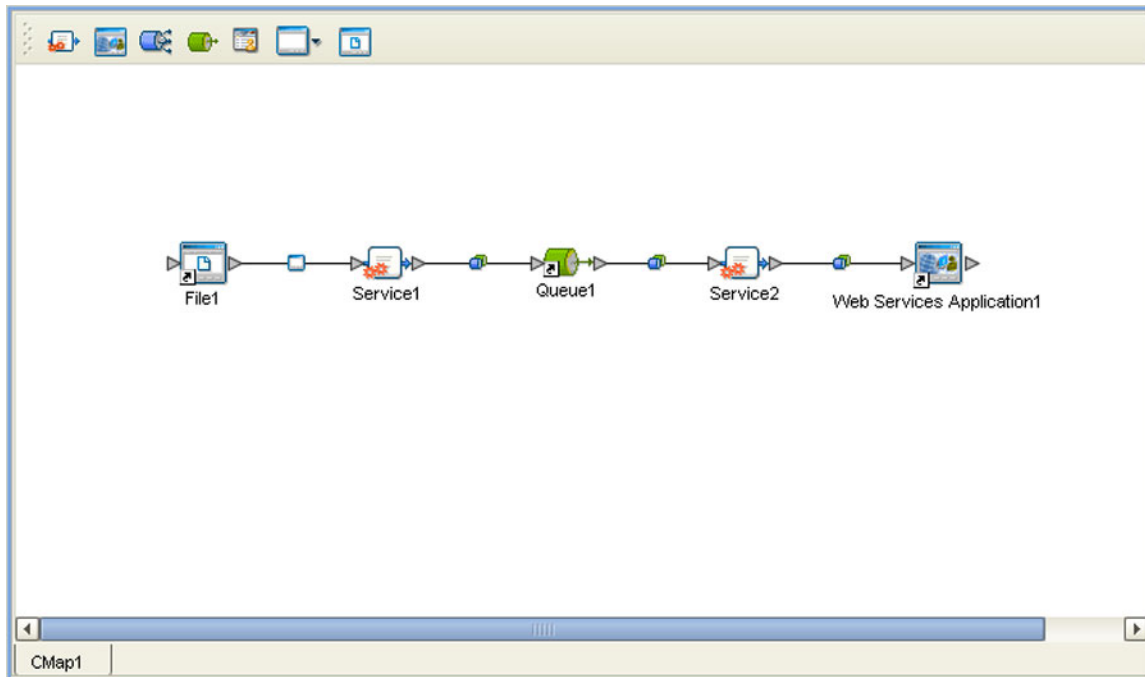
Table 18 Connectivity Map Menu Options

Command	Function
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.

5.3 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 51). 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.

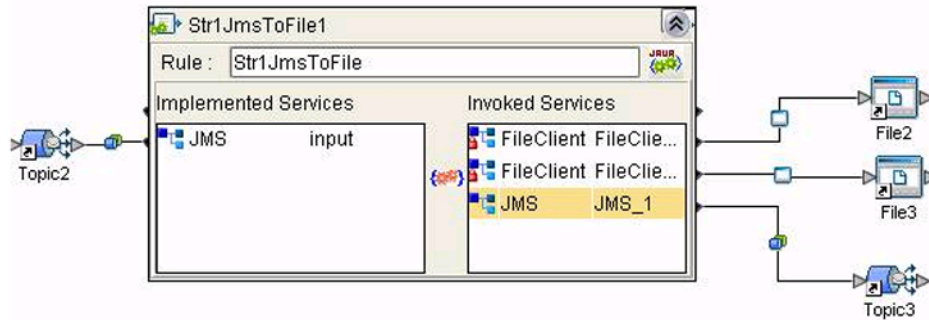
Figure 51 Connectivity Map Window



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.







When there are multiple destinations, as with a JMS topic, the Connectivity Map Editor cannot resolve which output port connects 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 52).

Figure 52 Linking JMS Topics



The Connectivity Map Editor toolbar contains the icons listed in Table 19, plus additional icons representing eGate add-ons and other ICAN components that you may have installed.

Table 19 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 81.
	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 SeeBeyond Web Services on page 282.
	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 82.
	Scheduler	Represents a scheduling component of the Connectivity Map. Use this component to set data transfer to occur at set intervals. See Schedulers on page 82.

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.

5.4 Services

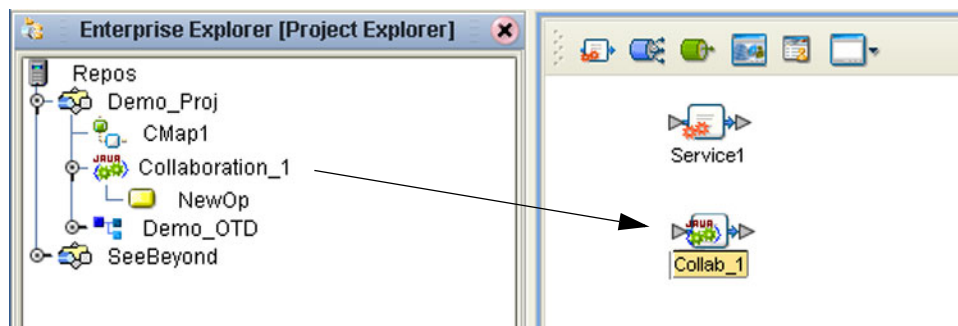
A service provides a framework for a process or a Collaboration, which contains the information required to execute a set of business rules.

5.4.1 Collaborations

A Collaboration is a logical operation performed between some combination of message destinations and external applications. The operation is defined by a Collaboration Definition, which can be encoded in either Java or XSLT.

The Collaboration acts as a service having a publication or subscription relationship with each linked entity. The link is provided by a JMS Client connection (see [Component Connections](#) on page 84). Dragging a Collaboration from the Project Explorer to the Service icon in the Connectivity Map defines the service as a Collaboration (see Figure 53).

Figure 53 Service Component



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.

Connection-related properties for the Collaboration (or other service) are configured in the adjoining JMS Client. These properties include:

- Concurrent or serial processing
- Transaction mode (transacted or XA)
- Security

All properties, and the procedures for configuring them, are detailed in the *eGate Integrator JMS Reference Guide*.

5.5 Message Destinations

A Message Destination is a container for stored data, and can follow either the topic or queue JMS model.

5.5.1 Topics

A *topic* is a message destination that conforms to the publish-and-subscribe messaging paradigm.

5.5.2 Queues

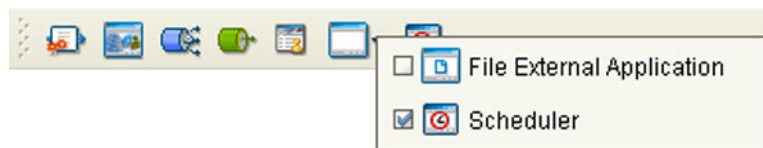
A *queue* is a message destination that conforms to the point-to-point messaging paradigm.

5.6 External Applications

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.

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. An example is shown in Figure 54.

Figure 54 External Application Drop-Down Menu



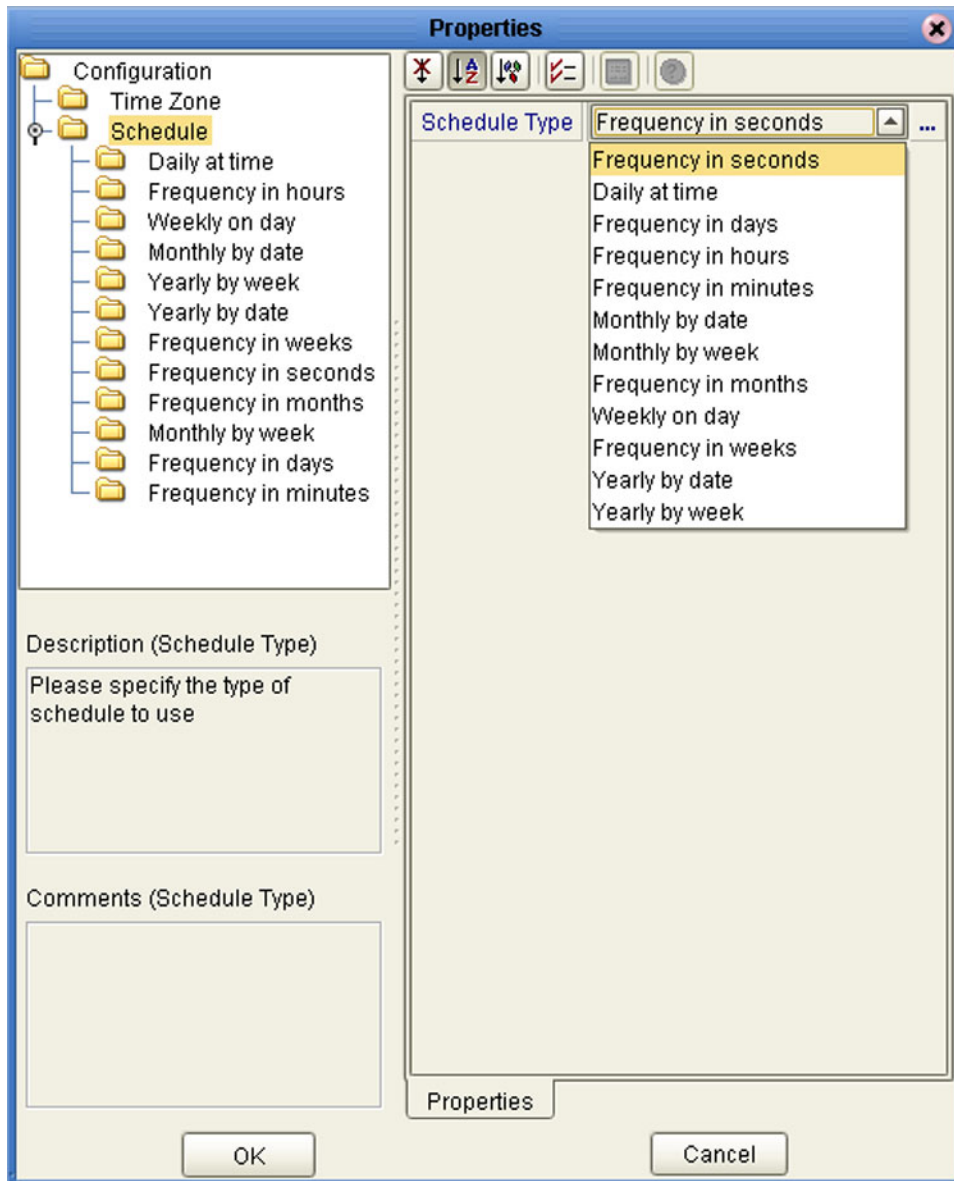
Selecting the check box beside an individual external application adds that icon to the toolbar; clearing the check box removes it from the toolbar.

5.6.1 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.

Once the scheduler is connected to a service in the Connectivity Map, double-clicking the JMS Client displays the Properties dialog box for that scheduler (see Figure 55).

Figure 55 Scheduler Properties Dialog Box



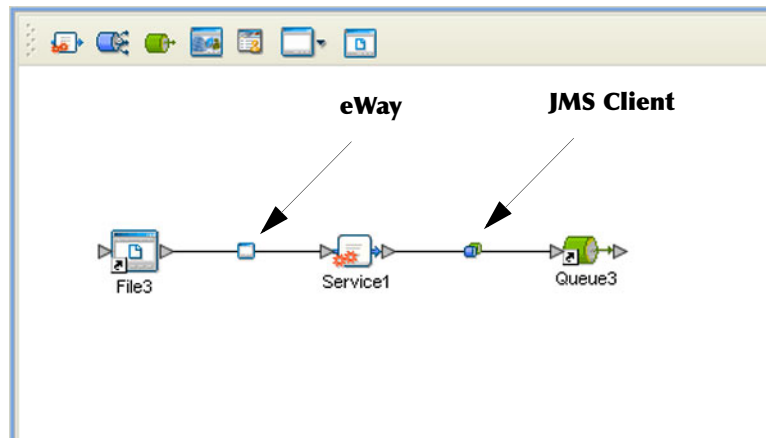
Selecting **Schedule** displays the **Schedule Type** property field which you set to the type of schedule you want to use. Selecting the corresponding node in the explorer tree displays the property field for that schedule type, in which you specify the desired value. The text in the *Description* box will include the appropriate units.

Selecting **Time Zone** displays the **Time Zone** property field in which you specify your local time zone, so that your schedule will be synchronized to the local time, if appropriate.

5.7 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 56).

Figure 56 Connection Icons in a Connectivity Map

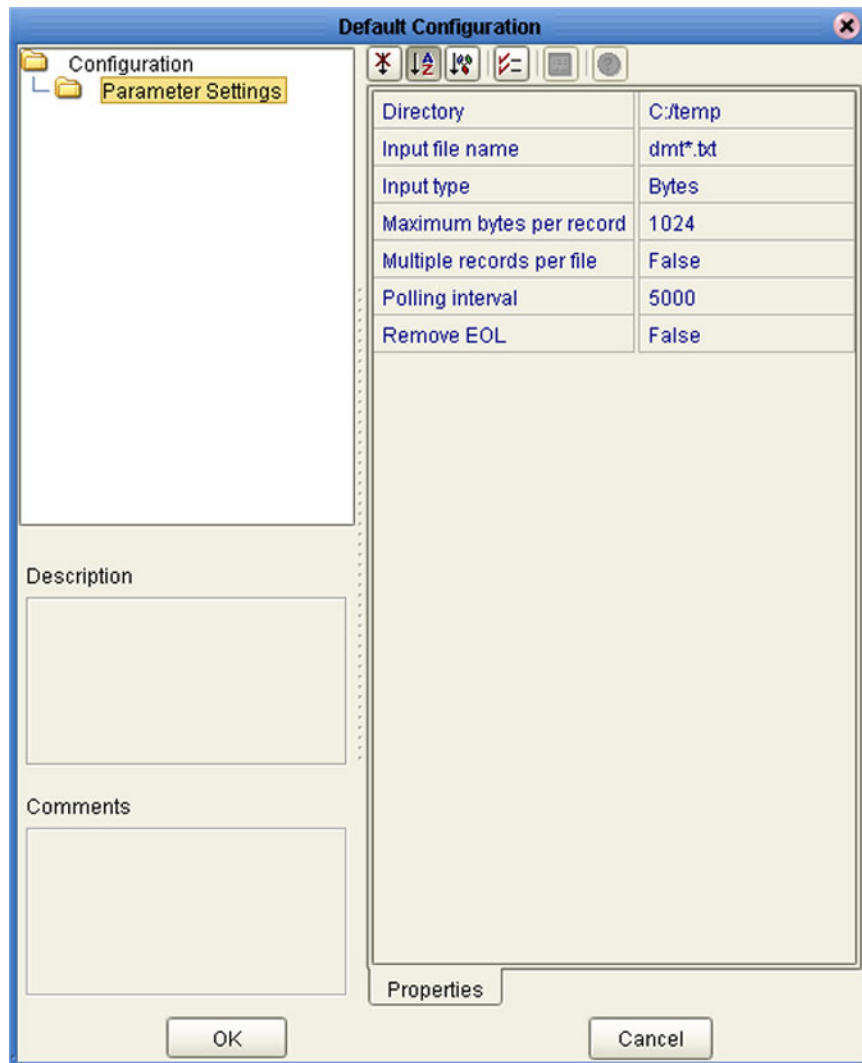


- When you link an external application with a Collaboration, the Enterprise Designer automatically adds an eWay Connection icon to the link. The eWay enables communication and movement of data between the external application and the eGate system. The eWay configuration specifies the logical connection 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.

5.7.1 Configuring a Connection

Double-clicking an eWay or JMS Client connection icon in the Connectivity Map displays the Default Configuration dialog box. As an example, Figure 57 shows a dialog box that lists the configuration properties for a File eWay.

Figure 57 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. 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 **Toolbar** contains a series of buttons used to sort and modify the information listed in the Properties folder, as described in Table 20.
- The **Description** box contains a brief description of the contents of the item currently selected in the Configuration Tree.
- The **Comments** box is for user comments about the item selected in the Configuration Tree.

Table 20 Configuration Dialog Box Toolbar Buttons

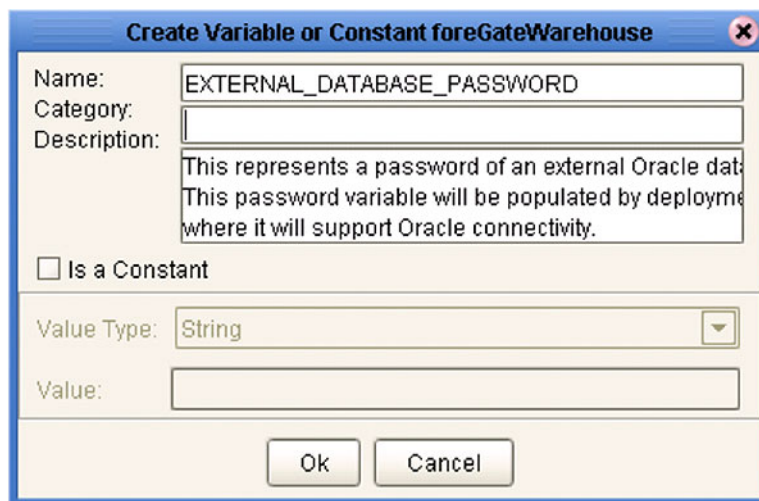
Button	Command	Function
	Unsorted	Displays configuration properties in their default order.
	Sort by Name	Sorts configuration properties alphabetically by name.
	Sort by Type	Displays configuration properties by property type.
	Show Editable Properties Only	Displays only the properties of an eWay or 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.

5.8 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 274). Project variable values can be literals or Environmental constants.

For example, Figure 58 shows a project variable defined to represent a password of a database user in a target environment. 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 58 Project Variable Creation



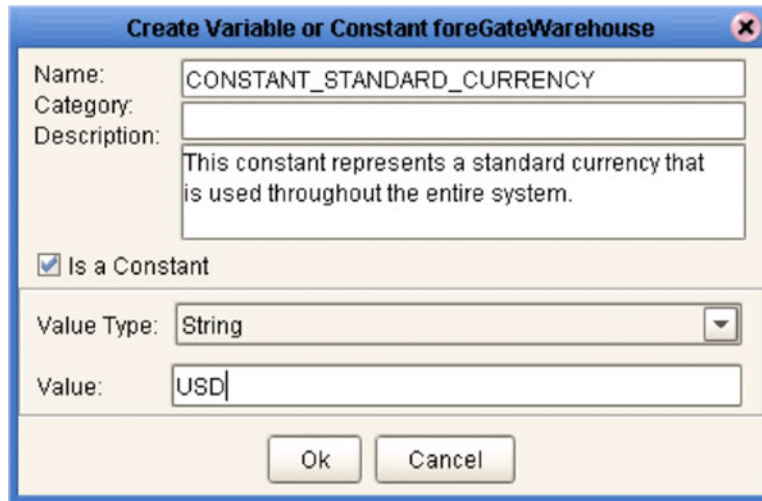
The screenshot shows a dialog box titled "Create Variable or Constant foreGateWarehouse". The dialog contains the following fields and controls:

- Name:** A text box containing "EXTERNAL_DATABASE_PASSWORD".
- Category:** An empty text box.
- Description:** A text box containing two lines of text: "This represents a password of an external Oracle dat..." and "This password variable will be populated by deployme... where it will support Oracle connectivity."
- Is a Constant:** A checkbox that is currently unchecked.
- Value Type:** A dropdown menu currently set to "String".
- Value:** An empty text box.
- Buttons:** "Ok" and "Cancel" buttons at the bottom.

Project constants are name/value pairs that are visible across the Project. For example, Figure 59 shows a standard currency defined to be used globally throughout the system.

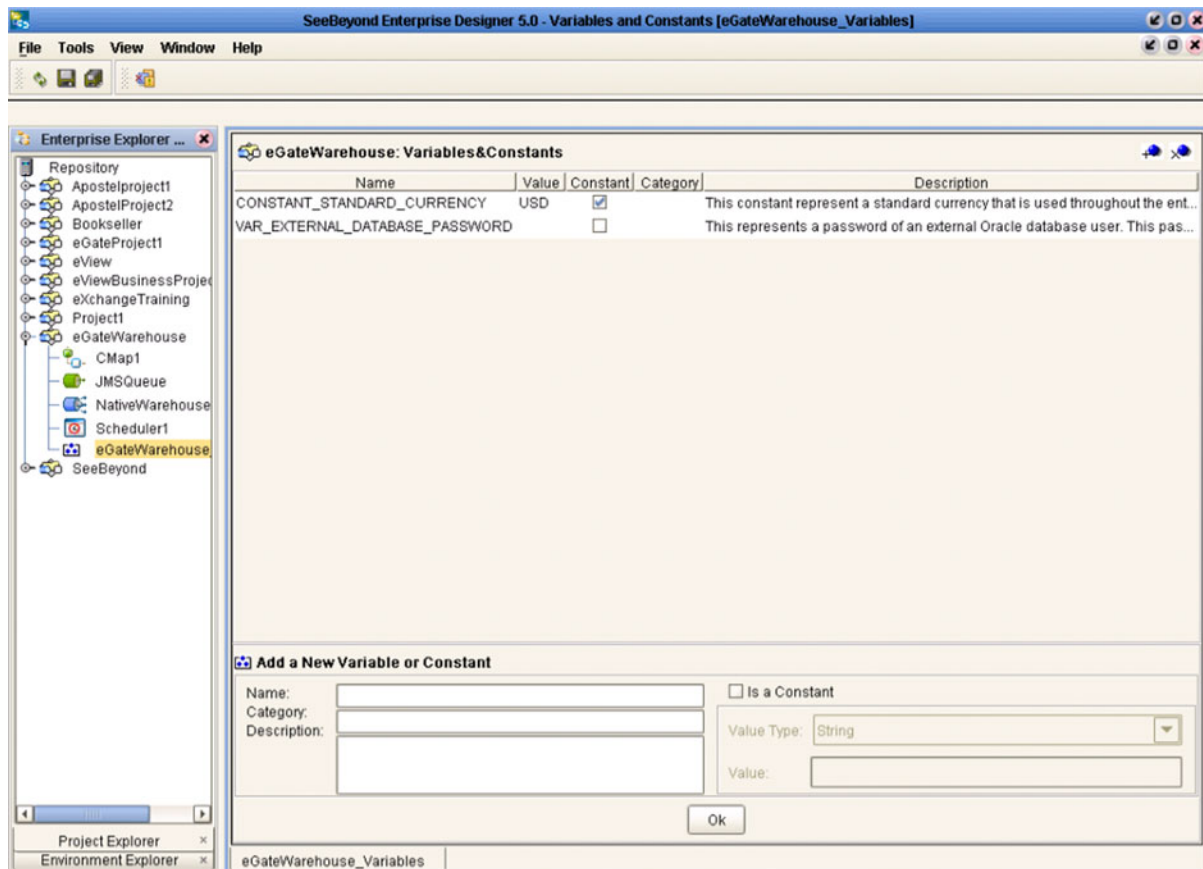
Note: When you create an Project constant, you assign a permanent value to it—which cannot be overridden.

Figure 59 Project Constant Creation



Constants and variables are automatically added to a Variables and Constants object group within the Project (see Figure 60).

Figure 60 Variables and Constants Object Group



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.

6.1 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-based 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. There is also a so-called generic interface (see [Appendix A](#)).

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.

6.1.1 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.

6.1.2 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.

6.1.3 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.

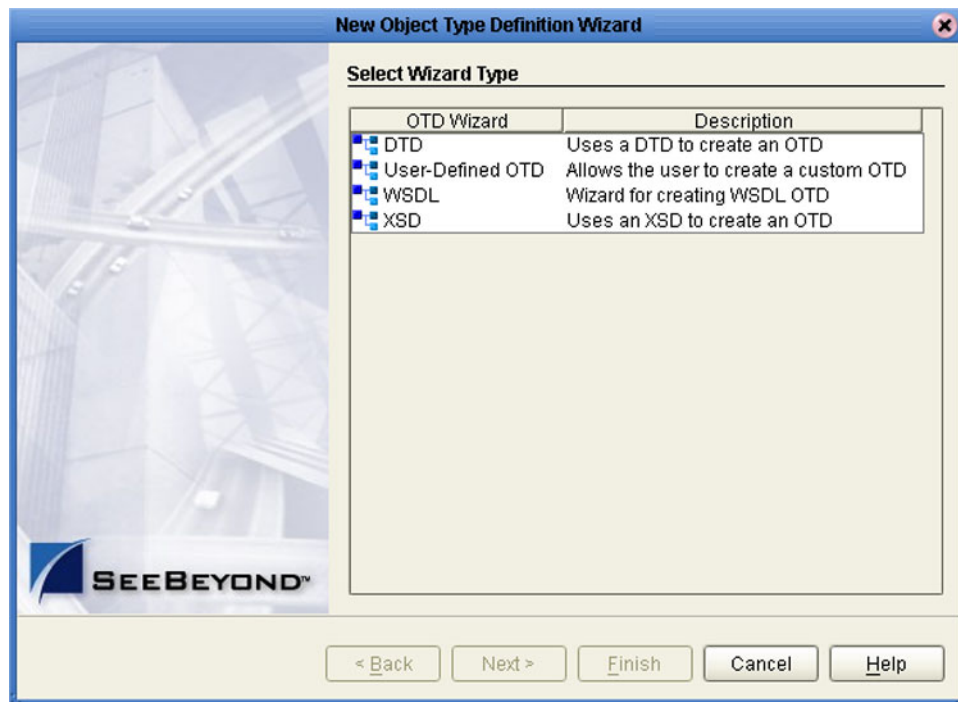
Note: *For compatibility with BPEL, the top element of an OTD must implement the `OtdRoot` and `OtdNode` interfaces; therefore, the top node cannot be a leaf node. As a result, OTDs cannot support a pure string-BLOB, since this would require the top element to be implemented as a `java.lang.String` object.*

6.2 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 61. The initial dialog allows you to select a specific OTD Wizard. The basic Wizards supplied with eGate Integrator are described in:

- [Using the DTD Wizard](#) on page 93
- [Using the User-Defined OTD Wizard](#) on page 108
- [Using the WSDL Wizard](#) on page 98
- [Using the XSD Wizard](#) on page 103

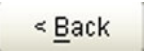

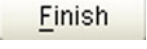
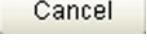

Figure 61 OTD Wizard Selection Dialog



Additional OTD Wizards are supplied with eGate add-on components, and are described in the User's Guides for the specific products. When these products are installed, the OTD Wizards are added to the list shown in Figure 61.

The OTD Wizards guide you through the initial phases of creating an Object Type Definition, and then invoke the OTD Editor. The user interface is highly self-explanatory, but details of the navigation buttons are listed in Table 21 for your reference.

Table 21 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.

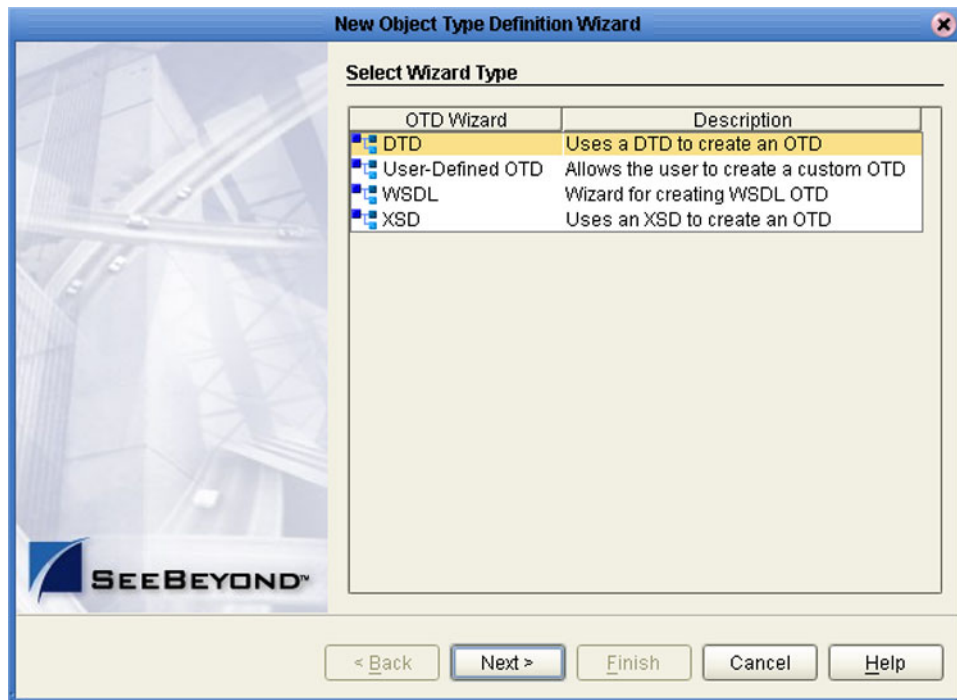
6.3 Externally-Defined OTDs

6.3.1 Using the DTD Wizard

To create an OTD file from a DTD file

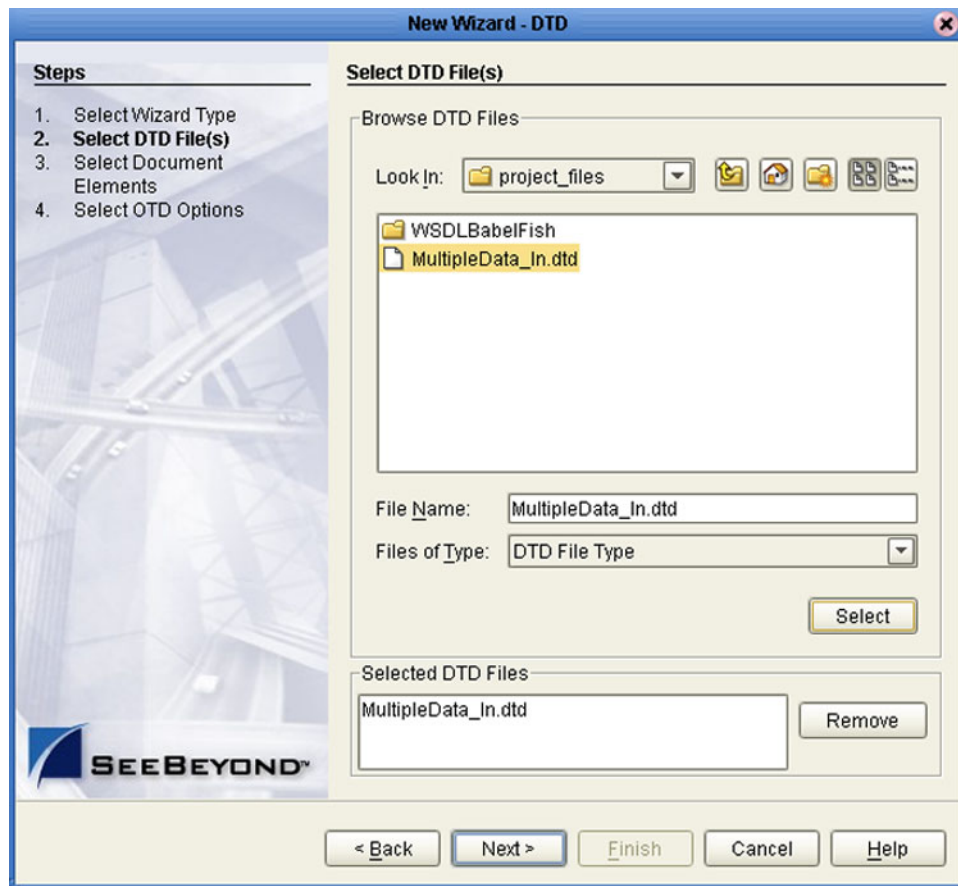
- 1 In the *Select Wizard Type* dialog, select **DTD** from the *OTD Wizard* list (see Figure 62) to create an OTD file from a Data Type Definition (DTD) file.

Figure 62 OTD Wizard Selection: DTD Wizard



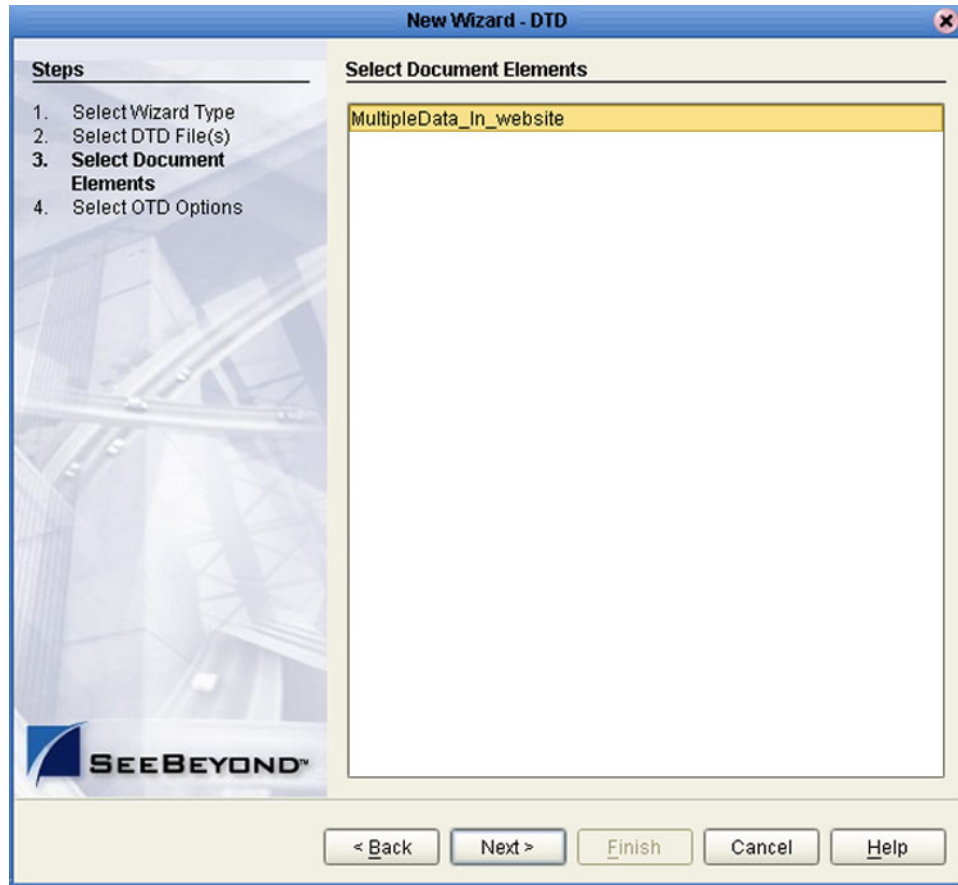
- 2 Click **Next** to display the *Select DTD File(s)* dialog box, shown in Figure 63.

Figure 63 Select DTD File(s) Dialog Box



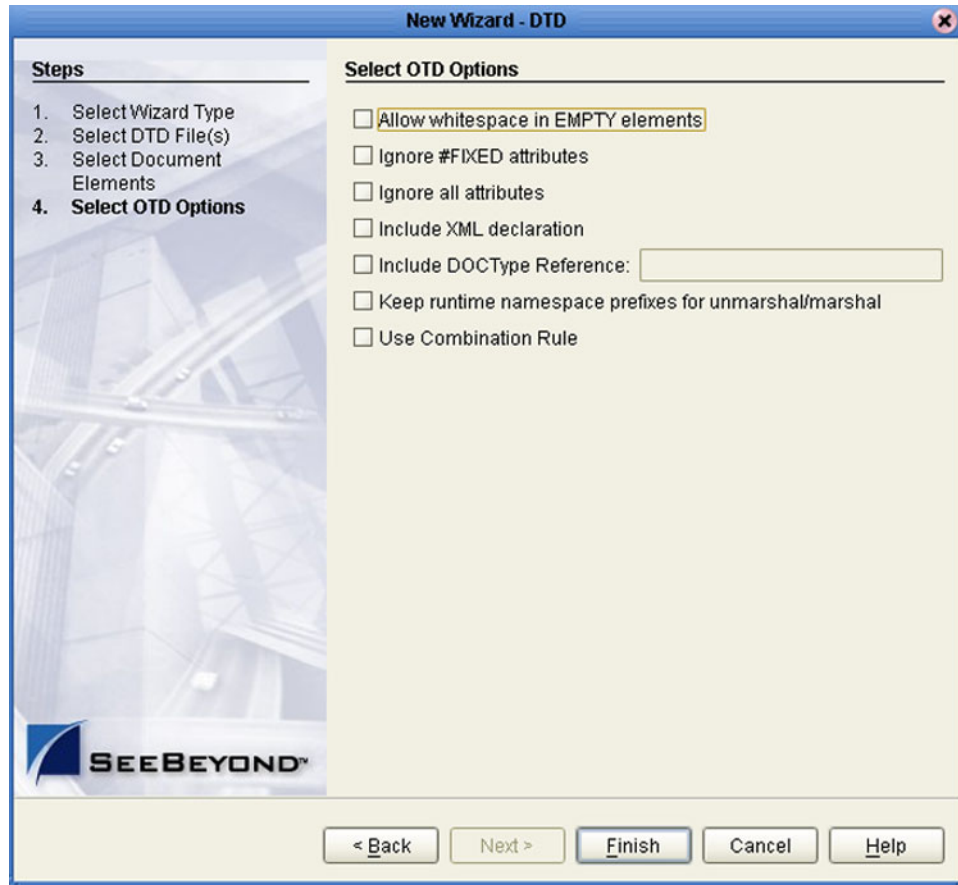
- 3 In the *Look In* drop-down list, navigate to the DTD file or files that you want to use to create the OTD. Click **Select** to add the files to the *List of Selected DTDs*.
- 4 Click **Next** to display the *Select Document Elements* dialog box, shown in Figure 64.

Figure 64 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 65.

Figure 65 Select OTD Options Dialog Box



7 Select the check boxes next to the OTD options you want to enable (see Table 22).

Table 22 DTD OTD Options

Option	Description
Allow whitespace in EMPTY elements	If an element is defined as EMPTY, this option controls whether or not white spaces are allowed within the element in the XML instance document forming the DTD.
Ignore #FIXED attributes	This option controls whether or not attributes defined as FIXED are ignored during the unmarshal and marshal processes. <ul style="list-style-type: none"> ▪ If this option is <i>not</i> selected, the attribute is recognized and saved into the OTD's runtime structure during the unmarshal process, and also appears in the output during the marshal process. ▪ If this option is selected, the attribute is ignored and neither of the above occurs.

Option	Description
Ignore all attributes	This option controls whether or not all attributes are ignored during the unmarshal and marshal processes. If both this option and the <i>Keep runtime namespace prefixes ...</i> option (below) are selected, only namespace attributes will be handled during the unmarshal process and consequently presented in the output during the marshal process. (The <i>namespace</i> attribute has the form xmlns:XX .)
Include XML declaration	This option controls whether or not the XML declaration <?xml version="1.0" encoding="....."?"> appears in the output during the marshal process.
Include DOC Type Reference	This option controls whether or not the "<!DOCTYPE ..." string appears in the output during the marshal process.
Keep runtime namespace prefixes for unmarshal/marshal	<p>This option controls whether or not the namespace prefixes used during the marshal process are identical to those used in the unmarshal process.</p> <ul style="list-style-type: none"> ▪ If this option is selected, all namespace attributes will be preserved once they appear in the XML instance document, and the namespace prefixes used in the marshal process will be exactly as they were presented in the XML document during the unmarshal process. ▪ If this option is <i>not</i> selected, then the namespace prefixes used in the marshal process might be different than the ones presented in the XML document during the unmarshal process (for example, the namespace prefixes that are presented in the XSD file might be used). <p>Note: A consequence of selecting this option is that if there is no unmarshal process performed before the marshal process, then there will be no namespace attributes presented in the output (see the comment for the option below).</p>
Use Combination Rule	Not currently used.

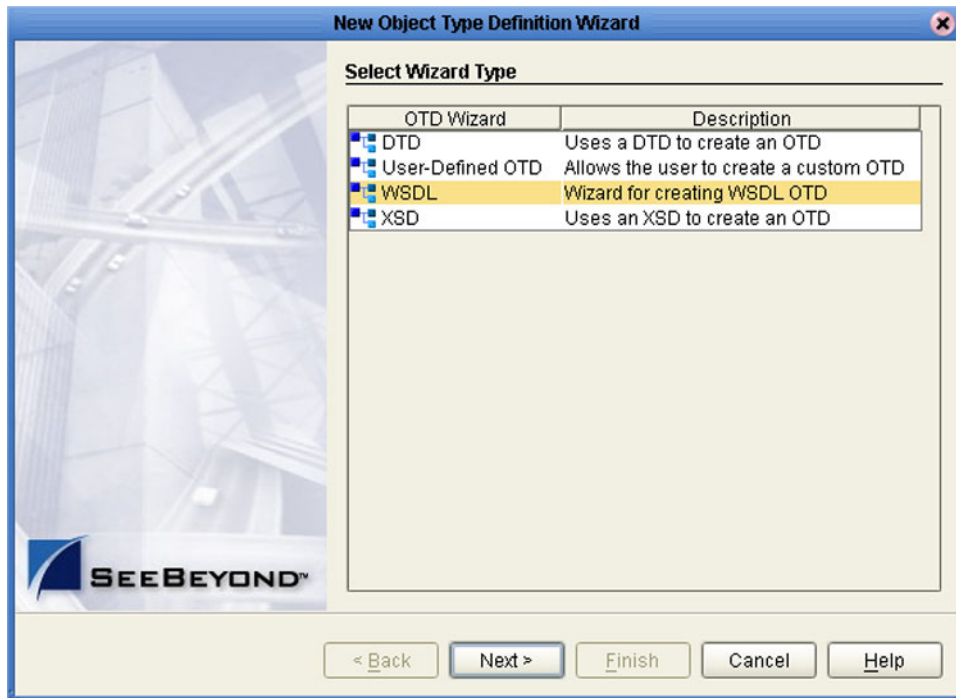
8 Click **Finish** to add the OTD to the Project.

6.3.2 Using the WSDL Wizard

To create an OTD file from a WSDL file

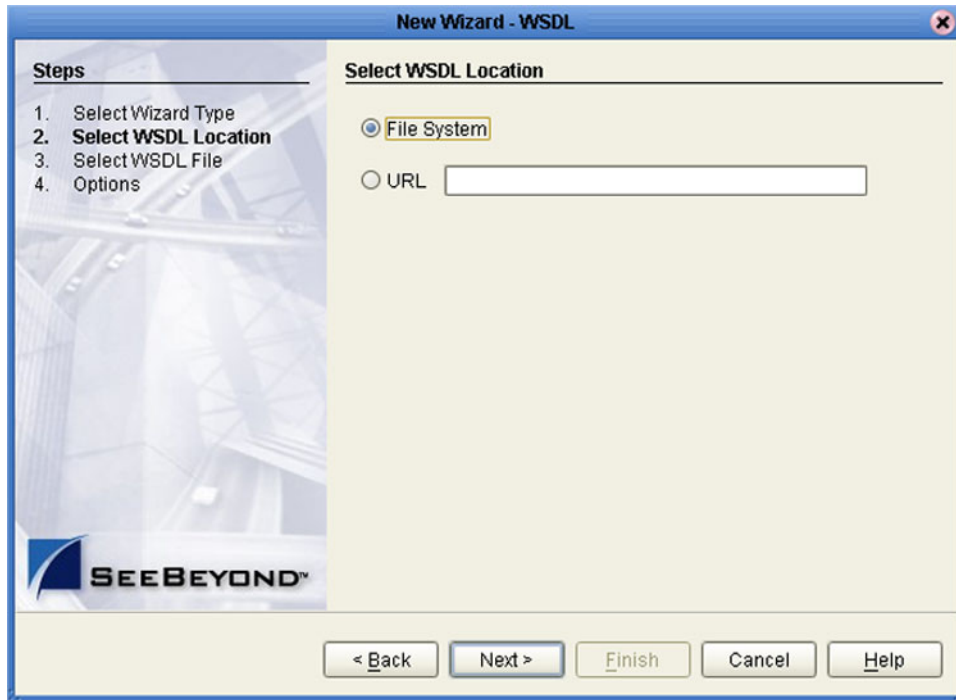
- 1 In the *Select Wizard Type* dialog, select **WSDL** from the *OTD Wizard* list (see Figure 66) to create an OTD file from an WSFL file.

Figure 66 OTD Wizard Selection: WSDL Wizard



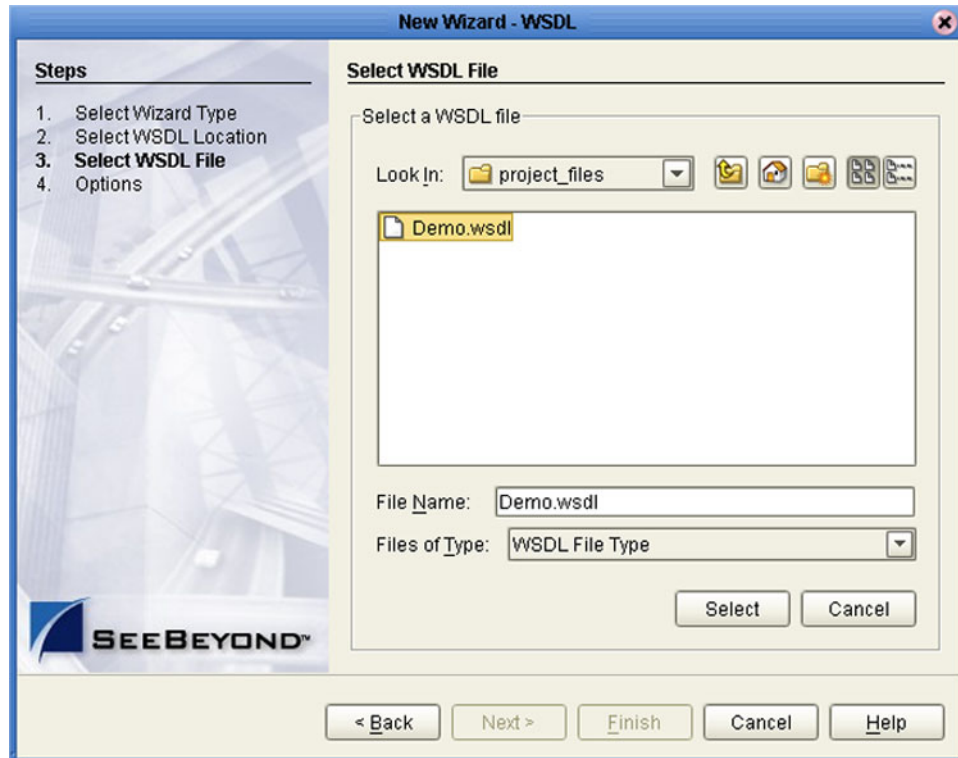
- 2 Click **Next** to display the *Select WSDL File Location* dialog, shown in Figure 67

Figure 67 WSDL Wizard: Select WSDL Location



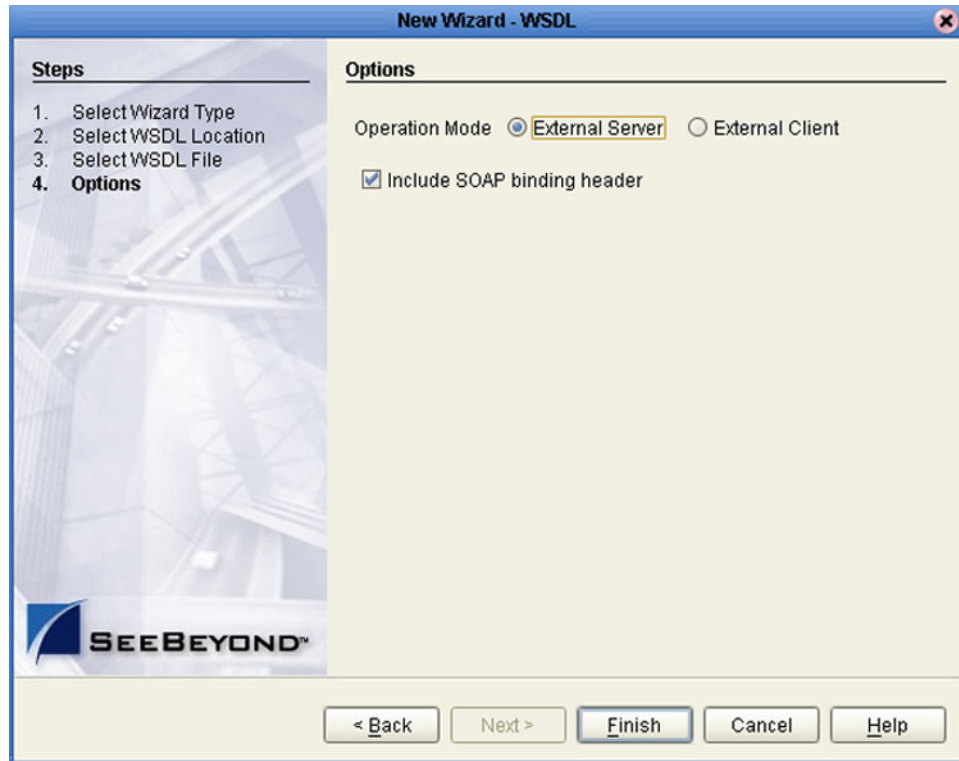
- 3 In the *Select WSDL Location* dialog, select **File System** or enter a **URL**, depending upon where your WSDL file is located.
- 4 Click **Next** to display the *Select WSDL File* dialog, shown in Figure 68.

Figure 68 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 *Options* dialog, shown in Figure 69.

Figure 69 WSDL Wizard: Select OTD Options



- 7 Select the check boxes next to the OTD options you want to enable:
 - ◆ If you are using a Web client, select **External Server**.
 - ◆ If you are using a Web server, select **External Client**.
 - ◆ To include the SOAP binding header in the WSDL file, select the check box.
- 8 Click **Finish** to add the OTD to the Project.

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.

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:

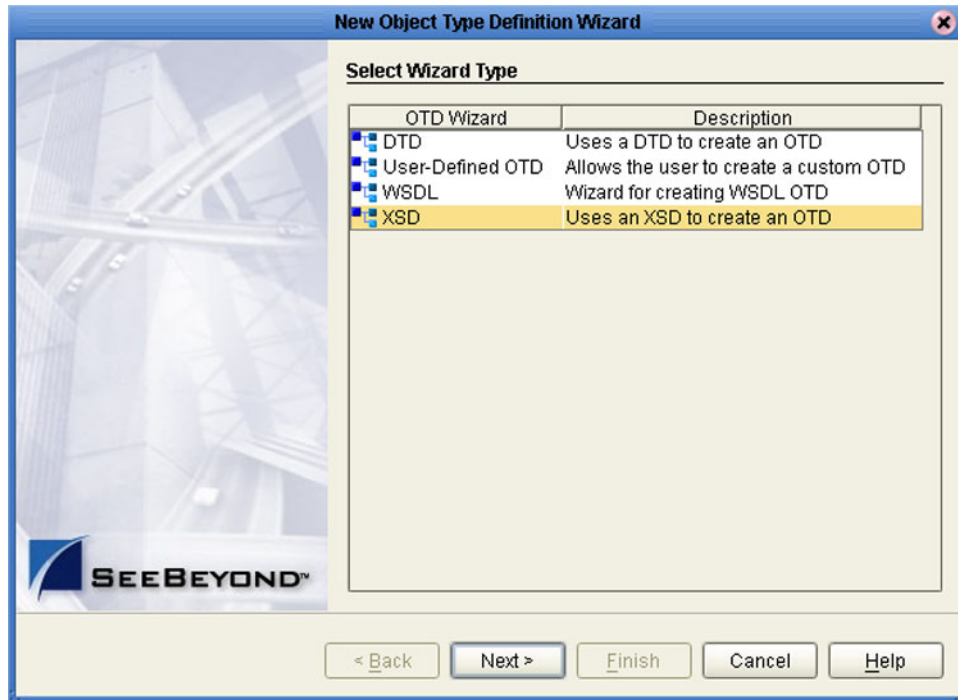
- **One-way:** The server receives a message from the client; also referred to as “fire and forget.”
- **Request-response:** The server receives a message from the client and sends a correlated message back

6.3.3 Using the XSD Wizard

To create an OTD file from an XSD file

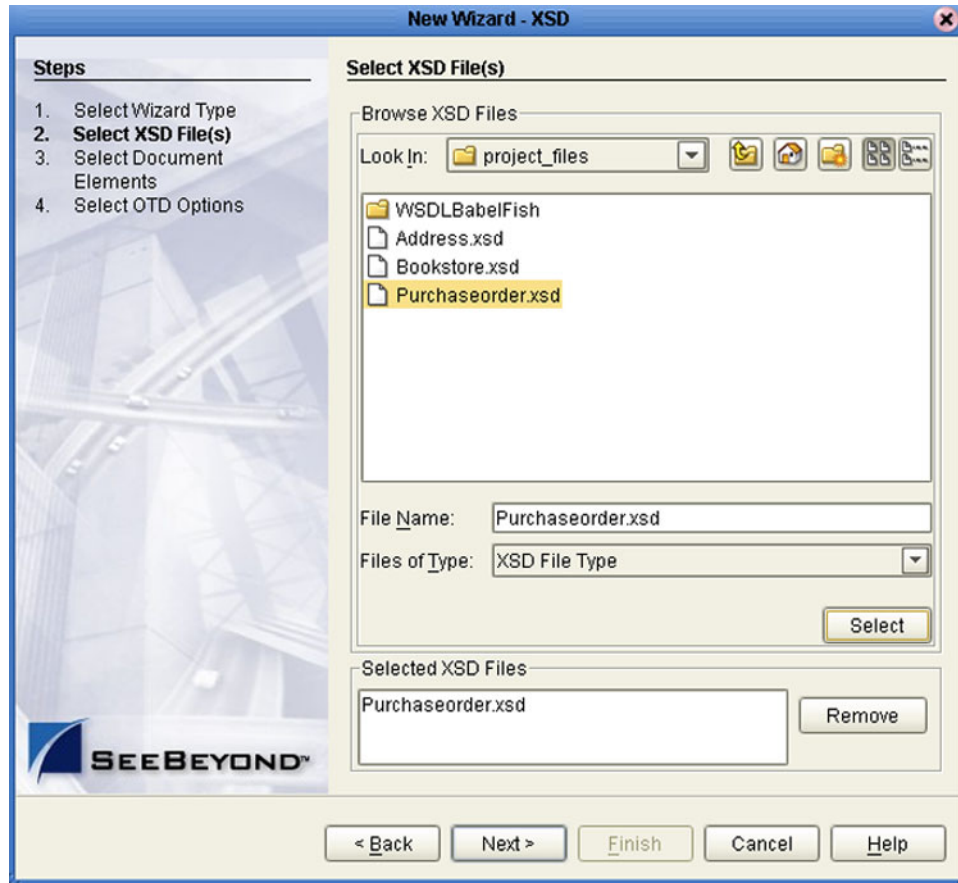
- 1 In the *Select Wizard Type* dialog, select **XSD** from the *OTD Wizard* list (see Figure 70) to create an OTD file from an XSD file.

Figure 70 OTD Wizard Selection: XSD Wizard



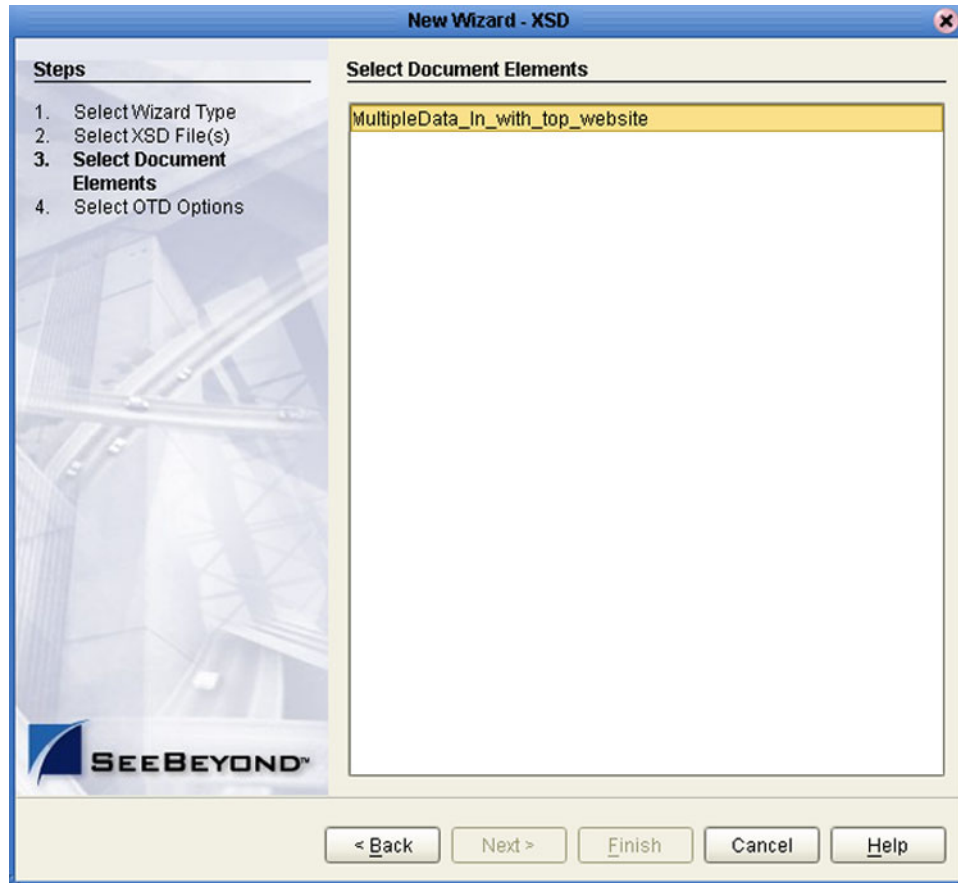
- 2 Click **Next** to display the Select XSD File(s) dialog box, shown in Figure 71.

Figure 71 XSD Wizard: Select XSD File(s)



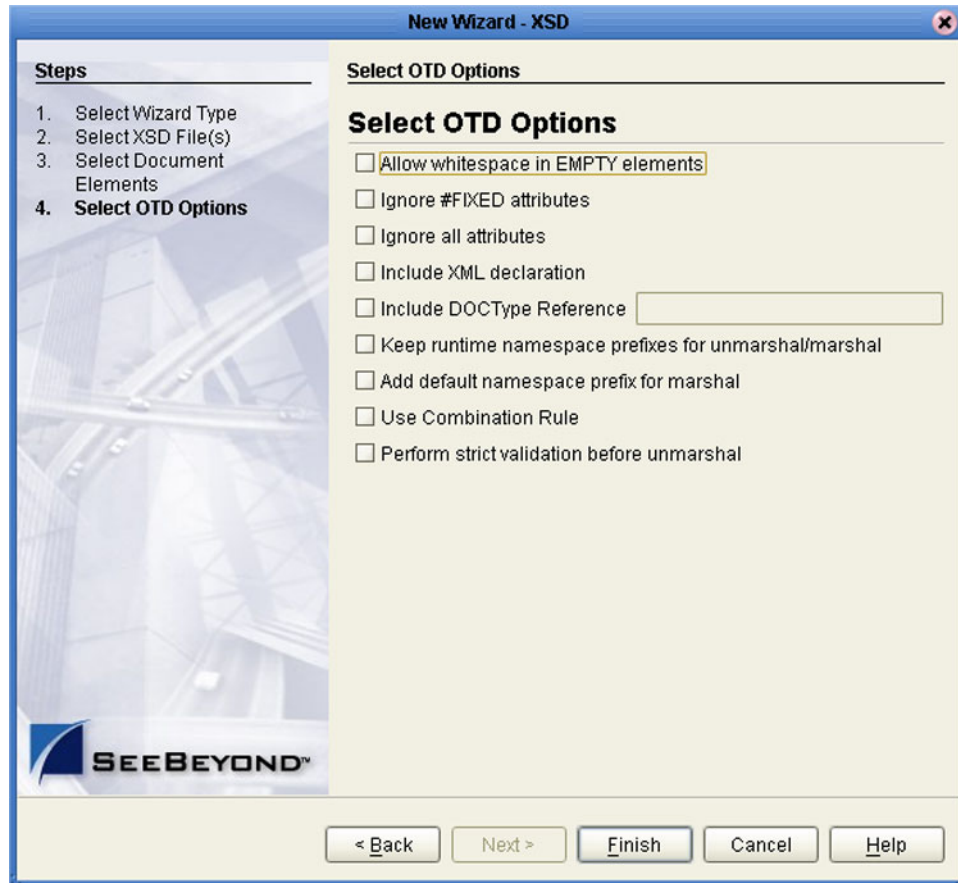
- 3 In the *Look In* drop-down list, navigate to the XSD file or files that you want to use to create the OTD. Click **Select** to add the files to the *List of Selected XSDs*.
- 4 Click **Next** to display the *Select Document Elements* dialog box, shown in Figure 64.

Figure 72 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 65.

Figure 73 Select OTD Options Dialog Box



7 Select the check boxes next to the OTD options you want to enable (see Table 23).

Table 23 XSD OTD Options

Option	Description
Allow whitespace in EMPTY elements	Not currently used for XSD OTDs.
Ignore #FIXED attributes	<p>This option controls whether or not attributes defined as FIXED are ignored during the unmarshal and marshal processes.</p> <ul style="list-style-type: none"> ▪ If this option is <i>not</i> selected, the attribute is recognized and saved into the OTD's runtime structure during the unmarshal process, and also appears in the output during the marshal process. ▪ If this option is <i>is</i> selected, the attribute is ignored and neither of the above occurs.

Option	Description
Ignore all attributes	This option controls whether or not all attributes should be ignored in the unmarshal and marshal processes. If both this option and the <i>Keep runtime namespace prefixes ...</i> option (below) are selected, only namespace attributes will be handled during the unmarshal process and consequently presented in the output during the marshal process. (The <i>namespace</i> attribute has the form xmlns:XX .)
Include XML declaration	This option controls whether or not the XML declaration <?xml version="1.0" encoding="....."?)> appears in the output during the marshal process.
Include DOC Type Reference	Not currently used for XSD OTDs.
Keep runtime namespace prefixes for unmarshal/marshal	<p>This option controls whether or not the namespace prefixes used during the marshal process are identical to those used in the unmarshal process.</p> <ul style="list-style-type: none"> ▪ If this option is selected, all namespace attributes will be preserved once they appear in the XML instance document, and the namespace prefixes used in the marshal process will be exactly as they were presented in the XML document during the unmarshal process. ▪ If this option is <i>not</i> selected, then the namespace prefixes used in the marshal process might be different than the ones presented in the XML document during the unmarshal process (for example, the namespace prefixes that are presented in the XSD file might be used). <p>Note: A consequence of selecting this option is that if there is no unmarshal process performed before the marshal process, then there will be no namespace attributes presented in the output (see the comment for the option below).</p>
Add default namespace prefix for marshal	<p>This option controls whether or not the prefix of the default target namespace of an element is applied to the element during the marshal process.</p> <ul style="list-style-type: none"> ▪ If both this flag and the <i>Keep runtime namespace prefixes ...</i> option (above) are selected, then the default target namespace of an element will be applied to the element during the marshal process, <i>if it is a root element</i>. ▪ If the <i>Keep runtime namespace prefixes ...</i> option is <i>not</i> selected, then the elements are qualified based on the XSD definition and this flag has no effect.
Use Combination Rule	Not currently used.
Perform strict validation before unmarshal	Not currently used.

8 Click **Finish** to add the OTD to the Project.

6.4 User-Defined OTDs

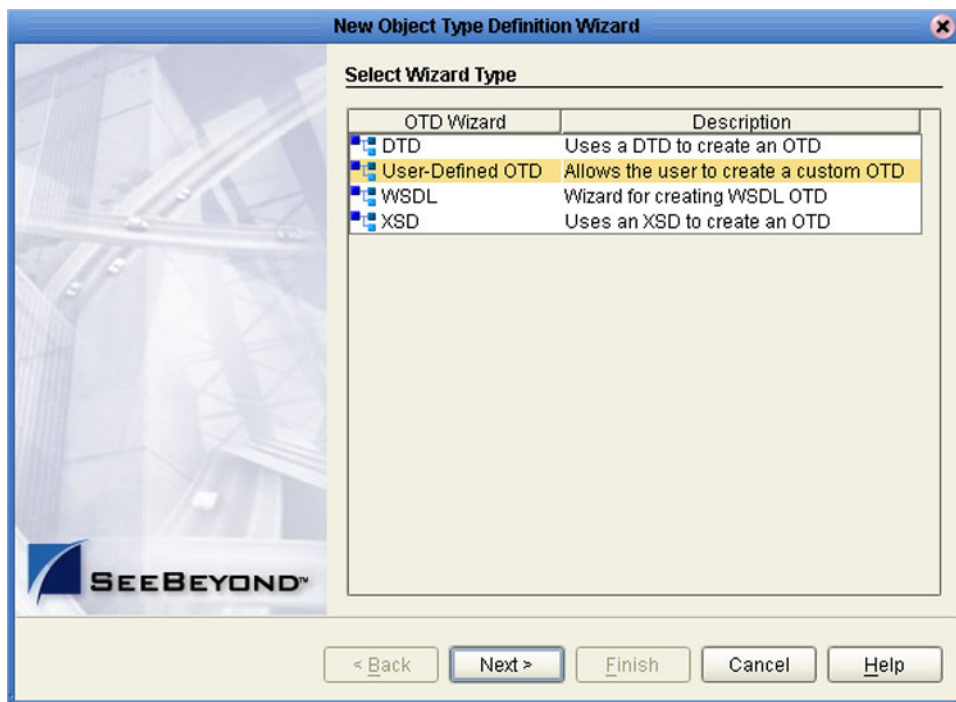
The User-Defined OTD Wizard creates an OTD from your specifications.

6.4.1 Using the User-Defined OTD Wizard

To create a User-Defined OTD

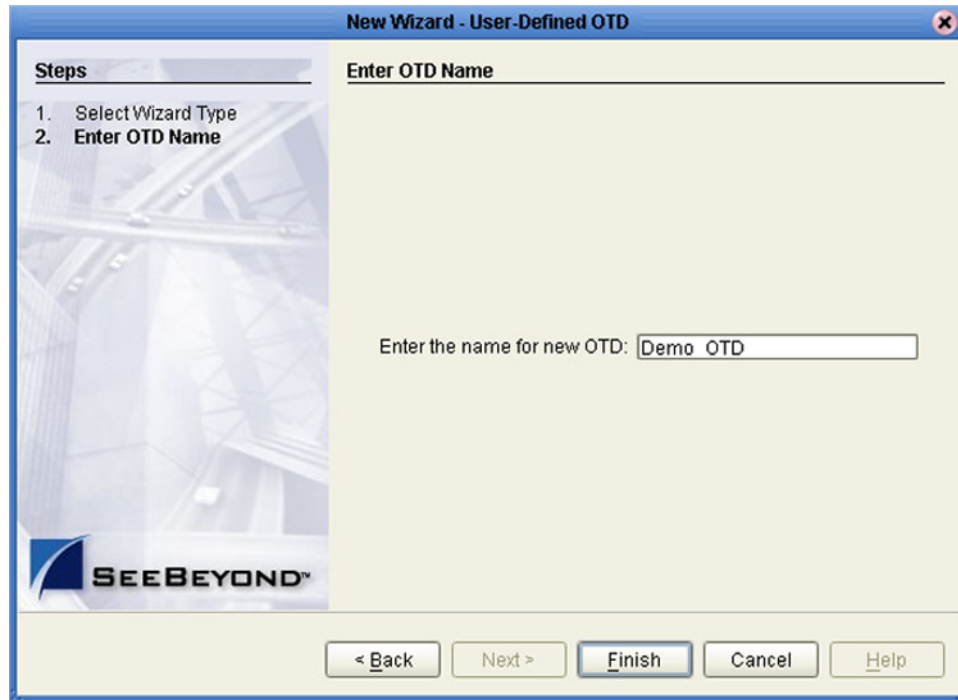
- 1 In the *Select Wizard Type* dialog, select **User-Defined OTD** from the *OTD Wizard* list (see Figure 74) to create an OTD file without using a source file.

Figure 74 OTD Wizard Selection: User-Defined OTD



- 2 Click **Next** to display the *Enter OTD Name* dialog box, shown in Figure 75.

Figure 75 Enter OTD Name



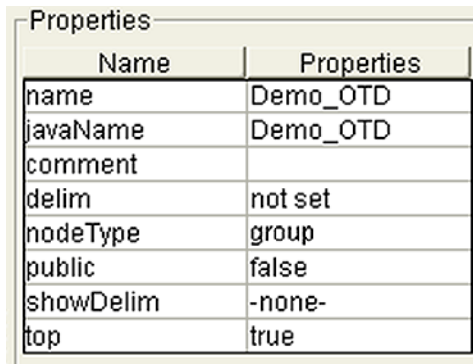
- 3 Enter a name for the OTD into the text box provided, then click **Finish** to add the OTD to the Enterprise Designer.

6.4.2 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 76.

Figure 76 User-Defined OTD Node Properties



Properties	
Name	Properties
name	Demo_OTD
javaName	Demo_OTD
comment	
delim	not set
nodeType	group
public	false
showDelim	-none-
top	true

Table 24 Node Properties

Name	Description
name	Node display name (see The Bean-like Interface on page 89).
javaName	Property accessor basename (see The Bean-like Interface on page 89).
comment	Free-form text (no run-time effect).
delim	Delimiter specification (see Specifying Delimiters on page 114).
nodeType	Governs the marshal/unmarshal format (see Specifying the Node Type on page 113)
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 77.

Figure 77 User-Defined OTD Element Properties

Properties	
Name	Properties
name	element_1
javaName	Element1
optional	false
repeat	false
comment	
delim	not set
nodeType	delim
showDelim	-none-

Table 25 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 114).
nodeType	Governs the marshal/unmarshal format.
showDelim	If nodeType is delimited,

Important: Do not modify the `javaName` property.

Field Properties

The set of properties associated with the field level is shown in Figure 78.

Figure 78 User-Defined OTD Field Properties

Name	Properties
name	field_2
javaName	Field2
javaType	java.lang.String
optional	false
repeat	false
comment	
delim	not set
encoding	
initial	
match	
nodeType	delim
showDelim	

Table 26 Field Properties

Name	Description
name	Field display name.
javaName	Property accessor basename.
javaType	Java type; can be either java.lang.String or byte array (byte[]).
optional	Flag: Can the field 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 114).
encoding	(Reserved for future use, in encoding of fields.)
initial	Initial field value, set when the parent node is created or reset. When provided, it is assigned to the node when node is not populated with any data.
match	If nodeType is <i>delimited</i> or <i>fixed</i> , performs exact match to the data.
nodeType	Governs the marshal/unmarshal format.
showDelim	If nodeType is <i>delimited</i> , shows the delimiter.

Important: Do not modify the *javaName* property.

6.4.3 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 79). Descriptions of the property options are listed in Table 27.

Figure 79 Node Type Property Options

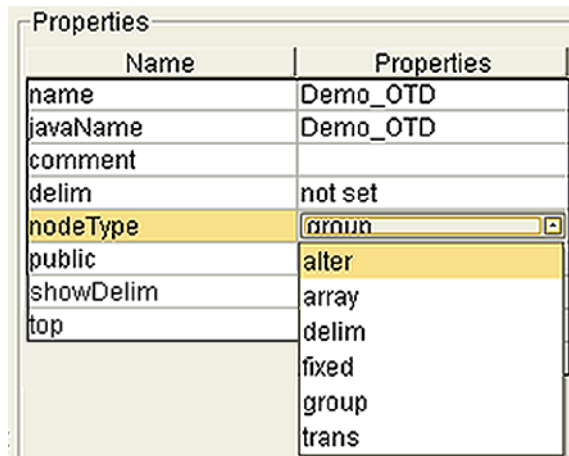


Table 27 Node Type Property Options

Option	Description	Element	Field	Internal
alter	Alter (<i>alternate</i>) selects one child or the other. One child is always present after the unmarshal operation. Applies only to elements.	Yes	No	choice
array	Array is a 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	Delim (<i>delimited</i>) structure. If repeated, occurrences are separated by a <i>normal</i> delimiter.	Yes	Yes	simple or group
fixed	Fixed indicates a fixed length, which is specified by non-negative integer (or zero to indicate end of parent node data).	Yes	Yes	simple or group
group	Group provides organizational grouping for purposes such as repetition. Applies only to elements.	Yes	No	group
trans	Trans (<i>transient</i>) appears only in an internal tree as a scratch pad field. It does not appear in external data representation, and can only have <i>trans</i> nodeTypes as children.	Yes	Yes	choice, simple, or group

Note: If you move a User-Defined OTD node, you must reset the nodeType for that node.

6.4.4 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, and you can specify a delimiter list on any node within the OTD—not only the root node as shown in the example. See [Creating a Delimiter List](#) on page 121 for a description of the procedure for creating a Delimiter List.

Delimiter Properties

Delimiters are defined using the Delimiter List Editor (see Figure 80).

Figure 80 Delimiter List Editor

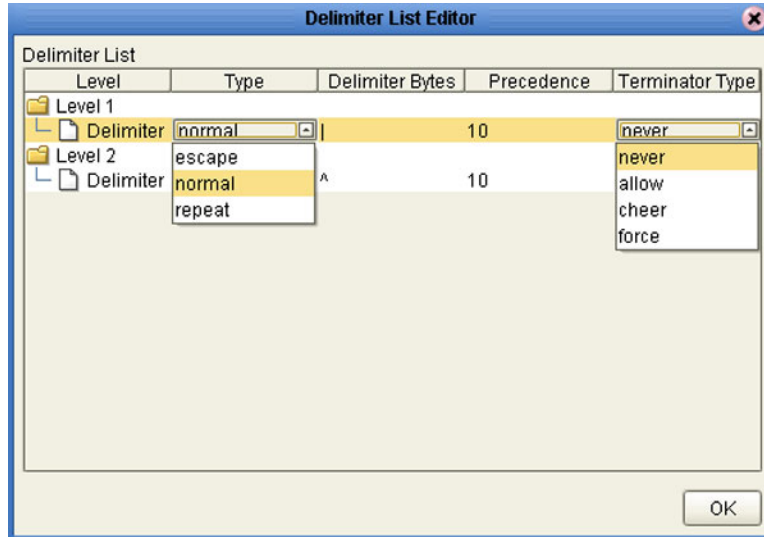


Table 28 Delimiter Properties and Value Options

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 120.
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 28 Delimiter Properties and Value Options

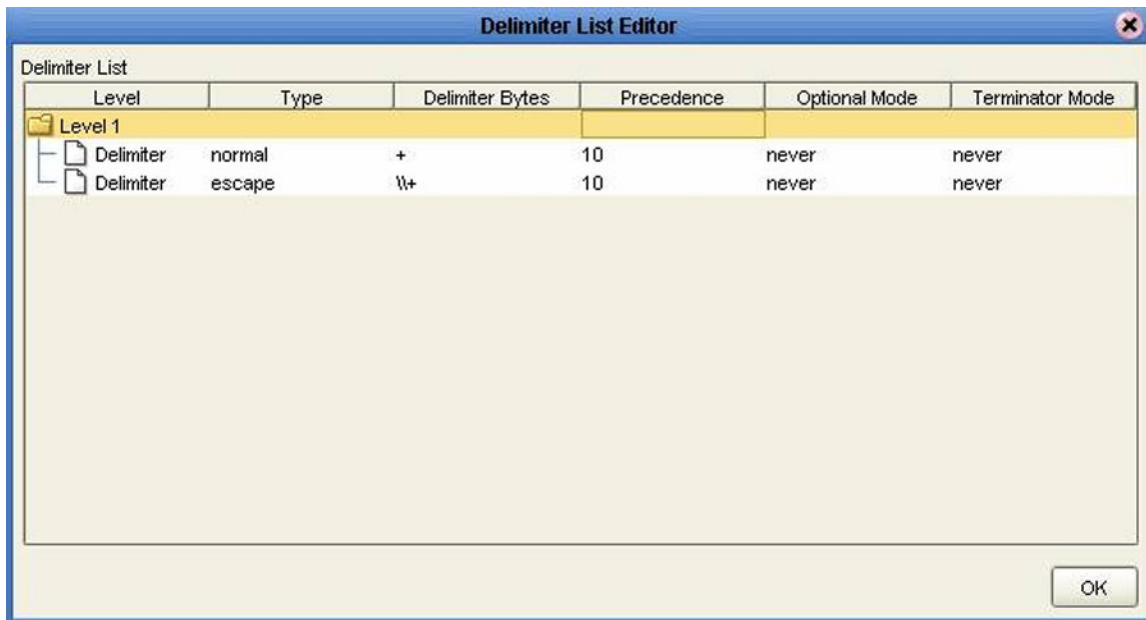
Property	Option	Description
Optional	never	Do not allow on input, do not emit on output (empty field between delimiters implies zero length data field).
	allow	Skip empty field if present; if absent, do not delimit on output.
	cheer	Skip empty field if present; if absent, do delimit on output.
	force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

Type Property - Escape Option

An *escape* delimiter is simply a sequence that is recognized *and ignored* during parsing. Its purpose is to allow the use of escape sequences to embed byte sequences in data that would otherwise be seen as delimiter occurrences.

For example, if there is a normal delimiter “+” at a given level, and we define an escape delimiter “\+” (see Figure 81), then **aaa+b\+c+ddd** will parse as three fields: **aaa**, **b\+c**, and **ddd**. If the escape delimiter were not defined, the sequence would then parse as four fields: **aaa**, **b**, **c**, and **ddd**.

Figure 81 Escape Type Delim

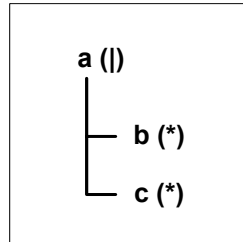


If there is *only* an escape delimiter on a given level, however, it presents a *no delimiter defined* situation for **delim** and **array** nodes.

Terminator Type Property

Consider the tree structure shown in Figure 82, where the node **a** has a pipe (|) as its delimiter, the sub-nodes **b** and **c** have asterisks (*) as their delimiters.

Figure 82 Terminal Type Property Example



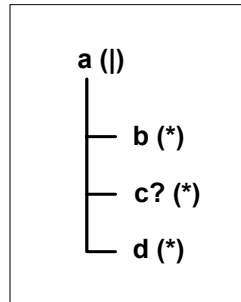
Option	Input	Output
never	c	c
allow	c or c*	c
cheer	c or c*	c*
force	c*	c*

Optional Property

Consider the tree structures shown in Figure 83 and Figure 84, 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.)

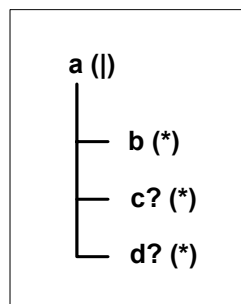
Figure 83 Optional Property (Example 1)



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 84 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**

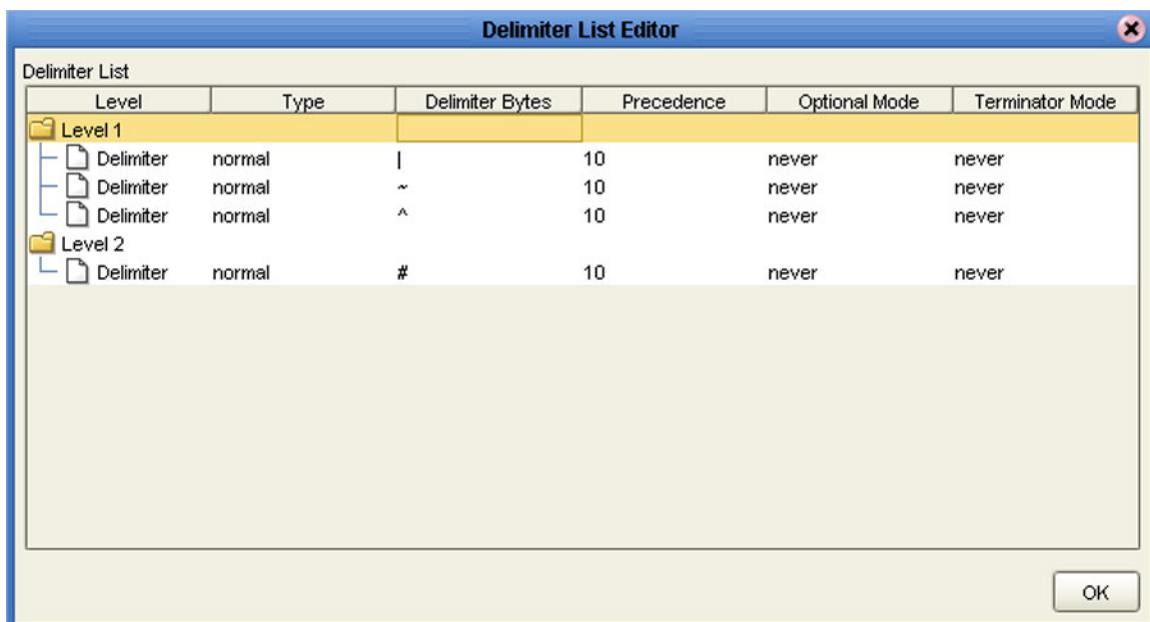
Multiple Delimiters

You can specify multiple delimiters at a given level; for example, if you specify |, ~, and ^ as delimiters for a specific level (see Figure 85), the parser will accept any of these delimiters:

- *root node*
 - ♦ element (delimiters = "|", "~", "^")
 - ♦ field (delimiter = "#")
 - ♦ field (delimiters = "|", "~", "^")

This will successfully parse the data `abc|def`, `abc~def`, and `abc^def`.

Figure 85 Multiple Delimiter Example



Escape Delimiters

The following escape delimiters are allowed.

Table 29 Escape Delimiters

Delimiter	Description
\b	Backspace
\n	Newline
\r	Carriage return
\t	Tab
\0HH	Hexadecimal number

Precedence

Precedence (see [Figure 93 on page 124](#)) 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)

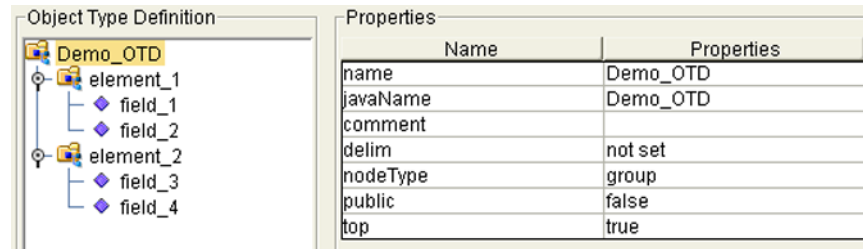
This will successfully parse the text 'abcde^zyxvuABCDEFGH'.

6.4.5 Creating a Delimiter List

To create a delimiter list

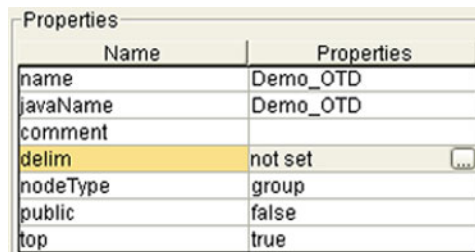
- 1 Create the structure of this example OTD in the User-Defined OTD editor as shown in Figure 86.

Figure 86 Example User-Defined OTD



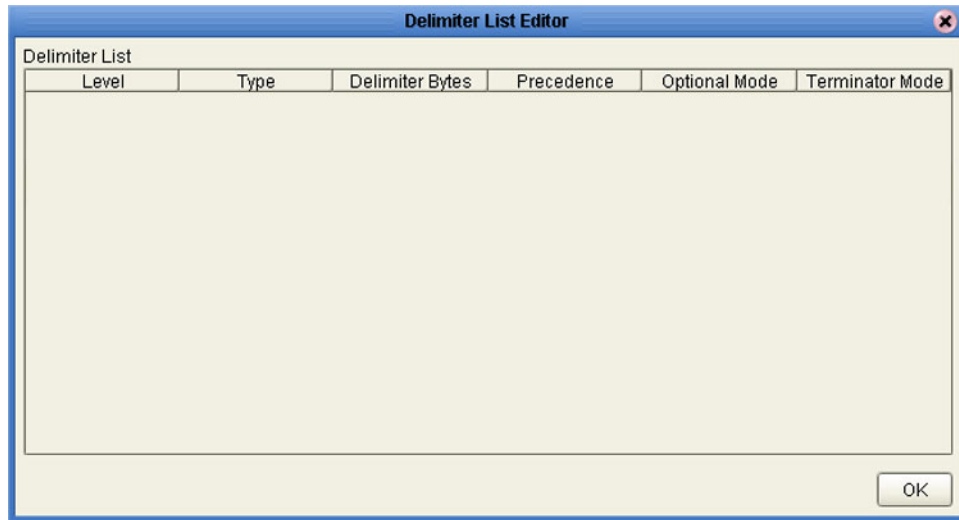
- 2 Select the node to which the delimiter list will apply.
- 3 Double-click on the value of the **delim** property to activate the property field for editing (see Figure 87). Initially it appears as *not set*.

Figure 87 Activate Field



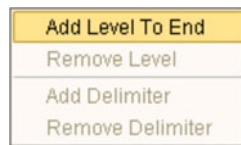
- 4 Click on the ellipsis (...) button to display the Delimiter List Editor, which is blank when the delimiter list is *not set* (see Figure 88).

Figure 88 Delimiter List Editor (1)



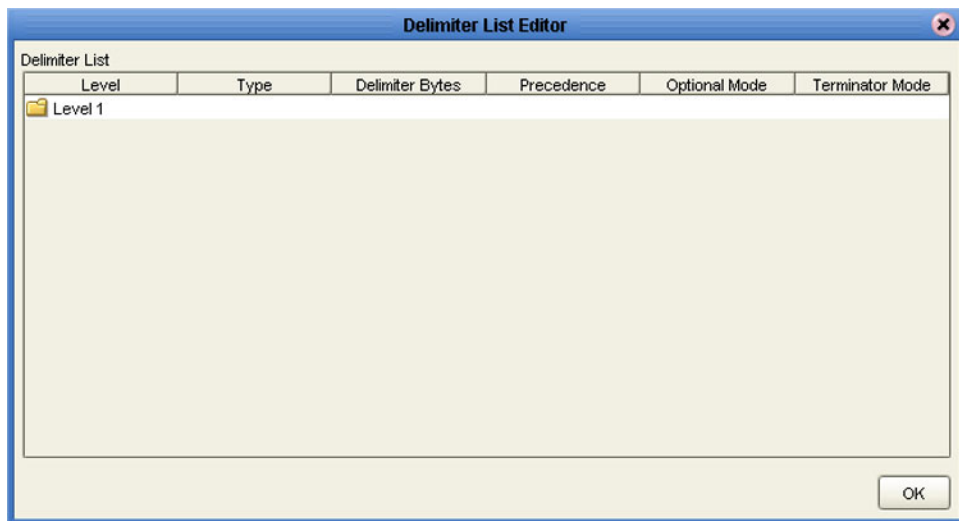
- 5 Right-click anywhere within the table area to invoke the action menu (see Figure 89).

Figure 89 Action Menu



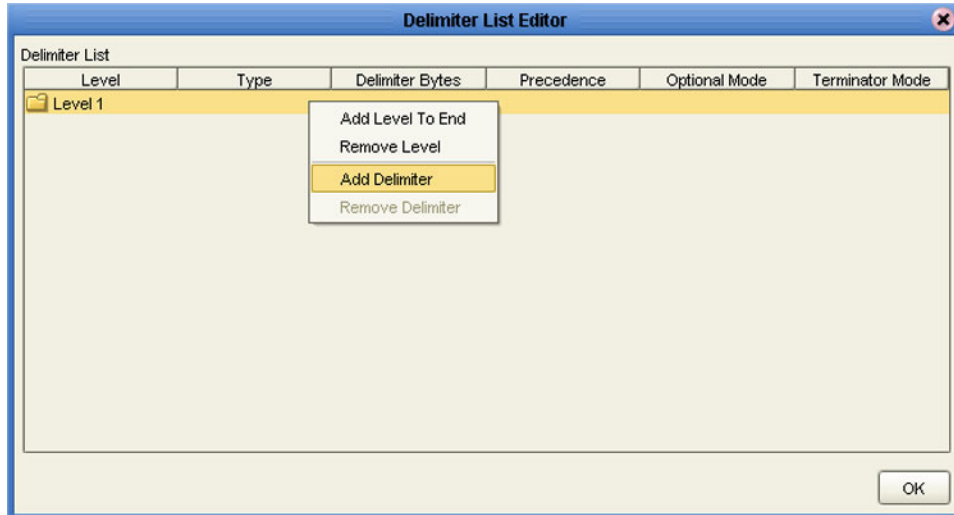
- 6 Levels can be added and removed in the delimiter list, and delimiters can be added or removed from under a given level, using the action menu options. Using the **Add Level To End** option, you can add a level the delimiter list as shown in Figure 93.

Figure 90 Delimiter List Editor (2)



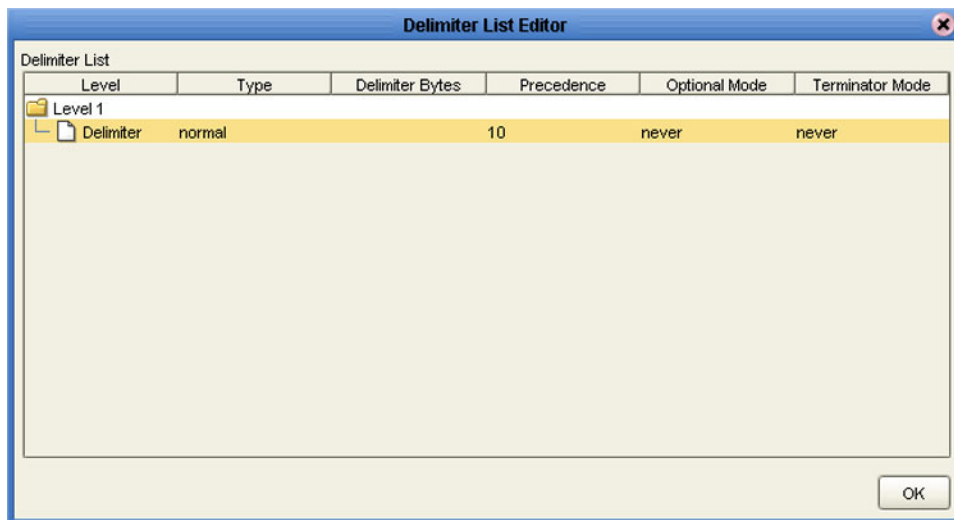
- 7 Select (left-click) a level and then right-click within the level to again display the action menu, as shown in Figure 91.

Figure 91 Delimiter List Editor (3)



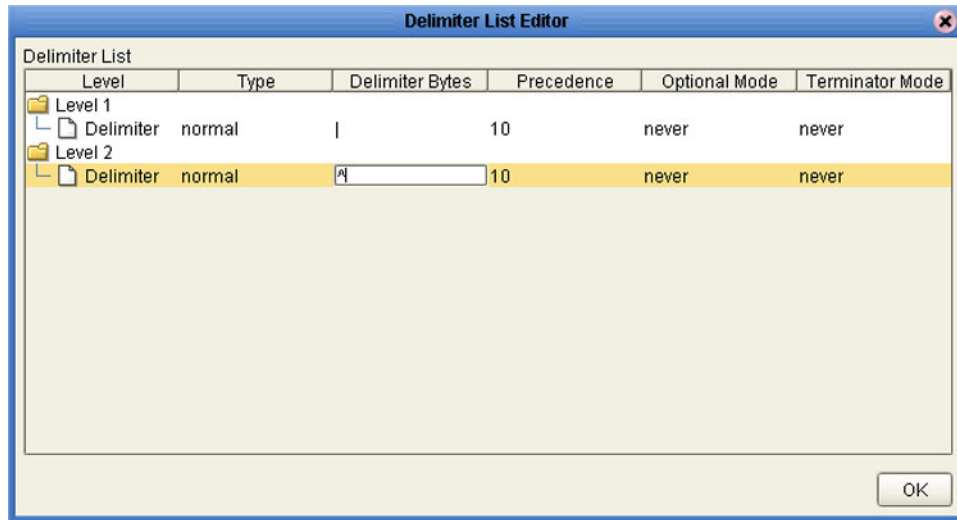
- 8 Select **Add Delimiter** from the menu options to add a delimiter to the selected level, as shown in Figure 92.

Figure 92 Delimiter List Editor (4)



- 9 Continue adding levels and delimiters as required. You define the delimiters by clicking under the **Delimiter Bytes** column to open the field for editing. An example delimiter list with delimiters defined is shown in Figure 93.

Figure 93 Delimiter List Editor (5)



- 10 Click the **OK** button to complete the edit. The **delim** property's value will now appear as *specified*, as shown in Figure 94.

Figure 94 Delimiter Specified

repeat	raise
comment	
delim	specified
nodeType	group
nonRootType	

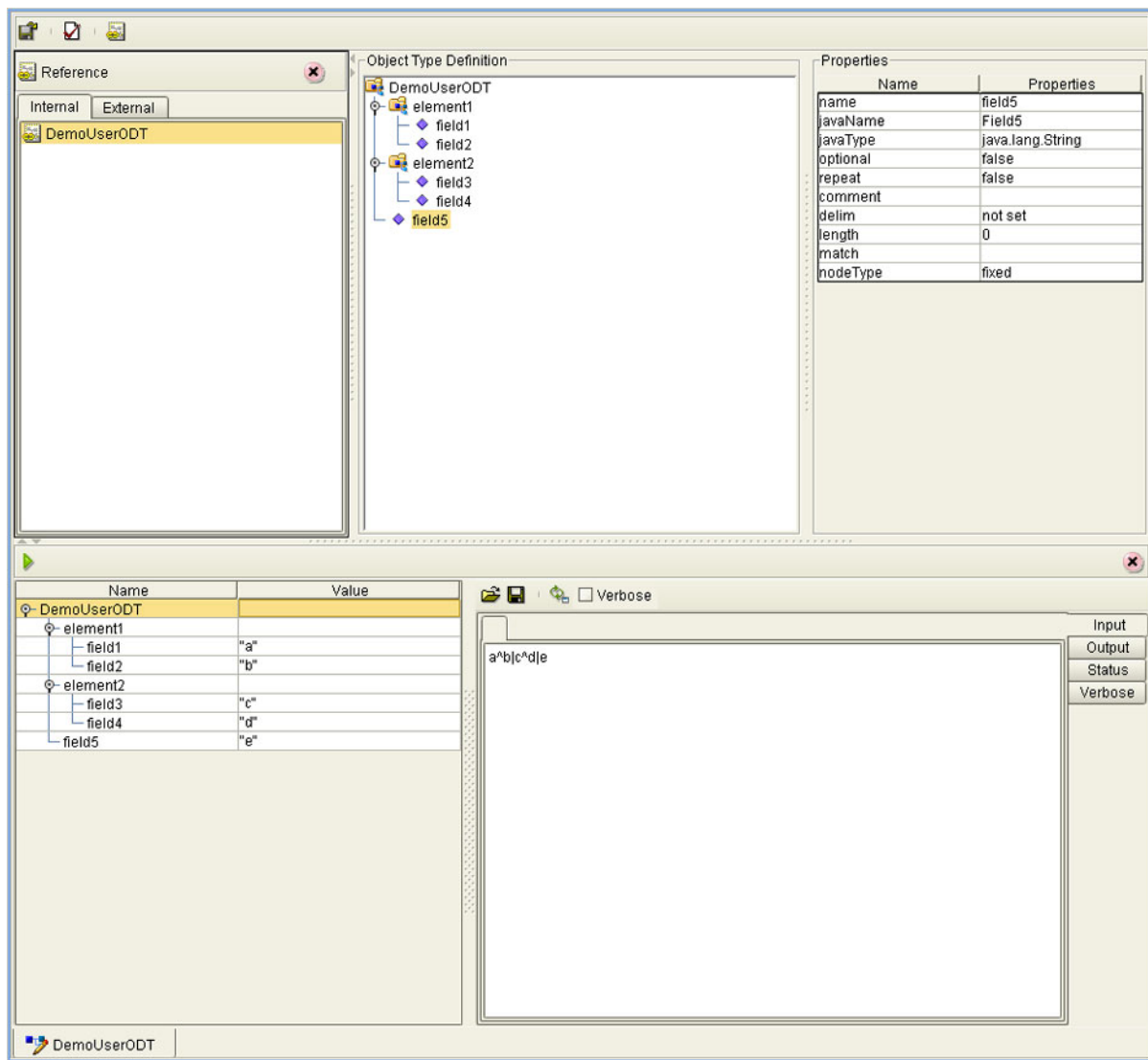
- 11 Test the OTD by invoking the OTD Tester, as described in [Using the OTD Tester](#) on page 128. Entering the data `a^b | c^d | e` will result in a successful parse, as shown in Figure 95.

6.5 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 95. 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 Analyzer](#) on page 67).









Figure 95 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 30.

Table 30 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.

6.5.1 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.

6.6 Using the OTD Tester

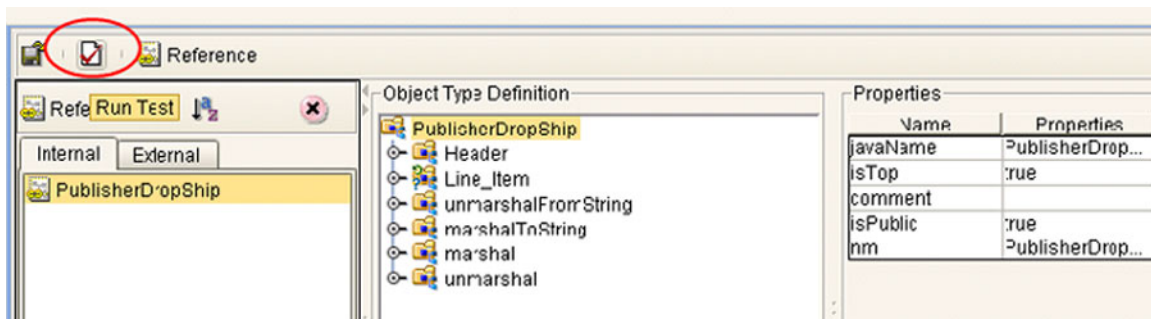
The OTD tester provides a facility to verify the correctness of OTDs, for example to:

- 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

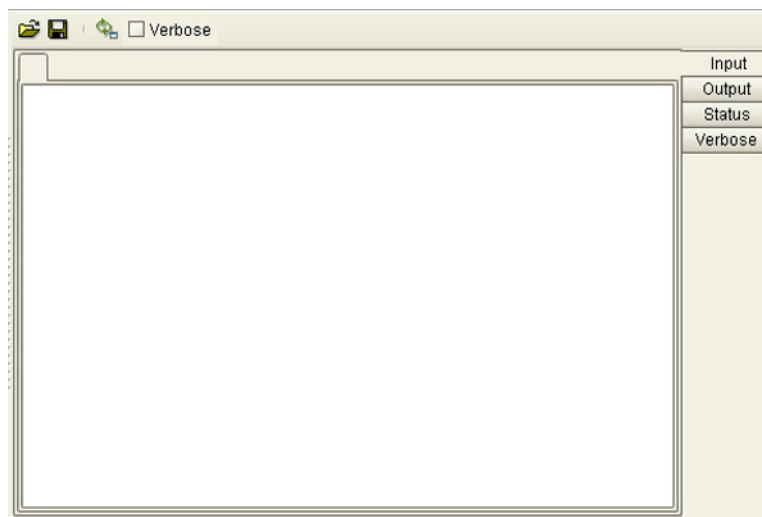
- 1 Open or create an OTD.
- 2 Click the **Tester** icon (see Figure 96).

Figure 96 OTD Tester



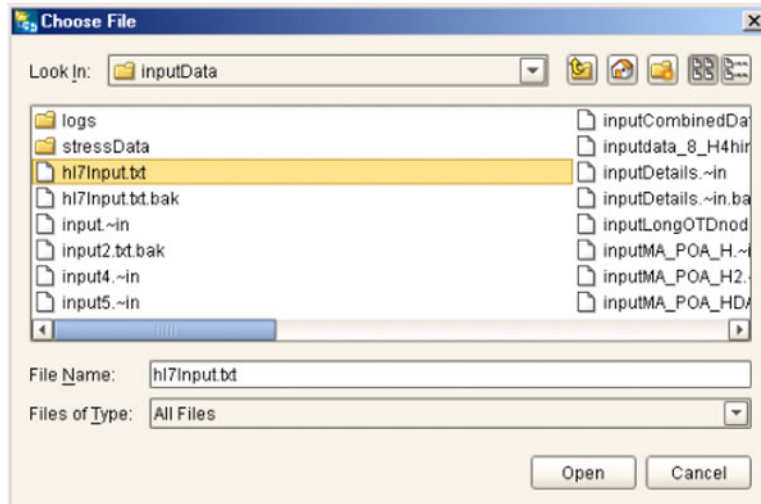
A test panel will appear below the OTD detail area of the editor. Note that there are four data display modes, selectable by tabs (see Figure 97). The Input tab is selected by default.

Figure 97 Test Panel Data Display



- 3 You can provide the input test data either by selecting a data file (see Figure 98), or by entering the data manually.

Figure 98 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 99).

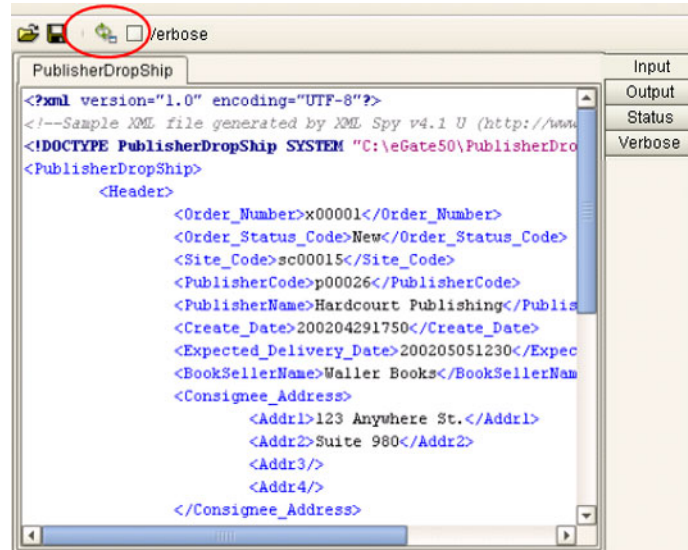
Figure 99 Object Elements and Values

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.

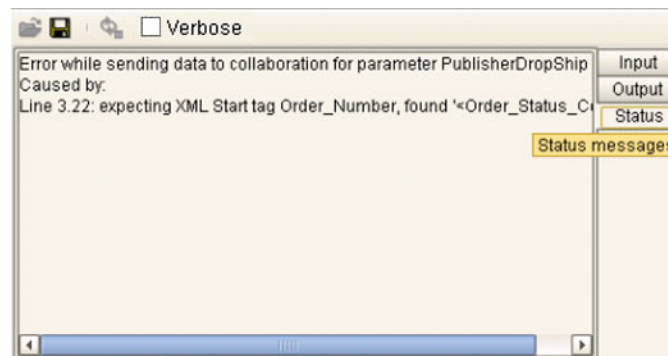
- 7 You can also change your test data in the Input data display, then re-test the OTD by clicking the **Refresh** icon (see Figure 100) to repopulate your OTD object elements with the new values.

Figure 100 Data Display: Refresh Icon



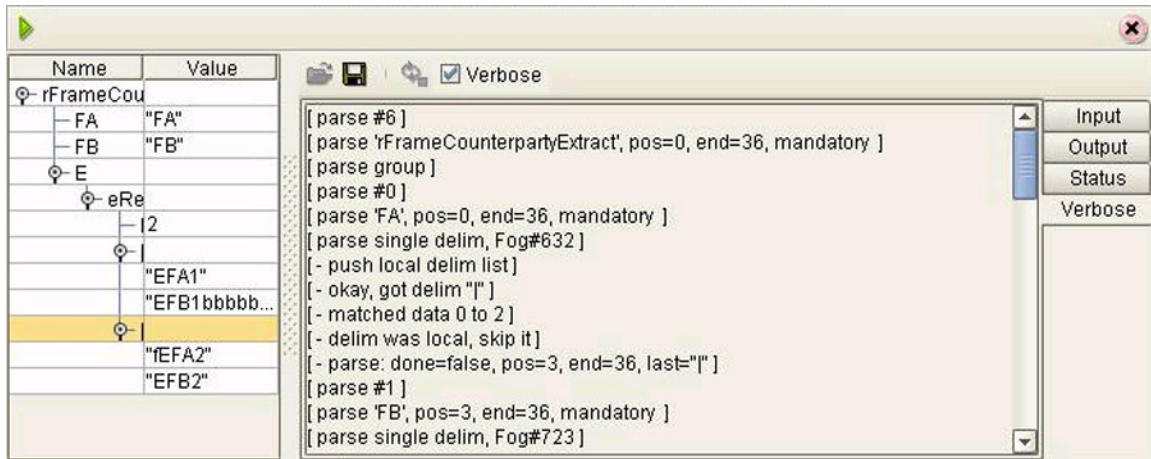
- 8 If there are errors in your input data, the **Status** data display is automatically invoked, showing the appropriate error messages (see Figure 101).

Figure 101 Status Data Display



- 9 For selected User-Defined OTDs, the **Verbose** option provides a trace of parsing actions during the *unmarshal* process to aid in debugging the OTD structure. When this option is available, the OTD will activate the **Verbose** check box. Selecting the **Verbose** check box causes parsing information to appear on the **Verbose** data display (see Figure 102). The format and content of the data display are OTD-specific.

Figure 102 Verbose Data Display



Collaboration Definitions (Java)

This chapter describes the process for building Java-based Collaboration Definitions.

7.1 Overview

Collaborations use Collaboration Definitions to define how data should be processed and 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 Collaboration Definition Wizard (Java) and Collaboration Editor (Java), that are used to create and customize your Java-based Collaboration Definitions. You must have OTDs available to use as the foundation for creating a Java-based Collaboration Definition. See [Object Type Definitions](#) on page 89 for more details.

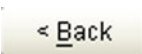






Important: *If you delete an OTD in the Project Explorer, any Java-based Collaboration Definitions that have been built using that OTD will be affected. It is recommended that you run the Impact Analyzer before attempting to delete any OTDs (see [Impact Analyzer](#) on page 67).*

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. To be safe, this should also be done before creating the Connectivity Map and Deployment Profile.*

7.2 Using the Collaboration Definition Wizard (Java)

The Collaboration Definition Wizard (Java) guides you through the initial phases of creating a Java-based Collaboration Definition, and then invokes the Collaboration Editor (Java). The user interface is highly self-explanatory, but details of the navigation buttons are listed in Table 31 for your reference.

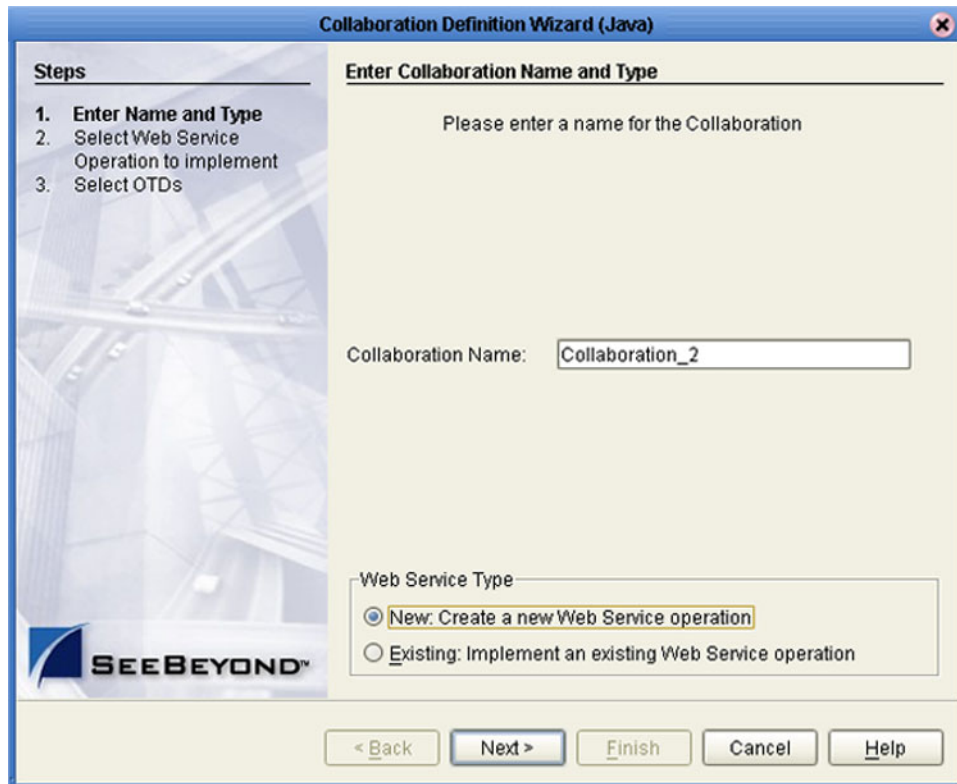
Table 31 Wizard Navigation Buttons

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.

7.2.1 Creating a Java-based Collaboration Definition

- 1 Right-click on a Project in the Enterprise Explorer to display the Project context menu.
- 2 Select **New > Collaboration Definition (Java)** to invoke the Collaboration Definition Wizard (Java).
- 3 Enter a **Name** for your Collaboration, as shown in Figure 103.

Figure 103 Initial Wizard Dialog



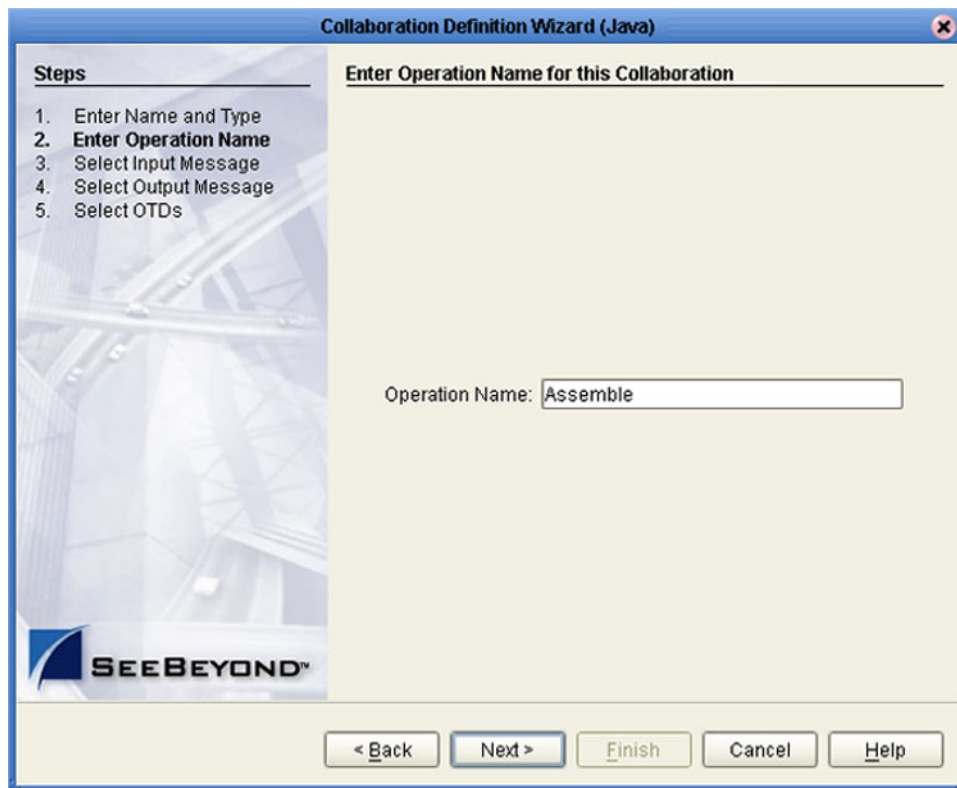
- 4 Select a Web service, which can be either:
 - A New Web Service.
 - An Existing Web Service (for example, an eInsight process or an OTD).
- 5 Click **Next** to proceed to the next Wizard dialog, which is dependent upon your Web Service selection.

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 104. This will become the *method* that can be used to invoke the Java-based Collaboration as a Web service.

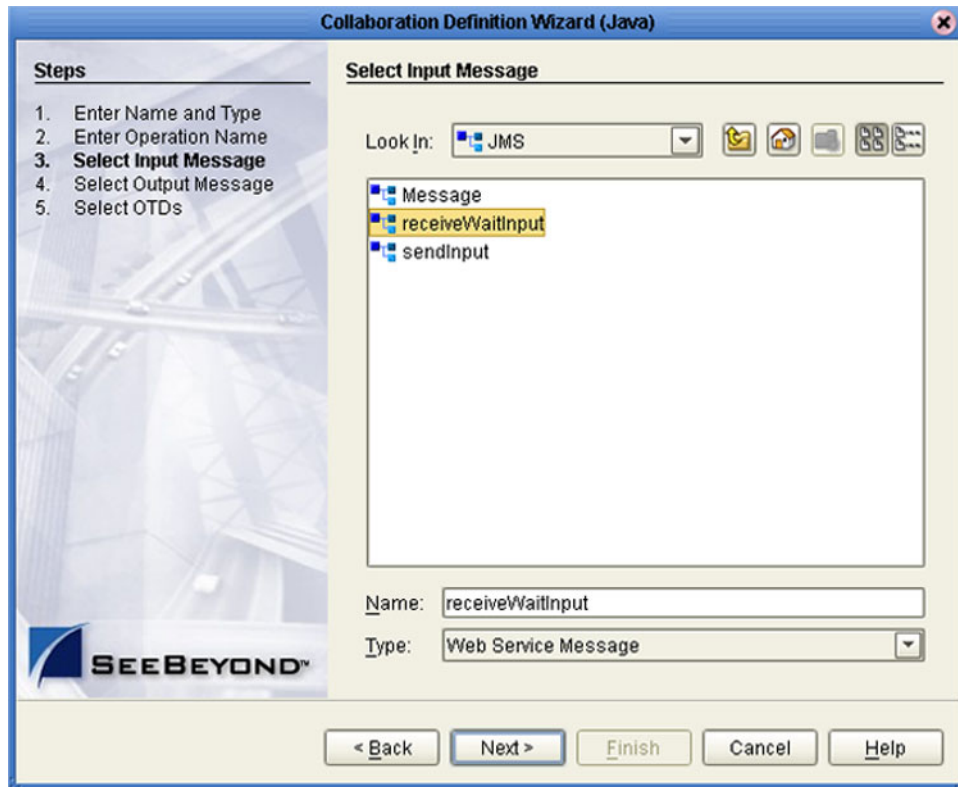
Figure 104 New Web Service: Operation Name



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

- 3 Select the input Web service message, as shown in Figure 105.

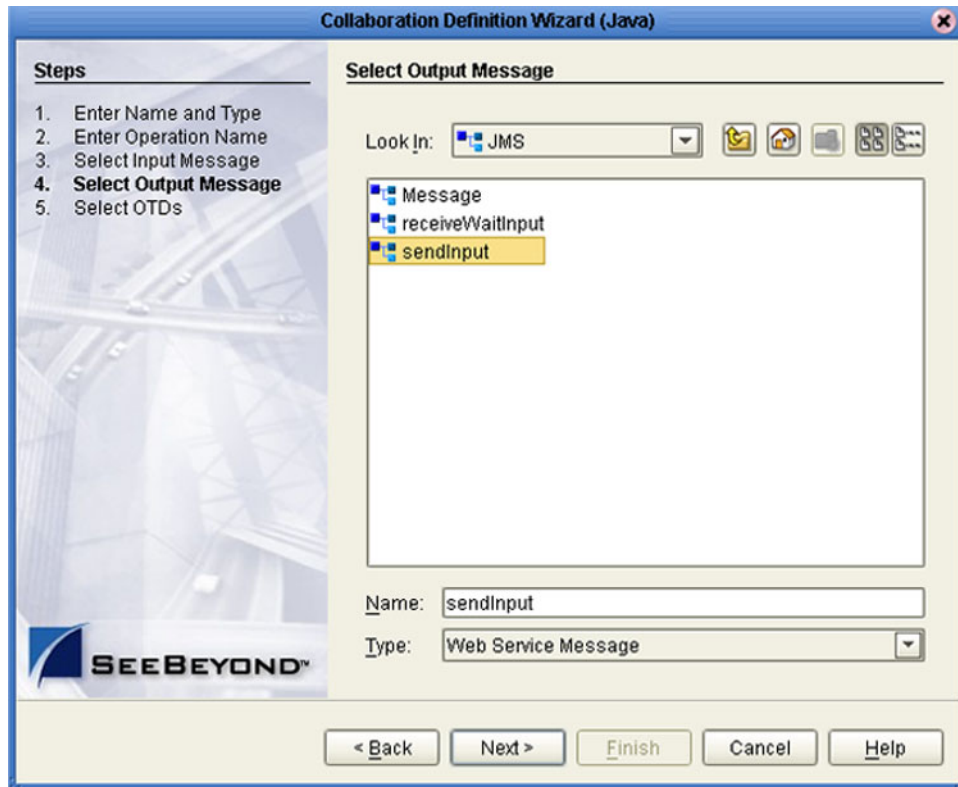
Figure 105 New Web Service: Input Message



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

- 5 Select the output Web service message, as shown in Figure 106.

Figure 106 New Web Service: Output Message

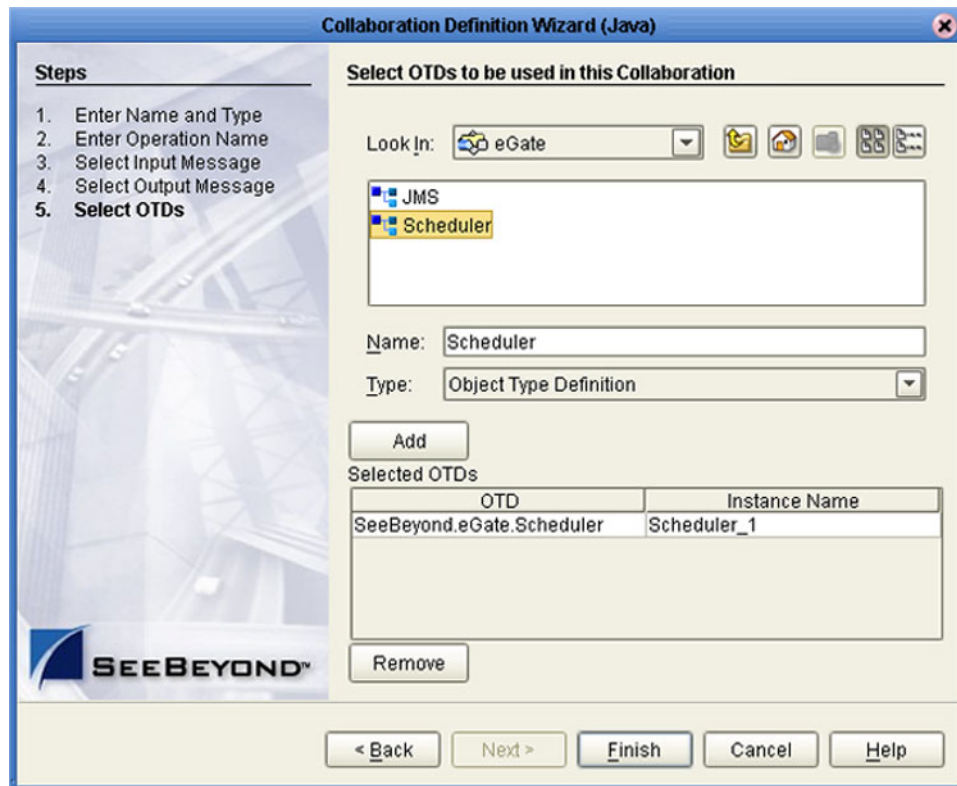


- 6 Click **Next** to proceed to the next Wizard dialog.

- 7 Select an auxiliary OTD, as shown in Figure 107. 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 107 New Web Service: Auxiliary OTD



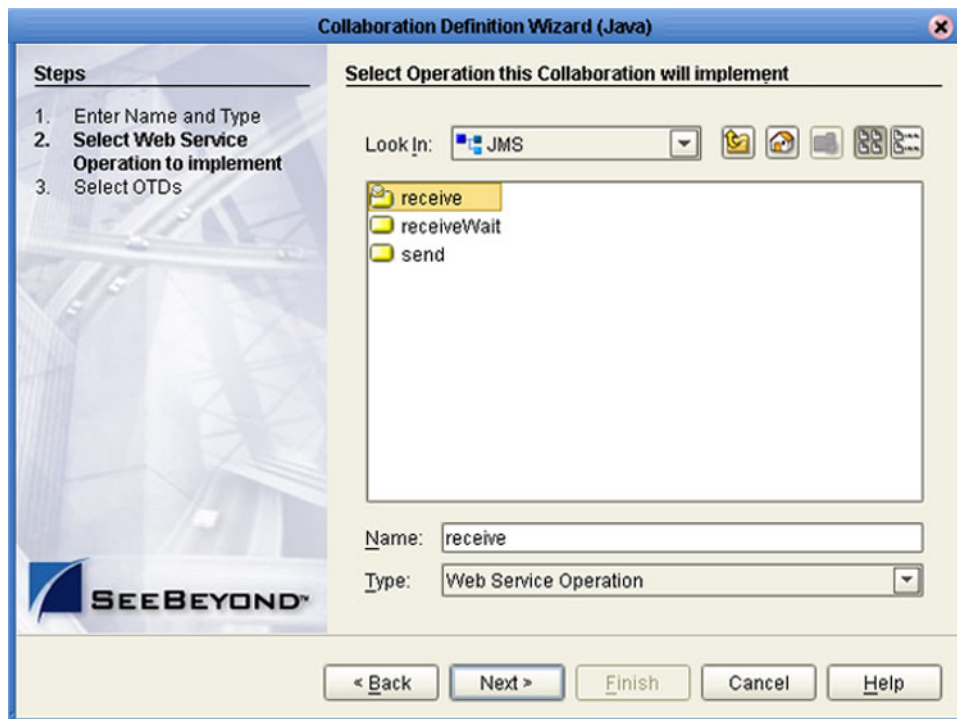
- 8 Click **Add** to add the OTD to the Collaboration Definition.
- 9 Click **Next** to proceed to the Collaboration Editor (Java).

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 108).
 - A custom Web Service (for example, something that has been created in an eGate Project).

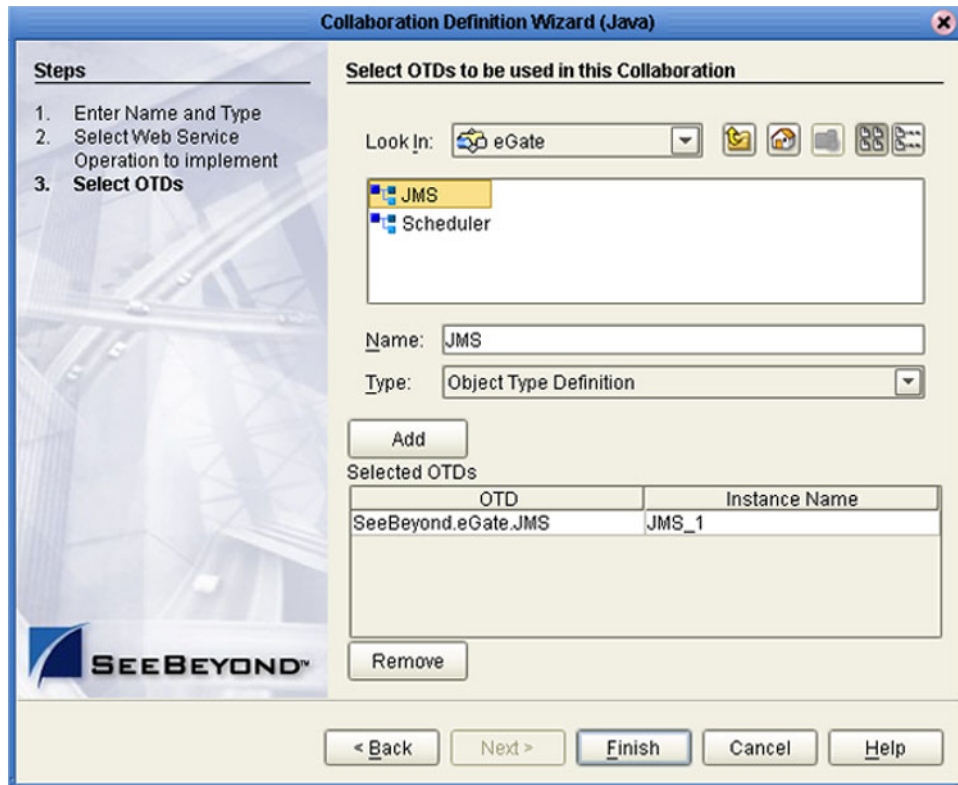
Figure 108 Existing Web Service: Select Operation



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

- 3 Select an OTD, as shown in Figure 109.

Figure 109 Existing Web Service: Select OTD



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

7.3 Collaboration Definition Properties

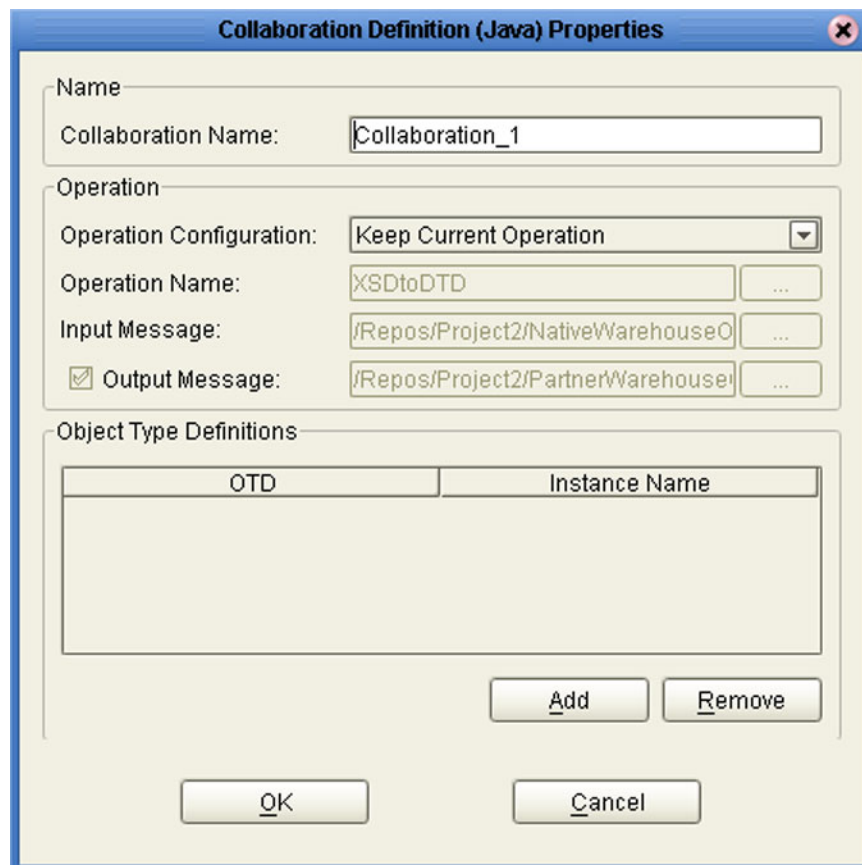
Right-clicking a Collaboration Definition (Java) in the Project Explorer displays the context menu shown in Figure 110.

Figure 110 Collaboration Definition (Java) Context Menu



Selecting **Properties** displays the Collaboration Definition (Java) Properties dialog box for the selected Collaboration Definition (see Figure 111). This dialog box closely resembles the Collaboration Definition Wizard, and shows the property values that were set previously. By default, the Operation Configuration field is set to **Keep Current Operation**; to edit the property values, you must select another command.

Figure 111 Collaboration Definition (Java) Properties

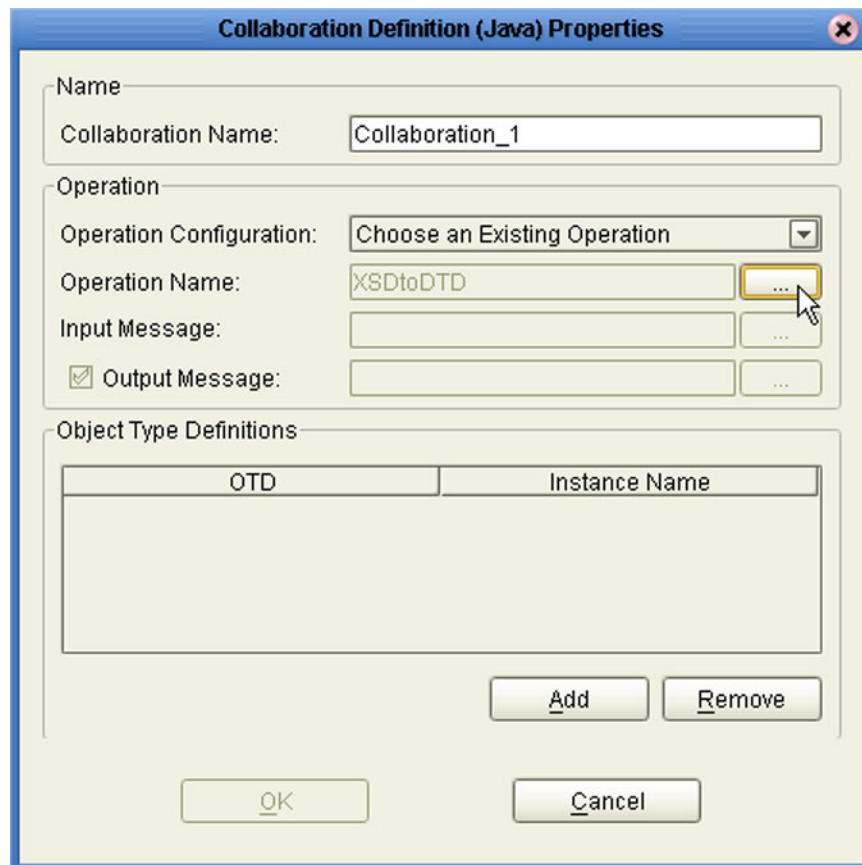


Important: If you change any Java-based Collaboration that implements existing Web services, be sure to reset the Web service in the Collaboration Definition (Java) Properties dialog box, as described in the following procedure.

To reset the Collaboration operation configuration

- 1 In the Explorer, right-click the Collaboration Definition that has been changed.
- 2 Select **Properties** from the Collaboration context menu (see Figure 110).
- 3 Select **Choose an Existing Operation** from the Operation Configuration list (see Figure 112).

Figure 112 Collaboration Definition (Java) Properties

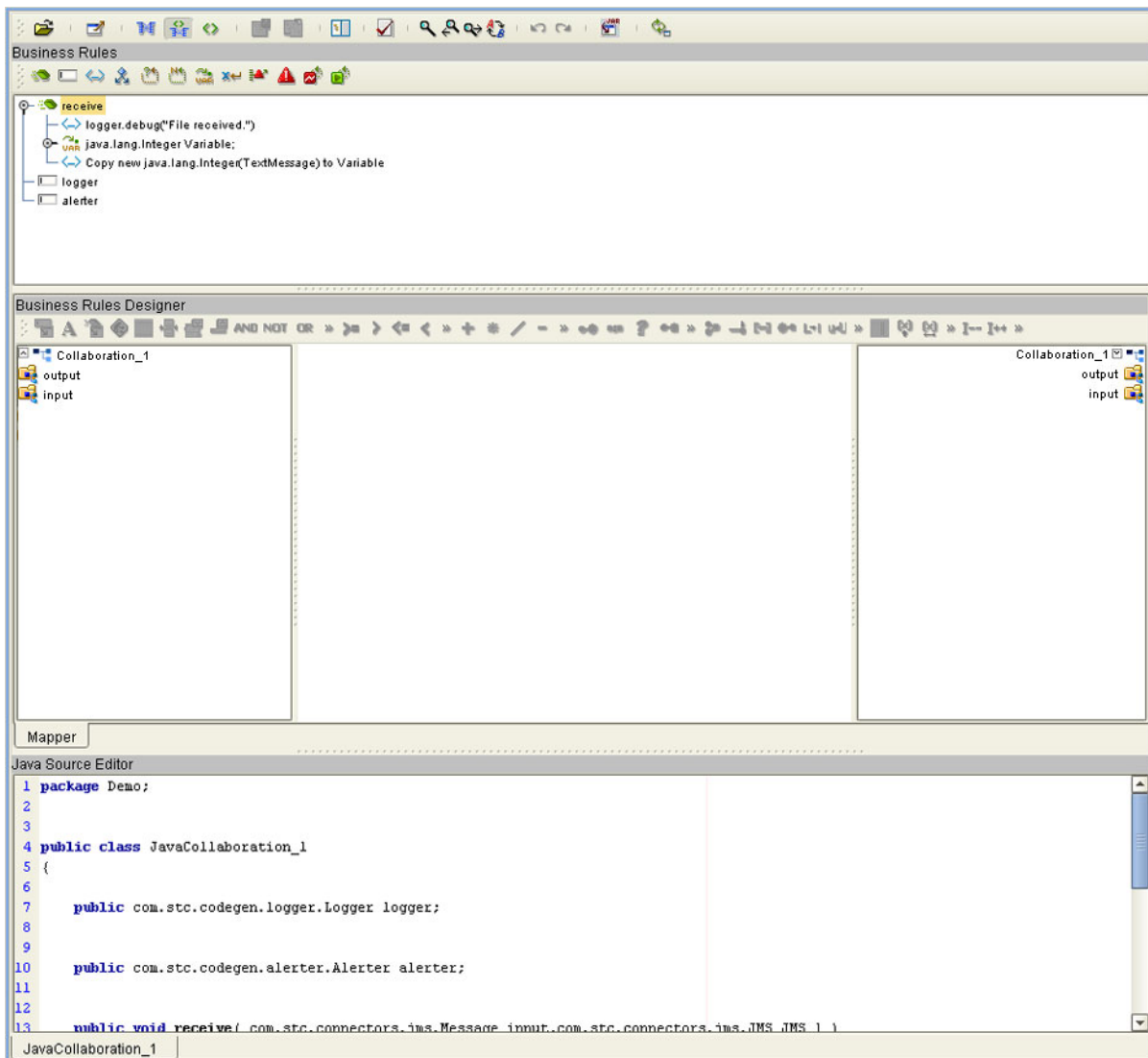


- 4 Select the same Web Service Operation as before in **Operation Name** (click the ellipsis [...] button, as shown in Figure 112, to display a dialog box for selecting the operation).
- 5 Select the appropriate input and output messages.
- 6 Click **OK** to finish.

7.4 Using the Collaboration Editor (Java)

After you create a Java-based Collaboration Definition file using the Collaboration Definition Wizard (Java), the Collaboration Editor (Java) appears in the editor panel of the Enterprise Designer (see Figure 113).

Figure 113 Collaboration Editor (Java)



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

- [Java Toolbar Icons](#) on page 144.
- [Business Rules Editor](#) on page 146.
- [Business Rules Designer](#) on page 149.
- [Java Source Editor](#) on page 156.

7.4.1 Java Toolbar Icons

Table 32 Java Toolbar Icons












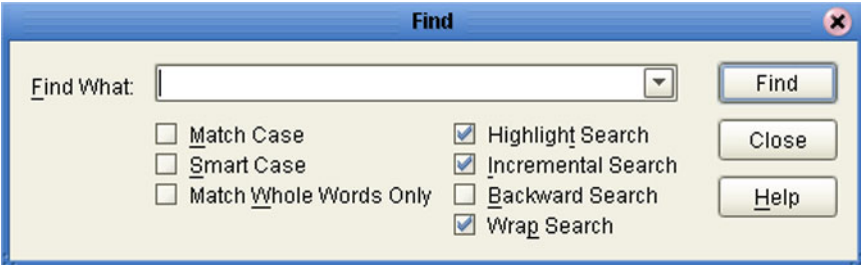








Icon	Command	Function
	Import a Local File	Displays the Open dialog box, which you can use to locate and select a Collaboration Definition (Java) to import. When you import a file, any previously generated code or rules are deleted. The imported code does not get appended to the existing Collaboration Rules.
	Export Java Rule to a Local File	Displays the Save dialog box, which you can use to save the selected Collaboration Definition (Java) 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-based 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.

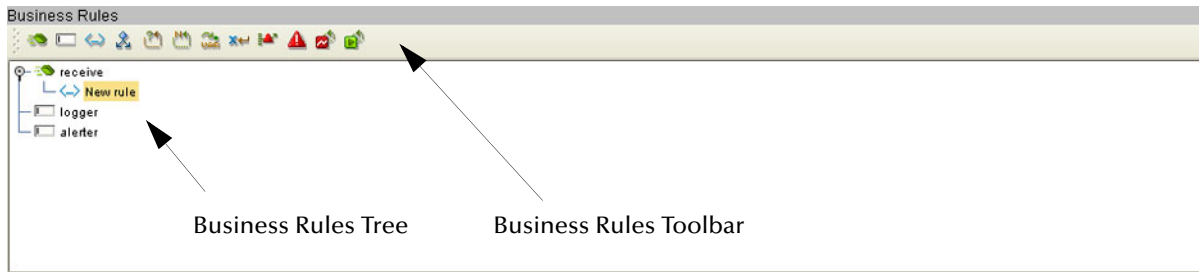
Table 32 Java Toolbar Icons

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.

7.4.2 Business Rules Editor

The Business Rules Editor lists each business rule included with the Java-based Collaboration Definition. This editor has its own toolbar which you use to add business rules to the Collaboration Definition (see Figure 114). 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 114 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-based 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 Java-based Collaborations).

The **Field** command is used when you want to create a field within a Java-based 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 33.

Business Rules Toolbar Icons

Table 33 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 34.

Table 34 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.

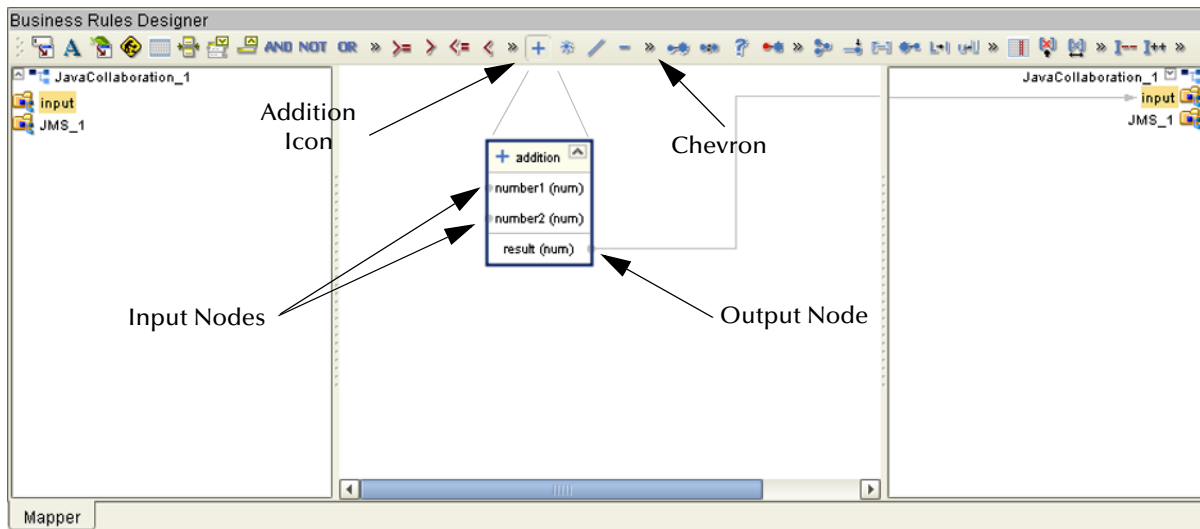
Note: When a subnode is moved to another method, or node, it becomes the first subnode listed under that node.

7.4.3 Business Rules Designer

The Business Rules Designer (see Figure 115) lists each field included in the Java-based 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 115 Business Rules Designer: Addition Method



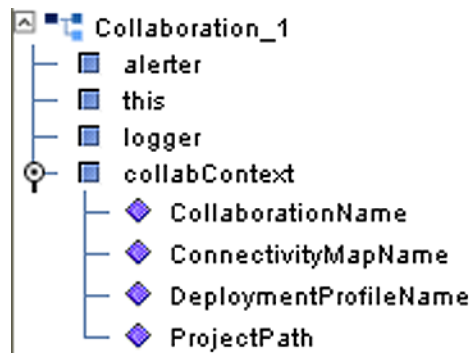
Note: If you encounter an error while using the addition, subtraction, multiplication, or division operators, try manually converting the string to integers.

Method-Access Nodes

When you create a Java-based Collaboration, several method-access nodes are added automatically in the Business Rules Designer (see Figure 116).

Note: You must double-click the Collaboration to display these nodes.

Figure 116 Method-Access Nodes



To call one of these methods, right-click on the node to display the context menu and click *Select a method to call ...*.

Table 35 Method Access Nodes

Node	Subnode	Description
alerter		Allows access to alert method, as described in Creating Alerts on page 203.
this		Allows you to call other methods defined for <i>this</i> Collaboration without having to edit the Java code.
logger		Allows access to logging method, as described in Creating Log Entries on page 206.
collabContext	CollaborationName	Returns the name of the selected Collaboration, as shown in the Connectivity Map.
	ConnectivityMapName	Returns the name of the Connectivity Map in which the selected Collaboration appears.
	DeploymentProfileName	Returns the name of the Deployment in which the selected Collaboration appears.
	ProjectPath	Returns the name of the Project in which the selected Collaboration appears.

Business Rules Designer Toolbar Icons

Table 36 Business Rules Designer Toolbar Icons


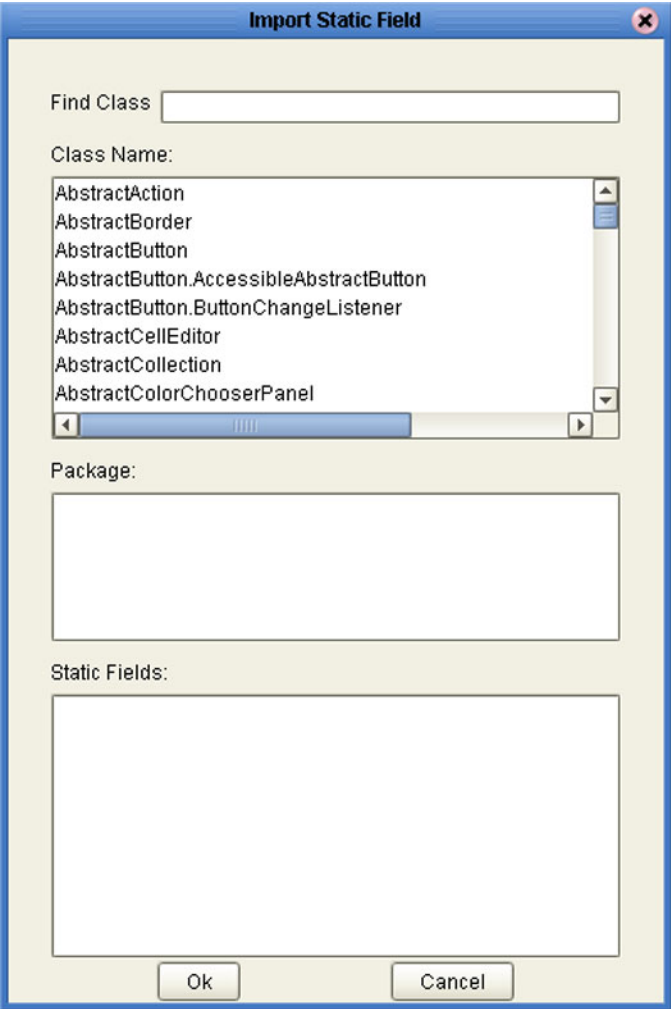
Icon	Command	Function
	Import Static Field	<p>Displays the <i>Import Static Field</i> dialog box, which you can use to select a field to add to the Collaboration Definition.</p> 

Table 36 Business Rules Designer Toolbar Icons


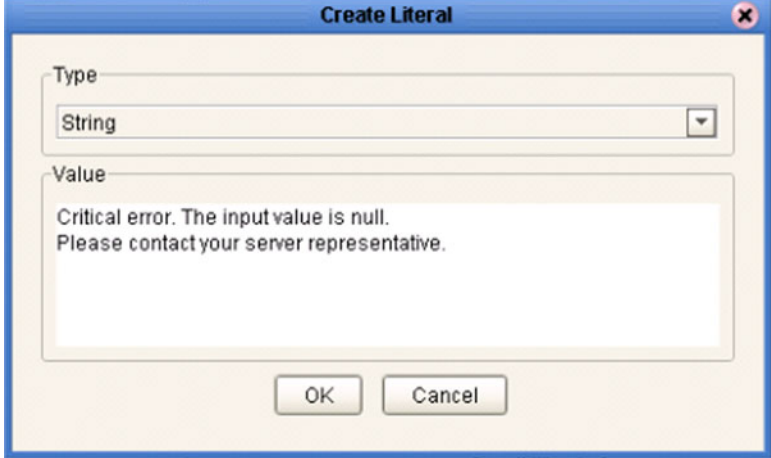


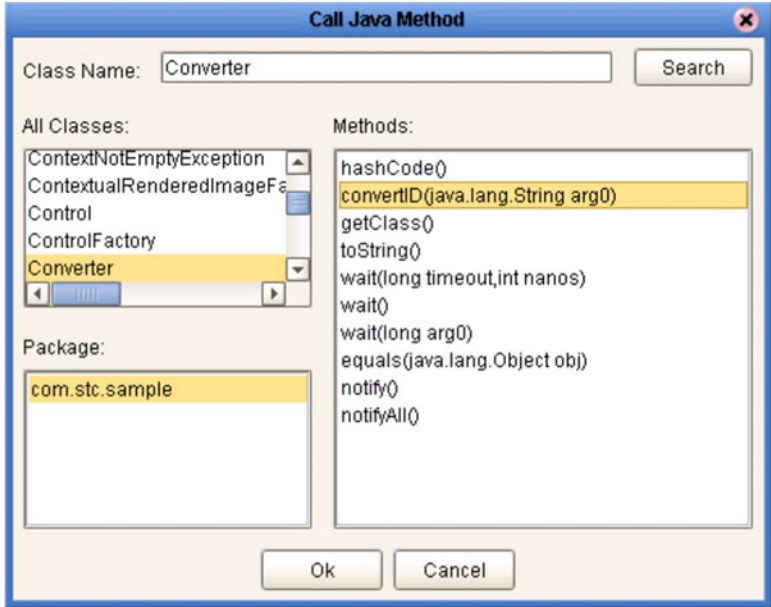
Icon	Command	Function
	<p>Create Literal</p>	<p>Displays the <i>Create Literal</i> dialog box, which you can use to select a literal and enter a value to add to the Collaboration Definition.</p>  <p>Note that a space character can be created by selecting Char as a type, and simply entering a space as a value. Although nothing will be visible in the Value field of the dialog box, the character will appear in the Business Rules canvas, as follows:</p> 
	<p>Call Java Method</p>	<p>Displays the <i>Call Java Method</i> dialog box, which you can use to select a Java method to add to the Collaboration Definition.</p> 

Table 36 Business Rules Designer Toolbar Icons


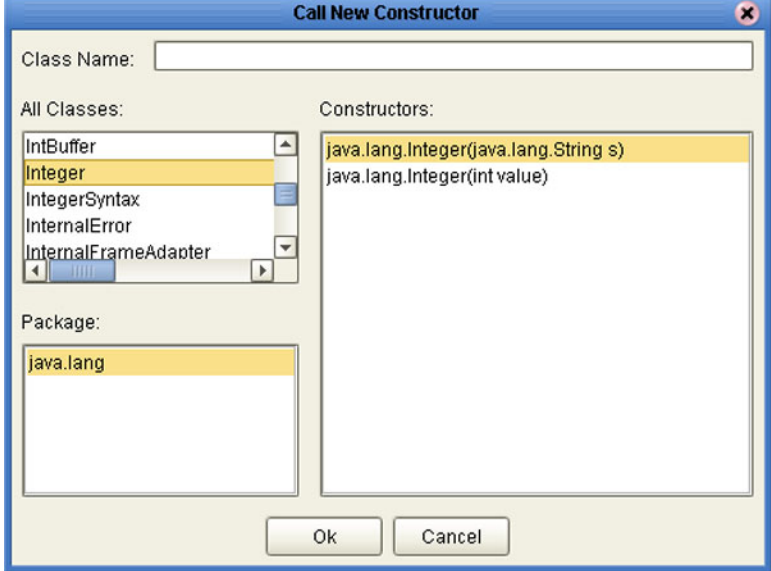

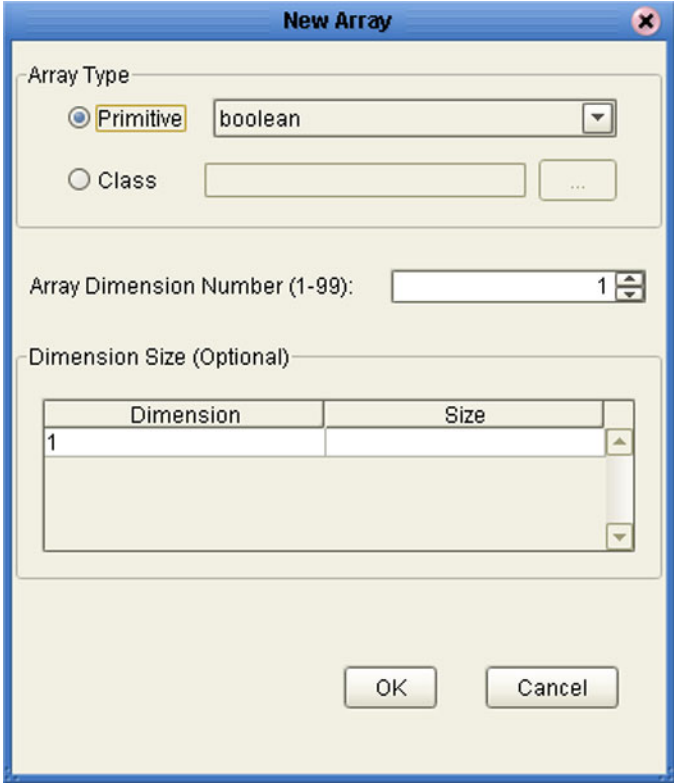



Icon	Command	Function
	<p>Call New Constructor</p>	<p>Displays the <i>Call New Constructor</i> dialog box, which you can use to select a Java class to the Collaboration Definition.</p> 
	<p>New Array</p>	<p>Displays the <i>New Array</i> dialog box, which you can use to add an array to the Collaboration Definition.</p> 

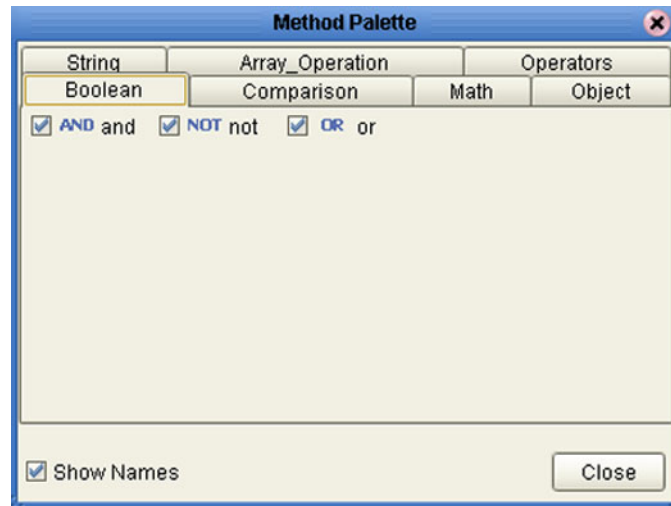
Table 36 Business Rules Designer Toolbar Icons

Icon	Command	Function
	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.

Collaboration Method Palette (Java)

The Collaboration Method Palette includes a series of method icons that you can drag onto the Business Rules Designer mapping area. Click the Chevron (>>) to the right of the method groups to display the dialog box shown in Figure 117. 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 [Collaboration Methods \(Java\)](#) on page 157.

Figure 117 Java Collaboration Method Palette Dialog Box



Collaboration Method Boxes (Java)

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 115 on page 149](#), 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 118); you can collapse them (see Figure 119) by clicking the caret (^) in the upper right corner of the box. Clicking the now-inverted caret expands the box. Some boxes expand further as needed to provide additional argument nodes.

Figure 118 Expanded Method Box

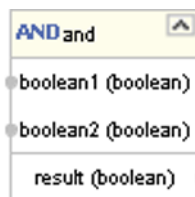


Figure 119 Collapsed Method Box



7.4.4 Java Source Editor

The Java Source Editor (see Figure 120) 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 120 Java Source Editor

A screenshot of the Java Source Editor window. The window title is "Java Source Editor". The code displayed 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
```

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.*

Important: *The Java Collaboration Editor does not support synchronized blocks.*

7.5 Collaboration Methods (Java)

7.5.1 Boolean Methods

Figure 121 Method Palette: Boolean Methods

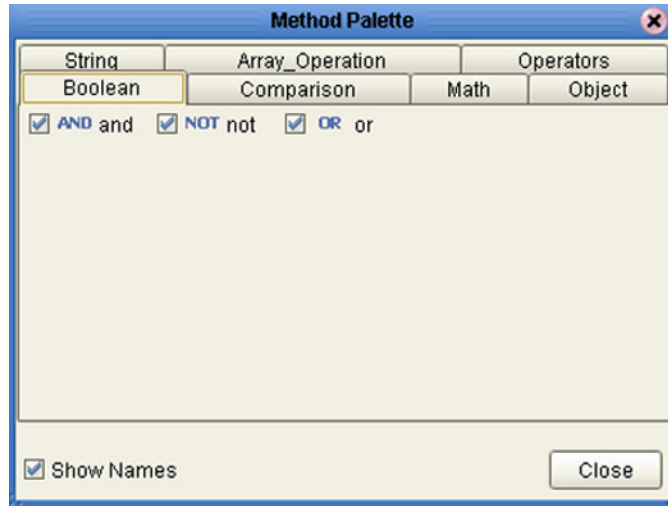


Table 37 Boolean Collaboration Methods (Java)

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> .

7.5.2 Comparison Methods

Figure 122 Method Palette: Comparison Methods

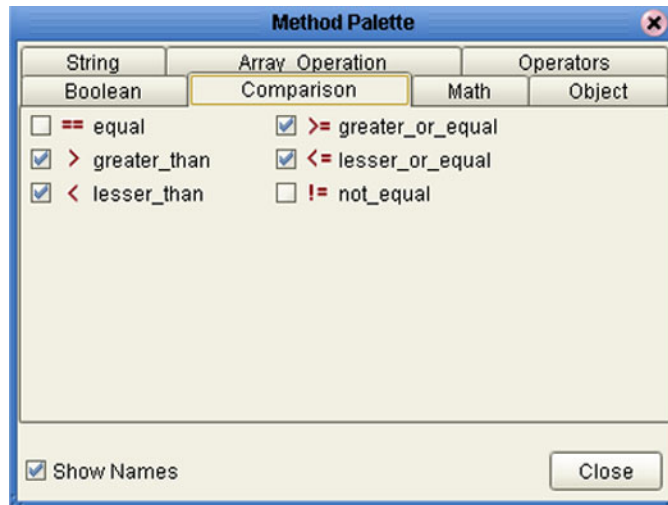
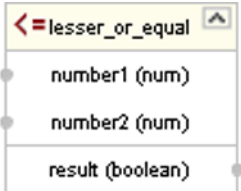
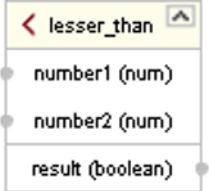
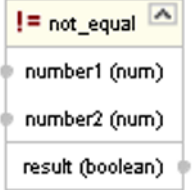


Table 38 Comparison Collaboration Methods (Java)

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 38 Comparison Collaboration Methods (Java)

Method Box	Description/Usage
 <p>The diagram shows a UML Method Box for the <code>lesser_or_equal</code> method. It features a title bar with a left-pointing arrow, the text <code>lesser_or_equal</code>, and a right-pointing arrow. Below the title bar are two parameter compartments: <code>number1 (num)</code> and <code>number2 (num)</code>. A return compartment at the bottom contains <code>result (boolean)</code>. Small circles are present on the left side of the parameter compartments and on the right side of the return compartment.</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 diagram shows a UML Method Box for the <code>lesser_than</code> method. It features a title bar with a left-pointing arrow, the text <code>lesser_than</code>, and a right-pointing arrow. Below the title bar are two parameter compartments: <code>number1 (num)</code> and <code>number2 (num)</code>. A return compartment at the bottom contains <code>result (boolean)</code>. Small circles are present on the left side of the parameter compartments and on the right side of the return compartment.</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 diagram shows a UML Method Box for the <code>not_equal</code> method. It features a title bar with a left-pointing arrow, the text <code>not_equal</code>, and a right-pointing arrow. Below the title bar are two parameter compartments: <code>number1 (num)</code> and <code>number2 (num)</code>. A return compartment at the bottom contains <code>result (boolean)</code>. Small circles are present on the left side of the parameter compartments and on the right side of the return compartment.</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>

7.5.3 Math Methods

Figure 123 Method Palette: Math Methods

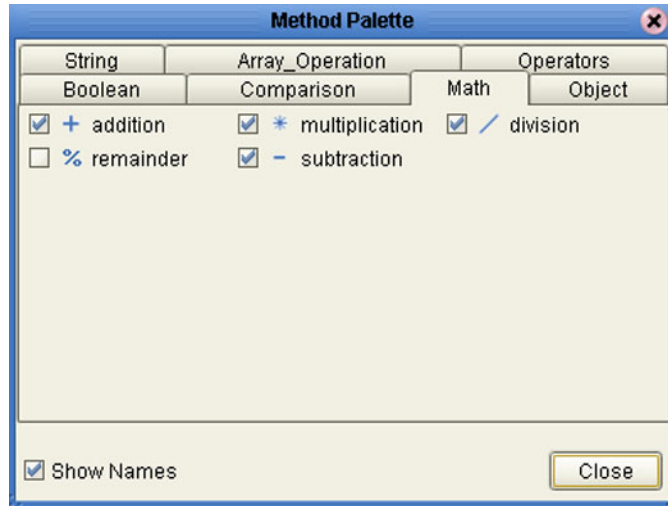
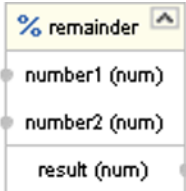
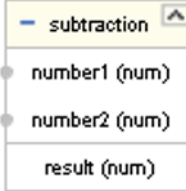


Table 39 Math Collaboration Methods (Java)

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

Table 39 Math Collaboration Methods (Java)

Method Box	Description/Usage
 A screenshot of a software interface for the 'remainder' method. It features a title bar with a blue percentage icon and the text '% remainder' followed by a small upward-pointing arrow icon. Below the title bar, there are two input fields: 'number1 (num)' and 'number2 (num)', each with a grey dot on its left side. At the bottom, there is an output field labeled 'result (num)' with a grey dot on its right side.	The remainder method divides the numerical value of <i>number1</i> by the numerical value of <i>number2</i> , returns the remainder.
 A screenshot of a software interface for the 'subtraction' method. It features a title bar with a blue minus sign icon and the text '- subtraction' followed by a small upward-pointing arrow icon. Below the title bar, there are two input fields: 'number1 (num)' and 'number2 (num)', each with a grey dot on its left side. At the bottom, there is an output field labeled 'result (num)' with a grey dot on its right side.	The subtraction method subtracts the numerical value of <i>number2</i> from the numerical value of <i>number1</i> , returns the difference.

7.5.4 Object Methods

Figure 124 Method Palette: Object Methods

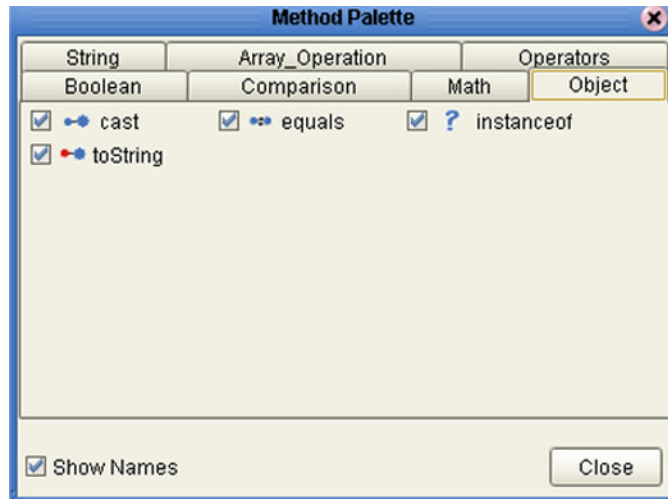


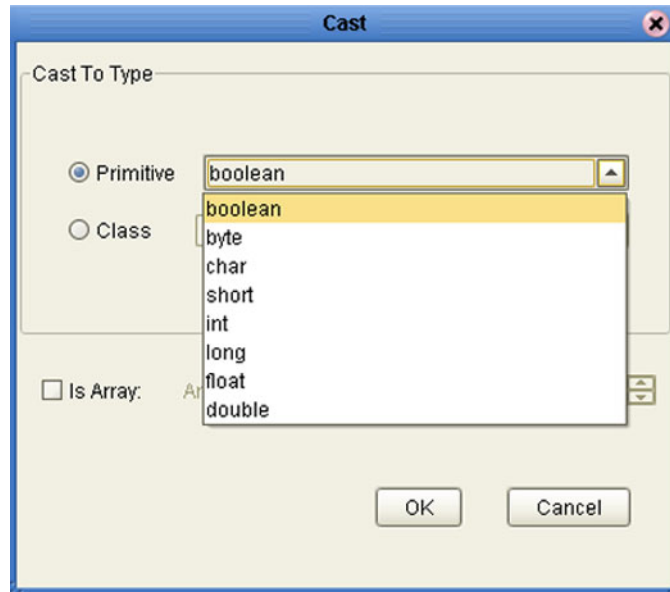
Table 40 Object Collaboration Methods (Java)

Method Box	Description/Usage
	<p>Dragging the icon first displays the <i>CAST</i> dialog box (see Figure 125 on page 163), 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 <i>CAST</i> 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 <i>InstanceOf</i> dialog box (see series beginning with Figure 126 on page 163), 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

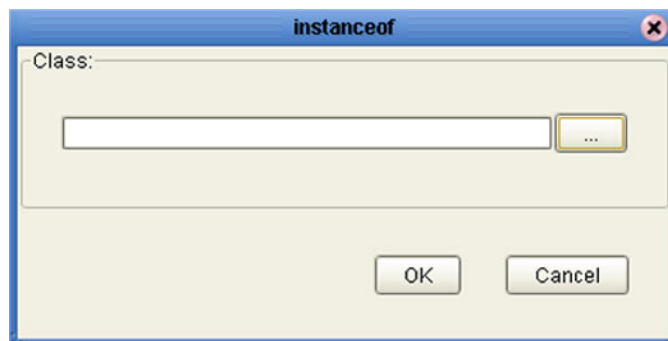
CAST Method

Figure 125 CAST Dialog Box



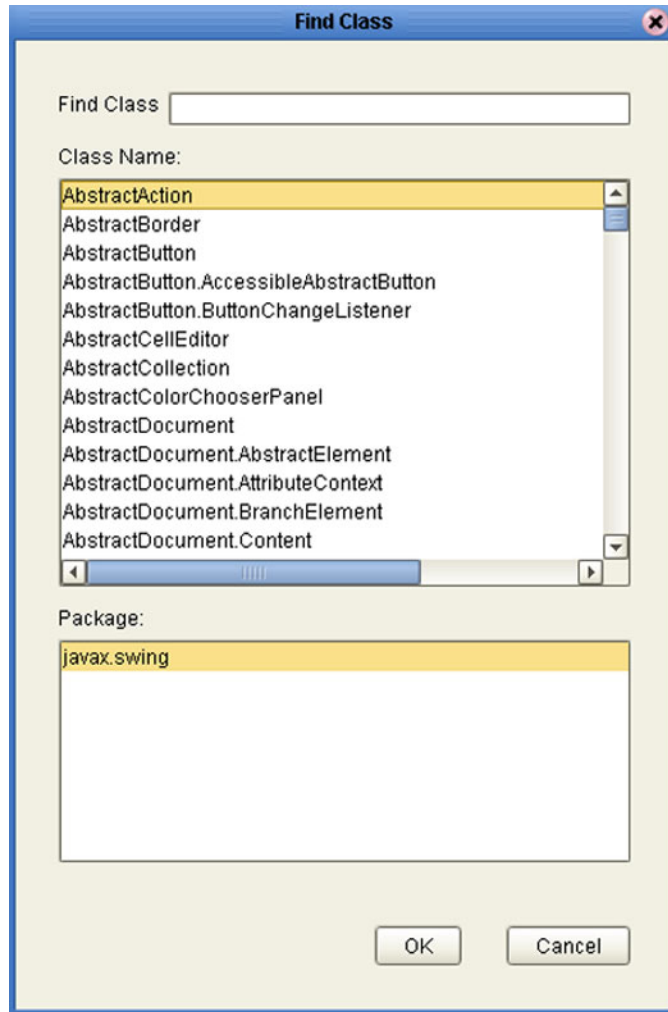
InstanceOf Method

Figure 126 InstanceOf Dialog Box (1)



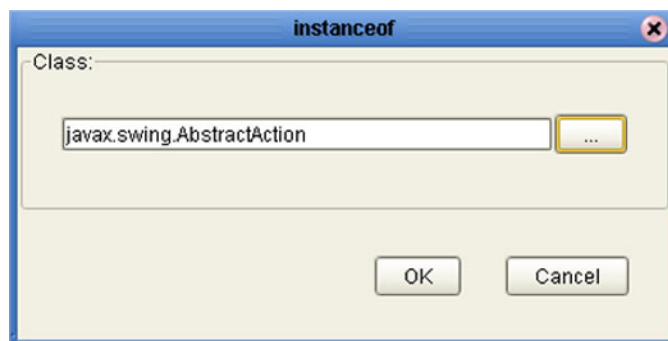
Clicking the ellipsis (...) button displays the *Find Class* dialog box (see [Figure 127 on page 164](#)), in which you select the desired class name.

Figure 127 Find Class Dialog Box



Clicking **OK** returns you to the *InstanceOf* dialog box (see Figure 128), with the field displaying *Package Name Class Name*.

Figure 128 InstanceOf Dialog Box (2)



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

7.5.5 String Methods

Figure 129 Method Palette: String Methods

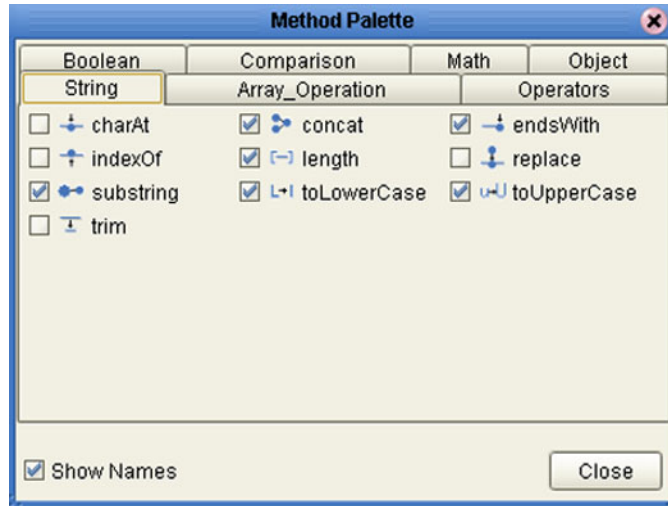


Table 41 String Collaboration Methods (Java)

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 41 String Collaboration Methods (Java)

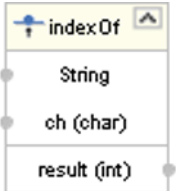


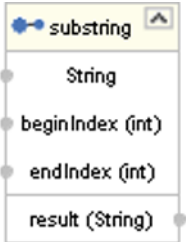
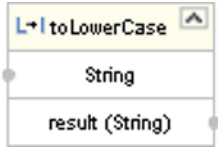
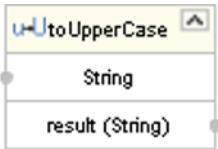
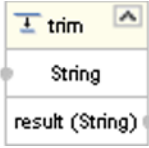
Method Box	Description/Usage
	<p>The indexOf method returns the index within <i>String</i> corresponding to the location of the specified character (<i>ch</i>), where the index represents the number of characters from the beginning of <i>String</i>.</p>
	<p>The length method returns the length (number 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 41 String Collaboration Methods (Java)

Method Box	Description/Usage
 A screenshot of a software tool's method box for the <code>trim</code> method. The box is divided into three sections: the top section contains the method name <code>trim</code> with a small icon to its left and a small upward-pointing arrow to its right; the middle section contains the parameter type <code>String</code> ; the bottom section contains the return type <code>result (String)</code> . Each section has a small grey dot on its left side.	The trim method trims leading and trailing whitespace from <i>String</i> .

7.5.6 Array Operation Methods

Figure 130 Method Palette: Array Operation Methods

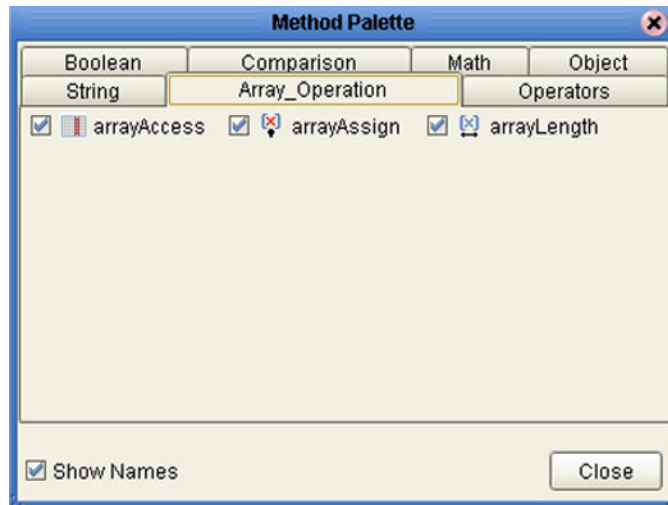
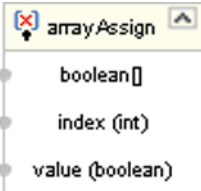
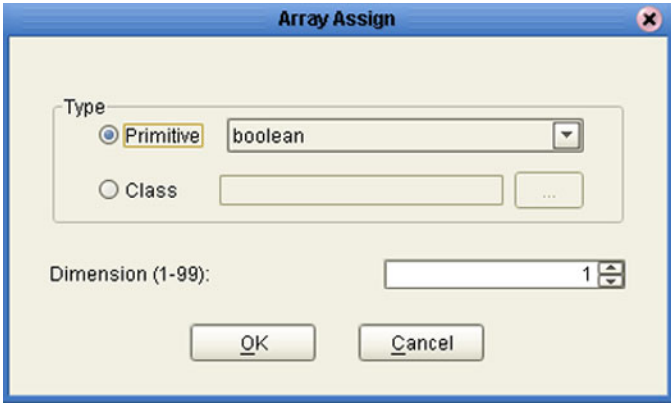
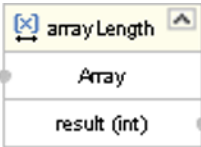


Table 42 Array Operation Collaboration Methods (Java)

Method Box	Description/Usage
	<p>Dragging the icon first displays the <i>Array Access</i> dialog box, 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>

Table 42 Array Operation Collaboration Methods (Java)

Method Box	Description/Usage
 <p>The method box for <code>arrayAssign</code> shows the following parameters: <code>boolean[]</code>, <code>index (int)</code>, and <code>value (boolean)</code>.</p>	<p>Dragging the icon first displays the <i>Array Assign</i> dialog box, 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 <i>Array Assign</i> dialog box has a title bar with a close button. It contains a 'Type' section with two radio buttons: 'Primitive' (selected) and 'Class'. The 'Primitive' option has a dropdown menu showing 'boolean'. The 'Class' option has an empty text field and a '...' button. Below this is a 'Dimension (1-99):' label with a text field containing '1' and a spinner control. At the bottom are 'OK' and 'Cancel' buttons.</p>
 <p>The method box for <code>arrayLength</code> shows the following parameters: <code>Array</code> and <code>result (int)</code>.</p>	<p>The arrayLength method returns the length (number of elements) of the array.</p>

7.5.7 Operators

Figure 131 Method Palette: Operators

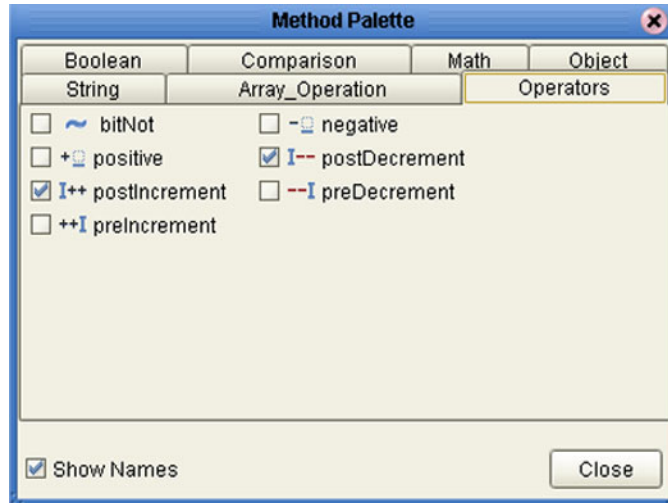
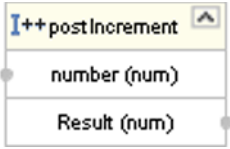
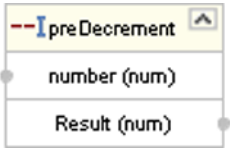
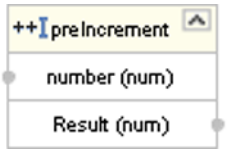


Table 43 Operators (Java)

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

Table 43 Operators (Java)

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

7.6 Version Control

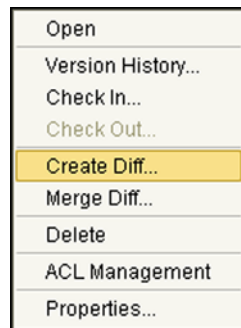
7.6.1 Creating a Modified Collaboration Definition (Java)

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

To create a *different* version of a Java-based Collaboration Definition file

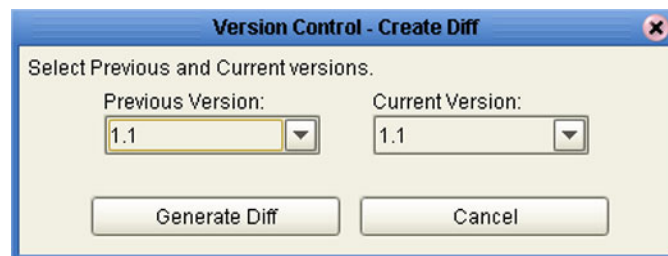
- 1 In Project Explorer, select a **Collaboration Definition (Java)** icon.
- 2 Right-click to display the context menu shown in Figure 132.

Figure 132 Collaboration Definition (Java) Context Menu



- 3 Select **Create Diff** to display the dialog box shown in Figure 133.

Figure 133 Version Control - Create Diff Dialog Box



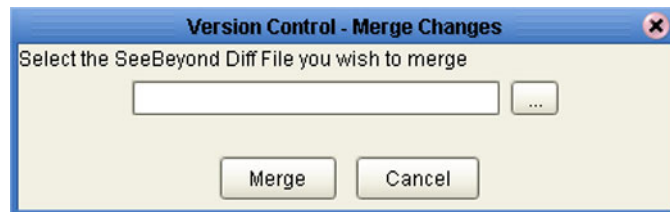
- 4 Click **Generate Diff** to display the *Specify Name ...* dialog box.
- 5 Enter a name for the Diff file in the **File Name** box (or use the default file name).
- 6 Click **Save** to save the Diff file.

7.6.2 Merging Two Versions of a Collaboration Definition (Java)

To merge two versions of a Java-based Collaboration Definition

- 1 From the Project Explorer tab, select a **Collaboration Definition (Java)** icon.
- 2 Right-click to display the Collaboration Definition (Java) context menu.
- 3 Select **Check Out**.
- 4 Right-click to again display the context menu.
- 5 Select **Merge Diff ...** to display the dialog box shown in Figure 134.

Figure 134 Version Control - Merge Changes Dialog Box



- 6 Click the **Ellipsis (...)** button to display the *Specify Name ...* dialog box.
- 7 Locate and select the Diff file.
- 8 Click **Open** add the Diff file to the dialog box shown in Figure 134.
- 9 Click **Merge** to merge the Diff file with the corresponding Collaboration Definition (Java) in your Project.
- 10 Resolve any conflicts from the merge.
- 11 **Commit** the merged Collaboration.
- 12 **Save** the merged Collaboration.

Important: *If you are merging code that contains references to third-party classes or .jar files, you must import these files into the destination Project before merging the code.*

7.7 Setting Up a Collaboration Definition Variable (Java)

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

7.7.1 Creating a Variable

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

To create the variable

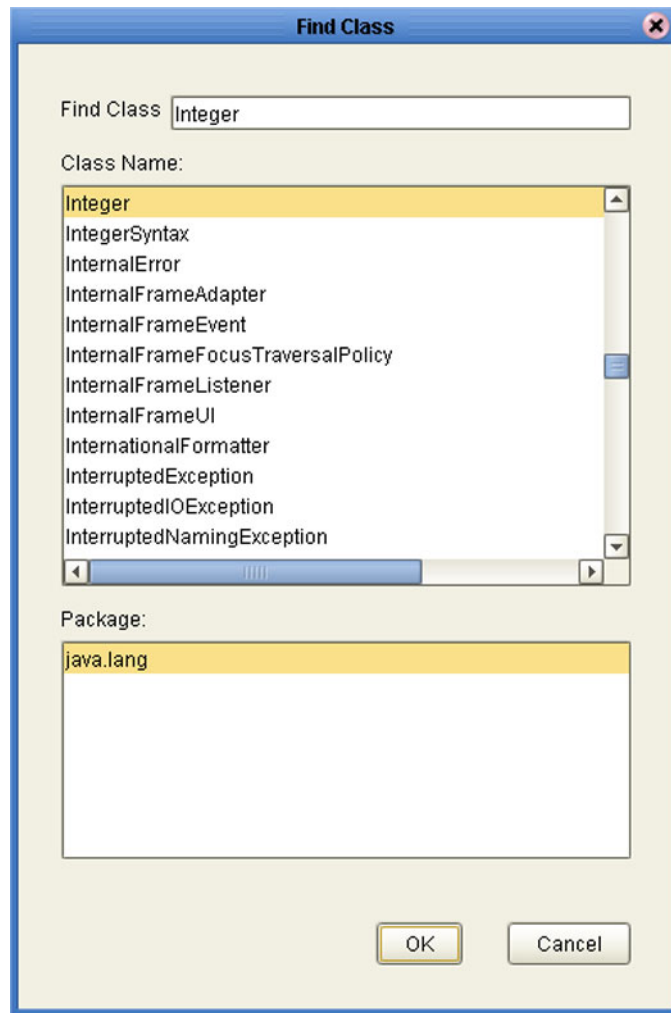
- 1 In the Business Rules area, click the **Local Variable** button to display the dialog box shown in Figure 135.

Figure 135 Create a Variable Dialog Box

The dialog box is titled "Create a Variable". It features a "Variable Name" field containing "Variable8". Under the "Method Modifier" section, there is an "Access" dropdown menu and checkboxes for "Static", "Transient", "Final", and "Volatile". The "Type" section includes radio buttons for "Primitive" and "Class", with "Class" selected. Next to "Class" is a text field containing "java.lang.Integer" and an ellipsis button "...". Below this is an "Is array" checkbox and an "Array Dimension" spinner set to "1". "OK" and "Cancel" buttons are at the bottom.

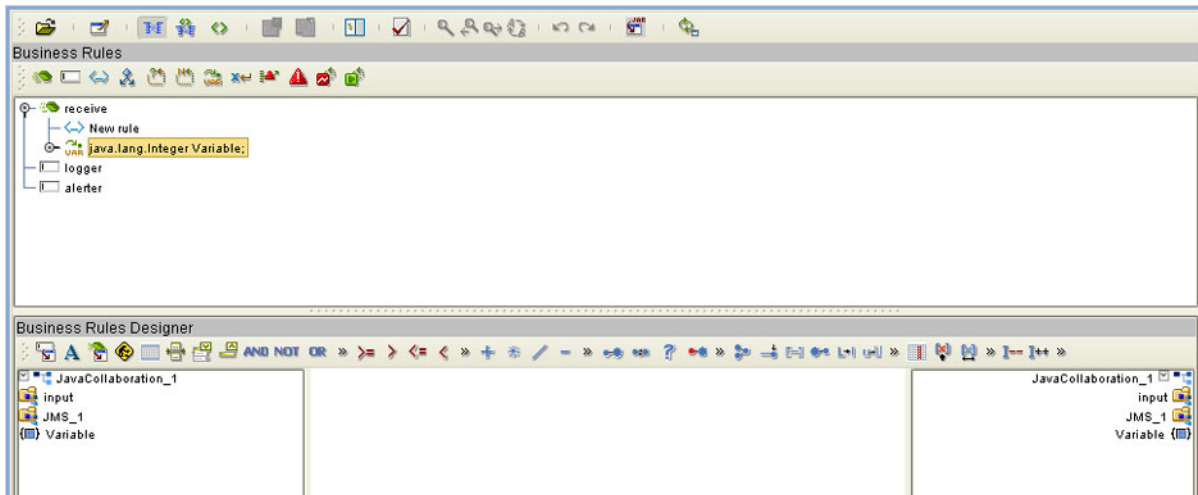
- 2 Enter a name for the variable in the **Variable Name** box.
- 3 Select the **Class** option button.
- 4 Click the **Ellipsis (...)** button to display the **Find Class** dialog box shown in Figure 136.

Figure 136 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 Collaboration Editor (Java) window as shown in Figure 137.

Figure 137 Collaboration Definition (Java) with Variable



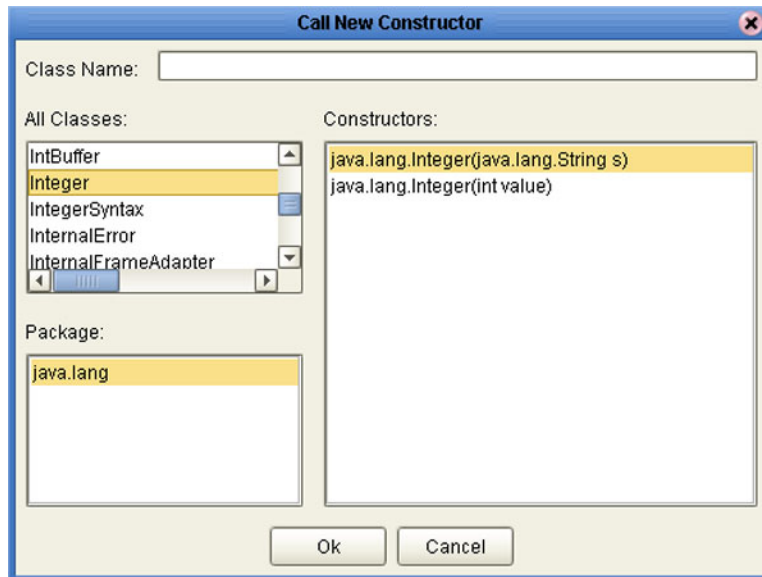
7.7.2 Invoking Variable Constructor

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

To invoke a constructor

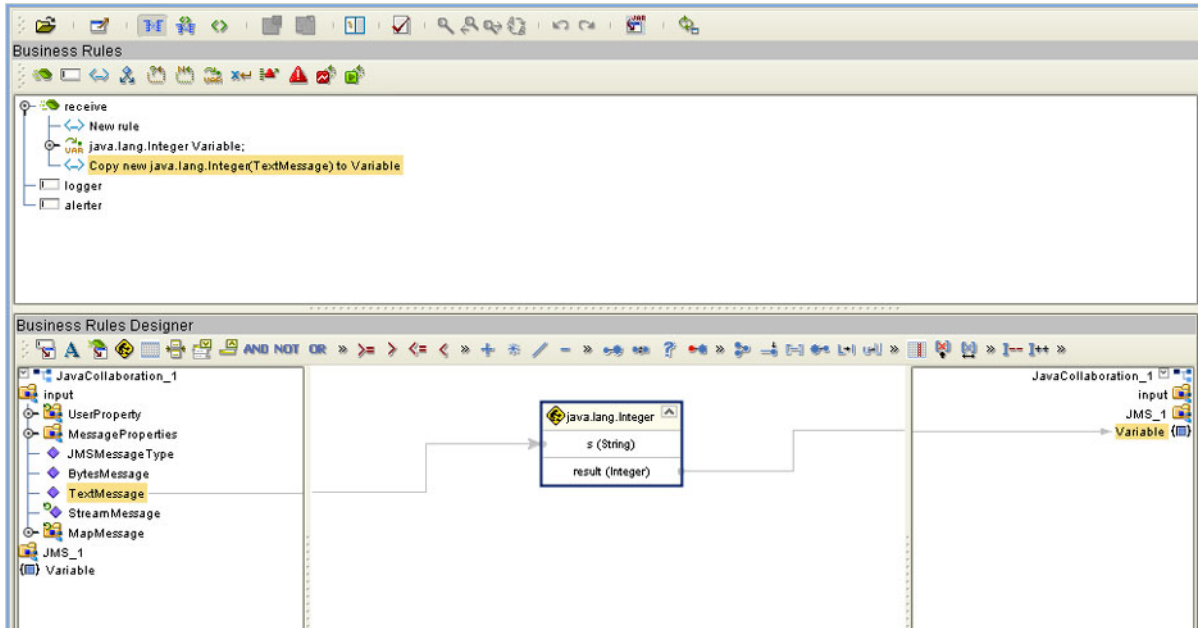
- 1 In the Business Rules Designer area, click the **Call New Constructor** button to display the dialog box shown in Figure 138.

Figure 138 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 139.
- 4 Link an item in the left column through the integer to an item in the right column.

Figure 139 Collaboration Definition (Java) with Constructor



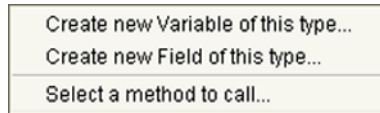
7.7.3 Displaying Method Classes

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 140

Figure 140 Variable Context Menu



- 3 Click **Select a Method to Call** to display a list of method classes as shown in Figure 141.

Figure 141 Method Selection List Box



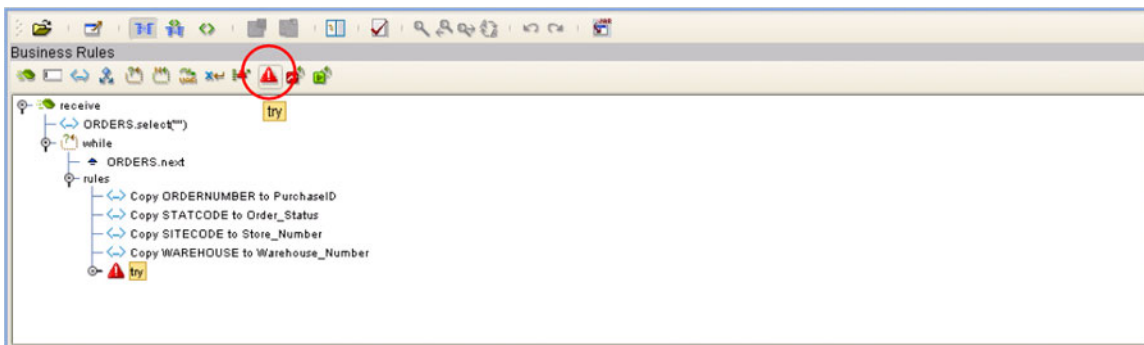
7.8 Using Try-Catch

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

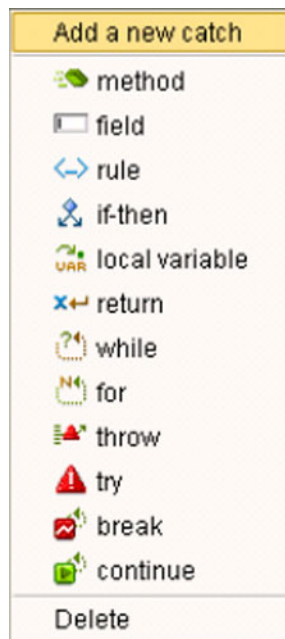
- 1 Click the **try** icon in the toolbar (see Figure 142), which adds the **try** rule to the business rules tree.

Figure 142 Try Icon



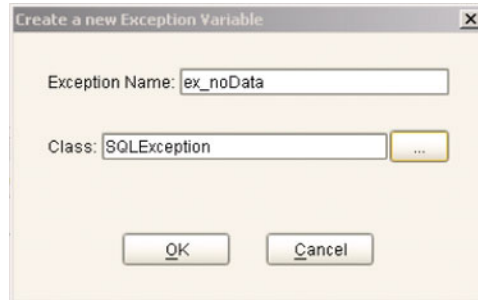
- 2 Right-click on the **try** rule to display the catch context menu (see Figure 143).

Figure 143 Catch Context Menu



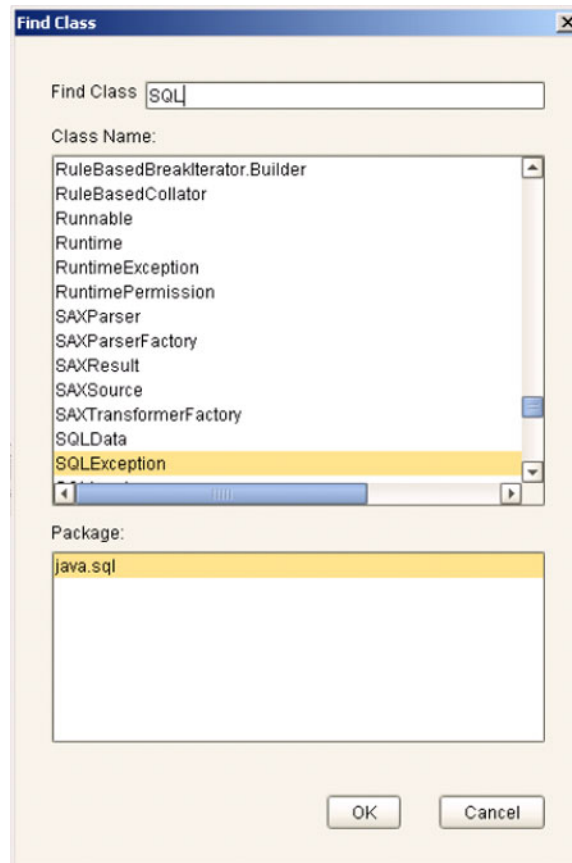
- 3 Select the **Add a new catch** option to add an exception variable, which displays a dialog box (see Figure 144).

Figure 144 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 145).

Figure 145 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.

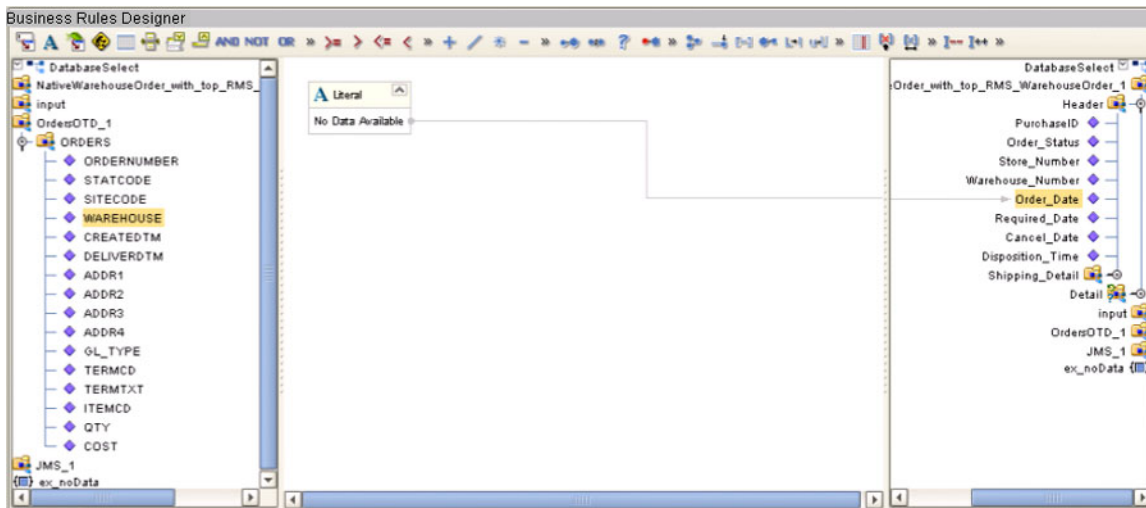
- 7 The catch rule is now added to the business rules tree (see Figure 146).

Figure 146 Catch SQLException Rule



- 8 The rule shown in Figure 146 contains a literal that represents the message that will be assigned to an outbound element Order_Date when the rule is executed (see Figure 147).

Figure 147 Exception Message



7.9 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 148) displays a file selection dialog box (see Figure 149).

Figure 148 Project Context Menu: New File

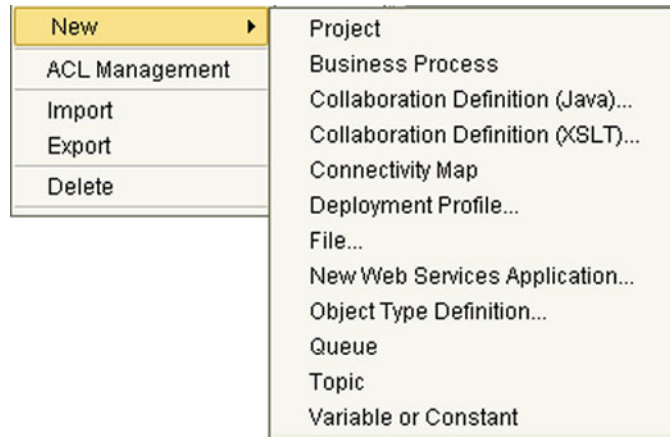
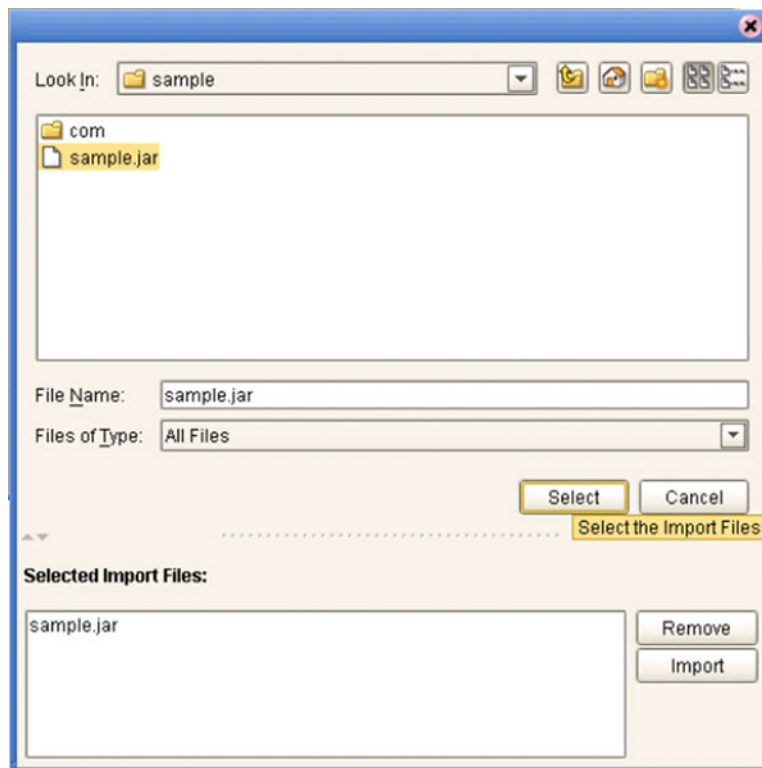


Figure 149 File Selection Dialog Box

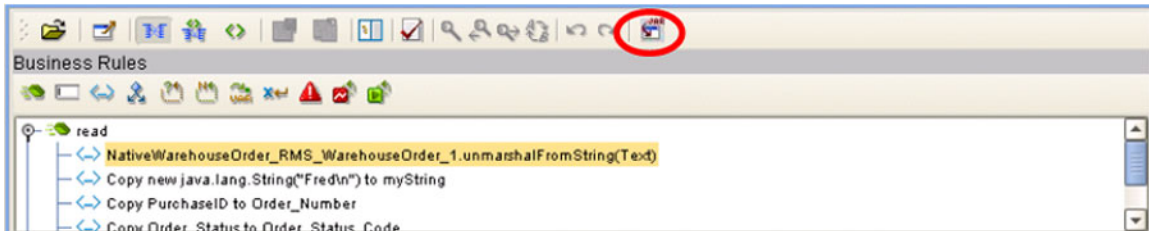


Locate the desired .jar files and click **Select** for each; then click **Import**.

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

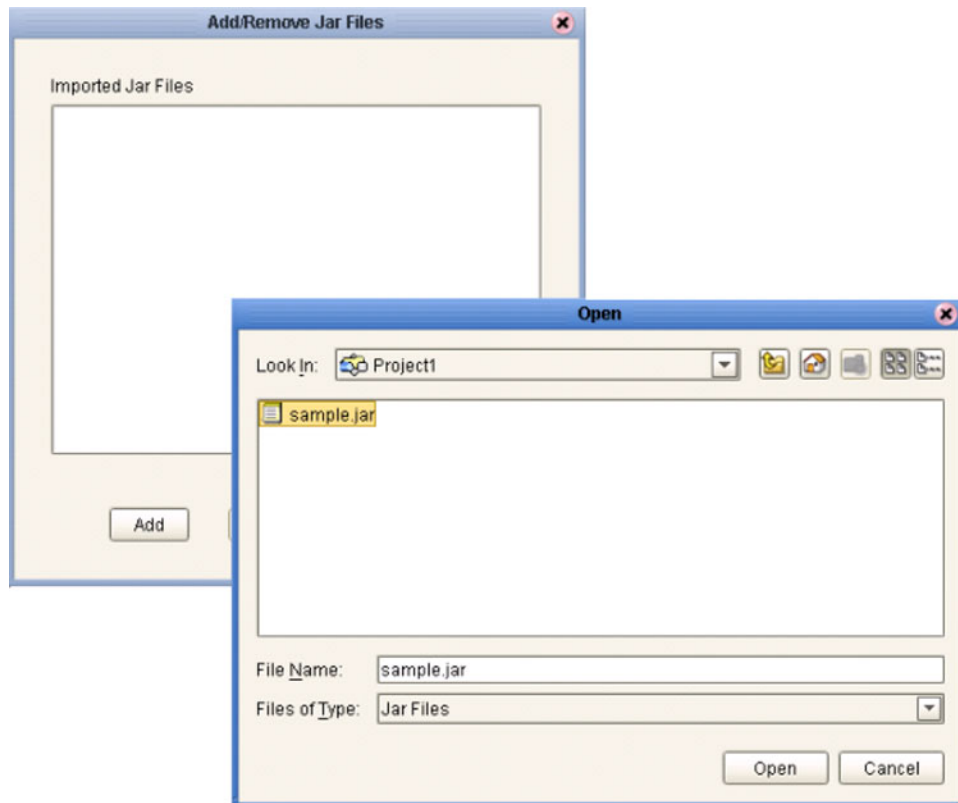
- 1 Open or create a Collaboration.
- 2 Click the **Import JAR File** icon in the toolbar (see Figure 150), which displays an *Add/Remove Jar Files* dialog box.

Figure 150 Import JAR File Icon



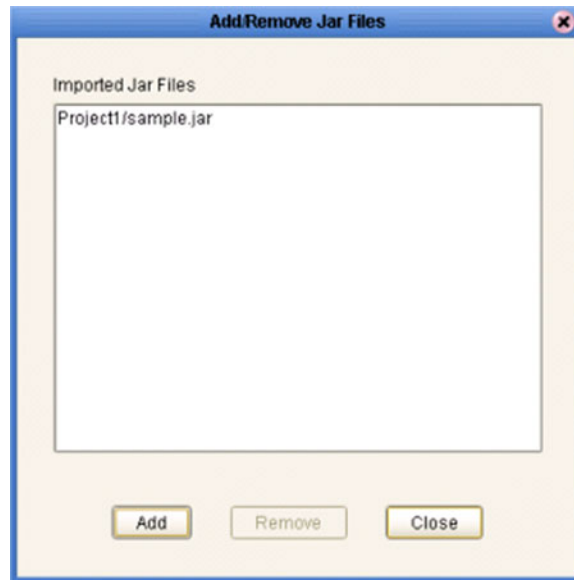
- 3 Click **Add** in the dialog box to display a file browser box (see Figure 152).
- 4 Locate the desired **.jar** files and click **Open** for each.

Figure 151 Add/Remove Jar File Dialog Box (1)



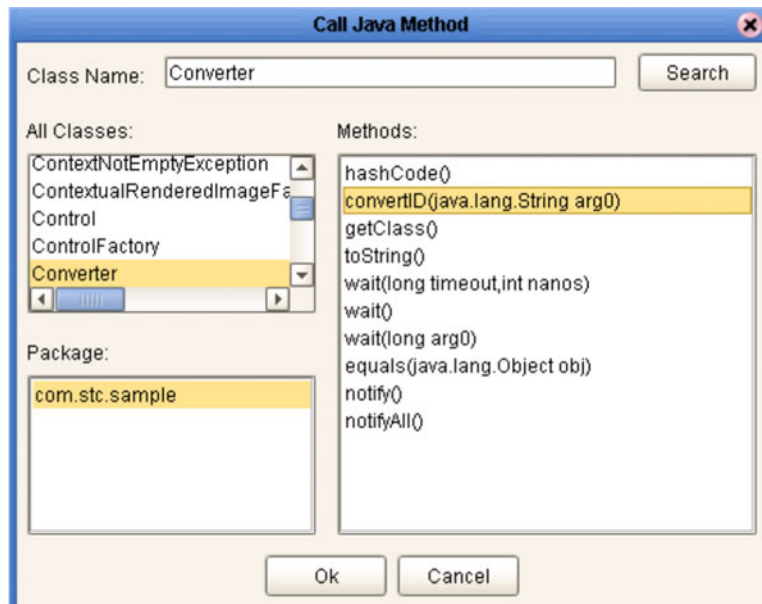
- When all the `.jar` files are listed in the *Add/Remove Jar Files* dialog box (see Figure 152), click **Close**.

Figure 152 Add/Remove Jar File Dialog Box (2)



- Click the **Call Java Method** icon in the Business Rules Designer, and a *Call Java Method* dialog box opens (see Figure 153).

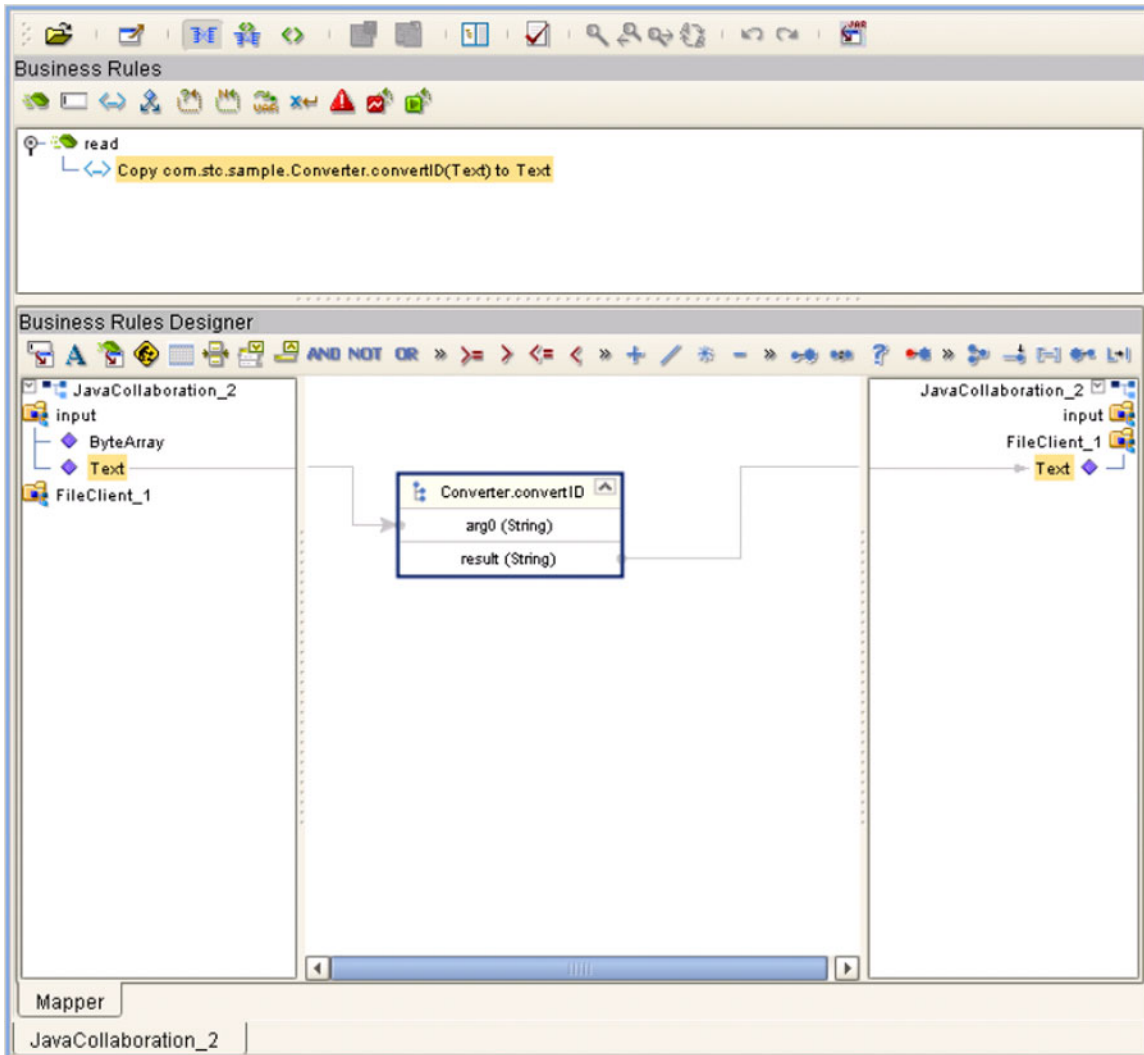
Figure 153 Call Java Method Dialog Box



- Select the desired class from the **All Classes** list.
- Select the desired method from the **Methods** list.
- 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 154).

Figure 154 Using the Third-Party Java Method



Important: If you are merging code that contains references to third-party classes or .jar files, you must import these files into the destination Project before merging the code.

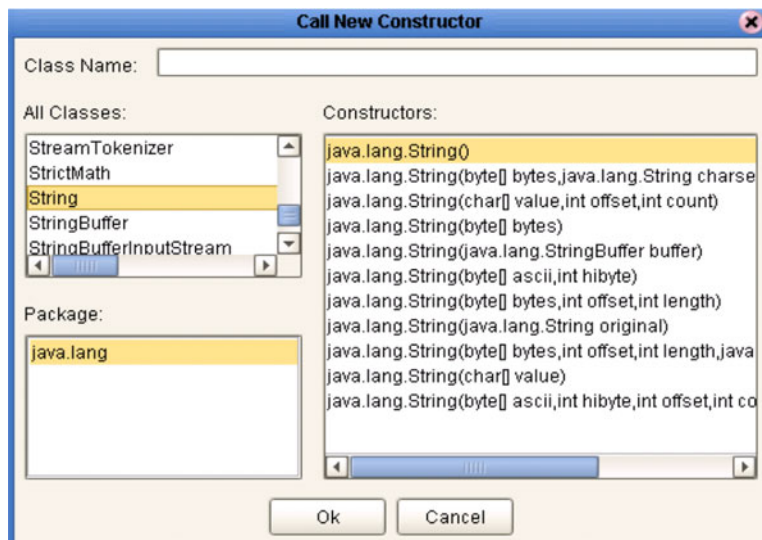
7.10 Adding Class Instances to a Collaboration

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 Java-based Collaboration Definition

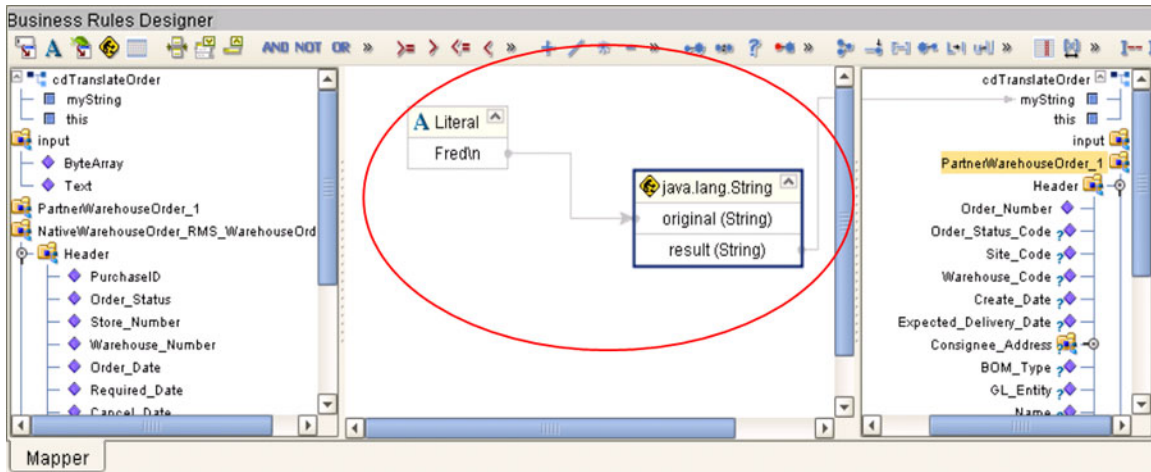
- 1 Create or open a Collaboration Definition and create a new rule.
- 2 Click the **Constructor** icon on the toolbar (see [Business Rules Designer Toolbar Icons](#) on page 151) to display the *Call New Constructor* dialog box (see Figure 155). This dialog box presents all available classes, including any third-party classes that have been uploaded.

Figure 155 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 156.

Figure 156 Constructor Example 1

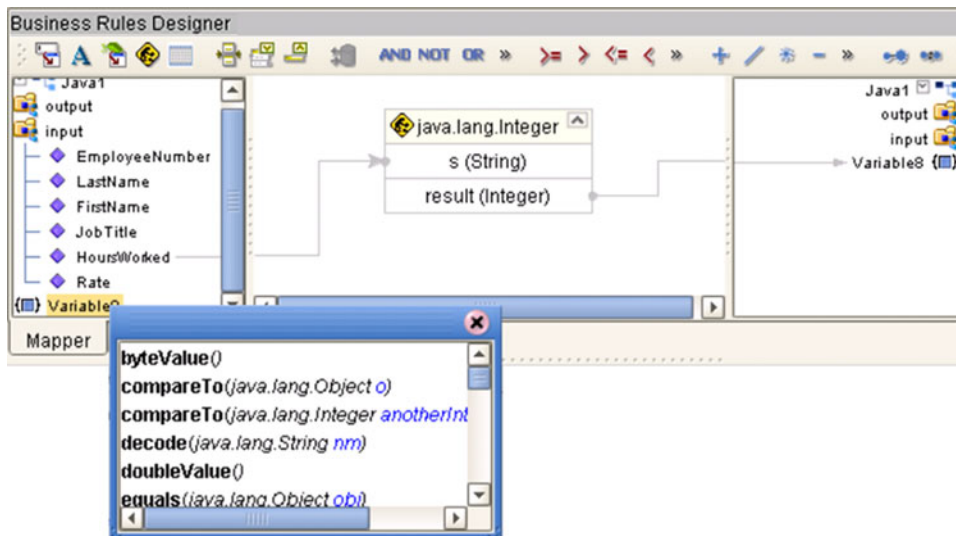


Alternative procedure to add a class instance to a Collaboration

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 157. The procedure is:

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

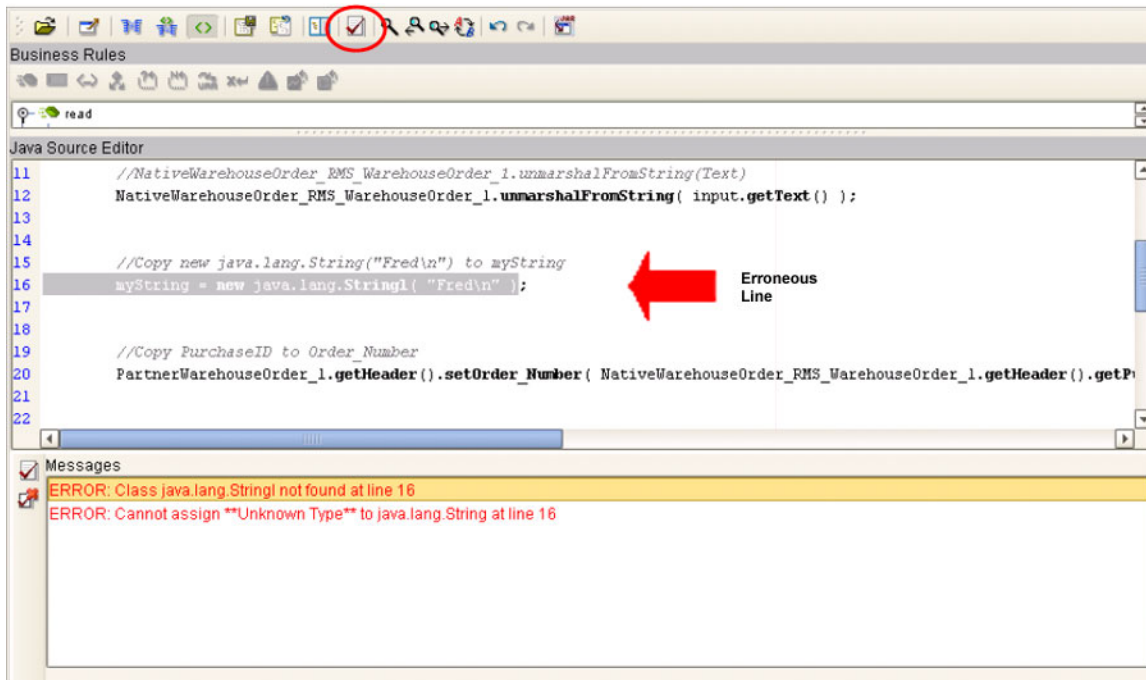
Figure 157 Constructor Example 2



7.11 Validating Java-based Collaborations

Clicking the **Validate** icon allows you to “precompile” the Java-based Collaboration Definition and display the errors in a validation panel, as shown in Figure 158. 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 158 Validating a Collaboration Definition



The JCE tester is enabled only when the OTDs involved in a particular Collaboration is are of the following types:

- XSD
- DTD
- User-defined

If the Collaboration contains any other type (for example, JMS receive) the tester is disabled.

7.12 Debugging Java-based Collaboration Definitions

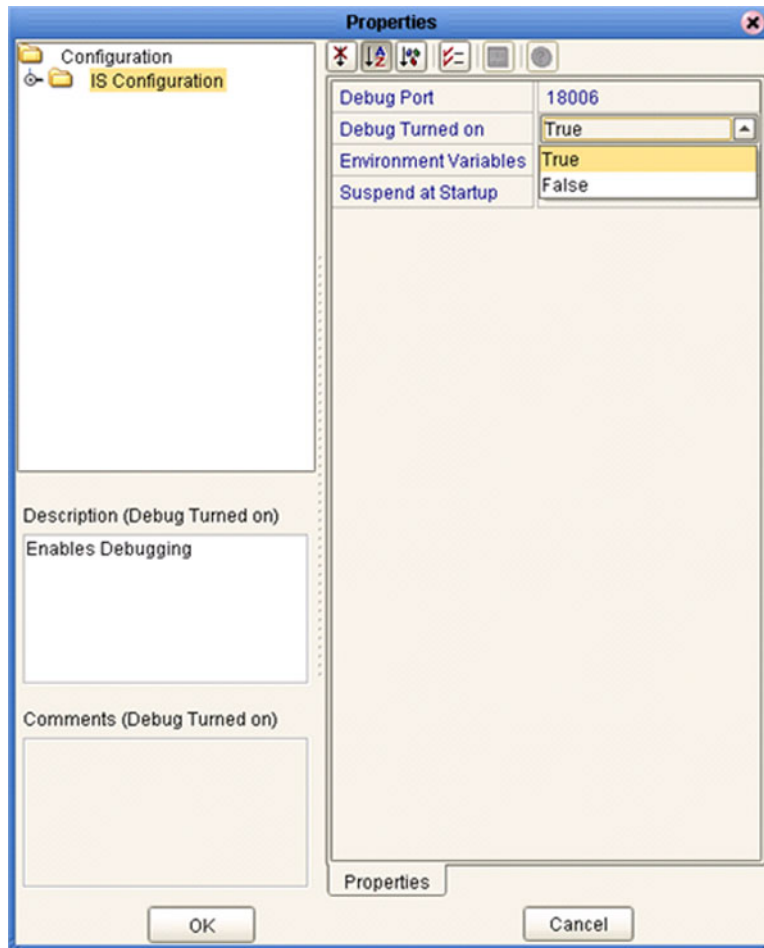
The Java Debugger enables you to debug Java-based Collaboration Definitions as deployed within an application server on a Logical Host, and offers an alternative to creating logs and warnings in an Java-based Collaboration and subsequently inspecting them via the Enterprise Manager.

7.12.1 Enabling the Debugger

To enable the Java Debugger

- 1 In Enterprise Explorer, select the appropriate application server.
- 2 Right-click to display the context menu.
- 3 Click **Properties** to display the Properties Dialog Box (see Figure 159).

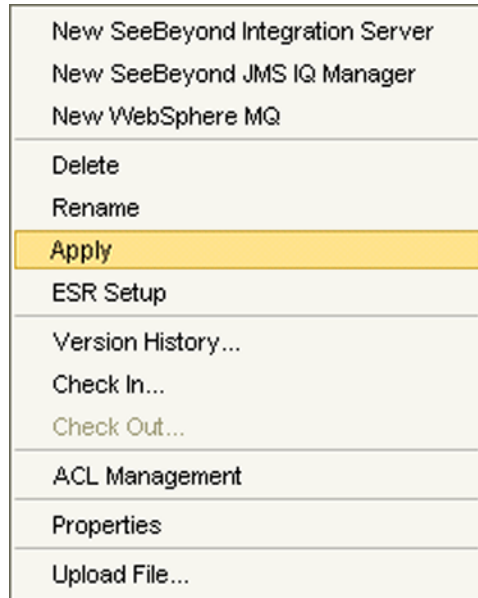
Figure 159 Application Server Properties Dialog Box



- 4 Set the value for *Debug Turned On* property to **True** and click **OK**.

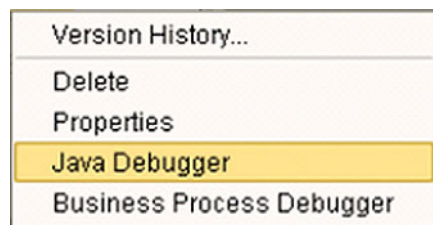
- 5 In Enterprise Explorer, select the Logical Host containing the integration server you have just configured.
- 6 Right-click to display the context menu and click **Apply** (see Figure 160).

Figure 160 Logical Host Context Menu



- 7 The **Java Debugger** option now is enabled in the Application Server context menu (see Figure 161).

Figure 161 Application Server Context Menu

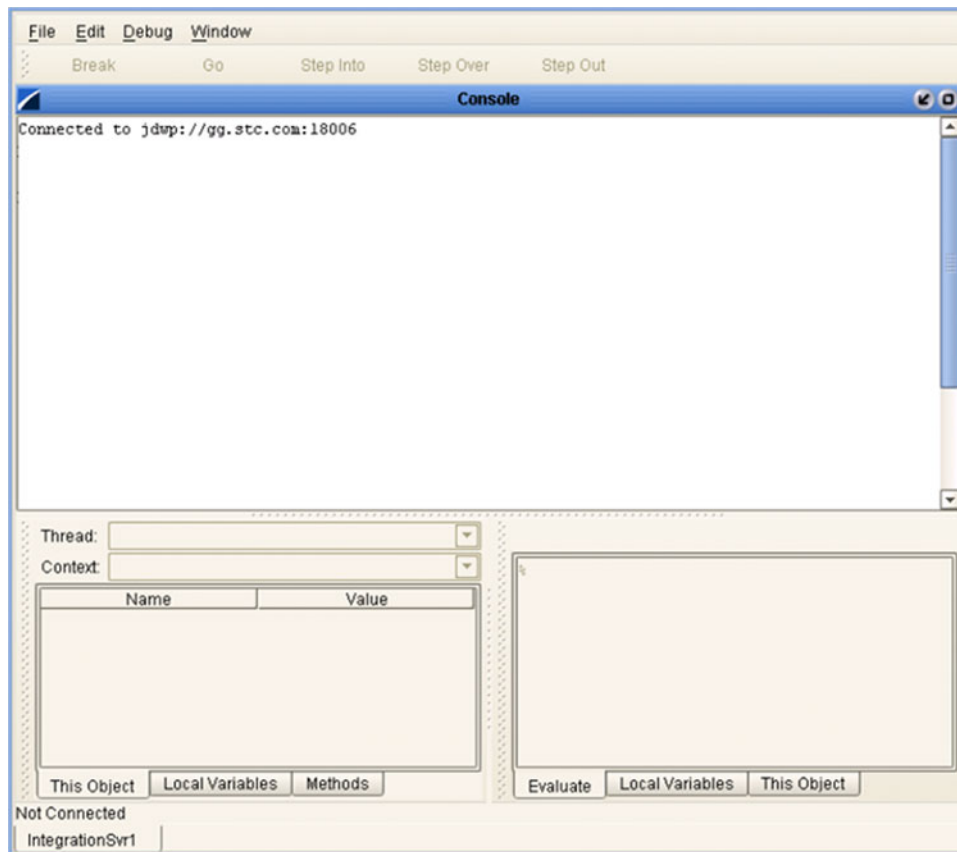


7.12.2 Invoking the Java Debugger

To invoke the Java Debugger

- 1 In Enterprise Explorer, select the appropriate integration server.
- 2 Right-click to display the context menu.
- 3 Click **Java Debugger** (see Figure 161), and the Java Debugger appears in the Enterprise Designer Editor panel (see Figure 162).

Figure 162 Java Debugger



Note: The Java Debugger appears whether or not the connection was successful. If there is no **Connected to ...** message, try the following procedure:

- A Select **Attach to JVM ...** from the File menu (see Figure 163), which displays the *Attach to JVM* dialog box (see Figure 164).

Figure 163 File Menu

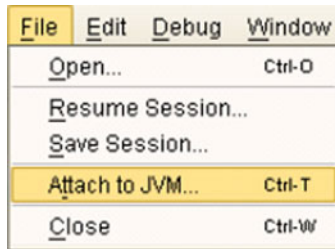
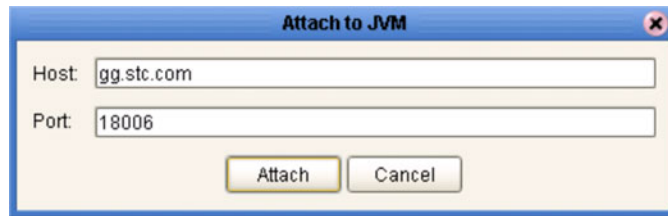
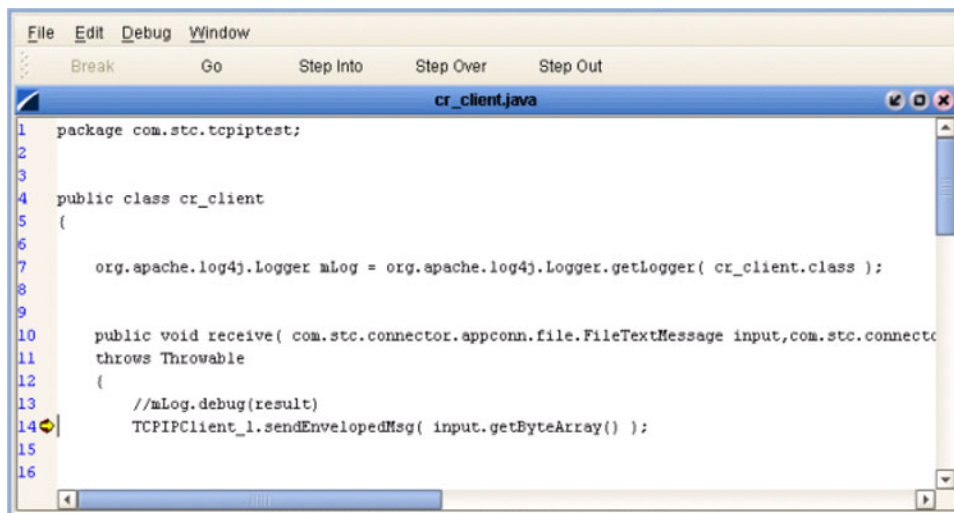


Figure 164 Attach to JVM Dialog Box



- 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-based Collaboration executes (see Figure 165).

Figure 165 Collaboration Source Code Display



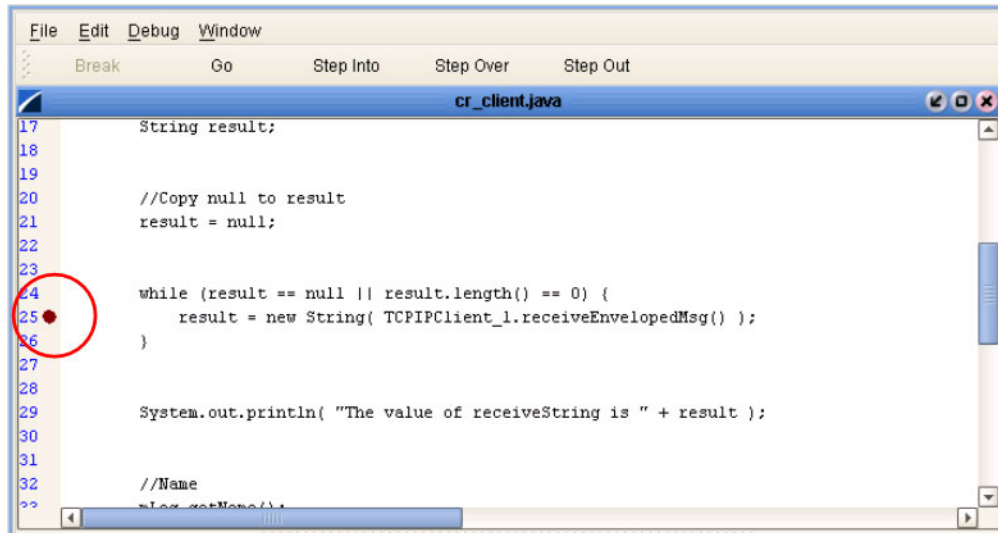
- 5 You can now set breakpoints to assist in examining and debugging the code.

7.12.3 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 166).

Figure 166 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 167). A *Stop in Method* dialog box is displayed, in which you can select the desired method (see Figure 168).

Figure 167 Debug Menu

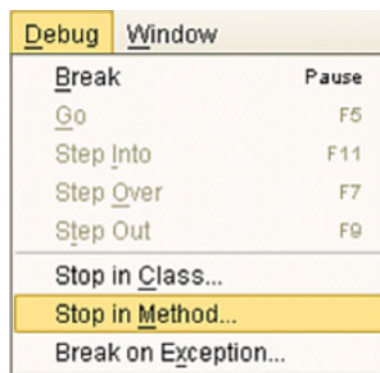
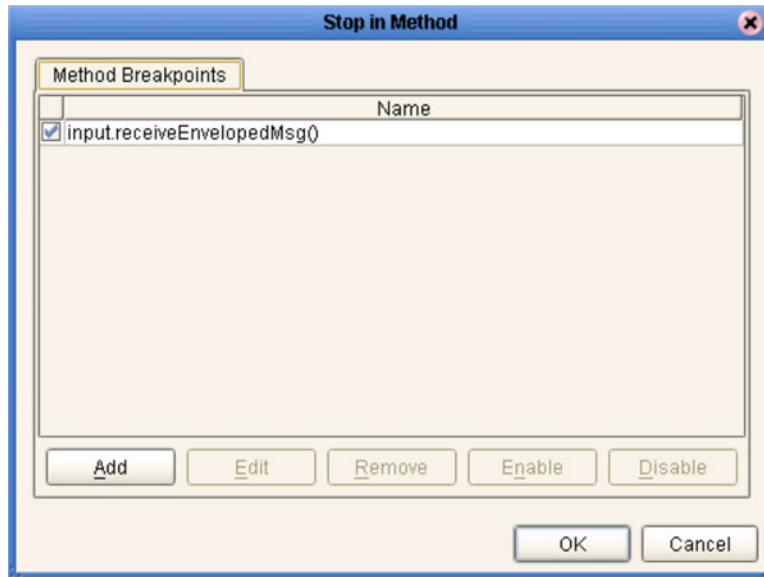


Figure 168 Stop in Method Dialog Box



- B To break on an exception, select **Break on Exception ...** from the Debug menu, which displays a *Choose Exception* dialog box (see Figure 169). All occurrences of the specified exception are then trapped and reported (see Figure 170).

Figure 169 Choose Exception Dialog Box

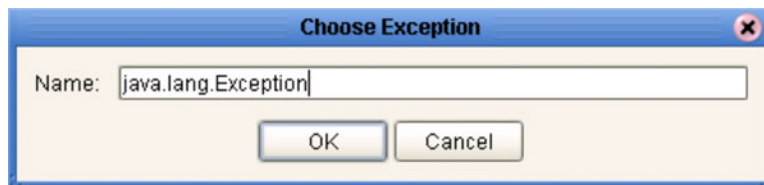


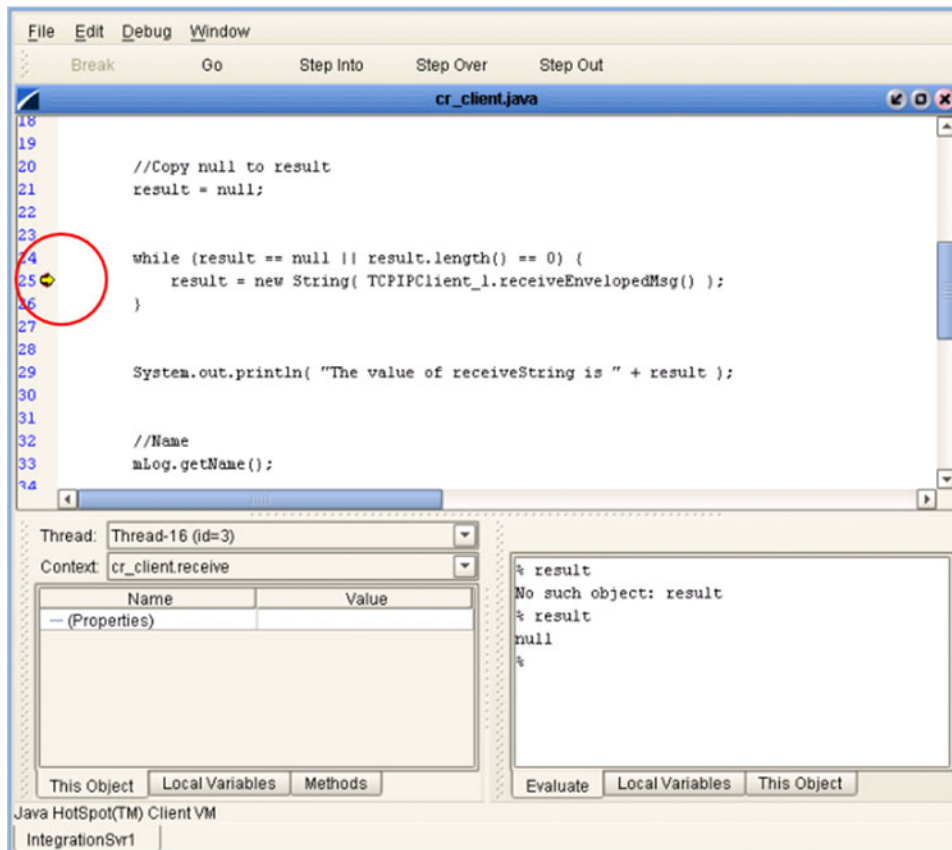
Figure 170 Break on Exception Dialog Box



7.12.4 Inspecting and Editing the Source Code

As soon as the execution of the Java-based Collaboration arrives at a set breakpoint, it stops executing and displays a right arrow indicator next to the line number in the source code (see Figure 171).

Figure 171 Breakpoint Indicator

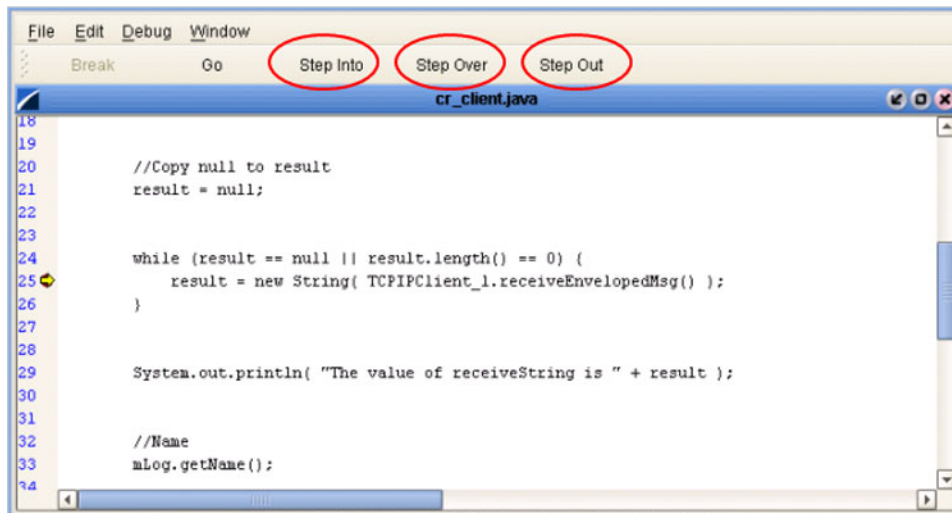


At this point, you can continue by (for example):

- ◆ Stepping Into
- ◆ Stepping Over
- ◆ Stepping Out
- ◆ Inspecting a local variable
- ◆ Setting a local variable

Stepping Into, Over, or Out

Figure 172 Stepping Into, Over, and Out Commands

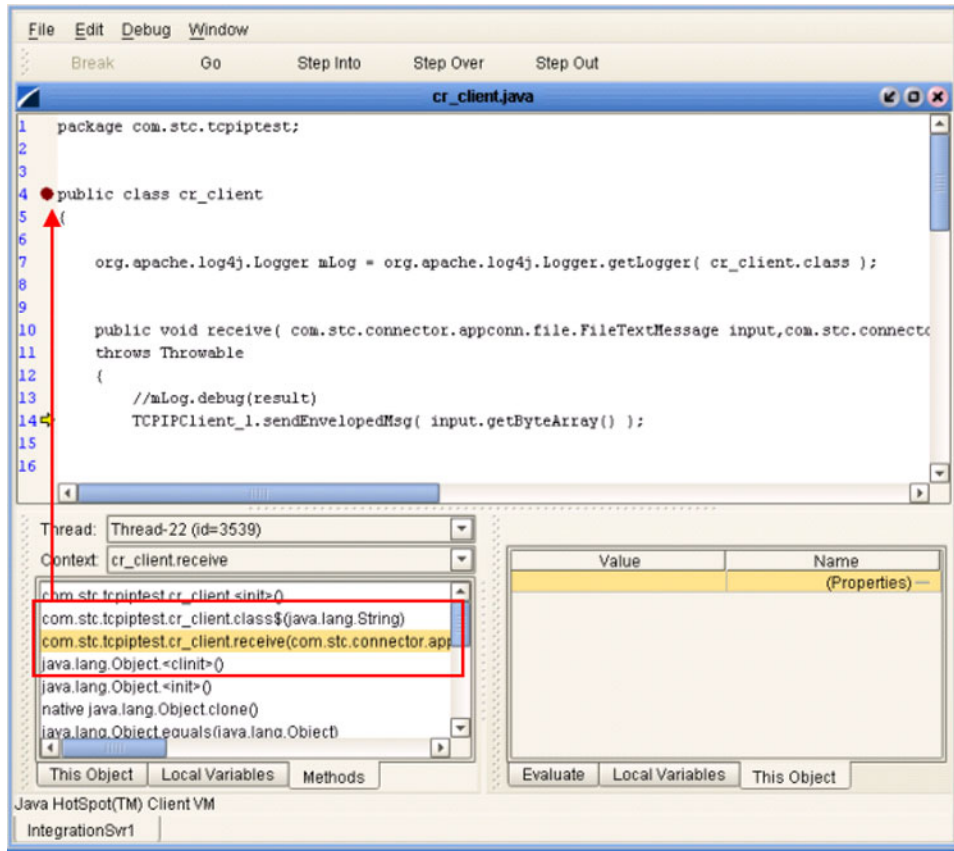


- By selecting the **Step Into** option (see Figure 172), the breakpoint is lifted and execution of the Collaboration will continue, *including* the line of code at the breakpoint.
- By selecting the **Step Over** option, the breakpoint is lifted and execution of the Collaboration will continue, *ignoring* 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 173).

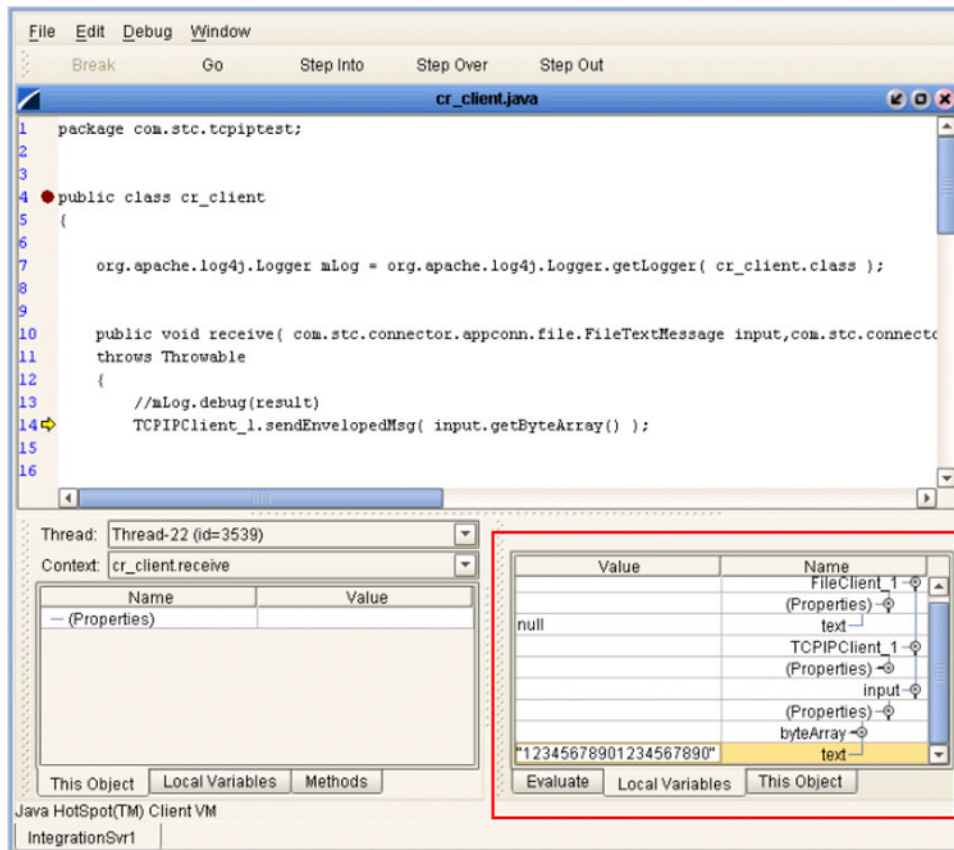
Figure 173 Java Thread and Method Display



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 174). 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 174 Local Variables Tab

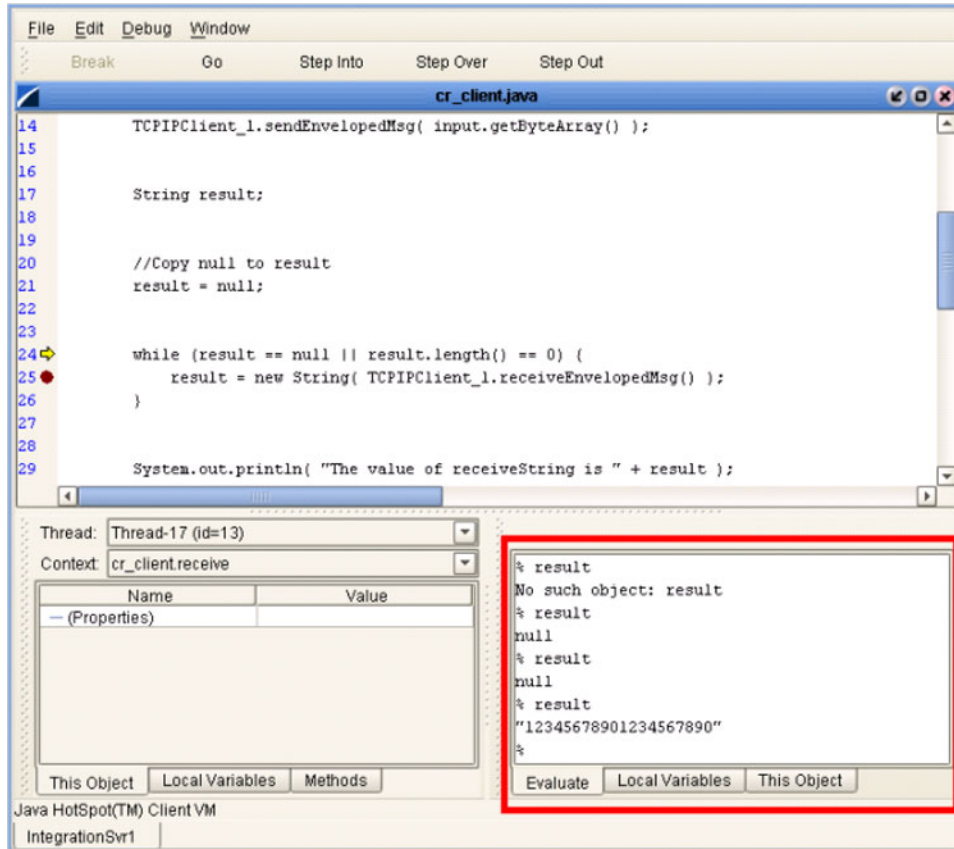


You also can inspect a local variable by selecting the **Evaluate** tab and entering the variable name in the panel using the following syntax:

```
% <variable_name>
```

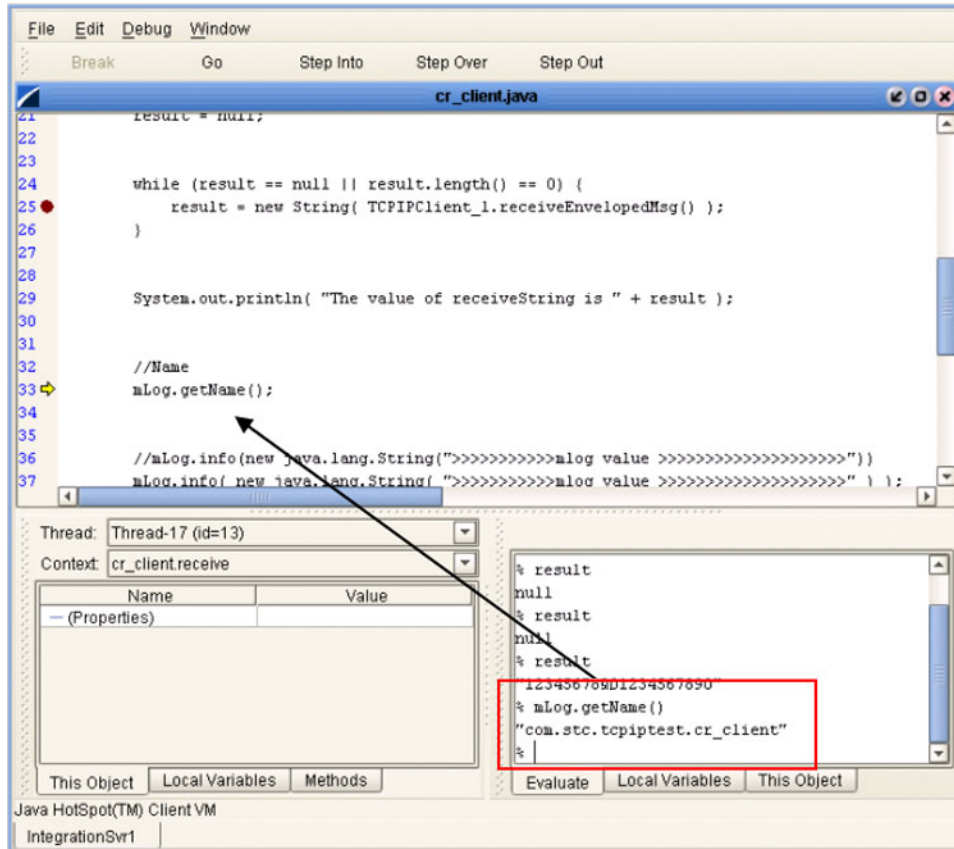
If the code has not initialized a variable, *No such object* is displayed; otherwise, the current value of the variable is displayed (see Figure 175).

Figure 175 Evaluate Local Variable



You can also inspect the result of a method by entering the method name, following the same syntax format (see Figure 176).

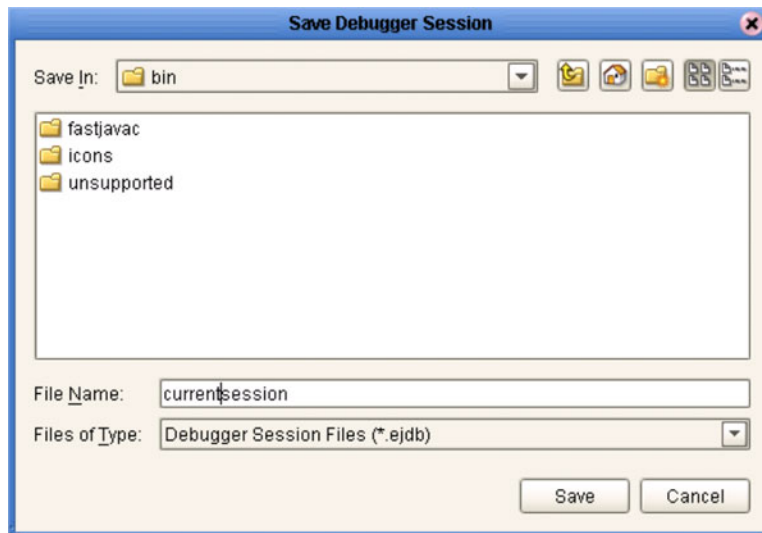
Figure 176 Evaluate Method



Saving and Resuming Debug Sessions

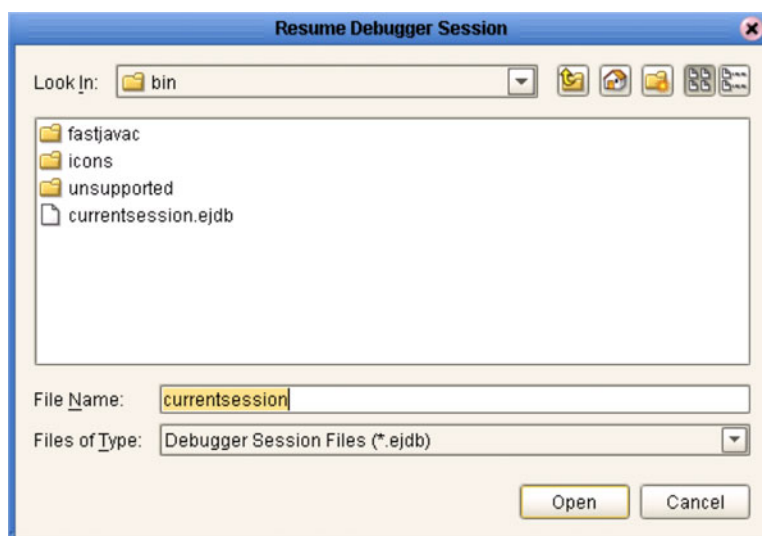
You can pause the debugging process by saving the session to a file. Selecting **Save Session** from the debugger File menu displays the *Save Debugger Session* dialog box (see Figure 177), in which you specify the file name and location.

Figure 177 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 178), in which you specify the file name and location.

Figure 178 Resume Debugger Session Dialog Box



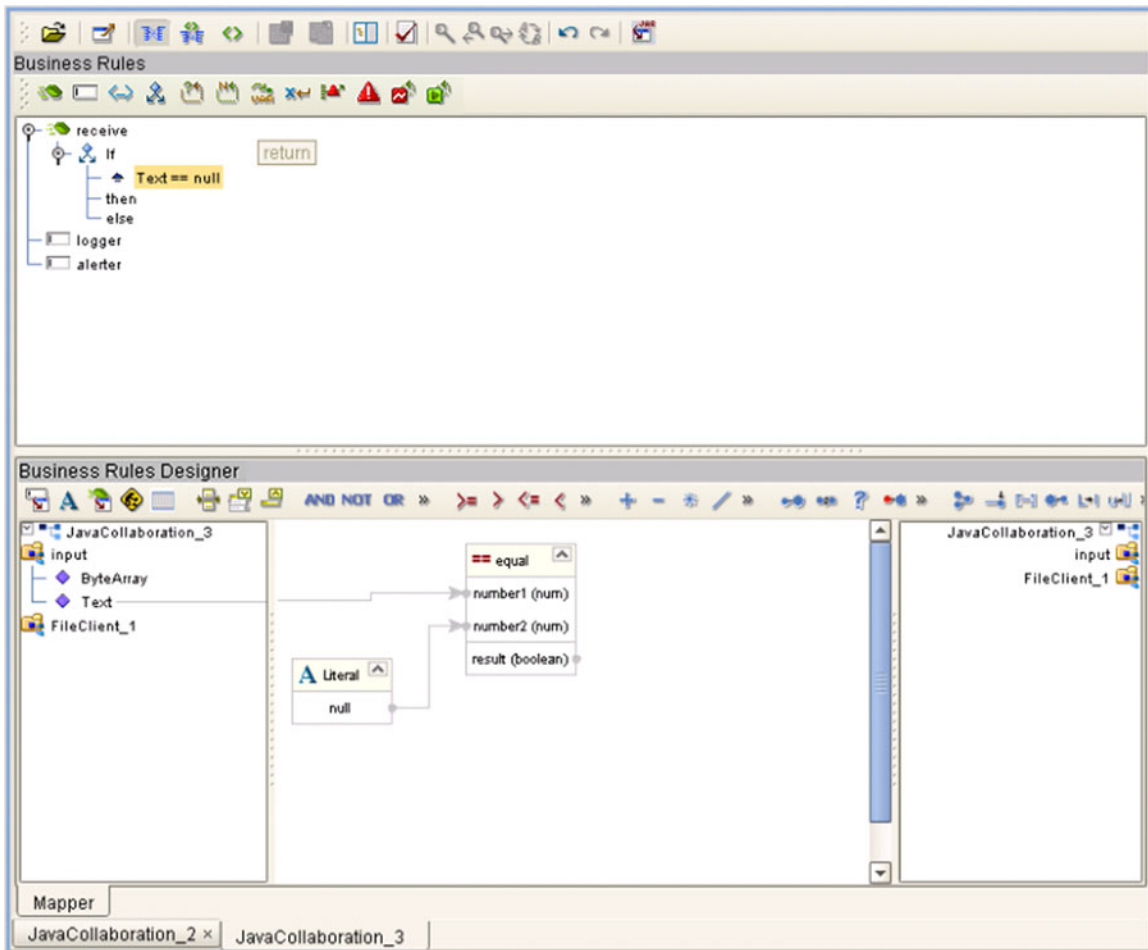
7.13 Creating Alerts

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

To create alerts using the JCE

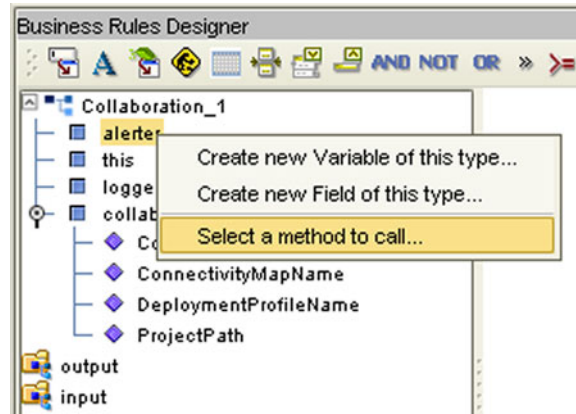
- 1 Create a new Java-based Collaboration.
- 2 Add your business rules. As an example, you might perform a test to determine whether or not a file is empty—if it is empty, then raise an alert (see Figure 179).

Figure 179 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 180).

Figure 180 Alert Menu



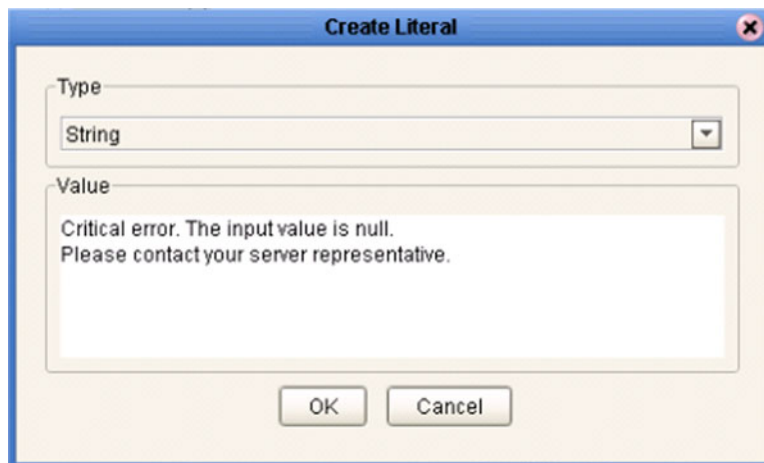
- 5 Click *Select a method to call ...*
- 6 Select the severity of the alert from the selection window (see Figure 181). As an example, we will select **critical**.

Figure 181 Alert Severity Selection Window



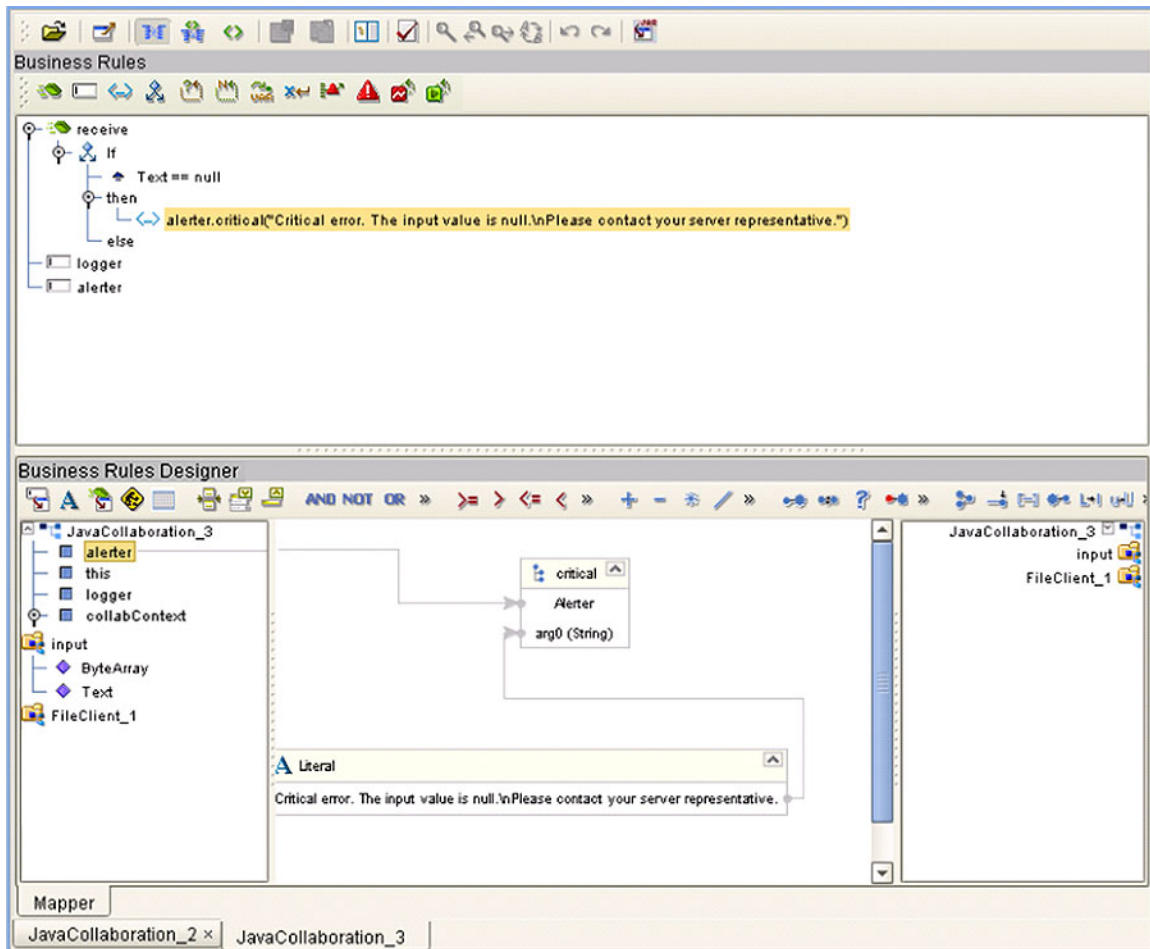
- 7 Create your alert message, which can be a literal, a constant, or an OTD field name. As an example, we will create a **literal** (see Figure 182).

Figure 182 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 183).

Figure 183 Pass Alert Message to Object Argument



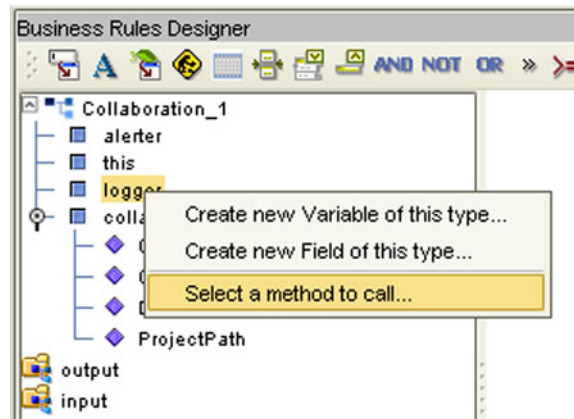
7.14 Creating Log Entries

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

To create log entries using the JCE

- 1 Create a new Collaboration.
- 2 Add your Collaboration rules and initiate a logging event.
- 3 Right-click the **logger** node for the Collaboration in the left pane of the Business Rules Designer, which displays a context menu (see Figure 184).

Figure 184 Logging Menu



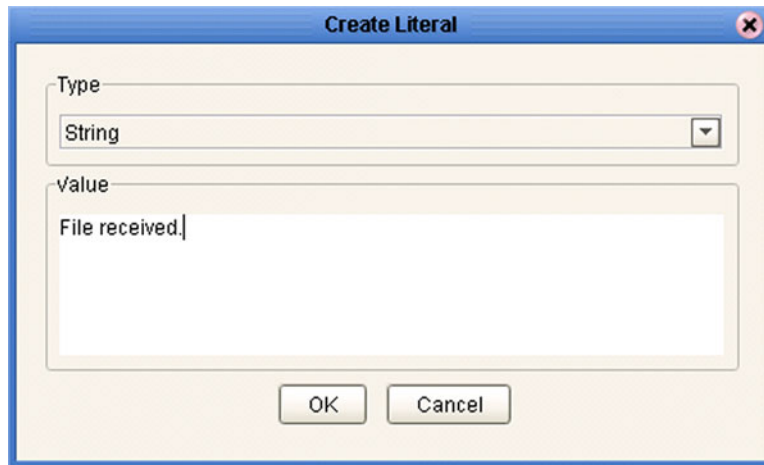
- 4 Click *Select a method to call ...*
- 5 Select the logging level with desired method from the selection window (see Figure 185). As an example, we will select **Debug**.

Figure 185 Logging Level/Method Selection Window



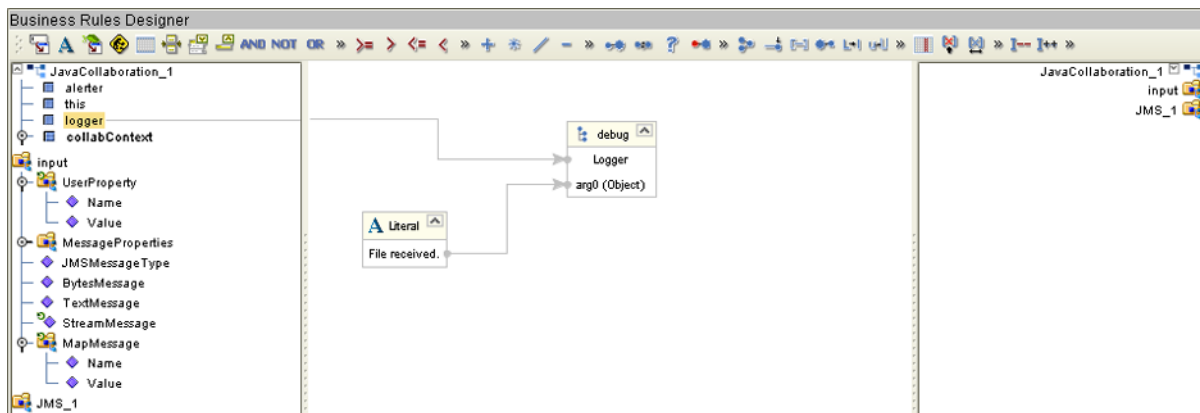
- 6 Create your log message, which can be a literal, a constant, or an OTD field name. As an example, we will create a **literal** (see Figure 186).

Figure 186 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 187).

Figure 187 Pass Log Message to Object Argument



Collaboration Definitions (XSLT)

This chapter describes the process for building XSLT-based Collaboration Definitions.

8.1 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 Collaboration Definition Wizard (XSLT) and Collaboration Editor (XSLT), that are used to create and customize your XSLT-based Collaboration Definitions. You must have OTDs available to use as the foundation for creating an XSLT-based Collaboration Definition. See [Object Type Definitions](#) on page 89 for more details.

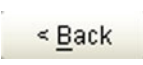




Important: *If you delete an OTD in the Project Explorer, any XSLT-based Collaboration Definitions that have been built using that OTD will be affected. It is recommended that you run the Impact Analyzer before attempting to delete any OTDs (see [Impact Analyzer](#) on page 67).*

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. This procedure should also be performed before creating the Connectivity Map and Deployment Profile.*

8.2 Using the Collaboration Definition Wizard (XSLT)

The Collaboration Definition Wizard (XSLT) guides you through the initial phases of creating an XSLT-based Collaboration Definition, and then invokes the Collaboration Editor (XSLT). The user interface is highly self-explanatory, but details of the navigation buttons are listed in Table 44 for your reference.

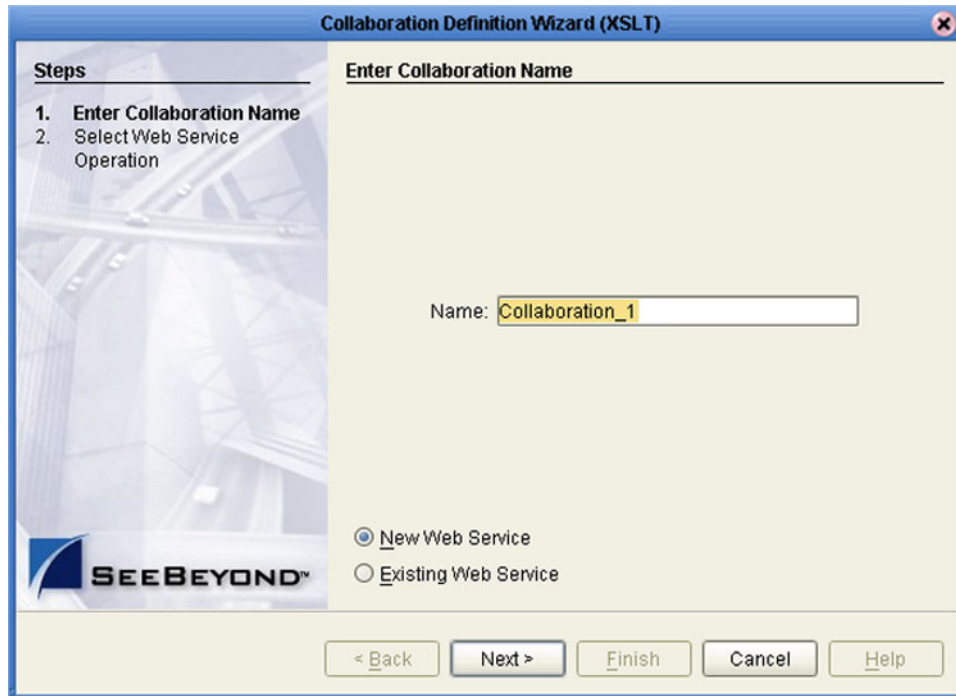
Table 44 Wizard Navigation Buttons

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.

8.2.1 Creating a Collaboration Definition (XSLT)

- 1 Right-click on a Project in the Enterprise Explorer to display the Project context menu.
- 2 Select **New > Collaboration Definition (XSLT)** to invoke the Collaboration Definition Wizard (XSLT).
- 3 Enter a **Name** for your Collaboration, as shown in Figure 188.

Figure 188 Collaboration Wizard (XSLT) Dialog Box



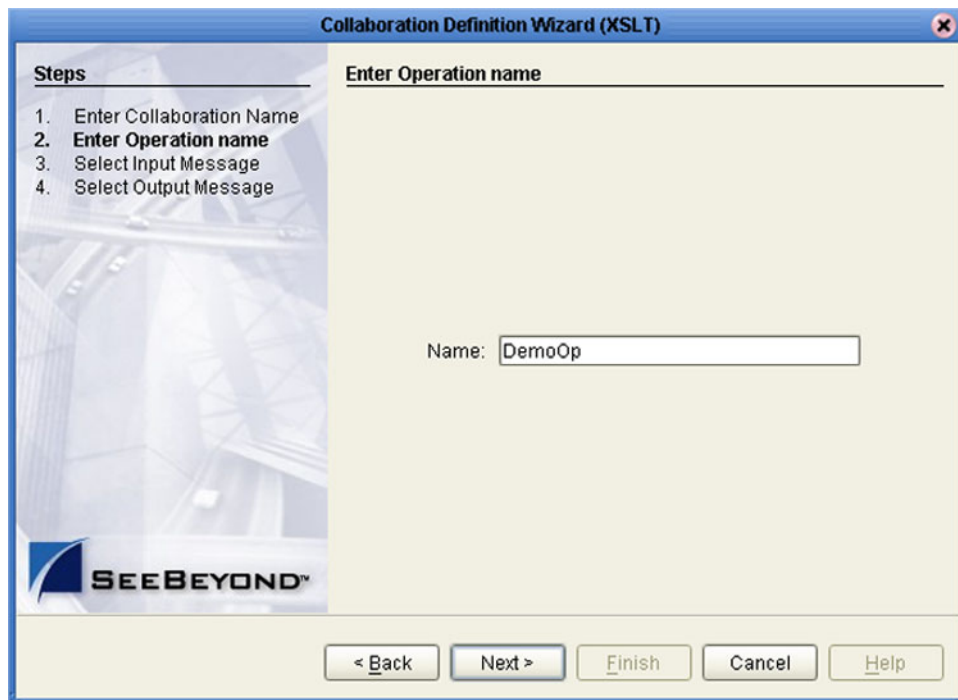
- 4 Select a Web service, which can be either:
 - A New Web Service.
 - An Existing Web Service (for example, an eInsight process or a Java Web Service Operation).
- 5 Click **Next** to proceed to the next Wizard dialog, which is dependent upon your Web Service selection.

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 189. This will become the *method* that can be used to invoke the XSLT-based Collaboration as a Web service.

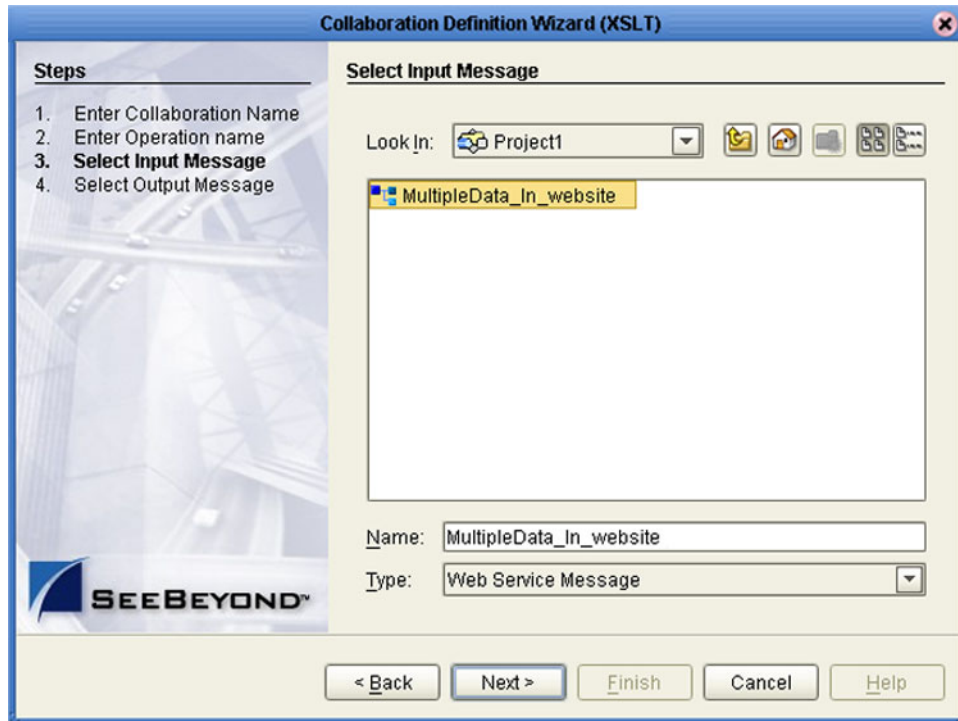
Figure 189 New Web Service: Operation Name



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

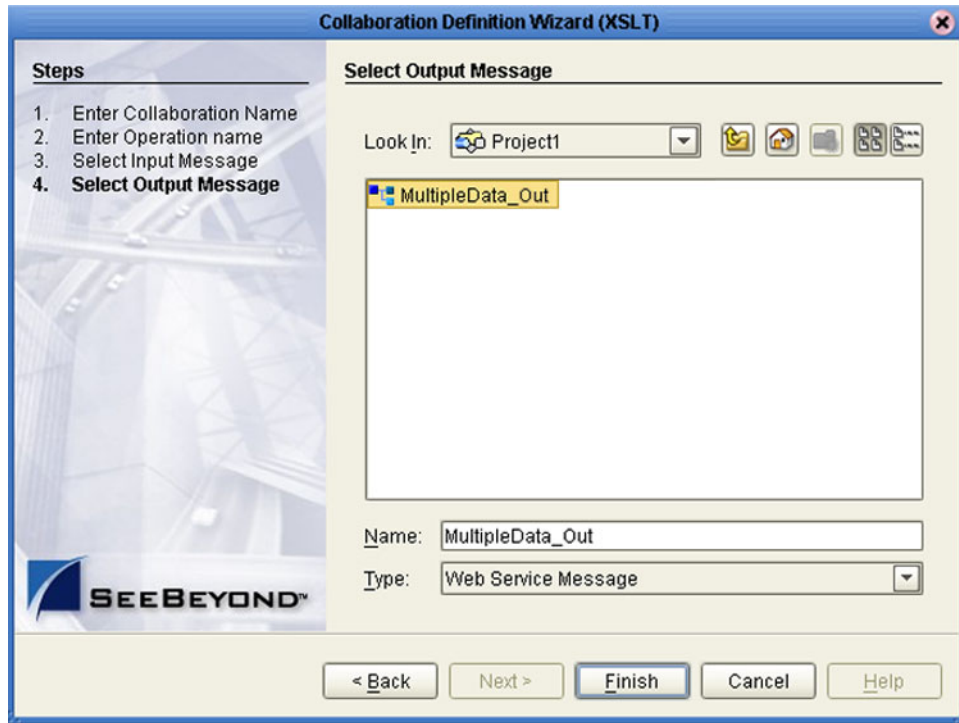
- 3 Select the input Web service message, as shown in Figure 190.

Figure 190 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 191.

Figure 191 New Web Service: Output Message



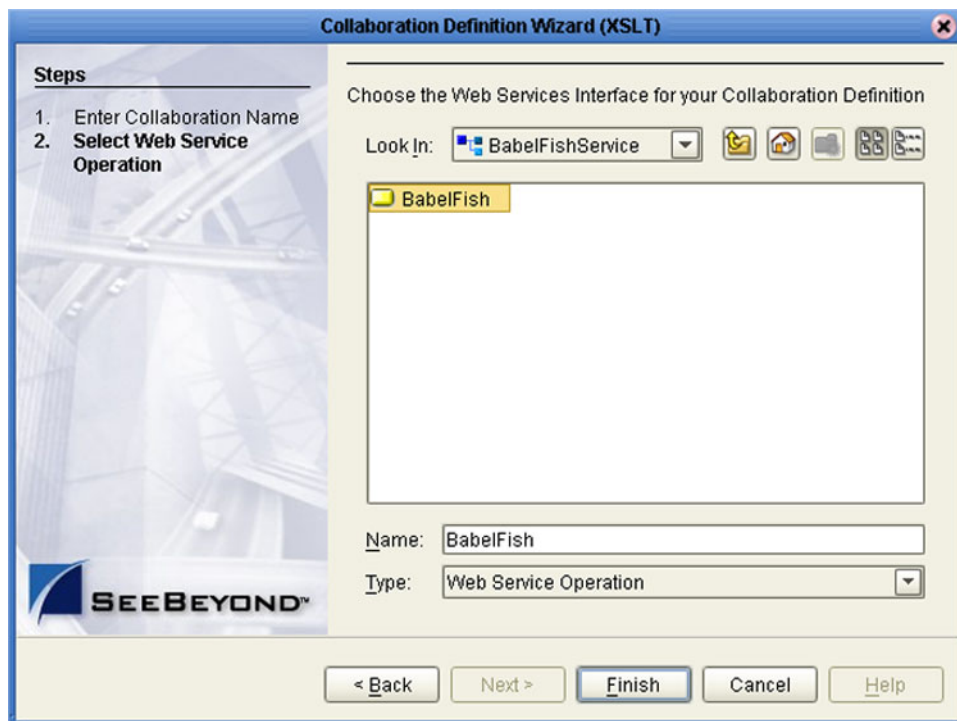
- 6 Click **Finish** to proceed to the Collaboration Editor (XSLT).

Existing Web Service

If you selected an Existing Web Service, you will be presented with the Wizard dialog shown in Figure 192.

- 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 192 Existing Web Service: Select Operation

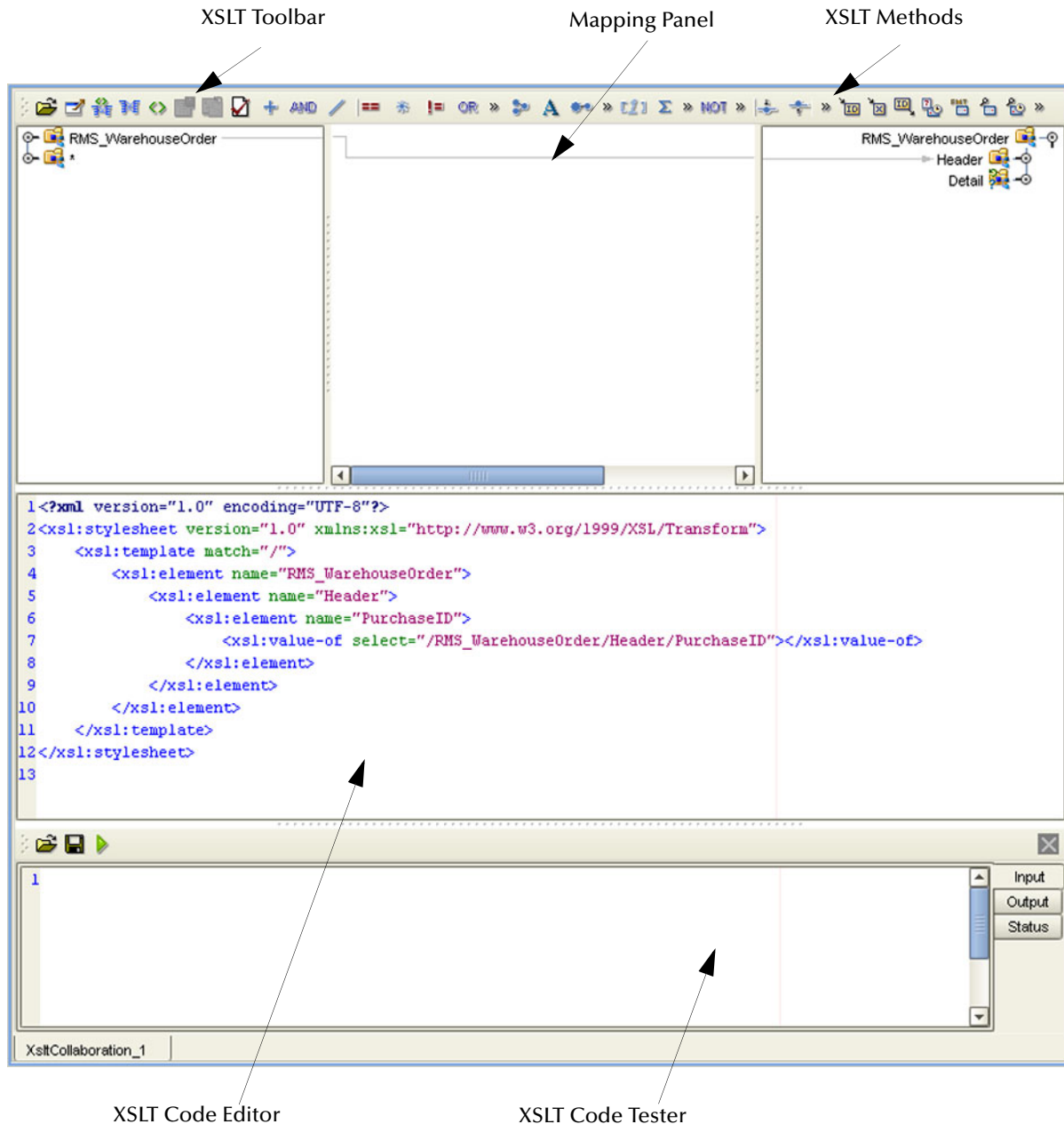


- 2 Click **Finish** to proceed to the Collaboration Editor (XSLT).

8.3 Using the Collaboration Editor (XSLT)

After you have created an XSLT-based Collaboration Definition using the Collaboration Definition Wizard (XSLT), the Collaboration Editor (XSLT) appears in the Editor panel of the Enterprise Designer. Major features of this window are identified in Figure 193.

Figure 193 Collaboration Editor (XSLT)












You can also invoke the Collaboration Editor (XSLT) by selecting **Open** in the context menu for an existing XSLT-based Collaboration Definition in the Enterprise Explorer.

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 45. 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. The Tester panel allows you to run the XSLT code without deploying the Project.

8.3.1 XSLT Toolbar Icons

Table 45 XSLT Toolbar Icons

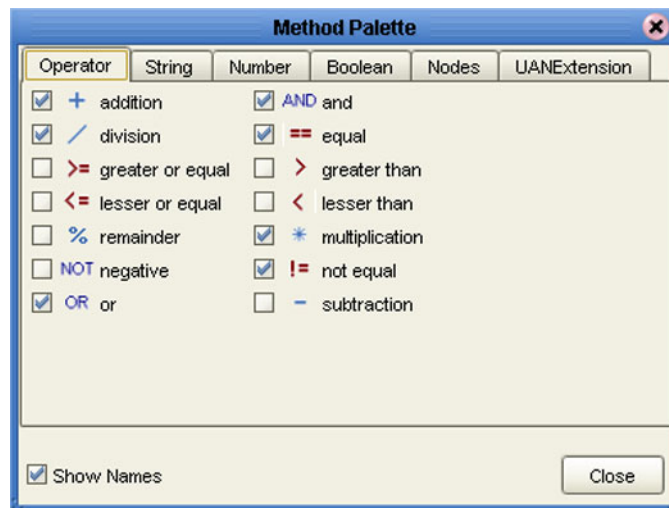
Icon	Command	Function
	Import XSLT from a Local File	Displays the Open dialog box, which you can use to locate and select an XSL file to import.
	Save XSLT to a Local File	Displays the Save dialog box, which you can use to save the selected XSLT-based Collaboration Definition to a file.
	Show Maps and 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	Cancels changes made to the XSLT code since the last time it was committed.
	Generate XSLT Code from Map Rules	Generates XSLT code from the mapping constructed in the Mapping panel.
	Test XSLT Code	Displays the XSLT Tester panel.

Collaboration Method Palette (XSLT)

The Collaboration Method Palette includes a series of method icons that you can drag onto the Mapping area. Click the Chevron (>>) to the right of the method groups to display the dialog box shown in Figure 194.

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 [Collaboration Methods \(XSLT\)](#) on page 218. Note that **UANExtension** is an optional add-on.

Figure 194 XSLT Collaboration Method Palette Dialog Box



Collaboration Method Boxes (XSLT)

The method boxes are placed in the mapping area by dragging the corresponding icon from the method palette toolbar. As shown in [Figure 193 on page 215](#), 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 195); you can collapse them (see Figure 196) by clicking the caret (^) in the upper right corner of the box. Clicking the now-inverted caret expands the box. Some boxes expand further as needed to provide additional argument nodes.

Figure 195 Expanded Method Box

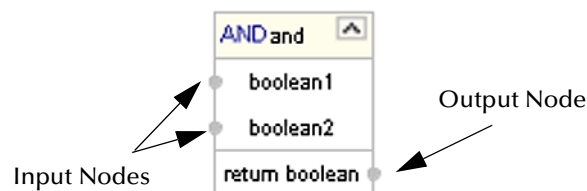


Figure 196 Collapsed Method Box



8.4 Collaboration Methods (XSLT)

8.4.1 Operator Methods

Figure 197 Method Palette: Operator Methods

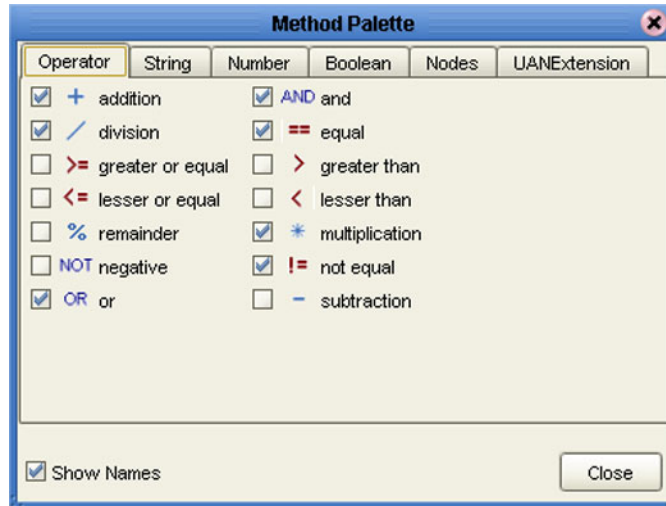


Table 46 Operator Collaboration Methods (XSLT)

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 46 Operator Collaboration Methods (XSLT)


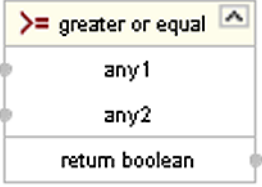
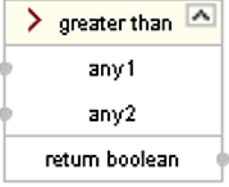
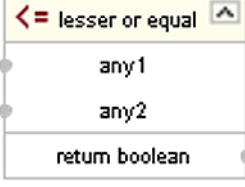
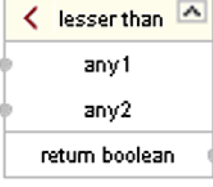
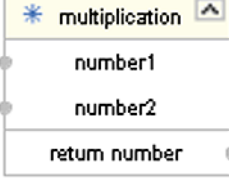
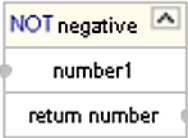
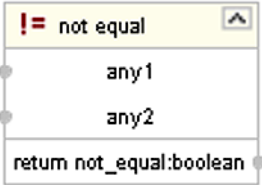
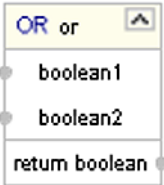
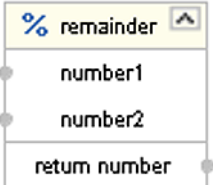
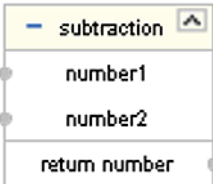
Method Box	Description/Usage
	<p>The equal method returns Boolean true if <i>any1</i> is equal to <i>any2</i>; otherwise, returns Boolean false.</p>
	<p>The greater_or_equal method returns Boolean true if <i>any1</i> is greater than or equal to <i>any2</i>; otherwise, returns Boolean false.</p>
	<p>The greater_than method returns Boolean true if <i>any1</i> is greater than <i>any2</i>; otherwise, returns Boolean false.</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 lesser_than method returns Boolean true if <i>any1</i> is less than <i>any2</i>; otherwise, returns Boolean false.</p>
	<p>The multiplication method multiplies the value of <i>number1</i> by the value of <i>number2</i>, returns the product.</p>

Table 46 Operator Collaboration Methods (XSLT)

Method Box	Description/Usage
 <p>The method box for 'NOT negative' has a title bar with the text 'NOT negative' and a small upward arrow icon. Below the title bar, there is a single input field labeled 'number1'. At the bottom of the box, there is a 'return number' field with a small dot to its right.</p>	<p>The negative method returns the arithmetic negation of <i>number1</i>.</p>
 <p>The method box for '!= not equal' has a title bar with the text '!= not equal' and a small upward arrow icon. Below the title bar, there are two input fields labeled 'any1' and 'any2'. At the bottom of the box, there is a 'return not_equal:boolean' field with a small dot to its right.</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' has a title bar with the text 'OR or' and a small upward arrow icon. Below the title bar, there are two input fields labeled 'boolean1' and 'boolean2'. At the bottom of the box, there is a 'return boolean' field with a small dot to its right.</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 '% remainder' has a title bar with the text '% remainder' and a small upward arrow icon. Below the title bar, there are two input fields labeled 'number1' and 'number2'. At the bottom of the box, there is a 'return number' field with a small dot to its right.</p>	<p>The remainder method divides the numerical value of <i>number1</i> by the numerical value of <i>number2</i>, and returns the remainder.</p>
 <p>The method box for '- subtraction' has a title bar with the text '- subtraction' and a small upward arrow icon. Below the title bar, there are two input fields labeled 'number1' and 'number2'. At the bottom of the box, there is a 'return number' field with a small dot to its right.</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>

8.4.2 String Methods

Figure 198 Method Palette: String Methods

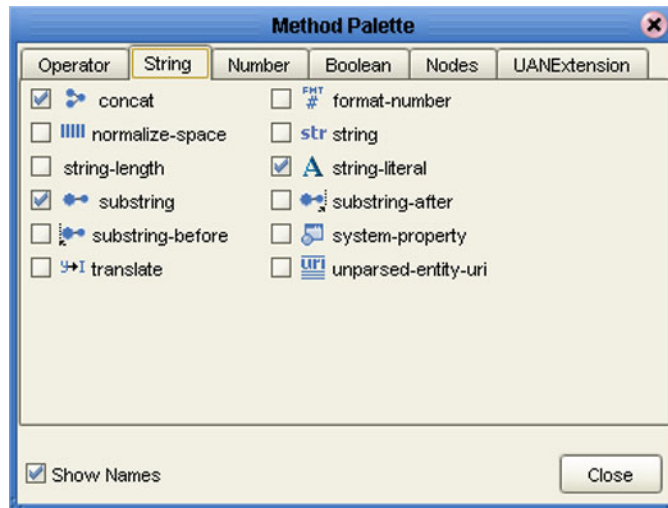


Table 47 String Collaboration Methods (XSLT)

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 47 String Collaboration Methods (XSLT)

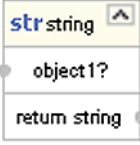
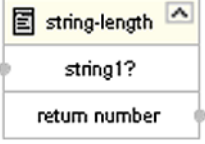


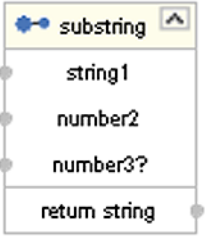
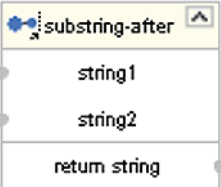
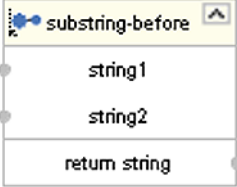
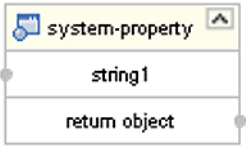

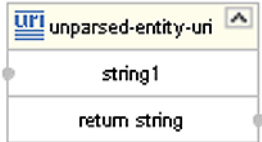
Method Box	Description/Usage
	<p>The string method returns a string representation of the input object.</p>
	<p>The string-length method returns the number of characters in the string.</p>
	<p>Dragging the icon first displays the <i>String Literal</i> dialog box, where you enter the literal value; for example, “Data received”:</p> <div data-bbox="711 802 1284 987" 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 47 String Collaboration Methods (XSLT)

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>

8.4.3 Number Methods

Figure 199 Method Palette: Number Methods

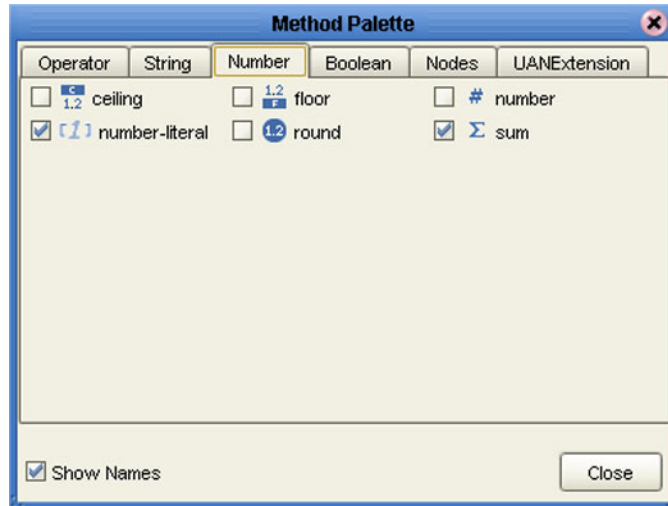
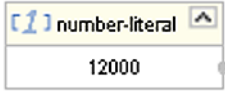
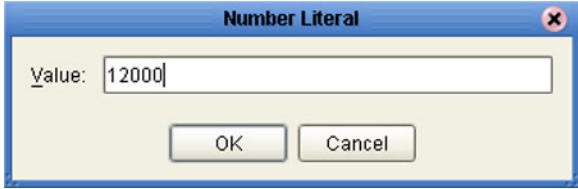

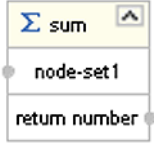


Table 48 Number Collaboration Methods (XSLT)

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 48 Number Collaboration Methods (XSLT)

Method Box	Description/Usage
 <p>The method box for 'number-literal' shows a blue icon with a number '1', the text 'number-literal', and a dropdown arrow. Below this, the value '12000' is entered 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 dialog box titled 'Number Literal' has a text input field containing '12000' and 'Value:' to its left. Below the field are 'OK' and 'Cancel' buttons.</p> <p>The number-literal method then returns a number having the specified value.</p>
 <p>The method box for 'round' shows a blue icon with '1.2', the text 'round', and a dropdown arrow. Below this, the argument 'number1' is shown in a text field, and 'return number' is shown in a separate field.</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 'sum' shows a blue icon with a summation symbol, the text 'sum', and a dropdown arrow. Below this, the argument 'node-set1' is shown in a text field, and 'return number' is shown in a separate field.</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>

8.4.4 Boolean Methods

Figure 200 Method Palette: Boolean Methods

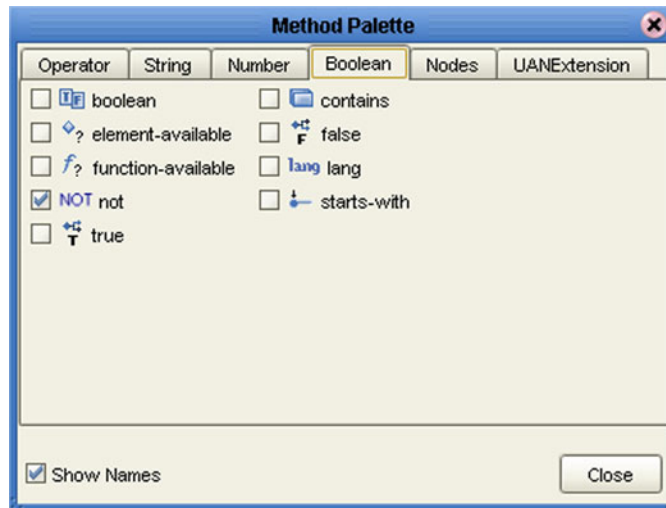
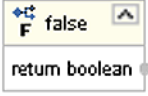
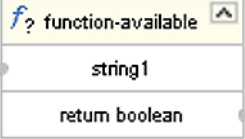
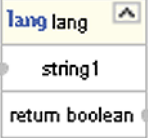
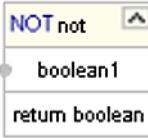
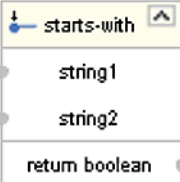
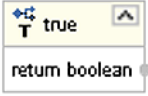


Table 49 Boolean Collaboration Methods (XSLT)

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 49 Boolean Collaboration Methods (XSLT)

Method Box	Description/Usage
 <p>The method box for 'false' shows a yellow header with a blue 'F' icon, the text 'false', and an upward arrow. Below the header is a white box containing the text 'return boolean' with a small grey dot to its right.</p>	<p>The false method returns Boolean false.</p>
 <p>The method box for 'function-available' shows a yellow header with a blue 'f?' icon, the text 'function-available', and an upward arrow. Below the header is a white box containing the text 'string1' and 'return boolean' with a small grey dot to its right.</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' shows a yellow header with a blue 'lang' icon, the text 'lang', and an upward arrow. Below the header is a white box containing the text 'string1' and 'return boolean' with a small grey dot to its right.</p>	<p>The lang (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' shows a yellow header with a blue 'NOT' icon, the text 'not', and an upward arrow. Below the header is a white box containing the text 'boolean1' and 'return boolean' with a small grey dot to its right.</p>	<p>The not method returns the inverse of <i>boolean1</i>.</p>
 <p>The method box for 'starts-with' shows a yellow header with a blue 'starts-with' icon, the text 'starts-with', and an upward arrow. Below the header is a white box containing the text 'string1', 'string2', and 'return boolean' with a small grey dot to its right.</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' shows a yellow header with a blue 'T' icon, the text 'true', and an upward arrow. Below the header is a white box containing the text 'return boolean' with a small grey dot to its right.</p>	<p>The true method returns Boolean true.</p>

8.4.5 Nodes Methods

Figure 201 Method Palette: Nodes Methods

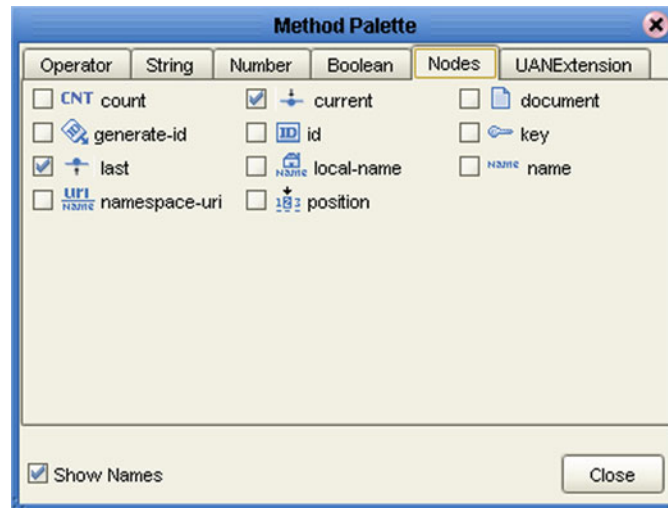


Table 50 Nodes Collaboration Methods (XSLT)

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 50 Nodes Collaboration Methods (XSLT)

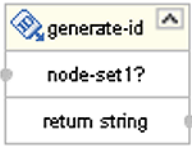
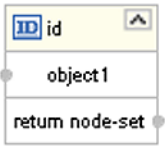
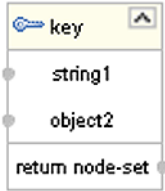
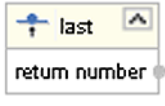
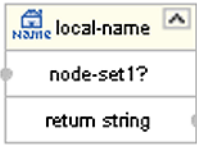
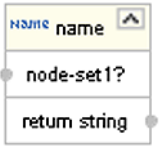
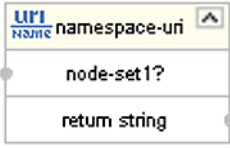
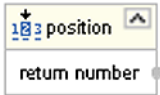
Method Box	Description/Usage
	<p>The generate-id method returns a string that uniquely identifies the node in the argument <i>node-set1?</i> that is first in document order. The unique identifier must consist of ASCII alphanumeric characters and must start with an alphabetic character. Thus, the string 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 is 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 50 Nodes Collaboration Methods (XSLT)

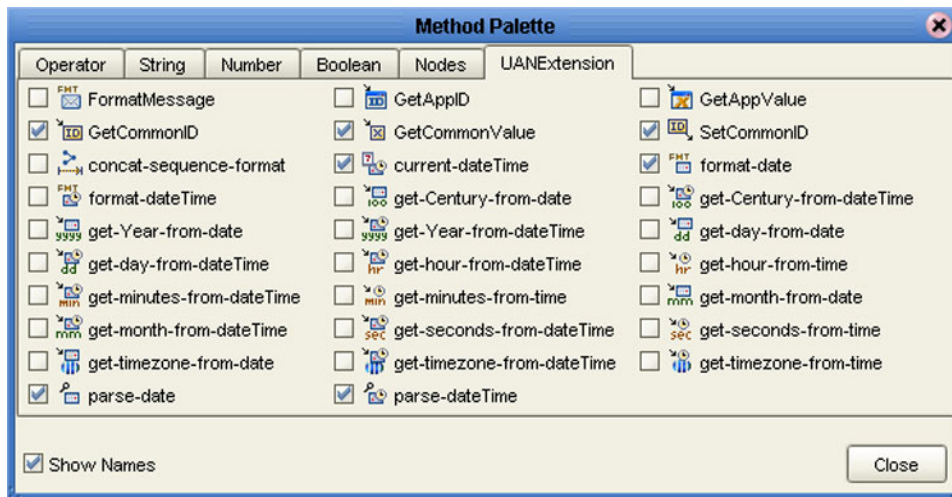
Method Box	Description/Usage
	The position method returns a number equal to the context position from the expression evaluation context.

8.4.6 UAN Extension Methods

These methods are used in conjunction with the Siebel UAN eWay for implementing UAN Extension functions during run time. Employing these methods causes the eWay to interact with the cross-reference (XRef) database to perform the specified request. Definitions of the parameters used in these methods are listed in Table 52.

Note: This Method Palette and the associated methods appear only if you have installed the SeeBeyond UANExtension add-on.

Figure 202 Method Palette: UAN Extension Methods



Note: Since these are run-time functions, their operation cannot be tested in the XSLT Tester.

Table 51 UAN Extension Collaboration Methods (XSLT)



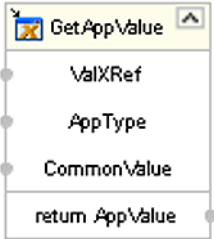

Method Box	Description/Usage
	<p>The formatMessage method function serves as a message packaging service in UAN. It is able to return a formatted message strings based on specifications given by the caller of the function. This function is able to retrieve pre-defined message text in repository, perform argument substitution and argument cross-referencing, and return the packaged message in the specified language.</p> <ul style="list-style-type: none"> ▪ Code is the code of the desired message; it is used to locate the desired message definition in the repository ▪ Lang is the desired language text of the message. ▪ ApplInst is the name of the application instance that is to receive the message. Passing a null value indicates id cross-referencing will not be performed. ▪ AppType is the type of the application that is to receive this message. Passing a null value indicates value cross-referencing will not be performed. ▪ ArgN is the value of the Nth argument. <p>Note: Only Code and Lang are required; all other parameters are optional.</p>
	<p>The getAppID method is used to obtain a specific application ID by giving the ID cross reference name (IDXRef), application instance name (ApplInst) and common ID (commonID). The return value of this method is the ID for the target application instance. If the application ID has not been registered, "" is returned.</p>
	<p>The getAppValue method returns the application value (AppValue) corresponding to the specified common value (CommonValue), given the application type (AppType) and type of object being cross-referenced (ValXRef).</p>
	<p>The getCommonID method is used to obtain the common ID corresponding to a specific application ID by giving the ID cross reference name (IDXRef), application instance name (ApplInst) and application ID (AppID). The return value of this method is the common ID for the target application instance. If the application ID has not been registered, "" is returned.</p>

Table 51 UAN Extension Collaboration Methods (XSLT)

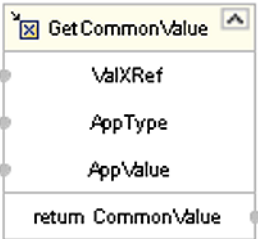

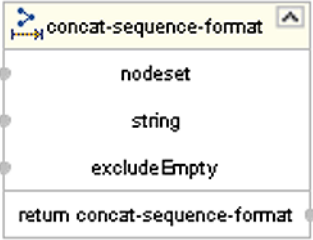
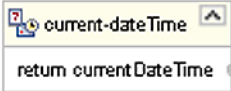
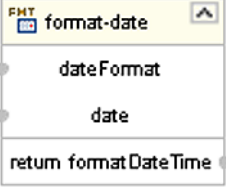
Method Box	Description/Usage
 <p>The screenshot shows a method box titled 'GetCommonValue'. It has three input parameters: ValXRef, AppType, and AppValue. The output is labeled 'return CommonValue'.</p>	<p>The getCommonValue method returns the common value (CommonValue) corresponding to the specified application value (AppValue), given the application type (AppType) and type of object being cross-referenced (ValXRef).</p>
 <p>The screenshot shows a method box titled 'SetCommonID'. It has four input parameters: IDXRef, AppInst, AppID, and CommonID. The output is labeled 'return CommonID'.</p>	<p>The setCommonID method is used to create, query, and establish the relationship between application ID (AppID) and common ID (CommonID).</p> <ul style="list-style-type: none"> ▪ To Create: If an application ID is being registered for the first time, the system will generate a new common ID to store the information. ▪ To Query: If an application ID is already registered, the common ID will be returned. ▪ To Establish: If the CommonID parameter is specified, the relationship between this common ID and the application ID will be established, linking the current application with an application ID. ▪ If the CommonID parameter is not specified, this method should generate a unique ID or return the existing common ID.
 <p>The screenshot shows a method box titled 'concat-sequence-format'. It has three input parameters: nodeset, string, and excludeEmpty. The output is labeled 'return concat-sequence-format'.</p>	<p>The concat-sequence-format method concatenates the string value of each node in order, using a string separator between entries. If the Boolean parameter excludeEmpty is <i>true</i>, empty strings are not included. If excludeEmpty is omitted, <i>false</i> is assumed.</p>
 <p>The screenshot shows a method box titled 'current-dateTime'. It has no input parameters. The output is labeled 'return currentDateTime'.</p>	<p>The currentDateTime method returns the current time as a dateTime string.</p>
 <p>The screenshot shows a method box titled 'format-date'. It has two input parameters: dateFormat and date. The output is labeled 'return formatDateTime'.</p>	<p>The formatDate method takes the given date string and converts it to an ISO 8601 dateTime string as specified by the dateFormat string.</p>

Table 51 UAN Extension Collaboration Methods (XSLT)

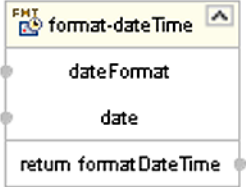






Method Box	Description/Usage
 <p>The method box for <code>format-dateTime</code> shows an icon with 'FMT', an input parameter <code>date</code>, and a return value <code>format DateTime</code>.</p>	<p>The formatDateTime method takes an ISO 8601 dateTime string and converts it to a dateTime string as specified by the dateFormat string.</p>
 <p>The method box for <code>get-Century-from-date</code> shows an icon with '100', an input parameter <code>date</code>, and a return value <code>CenturyFromDate</code>.</p>	<p>The getCenturyFromDate method extracts the century as a number from a string having an XML Schema date datatype, the encoding of which is based on the ISO 8601 reduced format (CCYY-MM-DD) with optional timezone.</p>
 <p>The method box for <code>get-Century-from-dateTime</code> shows an icon with '100', an input parameter <code>dateTime</code>, and a return value <code>CenturyFromDateTime</code>.</p>	<p>The getCenturyFromDateTime method extracts the century as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone.</p>
 <p>The method box for <code>get-Year-from-date</code> shows an icon with 'yyyy', an input parameter <code>date</code>, and a return value <code>YearFromDate</code>.</p>	<p>The getYearFromDate method extracts the year as a number from a string having an XML Schema date datatype, the encoding of which is based on the ISO 8601 reduced format (CCYY-MM-DD) with optional timezone.</p>
 <p>The method box for <code>get-Year-from-dateTime</code> shows an icon with 'yyyy', an input parameter <code>dateTime</code>, and a return value <code>YearFromDateTime</code>.</p>	<p>The getYearFromDateTime method extracts the year as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone.</p>
 <p>The method box for <code>get-month-from-date</code> shows an icon with 'mm', an input parameter <code>date</code>, and a return value <code>monthFromDate</code>.</p>	<p>The getMonthFromDate method extracts the month as a number from a string having an XML Schema date datatype, the encoding of which is based on the ISO 8601 reduced format (CCYY-MM-DD) with optional timezone.</p>
 <p>The method box for <code>get-month-from-dateTime</code> shows an icon with 'mm', an input parameter <code>dateTime</code>, and a return value <code>monthFromDateTime</code>.</p>	<p>The getMonthFromDateTime method extracts the month as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone.</p>
 <p>The method box for <code>get-day-from-date</code> shows an icon with 'dd', an input parameter <code>date</code>, and a return value <code>dayFromDate</code>.</p>	<p>The getDayFromDate method extracts the day as a number from a string having an XML Schema date datatype, the encoding of which is based on the ISO 8601 reduced format (CCYY-MM-DD) with optional timezone.</p>

Table 51 UAN Extension Collaboration Methods (XSLT)

Method Box	Description/Usage
 <p>The method box for <code>get-day-from-dateTime</code> shows an icon with 'dd', the method name, an input field for <code>dateTime</code>, and a return field for <code>return dayFromDateTime</code>.</p>	<p>The getDayFromDateTime method extracts the day as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone.</p>
 <p>The method box for <code>get-hour-from-dateTime</code> shows an icon with 'hr', the method name, an input field for <code>dateTime</code>, and a return field for <code>return hourFromDateTime</code>.</p>	<p>The getHourFromDateTime method extracts the hour as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone. The hour value ranges from 0 to 23, inclusive.</p>
 <p>The method box for <code>get-hour-from-time</code> shows an icon with 'hr', the method name, an input field for <code>date</code>, and a return field for <code>return hourFromDate</code>.</p>	<p>The getHourFromTime method extracts the hour as a number from a string with XML Schema time datatype, which is the left-truncated lexical representation for dateTime (hh:mm:ss.ssss) with optional timezone. The hour value ranges from 0 to 23, inclusive.</p>
 <p>The method box for <code>get-minutes-from-dateTime</code> shows an icon with 'min', the method name, an input field for <code>dateTime</code>, and a return field for <code>return minutesFromDateTime</code>.</p>	<p>The getMinutesFromDateTime method extracts the minutes as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone. The minutes value ranges from 0 to 59, inclusive.</p>
 <p>The method box for <code>get-minutes-from-time</code> shows an icon with 'min', the method name, an input field for <code>date</code>, and a return field for <code>return minutesFromDate</code>.</p>	<p>The getMinutesFromTime method extracts the minutes as a number from a string with XML Schema time datatype, which is the left-truncated lexical representation for dateTime (hh:mm:ss.ssss) with optional timezone. The minutes value ranges from 0 to 59, inclusive.</p>
 <p>The method box for <code>get-seconds-from-dateTime</code> shows an icon with 'sec', the method name, an input field for <code>dateTime</code>, and a return field for <code>return secondsFromDateTime</code>.</p>	<p>The getSecondsFromDateTime method extracts the seconds as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone. The seconds value ranges from 0 to 59, inclusive.</p>
 <p>The method box for <code>get-seconds-from-time</code> shows an icon with 'sec', the method name, an input field for <code>date</code>, and a return field for <code>return secondsFromDate</code>.</p>	<p>The getSecondsFromTime method extracts the seconds as a number from a string with XML Schema time datatype, which is the left-truncated lexical representation for dateTime (hh:mm:ss.ssss) with optional timezone. The seconds value ranges from 0 to 59, inclusive.</p>
 <p>The method box for <code>get-timezone-from-date</code> shows an icon with a globe, the method name, an input field for <code>date</code>, and a return field for <code>return timezoneFromDate</code>.</p>	<p>The getTimezoneFromDate method extracts the timezone as a number from a string having an XML Schema date datatype, the encoding of which is based on the ISO 8601 reduced format (CCYY-MM-DD) with optional timezone.</p>

Table 51 UAN Extension Collaboration Methods (XSLT)


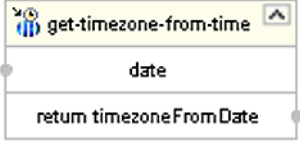
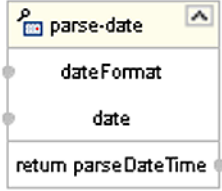
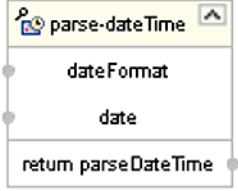
Method Box	Description/Usage
 <p>The screenshot shows a method box titled 'get-timezone-from-dateTime'. It has a single input parameter 'dateTime' and a return value 'return timezoneFromDateTime'.</p>	<p>The getTimezoneFromDateTime method extracts the timezone as a number from a string having an XML Schema dateTime datatype, the encoding of which is based on the ISO 8601 extended format (CCYY-MM-DDThh:mm:ss) with optional timezone.</p>
 <p>The screenshot shows a method box titled 'get-timezone-from-time'. It has a single input parameter 'date' and a return value 'return timezoneFromDate'.</p>	<p>The getTimezoneFromTime method extracts the timezone as a number from a string with XML Schema time datatype, which is the left-truncated lexical representation for dateTime (hh:mm:ss.ssss) with optional timezone.</p>
 <p>The screenshot shows a method box titled 'parse-date'. It has an input parameter 'dateFormat' and a return value 'return parseDateTime'.</p>	<p>The parseDate method parses the given dateValue string into an ISO 8601 date string as specified by the dateFormat string.</p>
 <p>The screenshot shows a method box titled 'parse-dateTime'. It has an input parameter 'dateFormat' and a return value 'return parseDateTime'.</p>	<p>The parseDateTime method parses the given date string into an ISO 8601 dateTime string as specified by the dateFormat string.</p>

Table 52 UAN Extension Collaboration Method Parameter Definitions

Parameter	Description/Usage
AppID	The key, having more than 50 characters, that is used to identify the object in the particular application.
AppInst	A string of no more than 50 characters that uniquely identifies the application instance.
AppType	A string of no more than 50 characters that uniquely identifies the application type.
AppValue	The value, having no more than 50 characters, that is used by the application type.
CommonID	The key, having no more than 50 characters, that is to be used in the neutral, common format.
CommonValue	The value, having no more than 50 characters, that is to be used in the neutral, common format.
IDXRef	A string of no more than 50 characters that identifies the type of object within the system that is being cross-referenced.

Table 52 UAN Extension Collaboration Method Parameter Definitions

Parameter	Description/Usage
ValueXRef	A string of no more than 50 characters that identifies the type of object within the application that is being cross-referenced.

8.5 Version Control

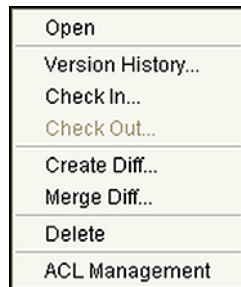
8.5.1 Creating a Modified Collaboration Definition (XSLT)

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

To create a *different* version of a XSLT-based Collaboration Definition file

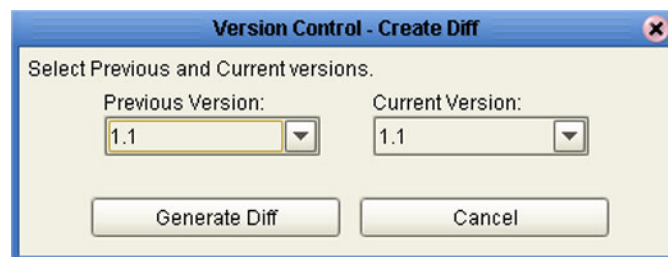
- 1 From Project Explorer, select a **Collaboration Definition (XSLT)** icon.
- 2 Right-click to display the context menu shown in Figure 203.

Figure 203 Collaboration Definition (XSLT) Context Menu



- 3 Select **Create Diff** to display the dialog box shown in Figure 204.

Figure 204 Version Control - Create Diff Dialog Box



- 4 Click **Generate Diff** to display the *Specify Name ...* dialog box.
- 5 Enter a name for the Diff file in the **File Name** box (or use the default file name).
- 6 Click **Save** to save the Diff file.

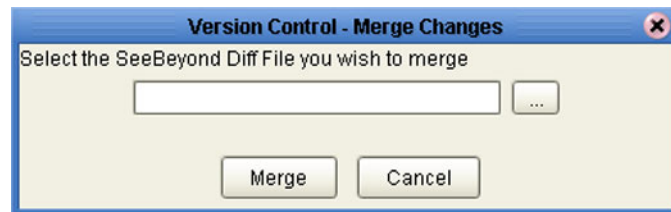
8.5.2 Merging Two Versions of a Collaboration Definition (XSLT)

To merge two versions of an XSLT-based Collaboration Definition

- 1 From the Project Explorer tab, select a **Collaboration Definition (XSLT)** icon.
- 2 Right-click to display the Collaboration Definition (XSLT) context menu.

- 3 Select **Check Out**.
- 4 Right-click to again display the context menu.
- 5 Select **Merge Diff...** to display the dialog box shown in Figure 205.

Figure 205 Version Control - Merge Changes Dialog Box



- 6 Click the **Ellipsis (...)** button to display the *Specify Name ...* dialog box.
- 7 Locate and select the Diff file.
- 8 Click **Open** add the Diff file to the dialog box shown in Figure 205.
- 9 Click **Merge** to merge the Diff file with the corresponding XSLT-based Collaboration Definition in your Project.
- 10 Resolve any conflicts from the merge.
- 11 **Commit** the merged Collaboration.
- 12 **Save** the merged Collaboration.

Environments

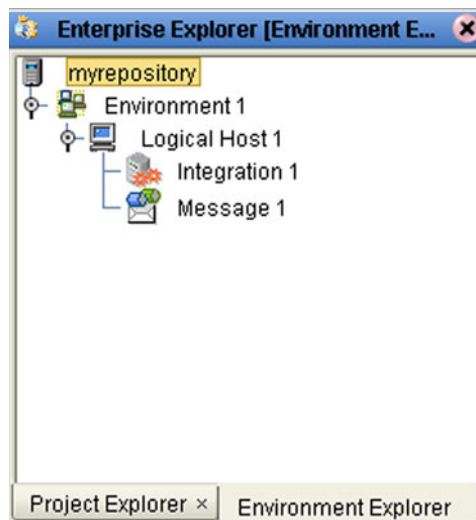
This chapter describes the process of defining eGate Environments, and the various components of an Environment.

9.1 Overview

Projects are run within *Logical Hosts*, which contain the logical resources required by the Project at run time. The Logical Hosts, in turn, are defined within *Environments*, which represent the physical resources required to implement the Project. The Environment also contains information about external systems with which the eGate Project interacts.

9.2 Environment Explorer








Figure 206 Enterprise Explorer: Environment Explorer View



9.2.1 Environment Explorer Icons

The icons described in Table 53 appear in the Environment Explorer.

Table 53 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 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.

9.2.2 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 242
- [Environment Menu](#) on page 243
- [Logical Host Menu](#) on page 244

Repository Menu

Figure 207 Repository Menu



Table 54 Repository Menu Options

Option	Function
New Environment	Displays a dialog box with which you can create a new Environment.
Configure SNMP Agent	Displays a dialog box in which you can modify the SNMP agent properties.
Save Changes to Repository	Saves all changes made in the Environment Explorer.
Refresh All from Repository	Refreshes the Project Explorer and Environment Explorer to display the current contents of the Repository. (Open editors are not refreshed.)

Environment Menu

Figure 208 Environment Menu

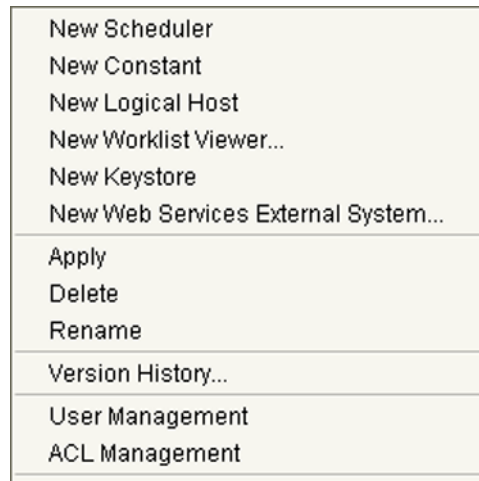


Table 55 Environment Menu Options

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. See Defining Environmental Constants on page 246.
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 Keystore	Adds a new keystore to the selected Environment.
New Web Service ...	Adds a third-party Web service application to the Project Explorer. See SeeBeyond Web Services on page 282.
Apply	Applies 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 Viewing a Component's Version History on page 71 for more information.
User Management	Displays a dialog box with which you can manage message server access. See the <i>eGate Integrator System Administration Guide</i> .

Table 55 Environment Menu Options

Option	Function
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 the <i>eGate Integrator System Administration Guide</i> .

Logical Host Menu

Figure 209 Logical Host Menu

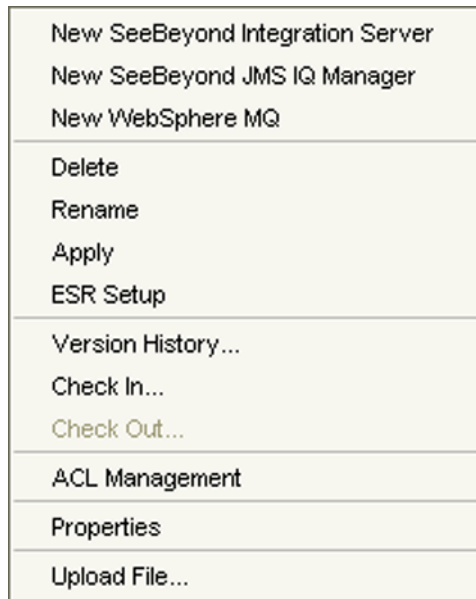


Table 56 Logical Host Menu Options

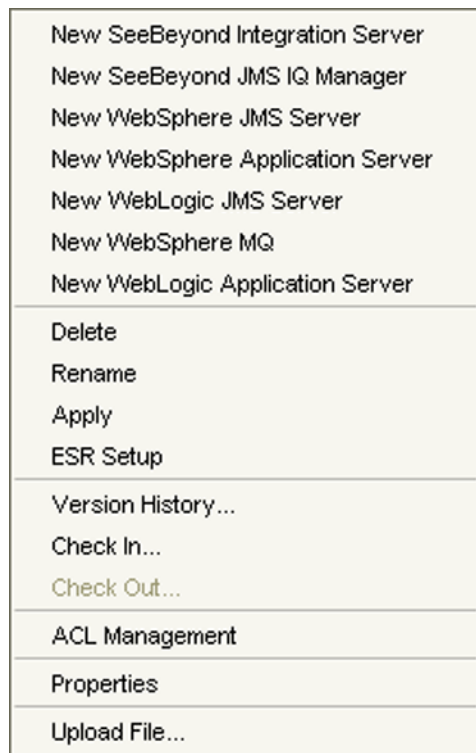
Option	Function
New SeeBeyond Integration Server	Adds a new SeeBeyond Integration Server to the selected Logical Host.
New SeeBeyond JMS IQ Manager	Adds a new SeeBeyond JMS IQ Manager to the selected Logical Host.
New WebSphere MQ	Adds a new IBM WebSphere MQ message server to the selected Logical Host.
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	Applies 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 Logical Host.

Table 56 Logical Host Menu Options

Option	Function
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 Viewing a Component's Version History on page 71 for more information.
Check In	Displays a dialog box, with which you can check in the current version of an Logical Host. Refer to Checking a Component In on page 69 for more details.
Check Out	Displays a dialog box with which you can check out a version of an Logical Host. See Checking a Component Out on page 70 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 Logical Host See the <i>eGate Integrator System Administration Guide</i> .
Properties	Displays a dialog box with which you can modify the default settings for the selected Logical Host.
Upload File	Allows you to upload third-party libraries (.jar files) to the Logical Host.

Note: *If you are using BEA WebLogic and/or IBM WebSphere, the Application Servers and JMS Message Servers for these products will also appear in the context menu (see Figure 210).*

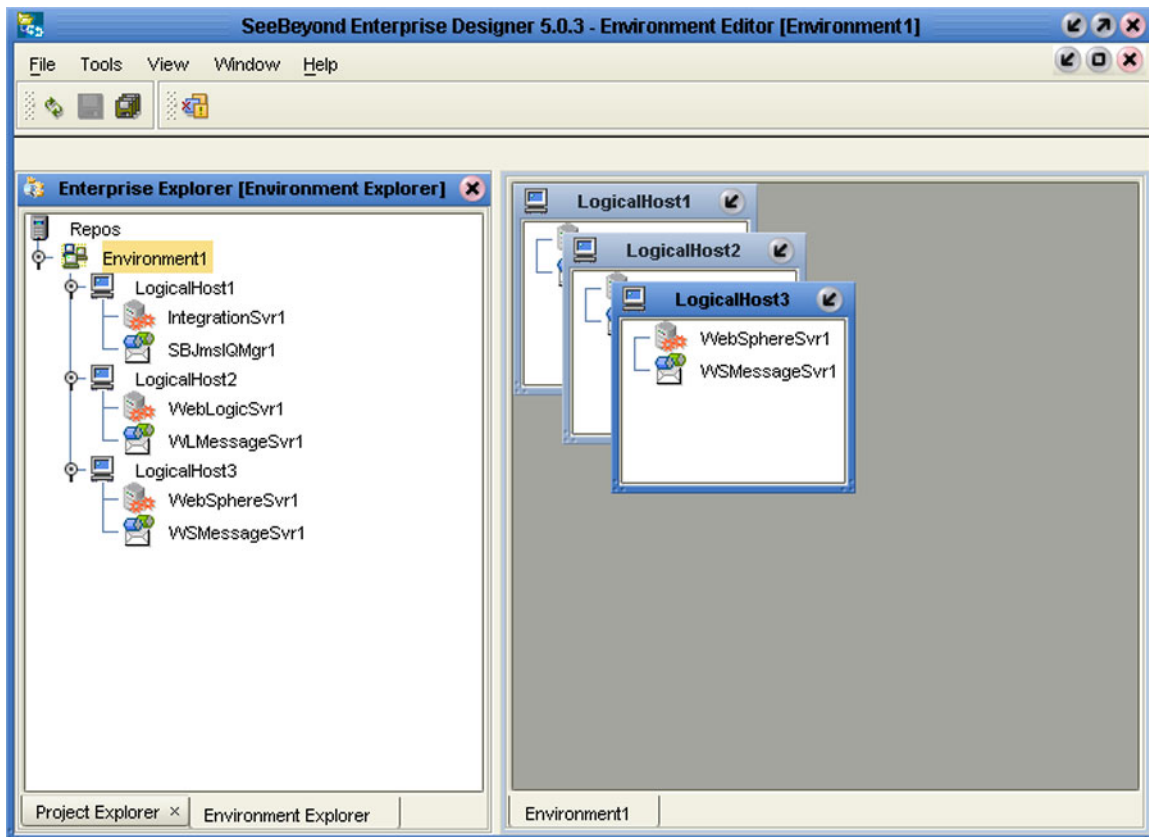
Figure 210 Logical Host Menu with Third-Party Servers



9.3 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 (see Figure 211).

Figure 211 Environment Editor

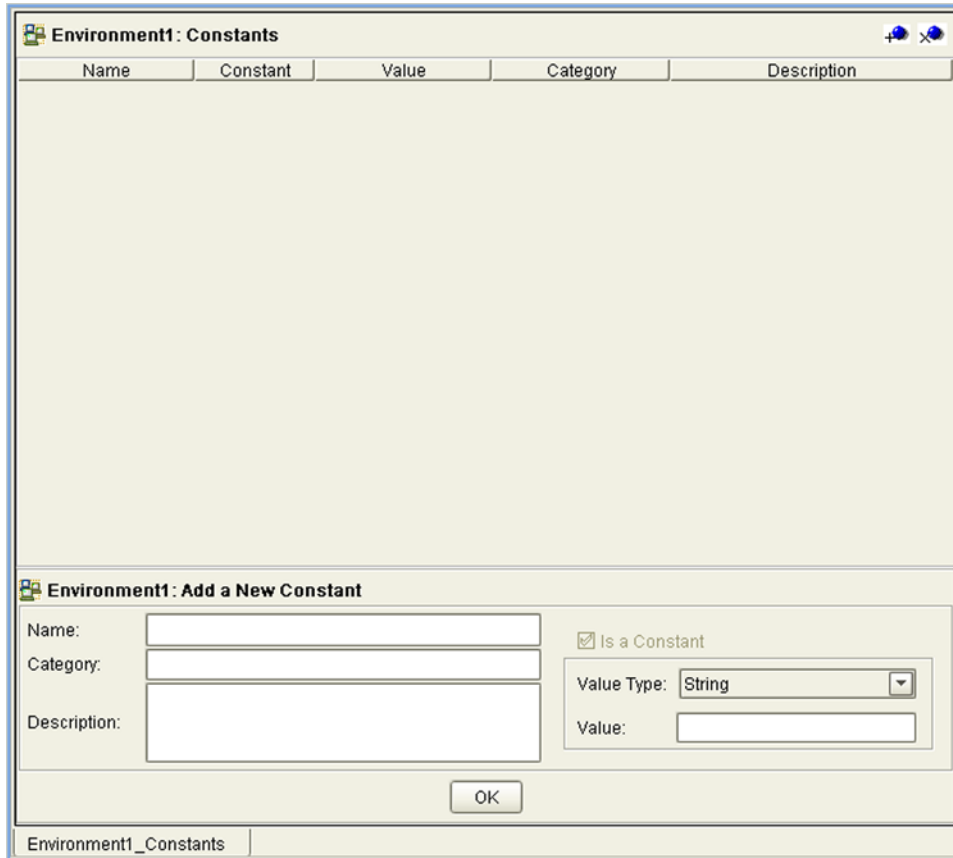


Here you can see the various components (Logical Hosts, servers, and external systems) included in the selected Environment. New Environments are added through the use of the Repository context menu (see [Repository Menu](#) on page 242). Components are added to the Environment by selecting options in the Environment and Logical Host context menus (see [Environment Menu](#) on page 243 and [Logical Host Menu](#) on page 244, respectively).

9.3.1 Defining Environmental Constants

Environmental constants are name/value pairs that are visible across the Environment. Selecting the **New Constant** option from the Environment context menu displays the Constants panel in the Environment Editor (see Figure 212).

Figure 212 Environmental Constants Panel



All constants defined for the specific Environment are listed in the *Constants* section of the panel, along with their various properties. New constants are added using the *Add a New Constant* section of the panel.

Note: *When you create an Environmental constant, you assign a permanent value to it—which cannot be overridden.*

Table 57 Environmental Constants Panel Icons

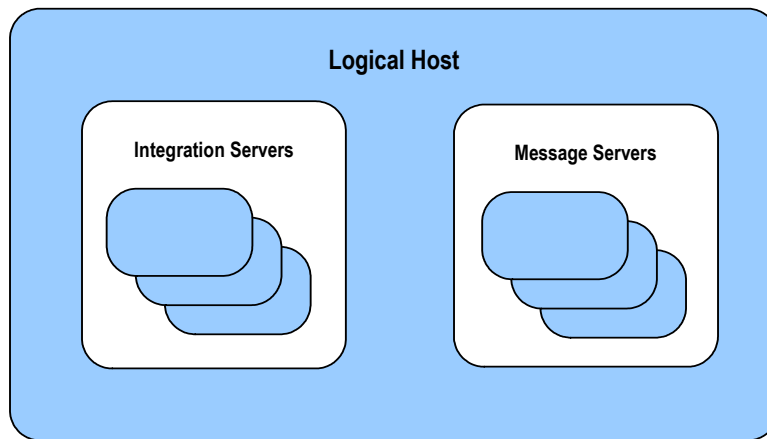
Icon	Name	Function
	Add a New Constant	Adds a new constant to the list.
	Delete a Highlighted Constant	Deletes the selected constant from the list.

9.4 Logical Hosts

9.4.1 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. The Logical Host contains both integration servers and message servers, as illustrated in Figure 213.

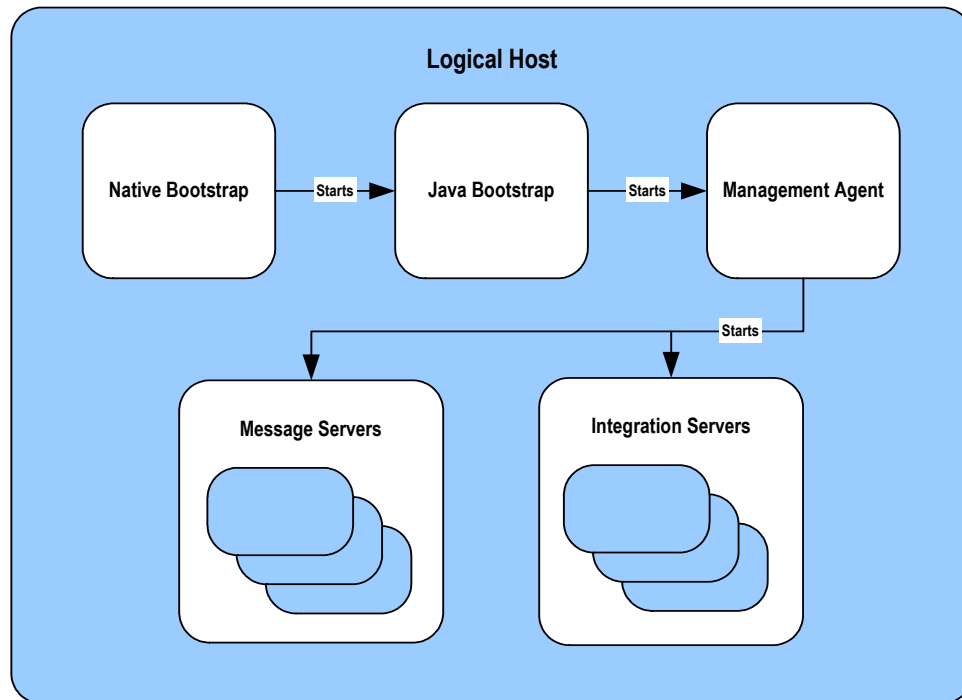
Figure 213 Logical Hosts



The master service of the Logical host is the Management Agent. This 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 message servers and integration servers.

At run time, a platform-specific bootstrap script starts the Java bootstrap program that downloads the Management Agent, message server, and integration server from the Repository. The Management Agent is then started, which in turn starts the message server(s) and integration server(s). Figure 214 illustrates this sequence.

Figure 214 Startup Sequence



Each Logical Host has a separate bootstrap process. The process is started from a batch file (*logical-host-root\bootstrap\bin\bootstrap.bat*) or script (*logical-host-root/bootstrap/bin/bootstrap.sh*). This file or script finds the Repository via command-line parameters or from the configuration file (*logical-host-root\bootstrap\config\logical-host.properties*). See the *eGate Integrator System Administration Guide* for additional information.

9.4.2 Configuring a Logical Host

To access the configuration properties for a Logical Host

- 1 Right-click a Logical Host in the Environment Explorer tree to display the context menu for that Logical Host instance.
- 2 Select **Properties** from the context menu to display the **Properties** dialog box.
- 3 Select the **Logical Host Configuration** node in the properties tree to display the Logical Host Configuration Section, which contains the top-level configuration properties for the Logical Host (see Figure 215).

Figure 215 Logical Host Configuration Properties

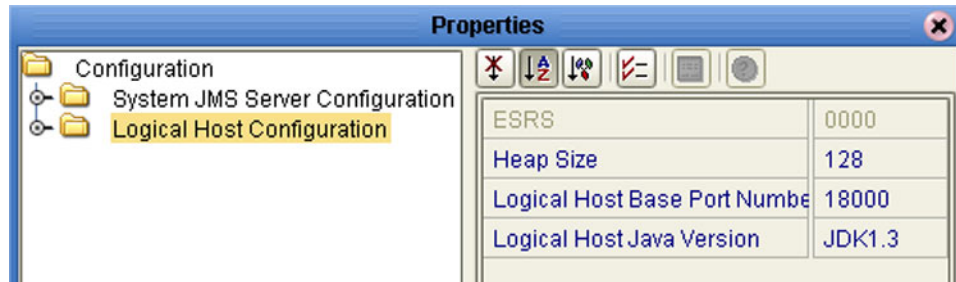


Table 58 Logical Host Configuration Properties List

Property	Description
ESRS	Shows a list of all installed Emergency Software Releases (ESRs).
Heap Size	Specifies the Heap size in Megabytes; the minimum size is 128 Mb, which is the default value. Note that this property is only for the bootstrap and management processes, and does not affect the integration server or any runtime components that are processing data.
Logical Host Base Port Number	Specifies the base port number for the Logical Host. The default value is 18000 . When multiple Logical Hosts reside on a single hardware platform, you must configure the base port numbers; see the following section.
Logical Host Java Version	Specifies the Java version being used to the eWay RAR file generation program, so that any generated file will be properly compatible. The default value is JDK1.3.

Configuring the Base Port Number

If multiple Logical Hosts concurrently run on the same computer, 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, the automatic numbering process will skip the component port number you have assigned manually (*be sure this port number is not used elsewhere*).

Note: While Windows will accept port numbers below 12000, UNIX cannot.

9.5 Integration Servers

The Logical Host contains one or more instances of a J2EE integration server, which is the engine that runs eGate Collaborations for processing business logic, and eWays that communicate with external applications. The integration server provides services for security, transactions, business rules execution, and connectivity management. eGate Integrator contains the SeeBeyond Integration Server, and also supports the use of third-party application servers such those supplied by BEA WebLogic and IBM WebSphere for this purpose (see [Deploying Projects to Third-Party Servers](#) on page 275).

9.5.1 Configuring an Integration Server

To access the configuration properties for an integration server

- 1 Right-click an integration server in the Environment Explorer tree to display the context menu for that instance.
- 2 Select **Properties** from the context menu to display the **Properties** dialog box.
- 3 Select the **IS Configuration** node in the properties tree to display the top-level IS configuration properties (see Figure 216).

Figure 216 Top-level IS Configuration Properties



Table 59 Top-level IS Configuration Properties List

Property	Description
Debug Port	This property is used only when Debug is enabled. The default depends upon the value of the Logical Host base port.
Debug Turned On	Enables/disables debugging for the IS. The default is False (disabled).
Environment Variables	Specifies user-defined Environment Variables. Each element has the format name=value. When present, these values override the system settings, so that <i>all</i> required variables must be set. There is no default.

Table 59 Top-level IS Configuration Properties List

Property	Description
JVM Args	Java Virtual Machine (JVM) arguments. Each element in the collection should specify one, and only one, argument.
Profiling Turned On	Enables/disables performance monitoring for the IS. The default is False (disabled). To enable, change to True (as shown) and configure the properties described in Performance Monitoring (Profiling) on page 255.
Suspend at Startup	Allows the VM to begin executing before the debugger application attaches. The default is n (do not suspend).

The IS Configuration node contains several sections, each containing detailed configuration properties for a particular IS component (including the integration server itself). These components are:

- Web Container
- IS Profiling Configuration
- Security Configuration Template
- eInsight Engine Configuration (if eInsight is installed)
- Application Manager Configuration Template
- Integration Server Configuration

You can access these properties by selecting **Properties** from the context menus for the appropriate nodes.

Web Container

Properties included here are used for setting up Web services.

Figure 217 Web Container Configuration Properties



Table 60 Web Container Configuration Properties List

Property	Description
Web Server Host Name	Specifies the host name; the default is localhost .

Default Web Server

The default Web server properties are contained in a subdirectory under the Web Container directory.

Figure 218 Default Web Server Properties

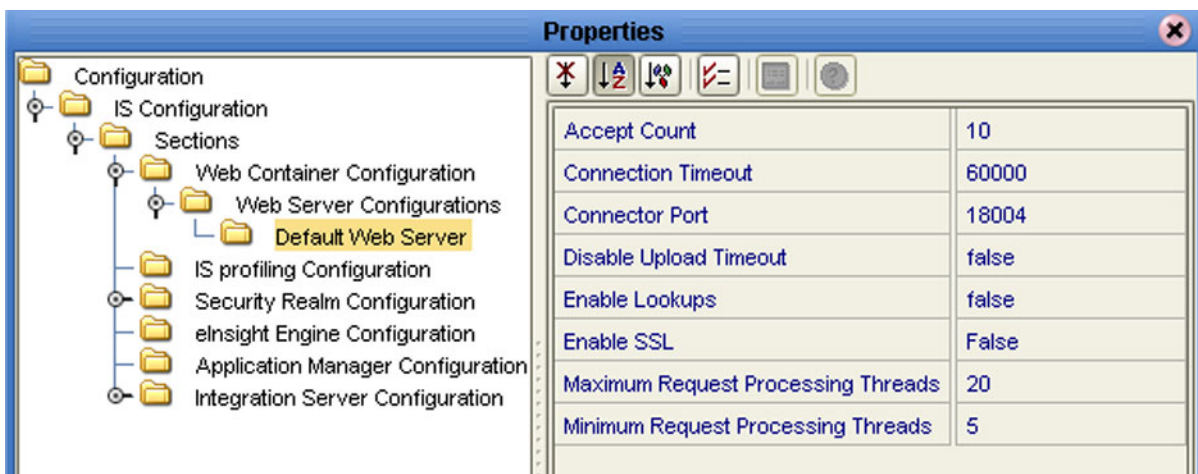


Table 61 Default Web Server Properties List

Property	Description
Accept Count	Specifies the maximum acceptable number of incoming connection requests when all possible request processing threads are in use. Any requests received beyond this number when the queue is full are refused. The default value is 10 .
Connection Timeout	Specifies the time period in milliseconds that this connector will wait for the request URI line to be presented, after accepting the connection. The default value is 80000 ms.
Connector Port	Specifies the connection port for the Web server. The default value is 18004 .
Disable Upload Timeout	Allows the servlet container to use a different, and longer, connection timeout while a servlet is executing. This gives the servlet a longer time to complete execution, and/or provides a longer timeout during data upload. The default value is false .
Enable Lookups	If set to true , calls are made requesting getRemoteHost() to perform DNS lookups in order to return the actual host name of the remote client. If set to false , the DNS lookup is bypassed and the IP address is returned in string form, thereby improving performance. The default value is false .
Enable SSL	Specifies whether or not to enable the Secure Sockets Layer (SSL) protocol. The default value is false .
Maximum Request Processing Threads	Specifies the maximum number of request processing threads to be created by this connector, thereby determining the maximum number of simultaneous requests that can be handled. The default value is 20 .
Minimum Request Processing Threads	Specifies the number of request processing threads to be created by this connector when it is first started. This value must be less than the value set for the Maximum Request Processing Threads property. The default value is 5 .

Performance Monitoring (Profiling)

You can monitor the performance of the integration server by using the built-in *Heap Analysis* tool, which is enabled and configured using the Profiling Configuration dialog box (see Figure 219).

Figure 219 Profiling Configuration Properties

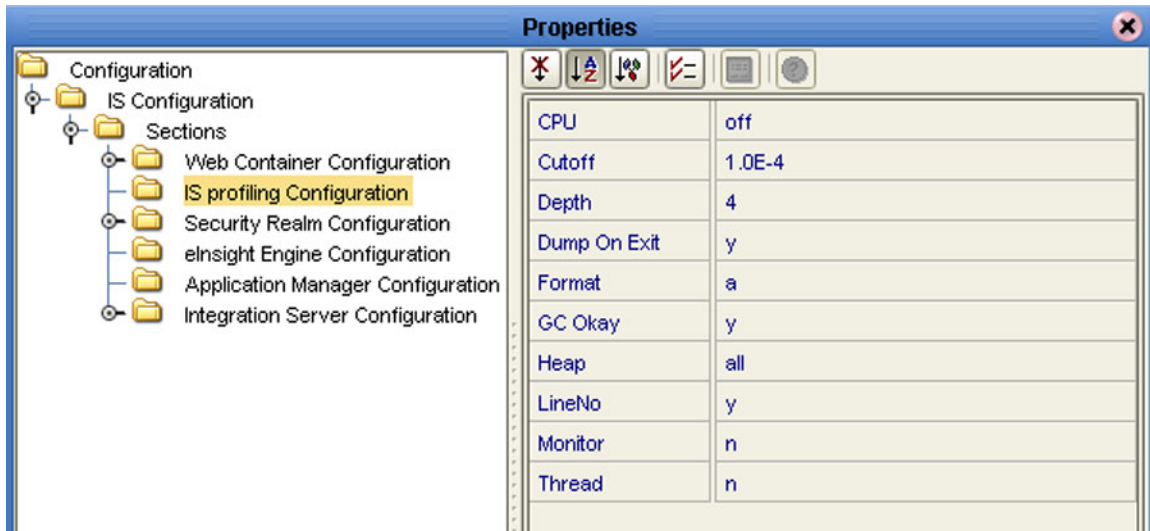


Table 62 Profiling Configuration Properties List

Property	Description
CPU	Specifies whether or not CPU usage is included in the trace. The default value is off .
Cutoff	Specifies the output cutoff point. The default value is 1.0E-4 .
Depth	Specifies the stack trace depth. The default value is 4 .
Dump on Exit	Specifies whether or not to dump on exit.
Format	Specifies ASCII (a) or binary (b) output. The default value is a (ASCII).
GC Okay	Specified whether or not to allow garbage collection (GC) during sampling. The default value is y (yes).
Heap	Specifies the blocks of memory to include in traces. The default value is all .
LineNo	Specifies whether or not to include line numbers in traces. The default value is y (yes).
Monitor	Specifies whether or not to include monitor contention. The default value is n (no).
Thread	Specifies whether or not to include the thread in traces. The default value is n (no).

Security Realm

These properties pertain to the Lightweight Directory Access Protocol (LDAP), if used. Subdirectories contain properties for SunONE Directory Server and Microsoft Active Directory Server. See the *eGate Integrator System Administration Guide* for information regarding Security Realm configuration.

Figure 220 Security Configuration Properties

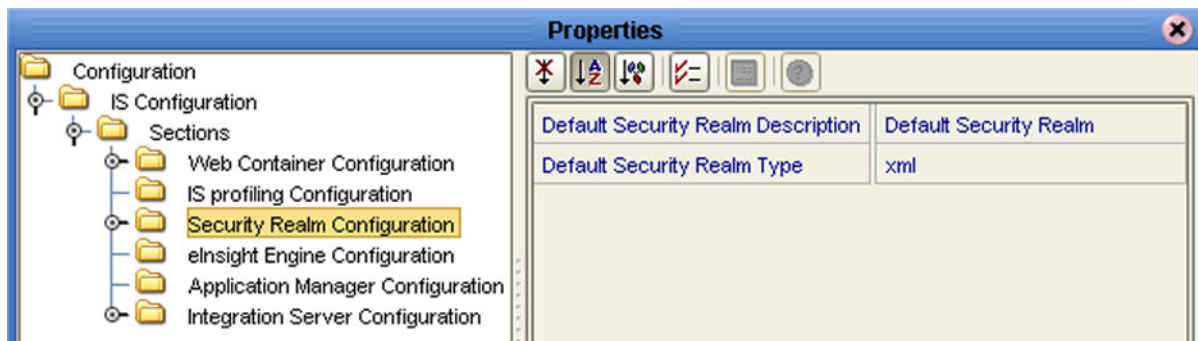


Table 63 Security Realm Configuration Properties List

Property	Description
Default Security Realm Description	Specifies the default LDAP Security Realm description. The default value is Default Security Realm .
Default Security Realm Type	Specifies the default LDAP Security Realm type. The default value is xml .

eInsight Engine

This configuration node is displayed only if you have eInsight Business Process Manager installed on your system. The configuration properties relate to the BPEL engine's database cache; see the *eInsight Business Process Manager User's Guide* for information regarding these properties (see Figure 221).

Figure 221 eInsight Engine Configuration Properties

The screenshot shows a 'Properties' dialog box with a tree view on the left and a table of properties on the right. The tree view shows a hierarchy: Configuration > IS Configuration > Sections > eInsight Engine Configuration. The table lists various configuration parameters and their values.

Cache Pruning Algorithm	Random
Cache Size (Instances)	5000
Database	SQL Server 2000
Database Host	<host>
Database Port	1521
Database User Name	<user>
Debug	true
Debug Port	4865
Enable Monitoring	false
Monitoring Thread Buffer Size	2
Monitoring Thread Buffer Time Lag (seconds)	30
Monitoring Thread Sleep Time (milliseconds)	5000
Password	
Persistence Mode	Memory
Recover During Startup	false
Reporting Thread Sleep Time (milliseconds)	180000
SID	<sid>

Description (eInsightConfig.xml)
Database/ cache configuration for BPEL engine

Application Manager

You can set integration server thread pool variables using the Application Manager Configuration Properties dialog box (see Figure 222).

Figure 222 Application Manager Configuration Properties

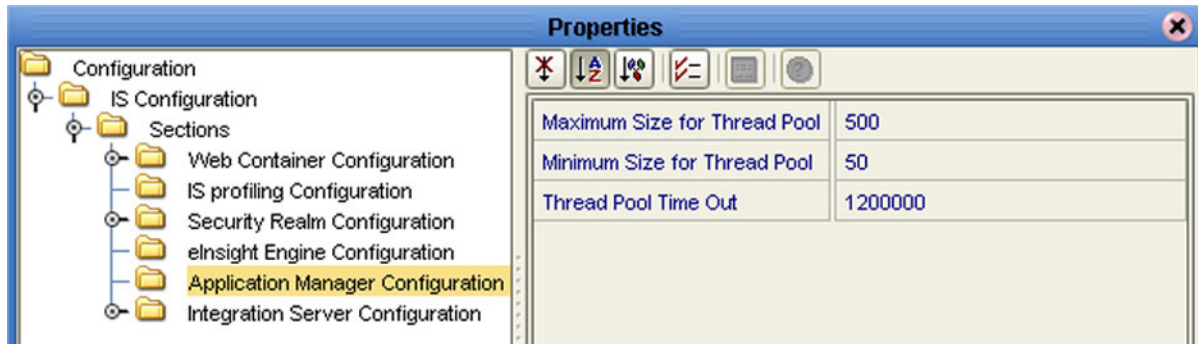


Table 64 Application Manager Configuration Properties List

Property	Description
Maximum Size for Thread Pool	Specifies the maximum size for the thread pool. The default value is 500 .
Minimum Size for Thread Pool	Specifies the minimum size for the thread pool. The default value is 1 .
Thread Pool Time Out	Specifies the timeout interval for the thread pool, measured in milliseconds. The default value is 60000 .

Integration Server

Detailed, low-level configuration of the integration server is performed using the Integration Server Configuration Properties dialog box (see Figure 223).

Figure 223 Integration Server Configuration Properties

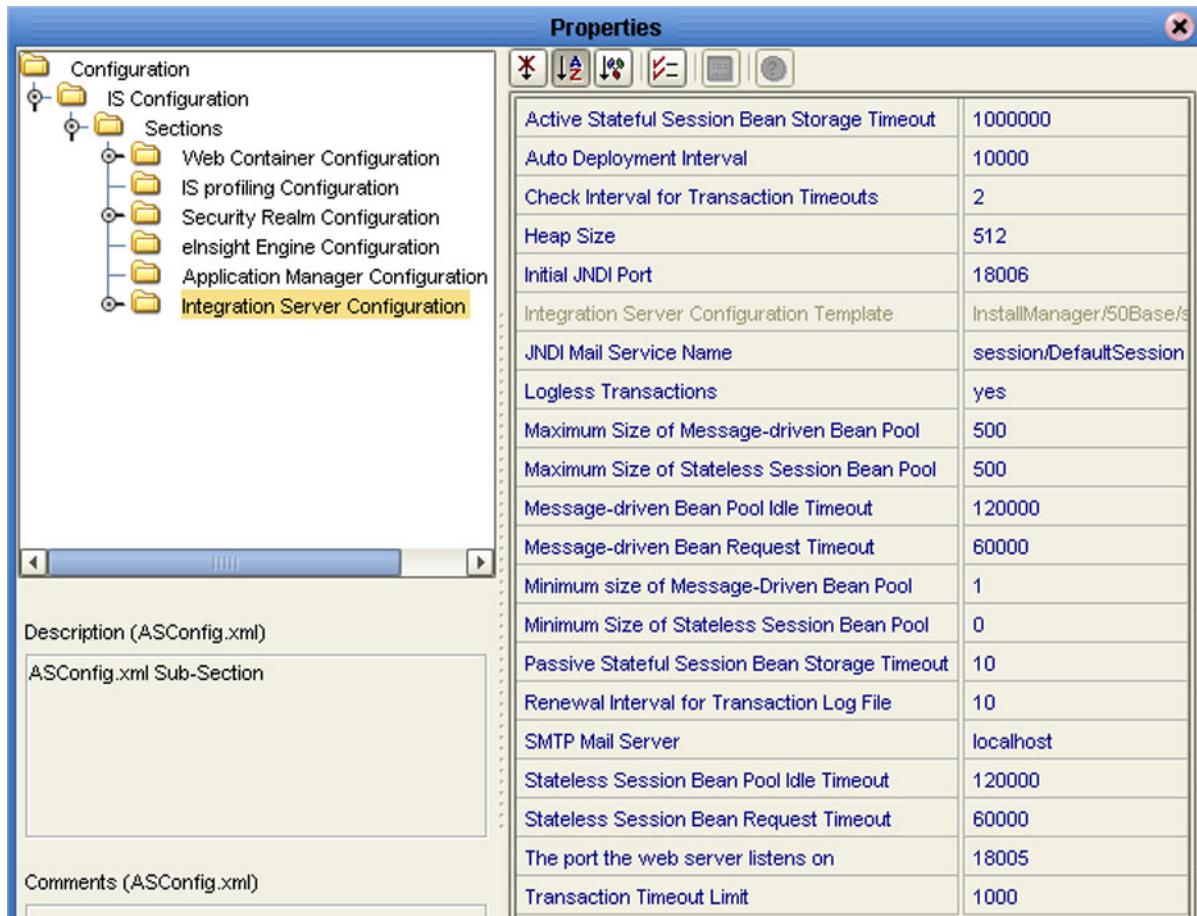


Table 65 Integration Server Configuration Properties List

Property	Description
Active Stateful Session Bean Storage Timeout	Specifies the interval after which an Active Stateful Session Bean is removed from storage, measured in minutes. The default value is 1000000 min, which ensures that it will not be removed unintentionally.
Auto Deployment Interval	Specifies the interval at which the auto-deployer checks the deployment directory for files, measured in milliseconds. The default value is 10000 ms.
Check Interval for Transaction Timeouts	Specifies the interval between checks for transaction timeouts, measured in minutes. The default value is 2 min.

Table 65 Integration Server Configuration Properties List

Property	Description
Heap Size	Specifies the Heap size in Megabytes; the minimum size is 512 Mb , which is the default value. Increasing this value increases the JVM size.
Initial JNDI Port	Specifies the initial port required by the Naming Service class for startup. The default value depends upon the value of the Logical Host base port.
JNDI Mail Service Name	Specifies the name of the JNDI mail service. The default value is session/DefaultSession .
Logless Transactions	Specifies whether or not logless transactions are allowed. The default value is yes .
Maximum Size of Message-driven Bean Pool	Specifies the maximum number of Message-driven Beans allowed in the Message-driven Bean pool at one time. The default value is 500 .
Maximum Size of Stateless Session Bean Pool	Specifies the maximum number of Stateless Session Beans allowed in the Stateless Session Bean pool at one time. The default value is 500 .
Message-driven Bean Pool Idle Timeout	Specifies the timeout interval for the Message-driven Bean pool, measured in milliseconds. The default value is 120000 ms .
Message-driven Bean Request Timeout	Specifies the interval after which a Message-driven Bean request times out, measured in milliseconds. The default value is 60000 ms .
Minimum Size of Message-driven Bean Pool	Specifies the minimum number of Message-driven Beans allowed in the Message-driven Bean pool at one time. The default value is 1 .
Minimum Size of Stateless Session Bean Pool	Specifies the maximum number of Stateless Session Beans allowed in the Stateless Session Bean pool at one time. The default value is 1 .
Passive Stateful Session Bean Storage Timeout	Specifies the interval after which a Passive Stateful Session Bean is removed from storage, measured in minutes. The default value is 10 min .
Renewal Interval for Transaction Log File	Specifies the interval for renewing the Transaction Service log file, measured in hours. The default value is 10 hr .
SMTP Mail Server	Specifies the name of the SMTP mail host server. The default value is localhost .
Stateless Session Bean Pool Idle Timeout	Specifies the timeout interval for the Stateless Bean pool, measured in milliseconds. The default value is 120000 ms .
Stateless Session Bean Pool Request Timeout	Specifies the interval after which a Stateless Bean request times out, measured in milliseconds. The default value is 60000 ms .
The Port the Web Server Listens On	Specifies the port the Web server listens on. The default value depends upon the value of the Logical Host base port.

Table 65 Integration Server Configuration Properties List

Property	Description
Transaction Timeout Limit	Specifies the time limit for transactions to time out, measured in seconds. The default value is 1000 sec.

Oracle JDBC Connection Pool

Connection Pool properties for an Oracle database associated with the integration server are specified in the Oracle JDBC Connection Pool dialog box (see Figure 224).

Figure 224 Oracle JDBC Connection Pool Properties

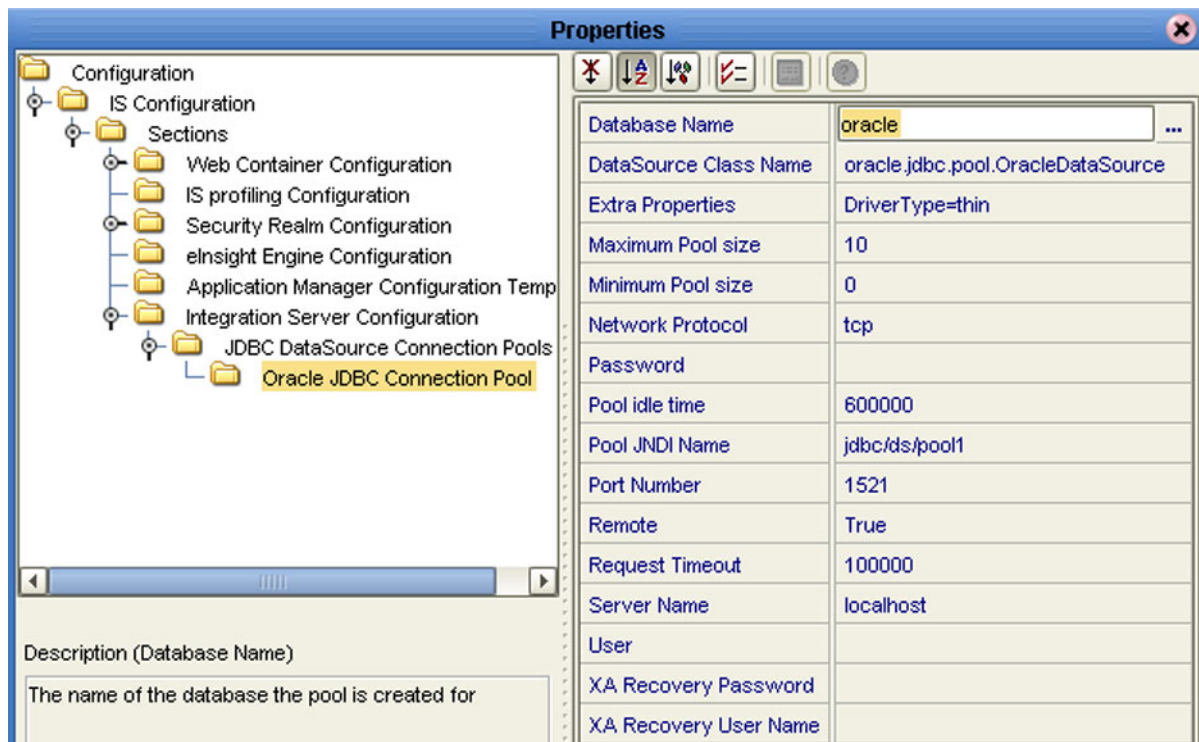


Table 66 Oracle JDBC Connection Pool Properties List

Property	Description
Database Name	Specifies the name of the database for which the pool is created. The default value is oracle .
DataSource Class Name	Specifies the name of the DataSource class. The default value is oracle jdbc pool OracleDataSource .
Extra Properties	Specifies custom properties for the DataSource, using semicolon-separated key-value pairs. The default value is DriverType=thin .
Maximum Pool Size	Specifies the maximum number of connections in the pool. The default value is 10 .

Table 66 Oracle JDBC Connection Pool Properties List

Property	Description
Minimum Pool Size	Specifies the minimum number of connections in the pool. The default value is 0 .
Network Protocol	Specifies the network protocol. The default value is tcp .
Password	Specifies the password for the connection. There is no default value.
Pool Idle Time	Specifies the maximum time period in milliseconds that a connection may remain unused before it is removed from the pool in order to reduce the pool size. The default value is 600000 ms.
Pool JINI Name	Specifies the unique JINI name of the DataSource pool. The pool is bound in the <code>java/namespac</code> for local access or into the global namespace for remote access. The default value is jdbc/ds/pool1 .
Port Number	Specifies the port number on which the server receives data. The default value is 1521 .
Remote	Specifies whether or not the DataSource should be bound into the global remote JINI namespace for access by remote clients. The default value is true .
Request Timeout	Specifies the maximum time period in milliseconds that a request for connection from the pool may block all other connections currently in use. The default value is 100000 ms.
Server Name	Specifies the host name of the database server or IP address where the database server is running. The default value is localhost .
User	Specifies the user name authorized for creating connections. There is no default value.
XA Recovery Password	For XA DataSources only, specifies the password to use for XA transaction recovery. There is no default value.
XA Recovery User Name	For XA DataSources only, specifies the user name to use for XA transaction recovery. There is no default value.

9.5.2 Deploying User-Defined Stateless Session Beans

User-defined stateless session beans can be deployed to the eGate Integration Server following the procedure outlined in this section.

Note: The deployment of stateful session beans, entity beans, and message-driven beans is not currently supported.

To deploy a stand-alone SLSB to the eGate Integration Server

- 1 Create and compile the EJB.
- 2 Write the `ejb-jar.xml` and `seebeyond-ejb.xml` deployment descriptors for your EJB.
- 3 Create a `.jar` file with the deployment descriptors in the `\META-INF` directory and the code in the root.
- 4 Move the `.jar` file into the `\logicalhost\stcis\deploy\new\integration_server_name` directory for deployment. The Integration Server will automatically pick up the `.jar` file from this location and deploy the EJB.

Examples of the EJBs and associated `.xml` files are as follows.

Example Remote Interface

```
package ejb.CustomApp;
import java.rmi.RemoteException;
import java.rmi.Remote;
import javax.ejb.*;

public interface CustomApp extends EJBObject, Remote
{
    public String getId() throws RemoteException;
}
```

Example Home Interface

```
package ejb.CustomApp;

import javax.ejb.*;
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.util.*;

public interface CustomAppHome extends EJBHome
{
    public CustomApp create() throws CreateException, RemoteException;
}
```

Example Stateless Session Bean (SLSB)

```
package ejb.CustomApp;

import javax.ejb.*;
import java.io.Serializable;
import java.util.*;
import java.rmi.*;
import javax.naming.Context;
import javax.naming.InitialContext;

// import additional classes as needed "CustomController"
```

```

public class CustomAppBean implements SessionBean
{
    private SessionContext ctx;
    private CustomController mCustom;

    public void setSessionContext( SessionContext context )
    {
        this.ctx = context;
    }

    public void ejbCreate()
    {
        try {
            javax.naming.Context context = new InitialContext();
            // lookup Custom application
            Object ref = context.lookup("ejb/CustomController");
            CustomControllerHome CustomHome =
            (CustomControllerHome)javax.rmi.PortableRemoteObject.narrow(ref,
            CustomControllerHome.class);
            mCustom = CustomHome.create();
        } catch (Exception e) {
            System.out.println( e.getMessage() );
        }
    }

    public String getId()
    {
        SystemObjectPK key = new SystemObjectPK( "SBYN", "0000000001" );
        String EUID= "Not Found";
        try {
            EUID = mCustom.getEUID( key );
        }
        catch (Exception e) {
            System.out.println("===> Exception: " );
            System.out.println( e.getMessage() );
        }

        return( EUID );
    }

    // add additional EJB methods
}

```

Example ejb-jar.xml file for the above SLSB

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE ejb-jar PUBLIC "-//Sun Microsystems, Inc.//DTD Enterprise
JavaBeans 2.0//EN" 'http://java.sun.com/dtd/ejb-jar_2_0.dtd'>
<!-- Generated XML! -->
<ejb-jar>
    <display-name>ServiceBeans</display-name>
    <enterprise-beans>
        <session>
            <description><![CDATA[Custom App Session Bean]]></
description>
            <display-name>Custom App</display-name>
            <ejb-name>CustomApp</ejb-name>
            <home>ejb.CustomApp.CustomAppHome</home>
            <remote>ejb.CustomApp.CustomApp</remote>
            <ejb-class>ejb.CustomApp.CustomAppBean</ejb-class>
            <session-type>Stateless</session-type>
            <transaction-type>Bean</transaction-type>
            <ejb-ref>
                <ejb-ref-name>ejb/CustomController</ejb-ref-name>

```



```
        <ejb-ref-type>Session</ejb-ref-type>
<home>com.stc.eindex.ejb.Custom.CustomControllerHome</home>
        <remote>com.stc.eindex.ejb.Custom.CustomController</
remote>
        <ejb-link>CustomController</ejb-link>
    </ejb-ref>
</session>
</enterprise-beans>
</ejb-jar>
```

Example seebeyond-*ejb.xml* file for the above SLSB

```
<sbyn-ejb-deployment-descriptor>
  <enterprise-beans>
    <session>
      <ejb-name>CustomApp</ejb-name>
      <jndi-name>ejb/CustomApp</jndi-name>
      <security>
        <authorize>no</authorize>
        <authenticate>no</authenticate>
        <security-audit>no</security-audit>
      </security>
      <pool-min>1</pool-min>
    </session>
  </enterprise-beans>
</sbyn-ejb-deployment-descriptor>
```

SLSB Deployment verification

Examine the log file `\logicalhost\logs\stc_is_`*integration_server_name*`.log`. You should find text such as “**CustomApp (EJB) was successfully deployed**” confirming deployment.

To remove a stand-alone SLSB from the eGate Integration Server

- 1 Shut down the Logical Host containing the Integration Server where the SLSB is deployed.
- 2 Remove the `.jar` file created in the preceding deployment procedure from the `\logicalhost\stcis\repository\applications\`*integration_server_name*`\EAR` directory.
- 3 Restart the Logical Host.

9.6 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.

Third-party application 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*.

Project Deployment

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

10.1 Deployment Profiles

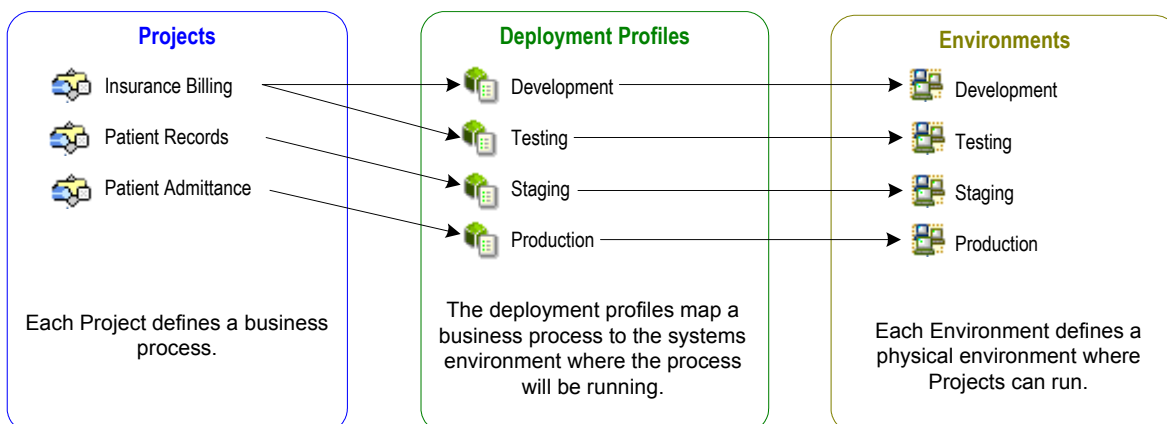
Deployment Profiles define specific instances of a Project in a particular Environment. A deployment profile contains information about the assignment of Services and Message Destinations to integration and message servers (JMS IQ Managers). It also contains version information for all relevant objects 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 2 from the [System Overview](#) on page 24:

Figure 225 eGate Integrator Implementation Model



10.2 The Deployment Editor

The Deployment Editor (see Figure 226) allows you to create a new Deployment Profile or edit an existing one. To create a new Deployment Profile, right-click on a Project in the Project Explorer to display its context menu. From the menu, select **New > Deployment Profile**. To edit an existing Deployment Profile, simply click on its icon.

Figure 226 Deployment Editor Window

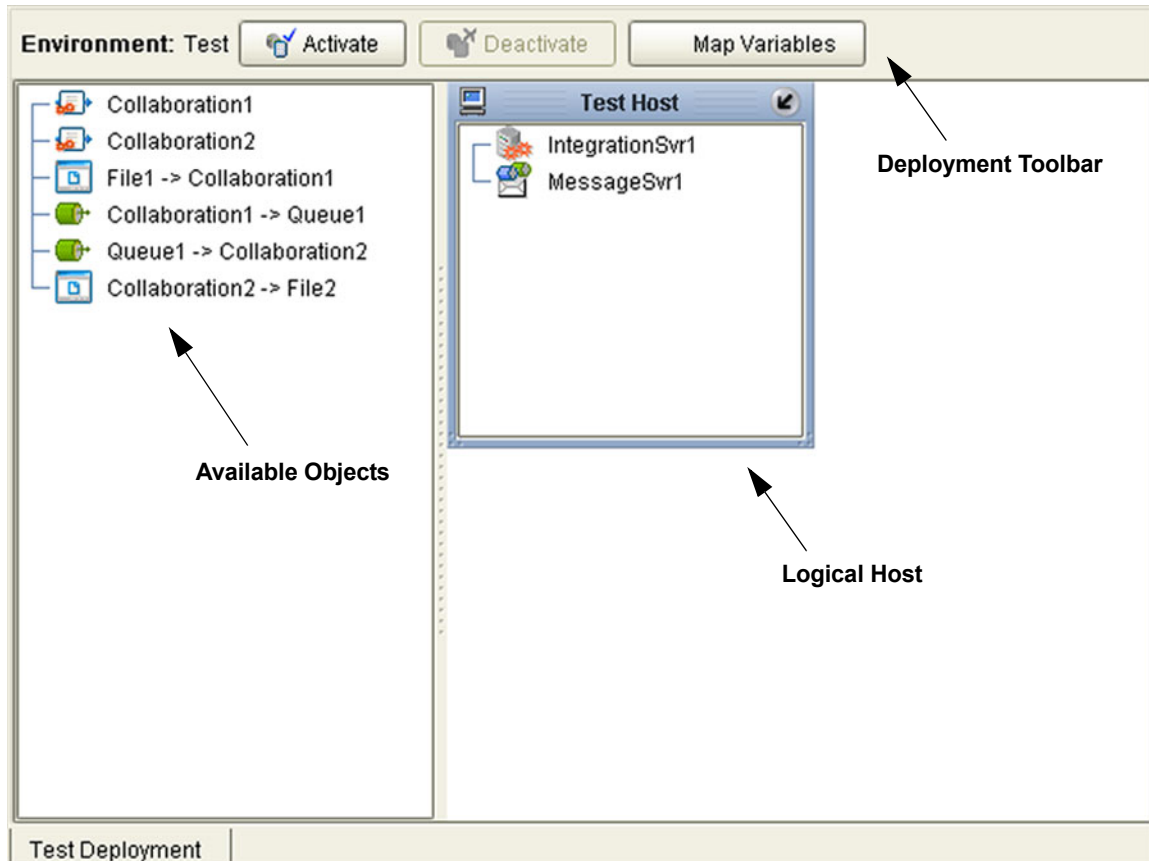




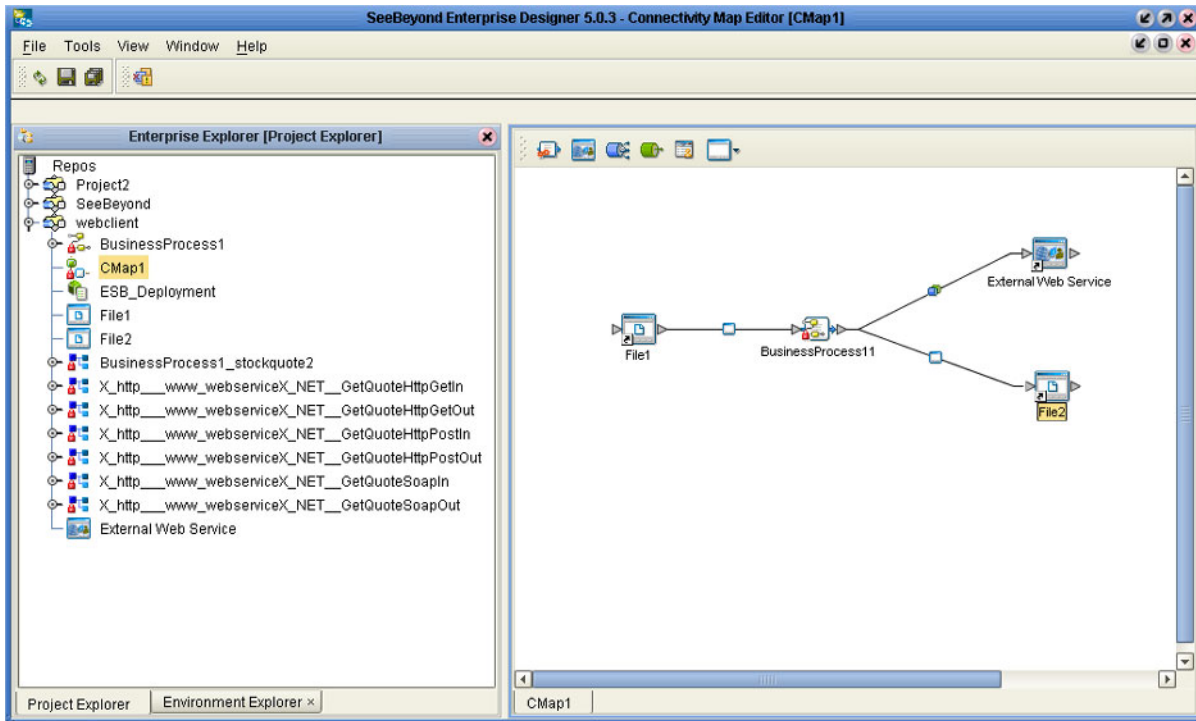
Table 67 Deployment Toolbar Buttons

Button	Function
 Activate	Starts the Project by creating an enterprise archive (EAR) file based on the Connectivity Map and linking this file with the application server. See Activating and Deactivating Deployment Profiles on page 272.
 Deactivate	Stops the Project by terminating the link between the EAR file and the application 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 274.

10.3 Creating a Deployment Profile

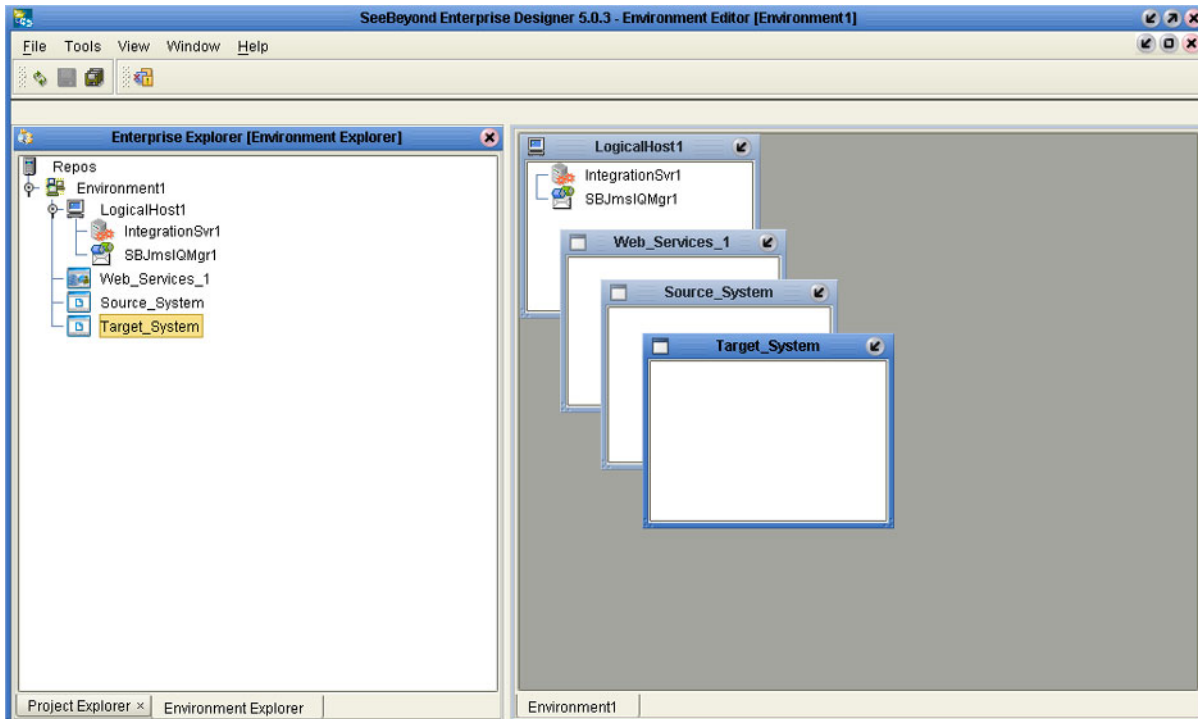
The Web Client Project shown in Figure 227 will be used as a deployment example.

Figure 227 Web Client Example Project



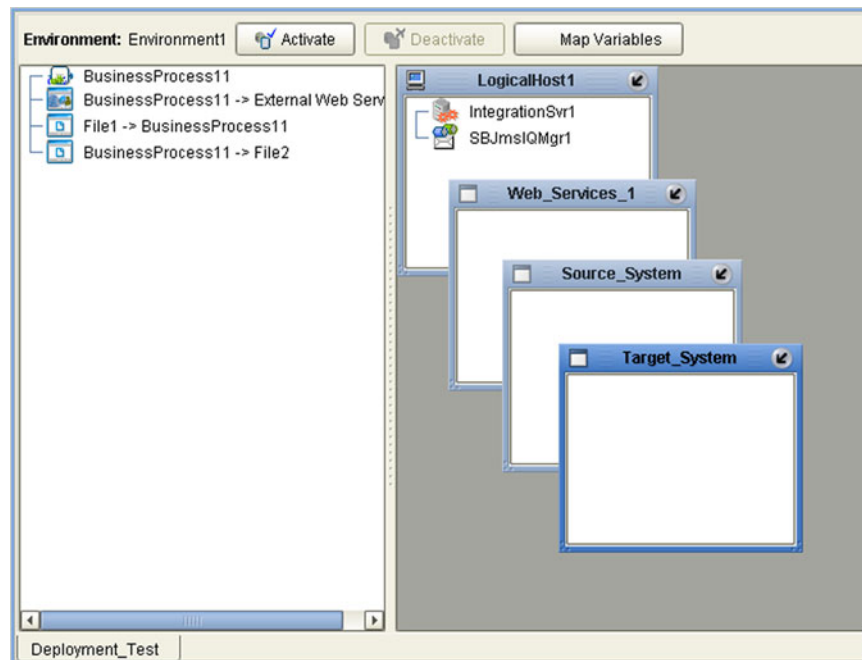
- 1 In the Environment Explorer, create an Environment and right-click on the Environment to display its context menu.
- 2 From the menu, select the Environment components you need and name them appropriately. They will appear as shown in Figure 228.

Figure 228 Web Client Example Environment



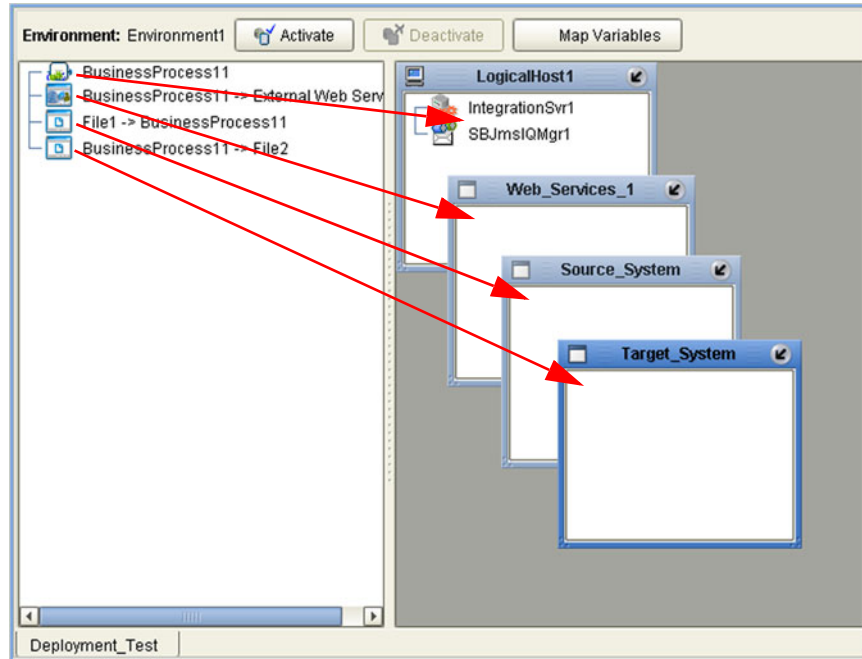
- 3 In the Project Explorer, right-click on the Project to display its context menu.
- 4 From the menu, select **New > Deployment Profile**. The Deployment Profile Editor appears, displaying the Environment you created (see Figure 229).

Figure 229 Example Deployment Profile (1)



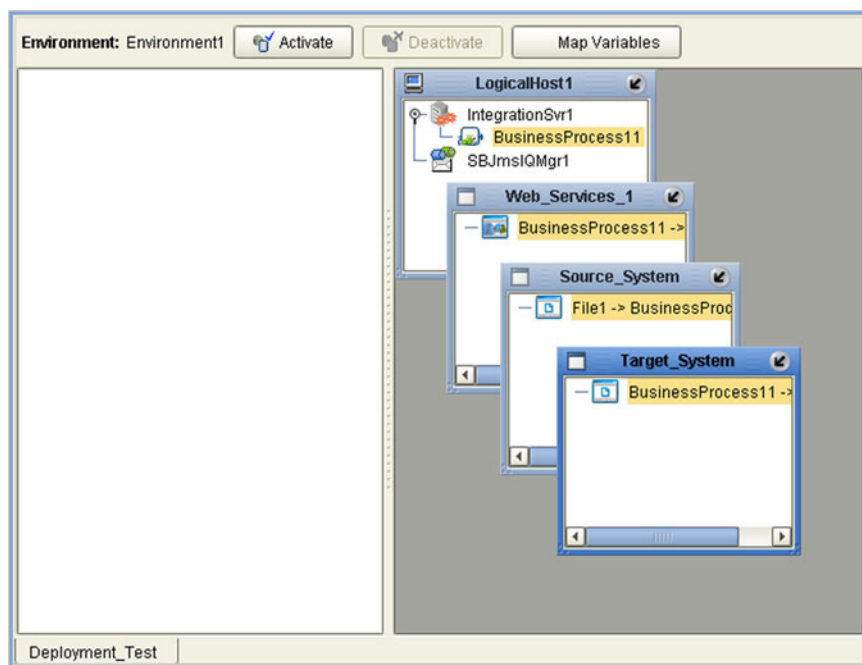
- 5 Drag the Project components from the left panel and drop them into the appropriate Environment components in the right panel, as illustrated in Figure 230.

Figure 230 Example Deployment Profile (2)



- 6 When the Environment components are fully populated, the left panel will be blank, as shown in Figure 231. You should now **Save** the profile.

Figure 231 Example Deployment Profile (3)



10.4 Activating and Deactivating Deployment Profiles

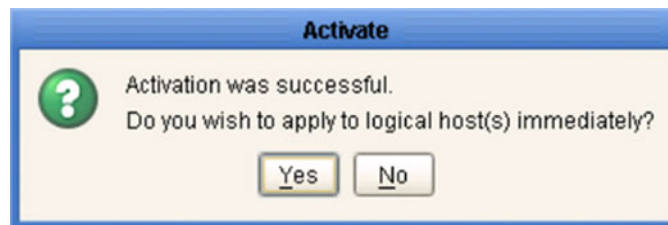
Using the Activate and Deactivate toolbar buttons, you have the option of immediately applying the changes to the Logical Host or deferring the changes to a later time. Activating the Deployment Profile without applying the changes checks the validity of the entire Deployment Profile.

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

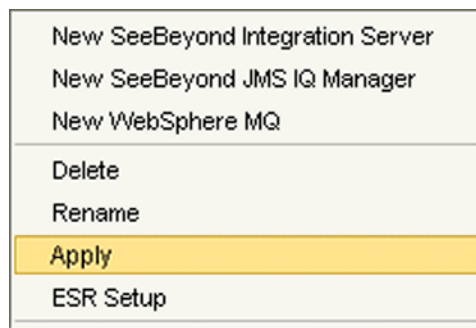
- 1 In the Deployment Profile, select the Deployment you wish to activate.
- 2 Click the **Activate** button. The following message appears:

Figure 232 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 233). This will apply all of the changes for that Logical Host.

Figure 233 Logical Host Context Menu - Apply

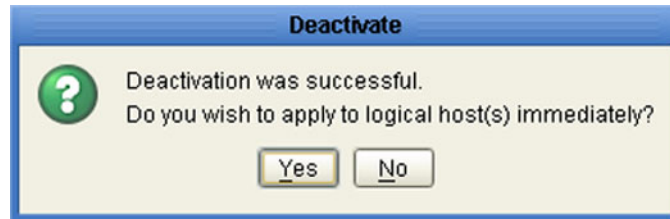


Note: The **Apply** action assumes that the Logic Host is running, since it invokes a trigger to the Logical Host causing it to download the latest settings from the Repository and deploy those settings to all components on the Logical Host.

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 234 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 233). This will apply all of the changes for that Logical Host. See the *Note* following the activation procedure in the preceding section.

10.4.1 Command-line Activation and Deactivation

A script named **CmdLineUtil.sh** (or **.bat**) allows you to deploy and undeploy a specific project via the command line. The *deploy* and *undeploy* commands function in the same way as the **Activate** (or **Deactivate**) and **Apply Changes to LogicalHost** commands in Enterprise Designer, with the one exception that the *deploy* command does not try to regenerate project file—it only copies the existing project file from the Repository to the Logical Host.

10.5 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 235) and values (see Figure 236).

Figure 235 Deployment Profile Mappings

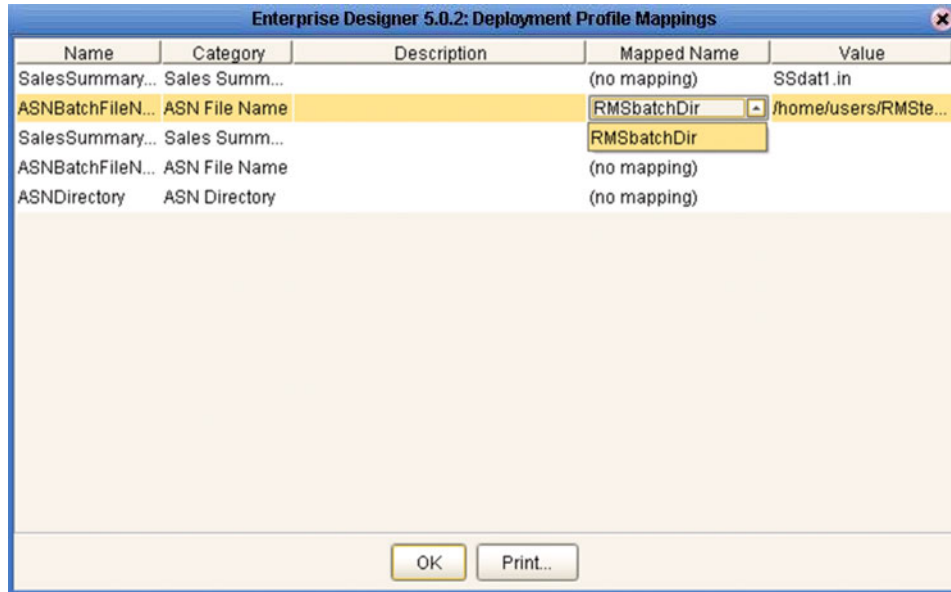
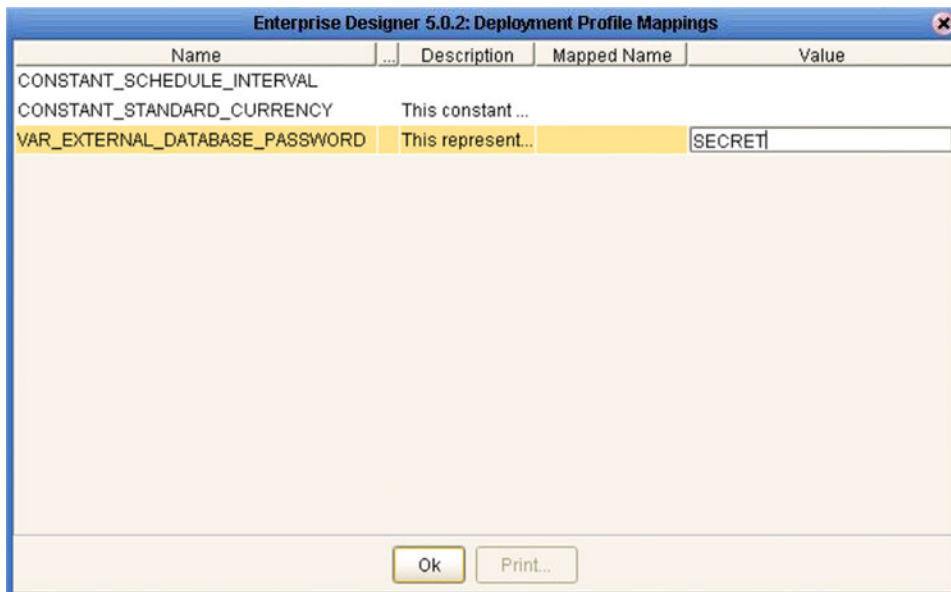


Figure 236 Project Variable Value Entry



10.6 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*.

Because of the versions of the Java Connection Architecture supported by WebLogic and WebSphere, the following restrictions apply:

- Services deployed to WebLogic or WebSphere are restricted to those internal to eGate Integrator itself (between message destinations), and those associated with outbound eWays.
- Not all SeeBeyond eWays support third-party servers. Check the individual eWay User's Guides regarding such support, and also any additional configuration that may be necessary for compatibility with WebLogic or WebSphere.

10.6.1 BEA WebLogic

Note: Before using the WebLogic JMS, you must install additional *.jar* files as described below. For additional information, see the *eGate Integrator JMS Reference Guide*.

To install additional *.jar* files

- 1 Download the **log4j.jar** file from the location below (location subject to change).
<http://jakarta.apache.org/log4j/docs>
- 2 Download the **xerces.jar** file from the location below (location subject to change).
<http://xml.apache.org/dist/xerces-j>
- 3 Copy the following files from *ICAN-root\edesigner\usrdir\modules*:
 - **com.stc.eventmanagementapi.jar**
 - **com.stc.eventmanagementimpl.jar**
 - **com.stc.jms.stcjms.jar**
- 4 Place all *.jar* files into the *ICAN-root\weblogic8x\server\lib* directory.
- 5 Add the *.jar* files to the *set CLASSPATH* segment of the **startWLS.cmd** file located in the *ICAN-root\weblogic8x\server\bin* directory. The text to be added is:

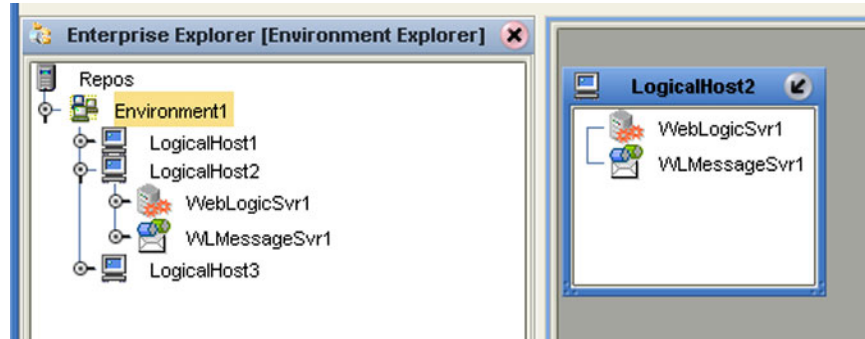
```
%WL_HOME%\server\lib\log4j.jar;%WL_HOME%\server\lib\xerces.jar;  
%WL_HOME%\server\lib\ com.stc.eventmanagementapi.jar;  
%WL_HOME%\server\lib\ com.stc.eventmanagementimpl.jar;  
%WL_HOME%\server\lib\ com.stc.jms.stcjms.jar
```

To deploy an eGate Project to a BEA WebLogic 8.0 or 8.1 environment

- 1 Create the following components in Enterprise Designer (see Figure 237):
 - A A new environment
 - B A Logical Host

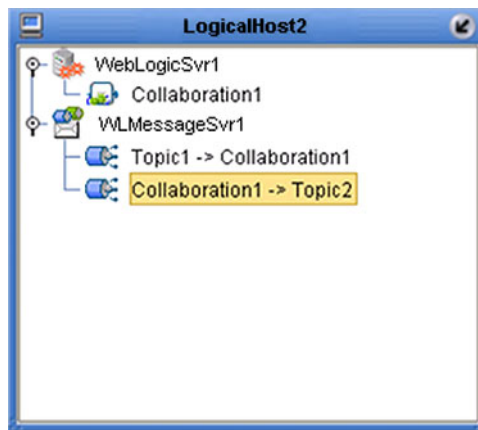
- C A WebLogic J2EE application server
- D A WebLogic JMS message server

Figure 237 WebLogic Deployment (1)



- 2 Create a new Deployment Profile to bind the Connectivity Map to the new WebLogic environment (see Figure 238).
 - A Drag the two topics and drop onto the WebLogic message server.
 - B Drag the Collaboration and drop onto the WebLogic application server.

Figure 238 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:

ICAN-root\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.

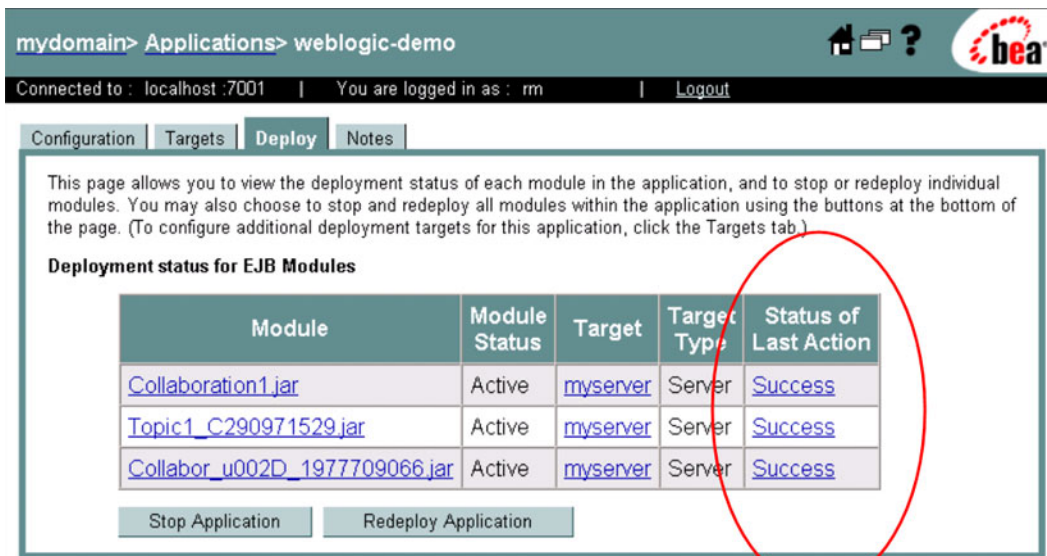
- 5 Navigate to **Server Administration Console > Deployments > Applications**.
- 6 Perform the following steps:
 - A Add a new JMS Connection Factory.
 - B Enter a JNDI name for the JMS Connection Factory:


```
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 239 , which shows a WebLogic 8.1 example).

Figure 239 WebLogic Deployment Verification



10.6.2 IBM WebSphere

Note: Before using the WebSphere JMS, you must install additional **.jar** files as described below. For additional information, see the *eGate Integrator JMS Reference Guide*.

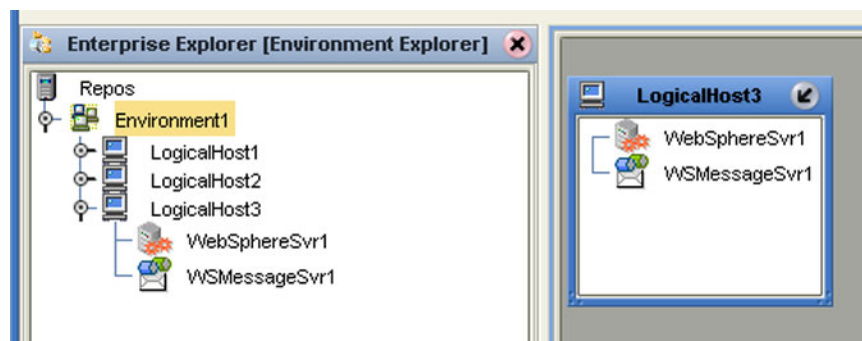
To install **log4.jar**

- 1 Download the **log4j.jar** file from the location below (location subject to change).
<http://jakarta.apache.org/log4j/docs>
- 2 Copy the following files from *ICAN-root\edesigner\usrdir\modules*:
 - **com.stc.antlrimpl.jar**
 - **com.stc.eventmanagementapi.jar**
 - **com.stc.eventmanagementimpl.jar**
 - **com.stc.jms.stc.jms.jar**
- 3 Place all **.jar** files into the *ICAN-root\WebSphere\AppServer\lib* directory.

To deploy an eGate Project to an IBM WebSphere 5.0 or 5.0.1 environment

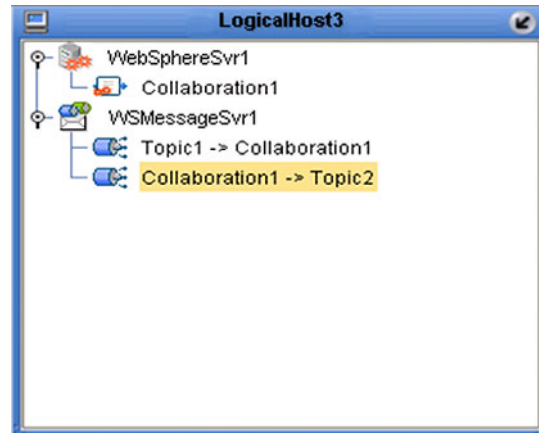
- 1 Create the following components in Enterprise Designer (see Figure 240):
 - A A new environment
 - B A Logical Host
 - C A WebSphere J2EE application server
 - D A WebSphere JMS message server

Figure 240 WebSphere Deployment (1)



- 2 Create a new Deployment Profile to bind the Connectivity Map to the new WebSphere environment (see Figure 241).
 - A Drag the two topics and drop onto the WebSphere message server.
 - B Drag the Collaboration and drop onto the WebSphere application server.

Figure 241 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:

```
ICAN-root\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.

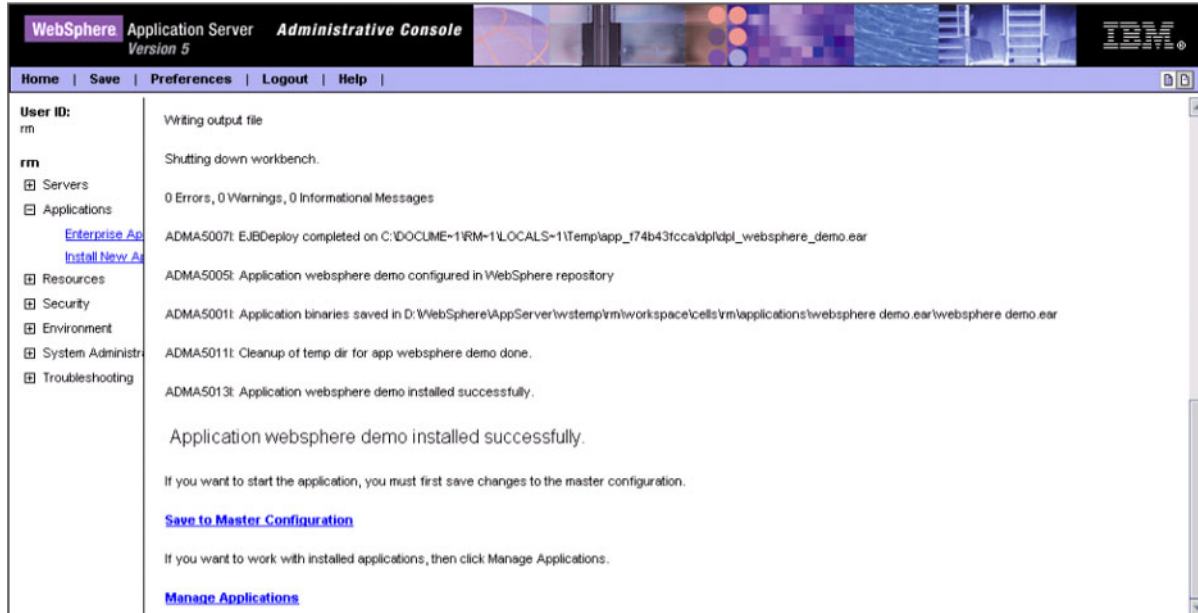
- 8 From the Administrative Console, navigate to **Applications > Enterprise Applications > Install New Application**.
- 9 In *Preparing for the application installation*:
 - A Enter the path for the EAR file described in step 3 and click **Next**.
 - B Select **Generate Default Bindings** and click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

- 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 242 , which shows a WebSphere 5 example).

Figure 242 WebSphere Deployment Verification



Web Services

This chapter describes the use of the Web Services capability of eGate Integrator, acting with other components of the ICAN Suite.

11.1 Overview

Basically, Web Services enables communication and data transfer between diverse applications using the Internet. In doing so, it provides a means for implementing EAI (Enterprise Application Integration) within an organization, or B2B (Business-to-Business) integration between partner organizations. This capability is achieved by wrapping back-end systems to present a common, standardized interface to the connecting network.

Four related technologies are used to transform and transport data within Web Services:

- **XML** (Extensible Markup Language)
Provides a language for defining both the data itself and the way to process it.
- **WSDL** (Web Services Description Language)
Defines the interfaces, data types, interactions, and mappings used in the Web Services. WSDL files are used to invoke and operate Web services on the Internet and to access and invoke remote applications and databases.
- **SOAP** (Simple Object Access Protocol)
Defines a communications envelope that is mappable to HTTP and provides a format for transmitting XML documents over a network.
- **UDDI** (Universal Description, Discovery, and Integration)
Provides a mechanism for storing and categorizing information that allows publication of services and discovery of external services.

11.2 SeeBeyond Web Services

eGate Integrator provides the capability to create either a client or a server to receive WSDL file from a remote server, or send WSDL files to a remote client. eGate works in conjunction with eInsight Business Process Manager, in which the associated business processes are developed. See [Building a Web Client](#) on page 285 and [Building a Web Server](#) on page 294.

The ICAN Suite contains the following components that implement the Web Services capability:

- **WSDL Wizard**

The WSDL Wizard creates an OTD from a WSDL file. See [Using the WSDL Wizard](#) on page 98.

- **WSDL Editor**

See the *eInsight Business Process Manager User's Guide*.

- **WSDL Interface Designer**

See the *eInsight Business Process Manager User's Guide*.

- **WSDL Viewer**

See the *eInsight Business Process Manager User's Guide*.

- **UDDI Repository**

All ICAN objects represented in the Repository that can be accessed as Web services are presented via a UDDI-compliant server. See [UDDI Repository](#) on page 283.

11.3 UDDI Repository

In general, all ICAN objects that expose themselves as a Web service (such as an eInsight business process) are presented via a UDDI-compliant server (see Figure 243). The URL of this server is:

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

Figure 243 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 244.

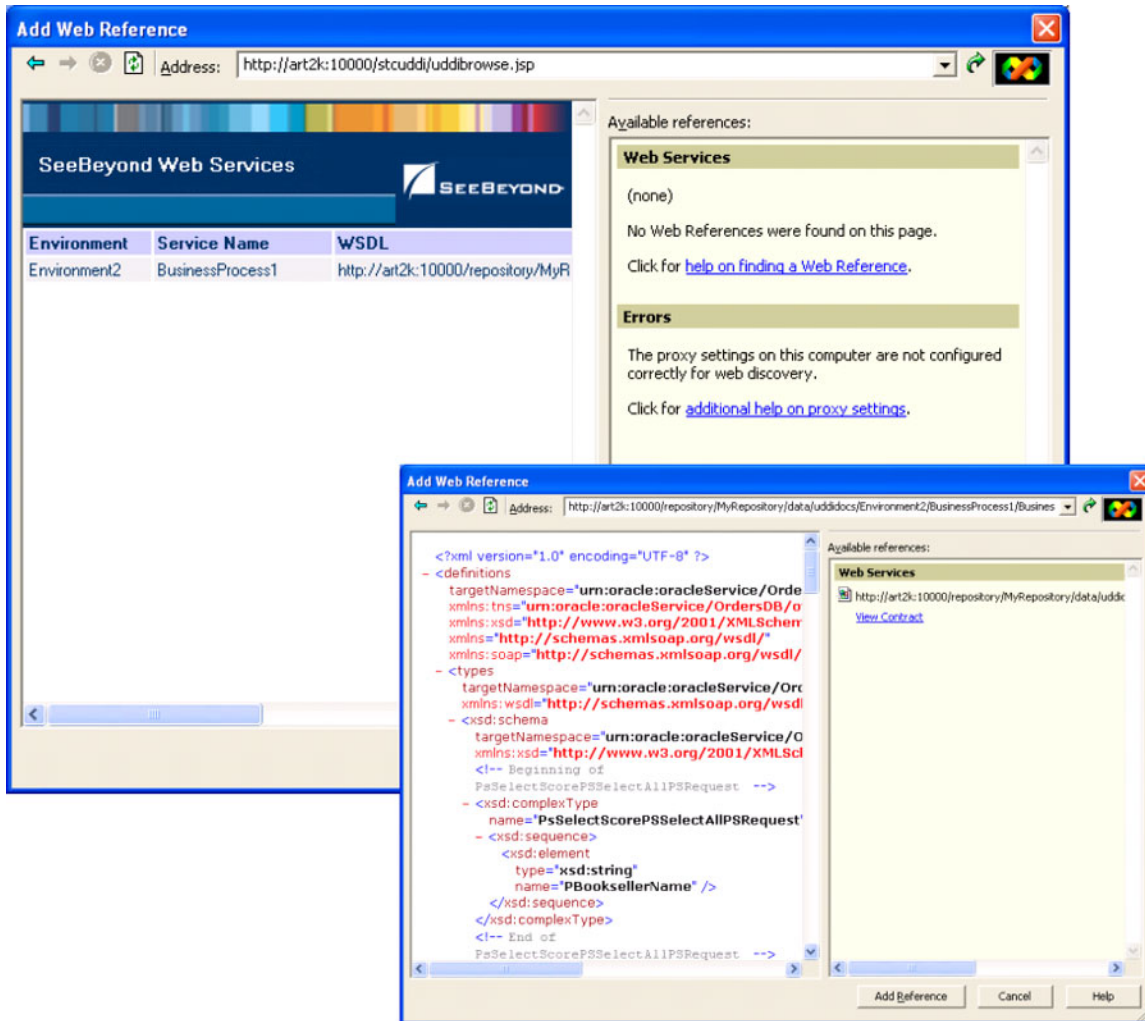
Figure 244 Example Web Service WSDL File

```
<?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 245). In this example, a so-called *Web reference* (to the UDDI Repository) is added to a C# project.

Figure 245 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 [Using the WSDL Wizard](#) on page 98).

11.4 Building a Web Client

Here we briefly demonstrate the procedure for building a Web client. The steps involved are:

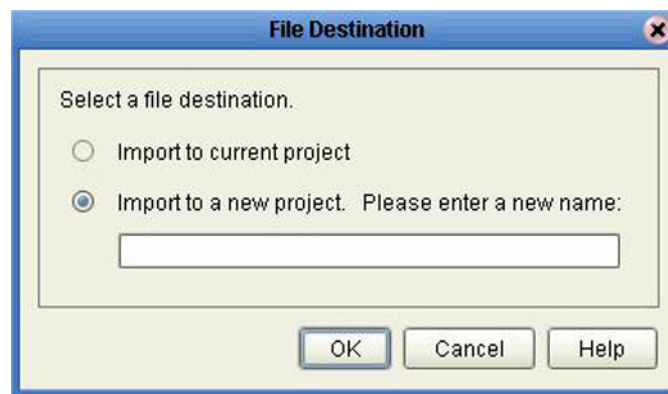
- 1 Build an Object Type Definition (OTD).
- 2 Develop a business process.
- 3 Create the eGate Project.
- 4 Deploy the Project to the selected Environment.

The Project used in the following example is available as **webclient.zip**, contained in the eGate User Guide Sample file included with this User's Guide.

To import the sample project

- 1 The sample files are uploaded with the User's Guide .sar file and downloaded from the Enterprise Manager's Documentation tab. Extract the samples from the Enterprise Manager to a local file.
- 2 From the Enterprise Designer's Project Explorer pane, right-click the Repository and click **Import Project** from the selection menu. The **Select File to Import** dialog box appears.
- 3 Browse to the directory that contains the sample project zip file. Select the sample file (**webclient.zip**) and click **Open**.
- 4 From the *File Destination* dialog box (see Figure 246), select **Import to a new Project**, enter the name of the Project, and click **OK**.

Figure 246 File Destination Dialog Box



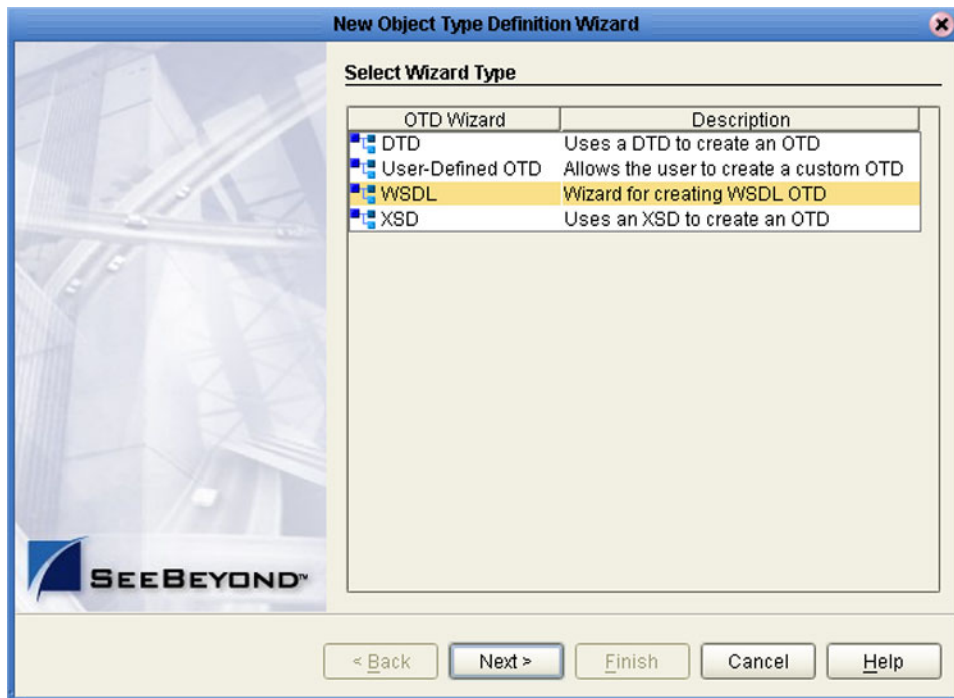
- 5 After the import has successfully completed, select the **Repository** in the Project Explorer and click **Refresh All from Repository**.

11.4.1 Object Type Definition

To create the WSDL OTD

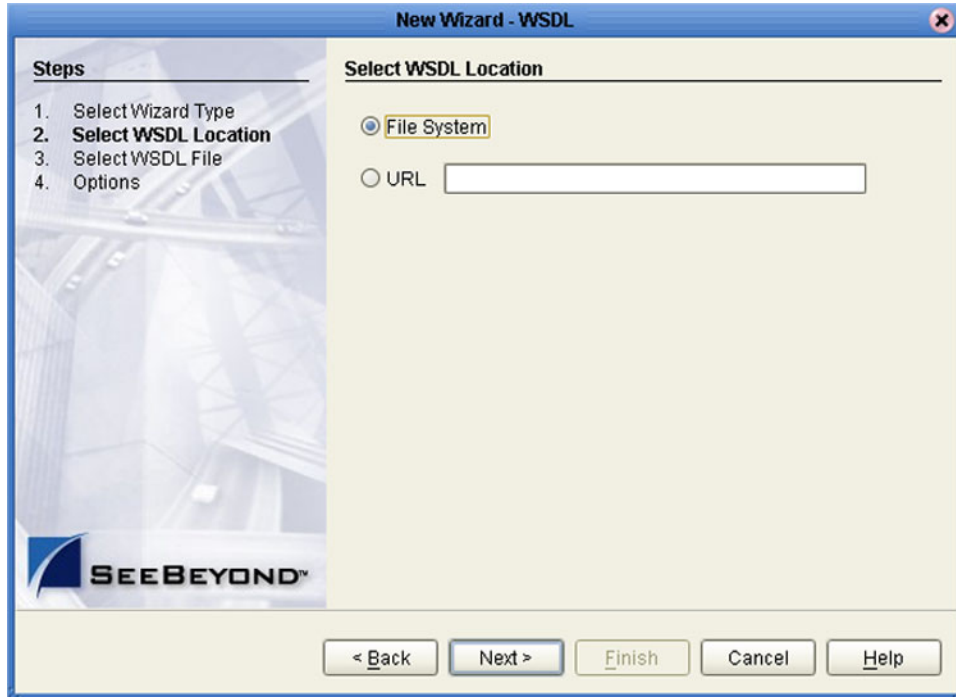
- 1 In Project Explorer, create a new OTD.
- 2 Select the **WSDL OTD Wizard** (see Figure 247).

Figure 247 Select WSDL Wizard



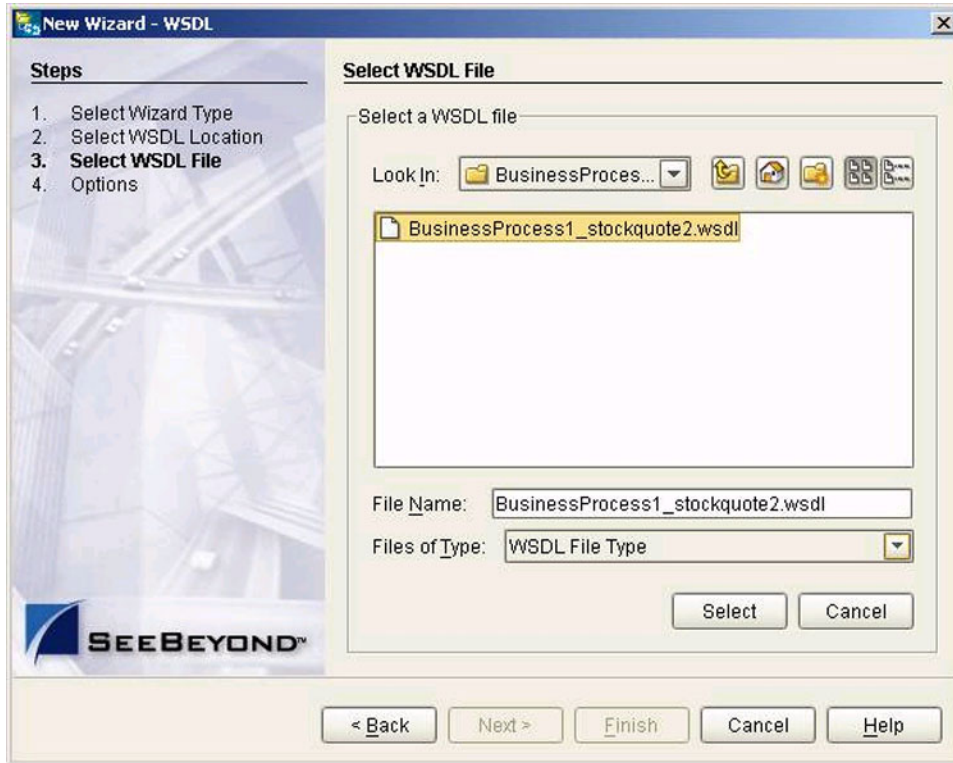
- 3 Select the WSDL file location and click **Next**. In this example, the file is located in the local file system (see Figure 248).

Figure 248 Select File Location



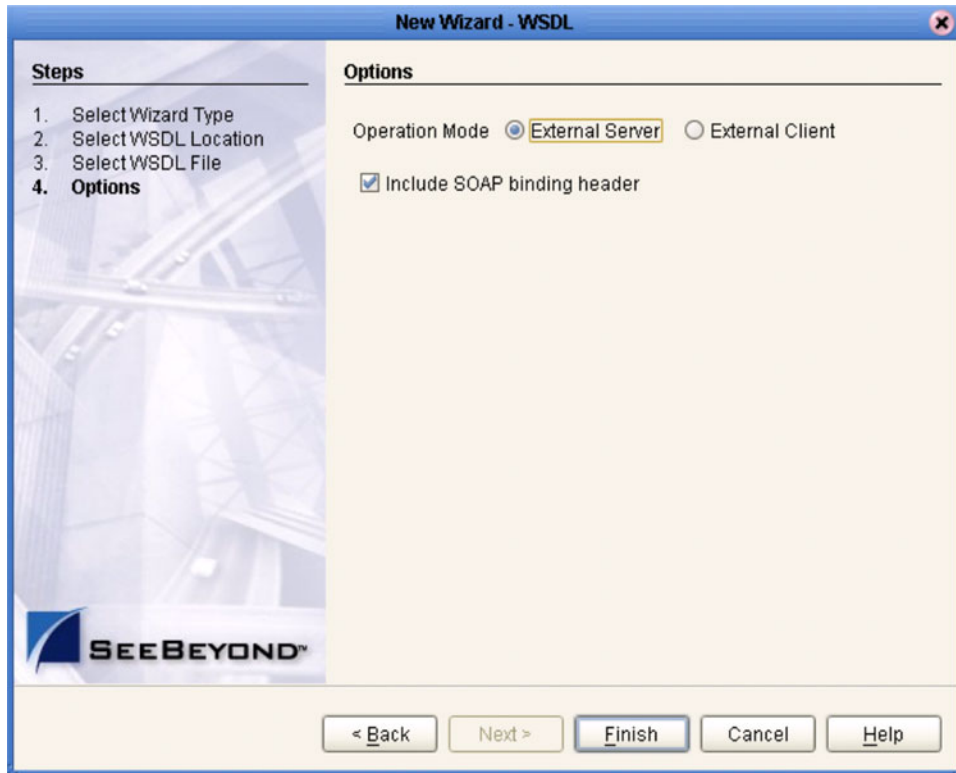
- 4 Select the WSDL file on which you want to base the OTD (see Figure 249). The file itself is shown in [Figure 254 on page 291](#).

Figure 249 Select WSDL File



- 5 For a Web Client, select the following options (see Figure 250):
 - A Select **External Server** as the Operation Mode.
 - B Select **Include SOAP binding header** .

Figure 250 Select External Server



- 6 Click **Finish**.

11.4.2 eInsight Business Process

The example business process, developed in eInsight Business Process Manager, is shown in Figure 251 (see the *eInsight Business Process Manager User's Guide* for details).

Figure 251 Web Client Business Process



The **receive** rule for the business process is shown in Figure 252, and the **write** rule is shown in Figure 253.

Figure 252 Web Client Business Process Receive Rule

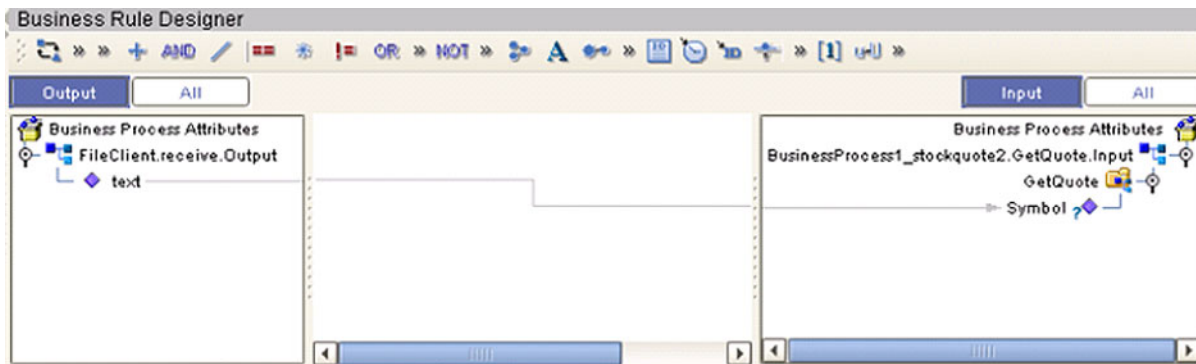
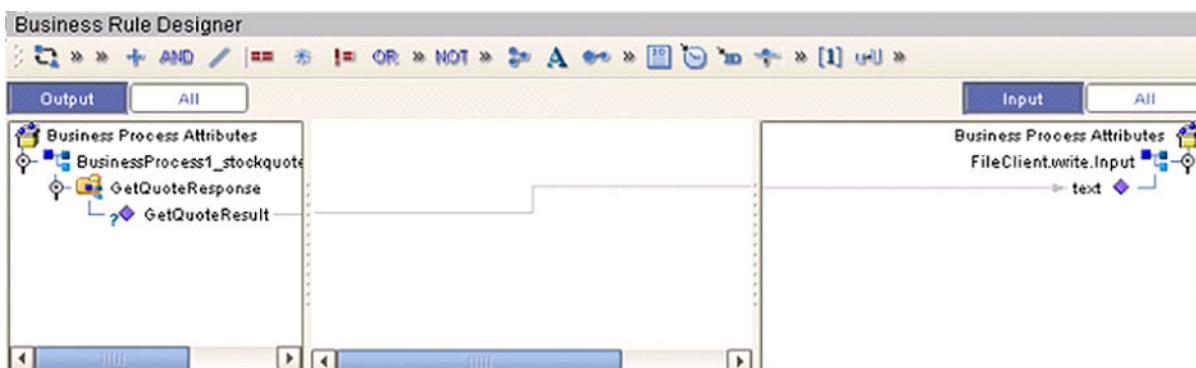


Figure 253 Web Client Business Process Write Rule



The WSDL file describing the business process is shown in Figure 254.

Figure 254 Sample WSDL File

```
1 <process name="BusinessProcess1"
2   targetNamespace="http://127.0.0.1:12000/repository/webclient/BusinessProcess1"
3   sbynpxp:end_YLoc="123.0"
4   sbynpxp:start_YLoc="120.0"
5   sbynpxp:linkStyle="angular"
6   sbynpxp:start_XLoc="50.0"
7   sbynpxp:end_XLoc="508.0"
8   xmlns:tns="http://127.0.0.1:12000/repository/webclient/BusinessProcess1"
9   xmlns:sbynpx="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/"
10  xmlns:slink="ServiceLinkTypes/SeeBeyond/eInsight/e32731:f8eaf3f6cf:-7fff"
11  xmlns:ns0="http://www.webserviceX.NET/"
12  xmlns:sbyruntime="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/runtime/"
13  xmlns:sbyncreation="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/creation"
14  xmlns:ns1="urn:fileservice"
15  xmlns:sbynpxp="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/presentation/"
16  xmlns="http://schemas.xmlsoap.org/ws/2002/07/business-process/"
17  xmlns:sbyntracing="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/tracing/"
18  xmlns:sbyninc="http://bpel.seebeyond.com/hawaii/5.0/privateExtension/incompleteModel">
19  <!-- partners definition -->
20  <partners>
21    <partner name="BusinessProcess1_stockmate?"
```

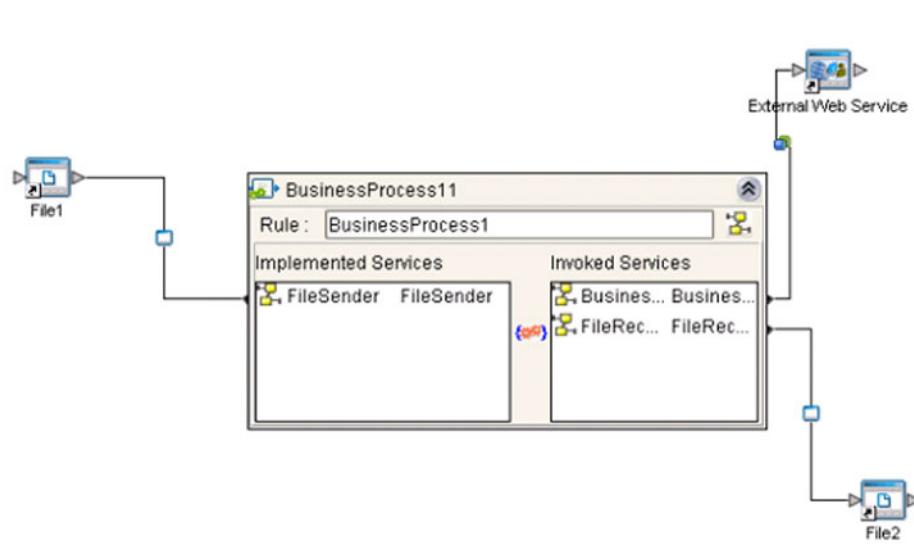
11.4.3 eGate Project

The Project components are created and mapped in the Enterprise Designer Connectivity Map Editor. The example Project contains:

- Two external files and accompanying File eWays.
- An External Web Service.
- A service, into which you drag and drop the eInsight business process from the Project Explorer.

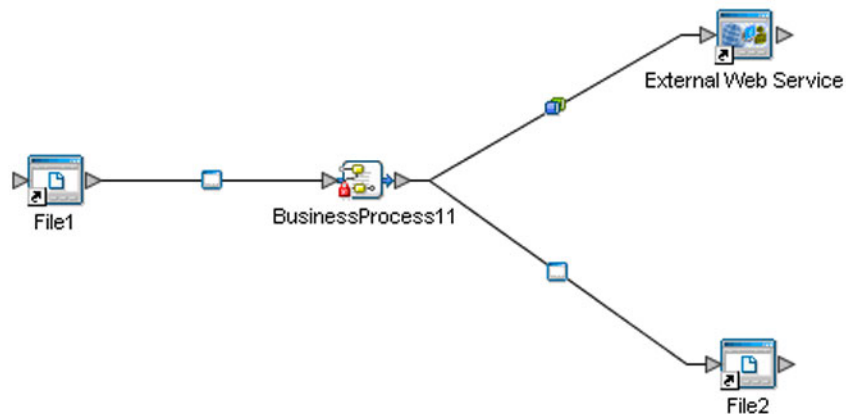
The business process is connected as shown in Figure 255.

Figure 255 Map Business Process



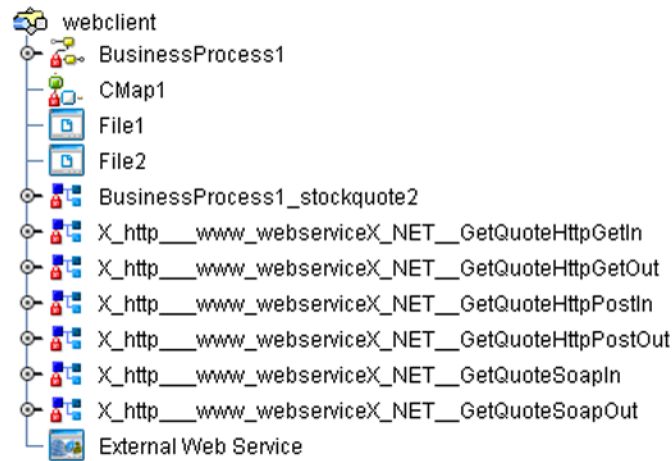
The completed Connectivity Map for the example Project is shown in Figure 256.

Figure 256 Web Client Connectivity Map



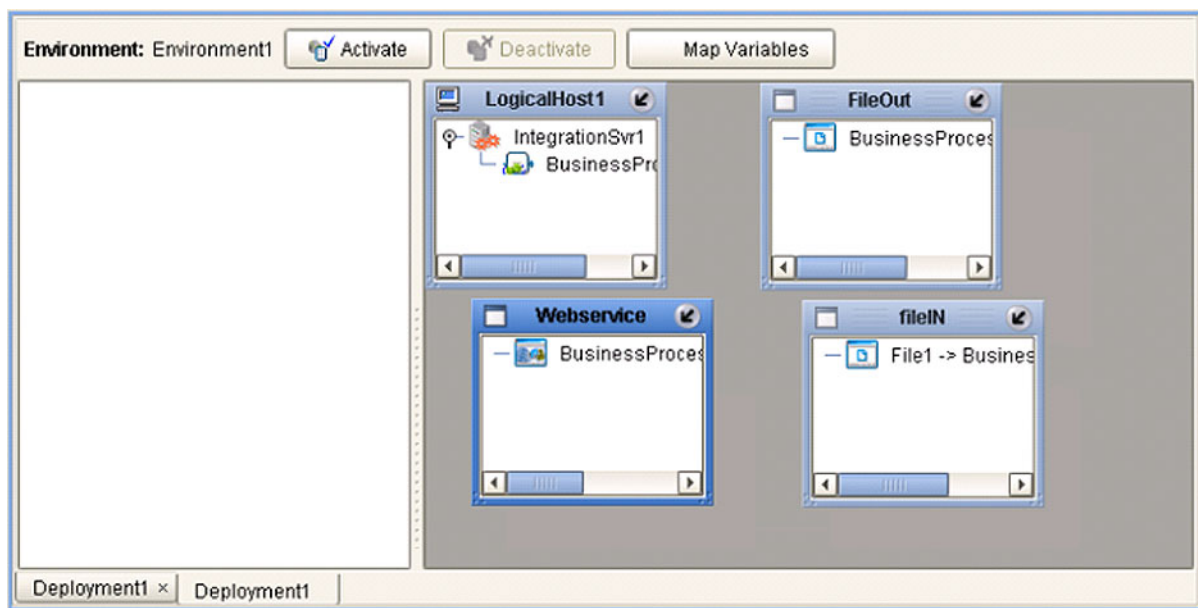
The Web client example Project appears in the Project Explorer as shown in Figure 257.

Figure 257 Web Client Example Project



The example Project is deployed as shown in Figure 258.

Figure 258 Project Deployment



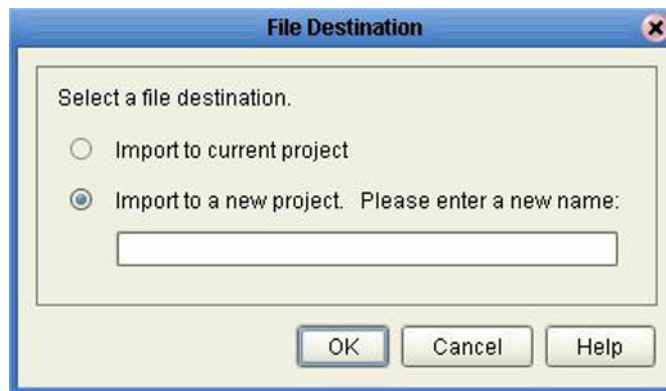
11.5 Building a Web Server

Here we briefly demonstrate the procedure for building a Web server. The Project used in the following example is available as **webserver.zip**, contained in the eGate User Guide Sample file included with this User's Guide.

To import the sample project

- 1 The sample files are uploaded with the User's Guide .sar file and downloaded from the Enterprise Manager's Documentation tab. Extract the samples from the Enterprise Manager to a local file.
- 2 From the Enterprise Designer's Project Explorer pane, right-click the Repository and click **Import Project** from the selection menu. The **Select File to Import** dialog box appears.
- 3 Browse to the directory that contains the sample project zip file. Select the sample file (**webserver.zip**) and click **Open**.
- 4 From the *File Destination* dialog box (see Figure 246), select **Import to a new Project**, enter the name of the Project, and click **OK**.

Figure 259 File Destination Dialog Box

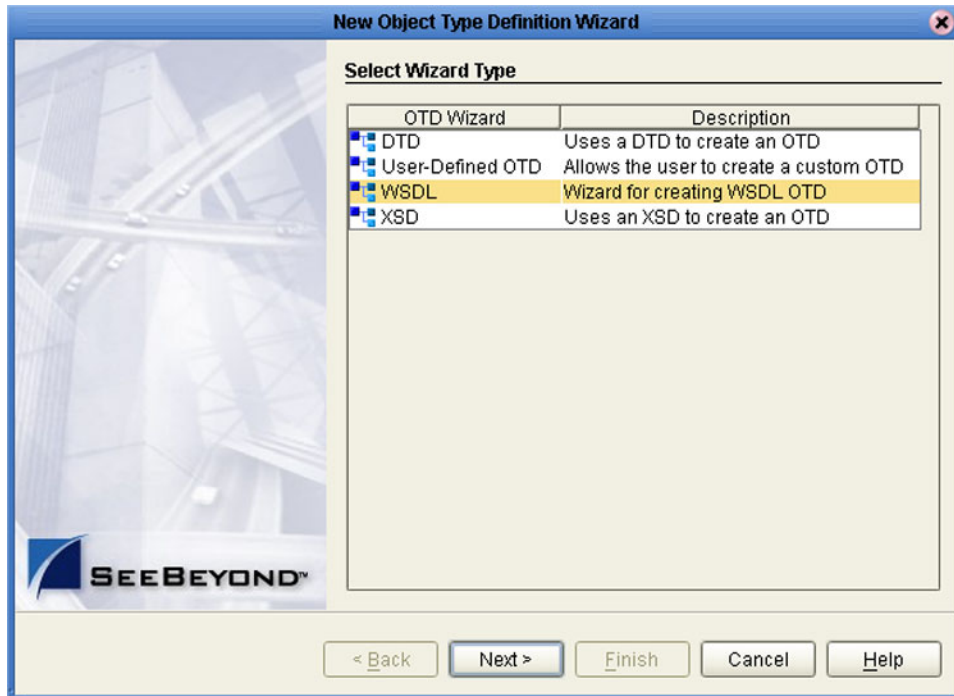


- 5 After the import has successfully completed, select the **Repository** in the Project Explorer and click **Refresh All from Repository**.

To build a Web Server Using the ICAN Suite

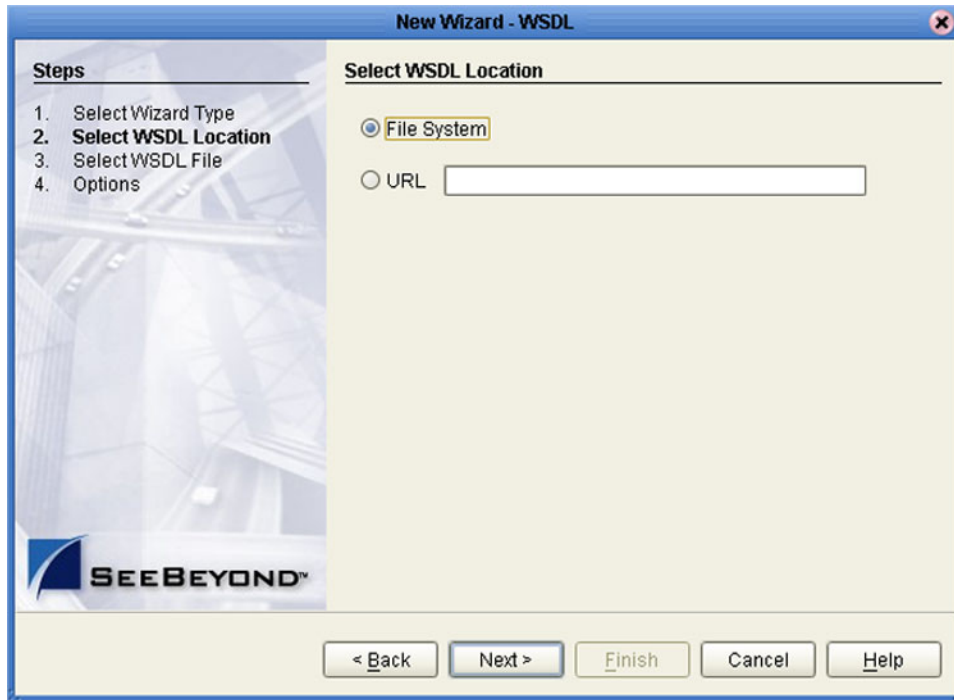
- 1 In Project Explorer, create a new OTD.
- 2 Select the WSDL OTD Wizard (see Figure 247).

Figure 260 Select WSDL Wizard



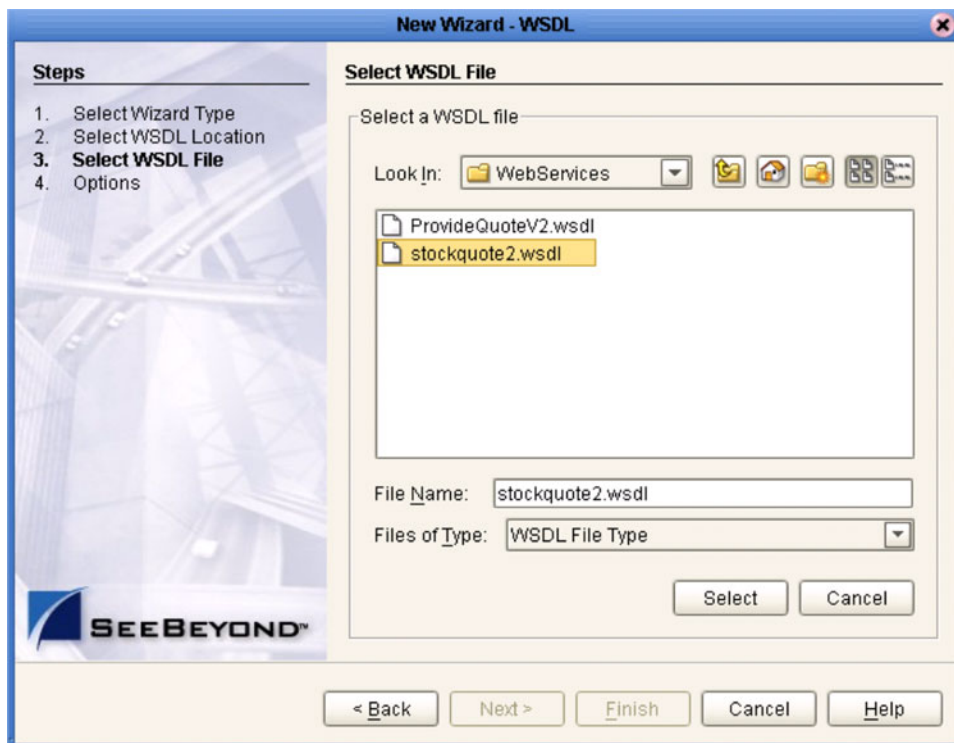
- 3 Select the WSDL file location (see Figure 261).

Figure 261 Select File Location



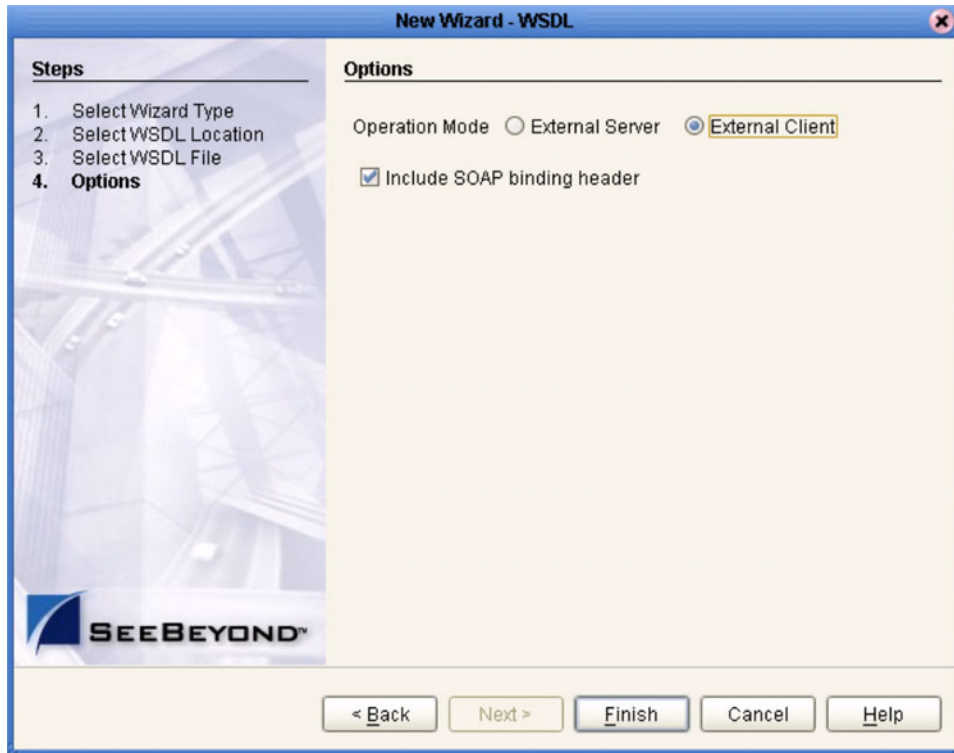
- 4 Select the WSDL file you want to use for the OTD (see Figure 262).

Figure 262 Select WSDL File



- 5 For a Web server, select External Client (see Figure 263).

Figure 263 Select External Client

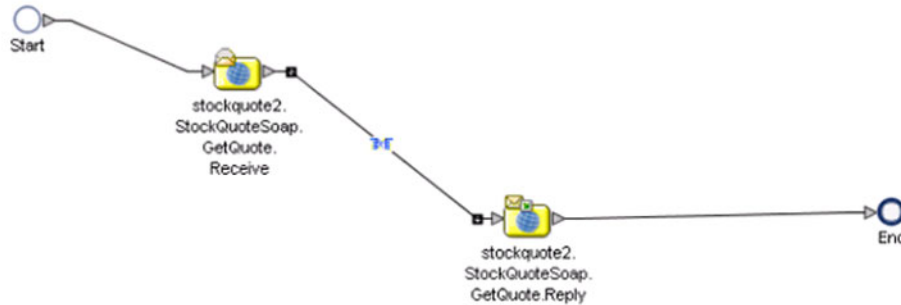


- 6 Click **Finish**.

11.5.1 eInsight Business Process

The example business process, developed in eInsight Business Process Manager, is shown in Figure 264 (see the *eInsight Business Process Manager User's Guide* for details).

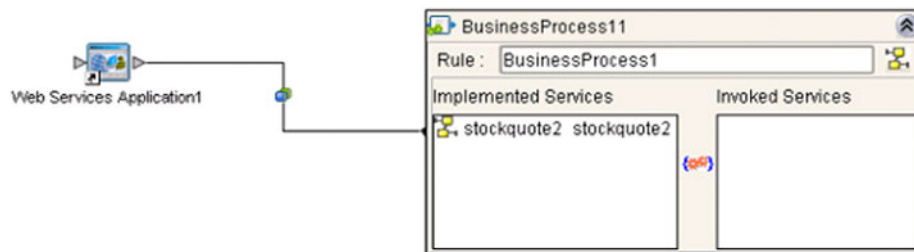
Figure 264 Web Server Business Process



11.5.2 eGate Project

The business process is connected as shown in Figure 265, using the Enterprise Designer Connectivity Map Editor.

Figure 265 Connectivity Map



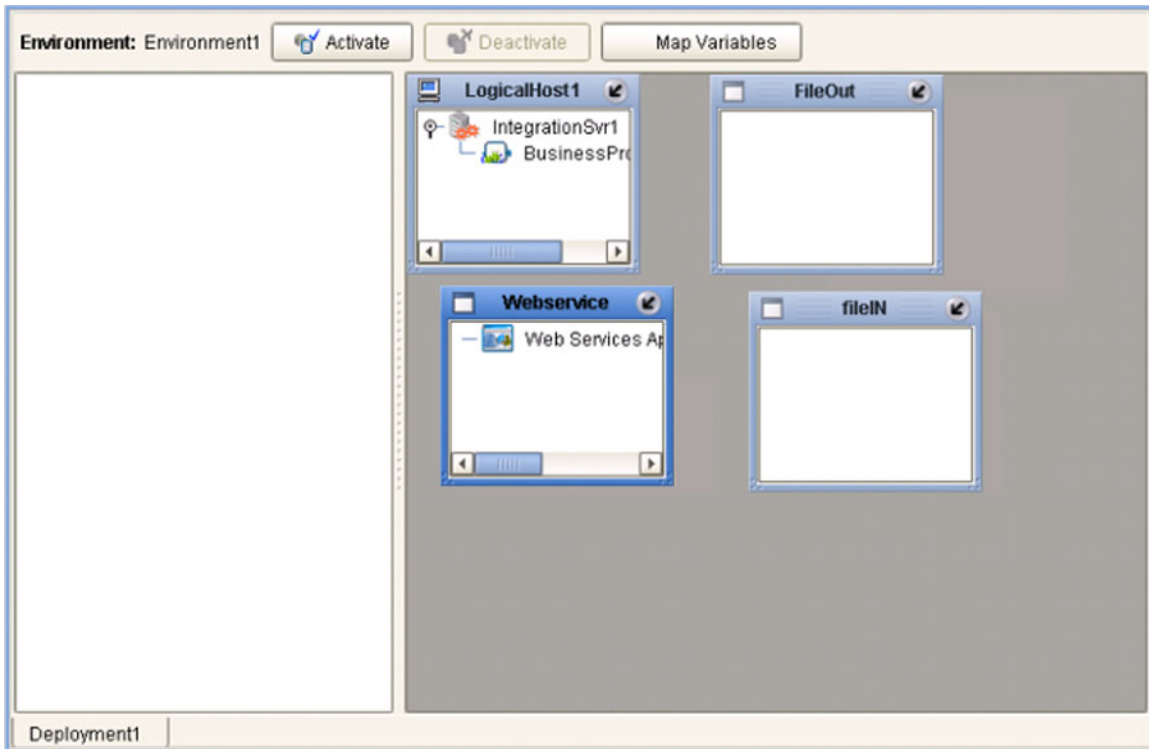
The Web server example Project appears in the Project Explorer as shown in Figure 266.

Figure 266 Web Server Example Project



The Project is deployed as shown in Figure 267.

Figure 267 Project Deployment



OTD Interfaces

A.1 The *Bean-like* Interface

At run time, an OTD instance is accessed directly from Java in a Java-based Collaboration, using the so-called *bean-like* accessors. 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.

This data structure is represented at run time by a set of generated Java classes, which 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.

Important: *Do not modify this Java name property.*

A.2 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.

Glossary

BI

Business integration (also Business Intelligence).

Collaboration

A logical operation performed between some combination of message destinations and external applications. The operation is defined by a Collaboration Definition, which can be encoded in either Java or XSLT.

Also 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**”). 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.

CRM

Customer Relations Management

Data Cleansing

Data must be cleansed of errors in structure and content before it is useful in data warehousing and integration; this means transforming data for accurate and effective use in a database or data management system by cleansing “dirty” or redundant data.

Data Dictionary

Defines the organization of a database and lists all files in the database, the number of records in each file, and the names and types of each field. The data dictionary is often hidden from end users. Although the dictionary doesn’t contain actual data, it does contain essential information for managing the database.

Data Integrity

Refers to the accuracy and validity of data. Data integrity can be compromised in many ways, including human error through data entry, or through faulty logic in programming. Computer viruses, software bugs and many other factors can also compromise data integrity.

Data Mapping

In relational databases (RDBMSs) data mapping is the relationship and data flow between source and target objects. Mapping involves structuring the relationship between source and target objects.

Data Mart

A smaller, focused, database designed to help managers make business decisions. (A data warehouse is a larger, enterprise, database(s).)

Data Mining

Used to synthesize or isolate unique data patterns to predict future behaviors or to filter data to select patterns that help discover previously unknown relationships among data. Commonly used by marketers who acquire and distill consumer information.

Data Transformation

Data transformation is necessary after extracting data from legacy data formats, or any format that requires cleansing. Data is transformed for efficient use for Business-to-Business Enterprise Data Integration.

Data Warehouse

A copy or view of enterprise transaction data (sometimes non-transaction data) that is used for reporting. The data is often summarized and always structured for queries and analysis.

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.

Dimension Table

Dimension tables describe the business entities of an enterprise; also called lookup or reference tables.

Dirty Data

Dirty data contains, but is not limited to, incorrect data including spelling errors, punctuation errors, incorrect data referencing, incomplete, inconsistent, outdated, and redundant data.

Drill Down

To move from summary to more detailed data by “drilling down” to get it. In database terminology this might mean starting with a general category and drilling down to a specific field in a record.

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.

EPR

Enterprise Resource Management

ETL

Extract, Transform, Load. Extract is the process of reading data from a source database and extracting the desired subset of data. Transform is the process of converting the extracted data from its previous form into the desired form. Load is the process of writing the data into a larger database.

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 in an eGate Project of an external application.

External System

A representation in an eGate Project of an external application system.

Extraction

Data are extracted from a source using software tools. This first step in ETL initially “gets” the data.

Fact Table

A fact table typically contains two types of columns: those containing facts and those that contain foreign keys to dimension tables. Fact tables contain detail facts and/or summary facts.

ICAN Suite

The SeeBeyond Integrated Composite Application Network Suite.

Integration Server

J2EE software platform that houses the business logic container used to run Collaborations and JCA connectors (eWays). Provides transaction services, persistence, and external connectivity.

JMS IQ Manager

JMS-compliant, guaranteed delivery store, forwarding, and queueing service.

Join

Matches records, which are joined by a common field, in two tables in a relational database. Often part of a Select query.

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.

Metadata

“Data about data.” Metadata describes “how,” “when,” and “who” about structure and format, of a particular set of data. ETL tools are used to generate and maintain a central metadata repository.

Non-normalized Data

Non-normalized data cannot be cross-referenced accurately, if at all, and causes manageability issues. Non-normalized data may be converted to normalized data.

Normalized Data

Normalization is a common database design process used to remove redundant or incorrect organization and data. The design and normalization of the database will create a maintainable data set that can be cross-referenced.

Normalized data is not only easier to analyze but also easier to expand. Normalization involves removing redundancy and correcting incorrect data structure and organization.

OLAP

Online analytical processing.

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.

Query

A request for information from a database. There are three query methods:

Choose – With this easy-to-use method, the database system presents a list of parameters from which you can choose. This method is not as flexible as other methods.

Query by example (QBE) – With this method, the system lets you specify fields and values to define a query.

Query language – With this method, you have the flexibility and power to make requests for information in the form of a stylized query using a query language. This is the most complex and powerful method.

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 JMS IQ Manager 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.

Raw Data

Data that has not been turned into "information," through processing. Although factual and "real," raw data is unorganized.

Relational Database (RDBMS)

Short for Relational Database Management System, most often referred to as RDBMS. Data is stored in related tables. Relational databases can be viewed in many different ways.

In this system a single database can be spread across several tables. (RDBMS differs from flat-file databases where each database is self-contained as a single file or table.)

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.

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.

Staging Data

Data that is to be processed before entering the warehouse.

Subproject

An independent Project that is included as part of another Project and listed on the Enterprise Explorer tree beneath the main Project icon.

Table

Refers to data arranged in rows and columns, like a spreadsheet. In relational database management systems, all information is stored in tables.

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 JMS IQ Manager publishes a message on a topic, it ensures that all subscribers receive the message.

Transformation

Data that are extracted from databases are transformed into a desired form, using various tools that cleanse, merge, purge, aggregate, calculate, audit, remove redundancy, standardize, etc.

XSLT

An acronym for Extensible Stylesheet Language Transformations. A file format used in eGate to generate Collaboration Definitions.

Warehouse

See “Data Warehouse”.

e*Gate 4.x Terms in eGate 5.0

Table 68 lists terminology that is new with eGate release 5.0 along with equivalent terms from eGate release 4.x, where applicable.

Table 68 Terminology Cross-Reference

eGate 5.0 Term	Equivalent e*Gate 4.x Term
Connection	e*Way Connection
Connectivity Map	Schema Network View (closest)
Deployment	Running the Control Broker
Deployment Profile	<none> (part of Schema)
Enterprise Designer	Enterprise Manager
Enterprise Manager	Enterprise Monitor
Environment	Schema (physical layer only)
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
ICAN Monitor	Enterprise Monitor
Integration Server	<none>
Link	JMS e*Way Connection
Linked Message Destination	<none>
Logical Host	Participating Host
Message Destination	Topic or queue
Message Server	MS IQ Manager
Object Type Definition (OTD)	Event Type Definition (ETD)
Process Manager	Control Broker
Project	Schema (logical layer only)
Queue	MS queue
Repository	Registry
Subproject	Schema (logical layer only)
Topic	JMS topic

Index

A

- ACL properties 77, 244–245
- Activate button 268
- activating
 - Deployment Profile 272
- addition Collaboration method 160, 218
- Advanced Mode command 144
- AND Collaboration method 157, 218
- arrayAccess Collaboration method 168
- arrayAssign Collaboration method 169
- arrayLength Collaboration method 169
- Auto Layout command 154
- Automerger 173, 238

B

- BEA WebLogic 275
- BI 302
- bitNot operator 170
- boolean Collaboration method 226
- Break command 147
- business rules
 - tree 148
- Business Rules on Left command 144
- Business Rules on Top command 144
- buttons
 - Activate 268
 - Deactivate 268
 - Map Variables 268

C

- Call Java Method command 152
- Call New Constructor command 153
- cast Collaboration method 162
- ceiling Collaboration method 224
- charAt Collaboration method 165
- Collaboration 302
 - derived 303
- Collaboration Definition 302
- Collaboration definition
 - Java 132
 - XSLT 208
- Collaboration method box

- Java 155
 - XSLT 217
- Collaboration method palette
 - Java 155
 - XSLT 217
- Collapse All Methods command 154
- Commit Changes command 144
- Commit Code Changes command 216
- concat Collaboration method 165, 221
- concat-sequence-format Collaboration method 233
- connection 302
- Connectivity Map 302
 - Editor 79
- constants 302
 - Environmental 246
- contains Collaboration method 226
- Continue command 147
- Control Broker 308
- conventions
 - path name separator 22
 - Windows 22
 - writing 22
- count Collaboration method 228
- Create Literal command 152
- CRM 302
- current Collaboration method 228
- currentDateTime Collaboration method 233
- customizer 86

D

- data cleansing 302
- data dictionary 302
- data integrity 303
- data mapping 303
- data mart 303
- data mining 303
- data transformation 303
- Data Type Definition (DTD) 93
- data warehouse 303, 307
- Deactivate button 268
- deactivating
 - Deployment Profile 272
- Deployment EditorEditors
 - Deployment 268
- Deployment Profile 267, 303
 - activating 272
 - creating 269
 - deactivating 272
 - map variables 274
- derived Collaboration 303
- Diff file
 - create 172, 238
 - generate 172, 238

- merge 173, 239
- dimension table 303
- dirty data 303
- division Collaboration method 160, 218
- document Collaboration method 228
- drill down 304
- DTD Wizard 93

E

- Editor
 - Connectivity Map 79
 - Java Collaboration 143
 - OTD 125
 - XSLT Collaboration 215
- eGate system 304
- element-available Collaboration method 226
- endsWith Collaboration method 165
- Enterprise Designer
 - enterprise explorer 49
 - menu bar 46
 - starting 45
- Enterprise Explorer
 - Environments 50, 240
 - Projects 49, 73
- Enterprise Manager
 - Documentation 36
 - Interface 34
 - starting 33
- Enterprise Monitor 37, 308
- Environment 28, 304
 - constants 246
- Environment Explorer 50, 240
- EPR 304
- equal Collaboration method 158, 219
- equals Collaboration method 162
- ETD 308
- ETL 304
- Event Type Definition 308
- eWay 304
- Expand All Methods command 154
- Export Java Rule command 144
- external
 - application 304
 - system 304
- External System 28
- extraction 304

F

- fact table 304
- false Collaboration method 227
- Field command 147
- Find command 145

- Find Next command 145
- Find Previous command 145
- floor Collaboration method 224
- For command 147
- format_number Collaboration method 221
- formatDate Collaboration method 233
- formatDateTime Collaboration method 234
- formatMessage Collaboration method 232
- function_available Collaboration method 227

G

- generate-id Collaboration method 229
- getAppID Collaboration method 232
- getAppValue Collaboration method 232
- getCenturyFromDate Collaboration method 234
- getCenturyFromDateTime Collaboration method 234
- getCommonID Collaboration method 232
- getCommonValue Collaboration method 233
- getDayFromDate Collaboration method 234
- getDayFromDateTime Collaboration method 235
- getHourFromDateTime Collaboration method 235
- getHourFromTime Collaboration method 235
- getMinutesFromDateTime Collaboration method 235
- getMinutesFromTime Collaboration method 235
- getMonthFromDate Collaboration method 234
- getMonthFromDateTime Collaboration method 234
- getSecondsFromDateTime Collaboration method 235
- getSecondsFromTime Collaboration method 235
- getTimezoneFromDate Collaboration method 235
- getTimezoneFromDateTime Collaboration method 236
- getTimezoneFromTime Collaboration method 236
- getYearFromDate Collaboration method 234
- getYearFromDateTime Collaboration method 234
- greater_or_equal Collaboration method 158, 219
- greater_than Collaboration method 158, 219

I

- IBM WebSphere 278
- ICAN Suite 304
- Icons
 - JCE Business Rules Designer toolbar 151
 - JCE Business Rules toolbar 147
 - JCE toolbar 144
 - XCE toolbar 216
- id Collaboration method 229
- If-Then command 147
- impact analyzer
 - overview 67

Import a Local File command 144
Import JAR File command 145
Import Static Field command 151
Import XSL from a Local File command 216
indexOf Collaboration method 166
instanceOf Collaboration method 162
Integration Server 304
Interfaces
 Enterprise Manager 34

J

Java Collaboration
 definitions 132
 Editor (JCE) 143
 method box 155
 method palette 155
 Wizard 133
Java Collaboration methods
 addition 160
 AND 157
 arrayAccess 168
 arrayAssign 169
 arrayLength 169
 cast 162
 charAt 165
 concat 165
 division 160
 endsWith 165
 equal 158
 equals 162
 greater_or_equal 158
 greater_than 158
 indexOf 166
 instanceOf 162
 length 166
 lesser_or_equal 159
 lesser_than 159
 multiplication 160
 NOT 157
 not_equal 159
 OR 157
 remainder 161
 replace 166
 substring 166
 subtraction 161
 toLowerCase 166
 toString 162
 toUpperCase 166
 trim 167
Java Collaboration operators
 bitNot 170
 negative 170
 positive 170

 postDecrement 170
 postIncrement 171
 preDecrement 171
 preIncrement 171
JCE commands
 Advanced Mode 144
 Auto Layout 154
 Break 147
 Business Rules on Left 144
 Business Rules on Top 144
 Call Java Method 152
 Call New Constructor 153
 Collapse All Methods 154
 Commit Changes 144
 Continue 147
 Create Literal 152
 Expand All Methods 154
 Export Java Rule 144
 Field 147
 Find 145
 Find Next 145
 Find Previous 145
 For 147
 If-Then 147
 Import a Local File 144
 Import JAR File 145
 Import Static Field 151
 Local Variable 147
 Method 147
 New Array 153
 Redo 145
 Refresh Collaboration 145
 Replace 145
 Return 147
 Roll Back Changes 144
 Rule 147
 Source Code Mode 144
 Standard Mode 144
 Throw 147
 Try 147
 Undo 145
 Validate 144
 While 147
JMS IQ Manager 304
join 305

K

key Collaboration method 229

L

language Collaboration method 227
last Collaboration method 229

Index

length Collaboration method 166
lesser_or_equal Collaboration method 159, 219
lesser_than Collaboration method 159, 219
link 305
linked message destination 305
Local Variable command 147
local-name Collaboration method 229
Logical Host 28, 305

M

Management Agent 305
map variables 274
Map Variables button 268
menu bar 46
message
 destination 305
metadata 305
method box
 Java Collaboration 155
 XSLT Collaboration 217
Method command 147
method palette
 Java Collaboration 155
 XSLT Collaboration 217
Monitor
 Enterprise 37
 SRE 38
multiplication Collaboration method 160, 219

N

name Collaboration method 229
namespace-uri Collaboration method 229
negative Collaboration method 220
negative operator 170
New Array command 153
non-normalized data 305
normalize_space Collaboration method 221
normalized data 305
NOT Collaboration method 157, 227
not_equal Collaboration method 159, 220
number Collaboration method 224
number-literal Collaboration method 225

O

Object Type Definition 306
 wizard 94, 100, 104
Object Type Definition (OTD) 89
OLAP 305
Open File command 126
OR Collaboration method 157, 220

OTD 306
 Editor 125
 tester 128
OTD Editor commands
 Open File 126
 Refresh OTD 126
 Run Tester 126
 Save as New Name 126
 Save File 126
 Sort by Name 126
 Tester 126
 Toggle Reference Tab Panel 126
OTD Wizard
 DTD 93
 User-Defined 108
 WSDL 98
 XSD 103

P

palette
 Java Collaboration methods 155
 XSLT Collaboration methods 217
parseDate Collaboration method 236
parseDateTime Collaboration method 236
Participating Host 308
position Collaboration method 230
positive operator 170
postDecrement operator 170
postIncrement operator 171
preDecrement operator 171
preIncrement operator 171
Profile, Deployment 267
Project 306
Project Explorer 49, 73

Q

query 306
queue 306

R

raw data 306
rdbm 306
Redo command 145
Refresh Collaboration command 145
Refresh OTD command 126
Registry 308
relational database 306
remainder Collaboration method 161, 220
replace Collaboration method 166
Replace command 145

Repository 28, 306, 308
 Return command 147
 Roll Back Changes command 144
 Roll Back Code Changes command 216
 round Collaboration method 225
 Rule command 147
 Run Tester command 126

S

Save as New Name command 126
 Save File command 126
 Save XSL to a Local File command 216
 Scheduler 82
 Schema 308
 Schema Runtime Environment 307
 Schema Runtime Environment (SRE) 38
 security 251
 Security Server 307
 setCommonID Collaboration method 233
 Show Mapping Only command 216
 Show Maps and Code command 216
 Show XSLT Code Only command 216
 Sort by Name command 126
 Source Code Mode command 144
 SRE 307
 Monitor 38
 staging data 307
 Standard Mode command 144
 starting
 Enterprise Designer 45
 Enterprise Manager 33
 starts_with Collaboration method 227
 string Collaboration method 222
 string_length Collaboration method 222
 string_literal Collaboration method 222
 subproject 307
 substring Collaboration method 166, 222
 substring-after Collaboration method 222
 substring-before Collaboration method 222
 subtraction Collaboration method 161, 220
 sum Collaboration method 225
 supporting documents 22
 system-property Collaboration method 223

T

table 307
 terminology cross-reference 308
 tester
 OTD 128
 Tester command 126
 Throw command 147
 Toggle Reference Tab Panel command 126

toLowerCase Collaboration method 166
 topic 307
 toString Collaboration method 162
 toUpperCase Collaboration method 166
 transformation 307
 translate Collaboration method 223
 trim Collaboration method 167
 true Collaboration method 227
 Try command 147

U

UANExtension 217
 UANExtension method
 concat-sequence-format 233
 currentDateTime 233
 formatDate 233
 formatDateTime 234
 formatMessage 232
 getAppID 232
 getAppValue 232
 getCenturyFromDate 234
 getCenturyFromDateTime 234
 getCommonID 232
 getCommonValue 233
 getDayFromDate 234
 getDayFromDateTime 235
 getHourFromDateTime 235
 getHourFromTime 235
 getMinutesFromDateTime 235
 getMinutesFromTime 235
 getMonthFromDate 234
 getMonthFromDateTime 234
 getSecondsFromDateTime 235
 getSecondsFromTime 235
 getTimezoneFromDate 235
 getTimezoneFromDateTime 236
 getTimezoneFromTime 236
 getYearFromDate 234
 getYearFromDateTime 234
 parseDate 236
 parseDateTime 236
 setCommonID 233
 Undo command 145
 unparsed-entity-uri Collaboration method 223
 User-Defined OTD Wizard 108

V

Validate command 144
 variables
 mapping 274
 Version Control
 Create Diff 172, 238

Index

Generate Diff 172, 238
Merge Diff 173, 239
version control 69

W

warehouse 307
WebLogic 275
WebSphere 278
While command 147
Wizard
 Java Collaboration 133
 OTD (DTD) 93
 OTD (User-Defined) 108
 OTD (WSDL) 98
 OTD (XSD) 103
 XSLT Collaboration 209
WSDL Wizard 98

X

XCE commands
 Commit Code Changes 216
 Import XSL from a Local File 216
 Roll Back Code Changes 216
 Save XSL to a Local File 216
 Show Mapping Only 216
 Show Maps and Code 216
 Show XSLT Code Only 216
XSD Wizard 103
XSLT 307
XSLT Collaboration
 definitions 208
 Editor 215
 method box 217
 method palette 217
 Wizard 209
XSLT Collaboration methods
 addition 218
 AND 218
 boolean 226
 ceiling 224
 concat 221
 concat-sequence-format 233
 contains 226
 count 228
 current 228
 currentDateTime 233
 division 218
 document 228
 element-available 226
 equal 219
 false 227
 floor 224

format_number 221
formatDate 233
formatDateTime 234
formatMessage 232
function_available 227
generate-id 229
getAppID 232
getAppValue 232
getCenturyFromDate 234
getCenturyFromDateTime 234
getCommonID 232
getCommonValue 233
getDayFromDate 234
getDayFromDateTime 235
getHourFromDateTime 235
getHourFromTime 235
getMinutesFromDateTime 235
getMinutesFromTime 235
getMonthFromDate 234
getMonthFromDateTime 234
getSecondsFromDateTime 235
getSecondsFromTime 235
getTimezoneFromDate 235
getTimezoneFromDateTime 236
getTimezoneFromTime 236
getYearFromDate 234
getYearFromDateTime 234
greater_or_equal 219
greater_than 219
id 229
key 229
language 227
last 229
lesser_or_equal 219
lesser_than 219
local-name 229
multiplication 219
name 229
namespace-uri 229
negative 220
normalize_space 221
NOT 227
not_equal 220
number 224
number-literal 225
OR 220
parseDate 236
parseDateTime 236
position 230
remainder 220
round 225
setCommonID 233
starts_with 227
string 222

Index

string_length 222
string_literal 222
substring 222
substring-after 222
substring-before 222
subtraction 220
sum 225
system-property 223
translate 223
true 227
unparsed-entity-uri 223