



Composer 8.1

Routing Applications

User's Guide

The information contained herein is proprietary and confidential and cannot be disclosed or duplicated without the prior written consent of Genesys Telecommunications Laboratories, Inc.

Copyright © 2009–2013 Genesys Telecommunications Laboratories, Inc. All rights reserved.

About Genesys

Genesys is the world's leading provider of customer service and contact center software—with more than 4,000 customers in 80 countries. Drawing on its more than 20 years of customer service innovation and experience, Genesys is uniquely positioned to help companies bring their people, insights and customer channels together to effectively drive today's customer conversation. Genesys software directs more than 100 million interactions every day, maximizing the value of customer engagement and differentiating the experience by driving personalization and multi-channel customer service—and extending customer service across the enterprise to optimize processes and the performance of customer-facing employees. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Customer Care website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

Although reasonable effort is made to ensure that the information in this document is complete and accurate at the time of release, Genesys Telecommunications Laboratories, Inc., cannot assume responsibility for any existing errors. Changes and/or corrections to the information contained in this document may be incorporated in future versions.

Your Responsibility for Your System's Security

You are responsible for the security of your system. Product administration to prevent unauthorized use is your responsibility. Your system administrator should read all documents provided with this product to fully understand the features available that reduce your risk of incurring charges for unlicensed use of Genesys products.

Trademarks

Genesys and the Genesys logo are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other company names and logos may be trademarks or registered trademarks of their respective holders. © 2012 Genesys Telecommunications Laboratories, Inc. All rights reserved.

The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

If you have purchased support from a value-added reseller (VAR), please contact the VAR for technical support.

Customer Care from Genesys

If you have purchased support directly from Genesys, please contact [Genesys Customer Care](#). Before contacting Customer Care, please refer to the [Genesys Care Program Guide](#) for complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the [Genesys Licensing Guide](#).

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 81composer_us-ra_07-2013_v8.1.301.00



Table of Contents

List of Procedures	7
Preface	9
About Composer.....	9
Intended Audience.....	10
Making Comments on This Document	11
Contacting Genesys Customer Care.....	11
Document Change History	11
New in Document Version 8.1.301.00	11
New in Document Version 8.1.201.00	12
New in Document Version 8.1.101.00	12
New in Document Version 8.1.001.00	12
Chapter 1	SCXML-Based Routing Applications..... 15
What is Routing?	15
Routing Applications	16
Routing Application Structure	16
Interaction Process Diagrams.....	16
Workflow Diagram-Building Blocks.....	17
Application Element Relationships	18
Moving Interactions Between IPDs.....	18
Projects.....	19
Project Workspace.....	20
Project Folders.....	20
SCML Code Editor.....	22
Chapter 2	Composer GUI 25
Diagram Editor.....	26
IPD in Canvas Area	26
Properties View.....	27
Resource Property Points to Workflow Diagram.....	28
Viewing/Defining Block Properties.....	29

	Exception/Error Handling.....	31
	Variables: Project and Application	34
	Workflow (Application) Variables	34
	Project Variables	34
	Expression Builder.....	35
	ECMAScript Expressions.....	36
	Orchestration Server Functions	37
	Statistics Manager/Builder	38
	List Objects Manager/Builder	39
	Skill Expression Builder	40
	Toolbar Buttons for Routing	41
	Menu Bar	41
	Preferences	42
Chapter 3	Summary of Tasks.....	43
	Tasks: Planning & Preparation	43
	Task: Creating a New Project	44
	Tasks: Creating the Workflow Diagram(s)	45
	Tasks: Creating the IPD	45
	Tasks: Code Generation, Testing, and Deployment	47
Chapter 4	Planning & Preparation.....	49
	Planning the Design	49
	Stages in Multimedia Processing	49
	One or Multiple IPDs?.....	51
	Summary of Planning/Preparation Process	51
	Installing the Required Software.....	52
	Preconfiguring Database Objects	52
	eService Objects.....	52
	Business Attributes	53
	Using Context Services	53
	Mapping Context Services Attributes.....	54
	Managing/Orchestrating Conversations	55
	Reviewing the Samples	56
	Enabling/Disabling Functionality.....	58
Chapter 5	Creating a Routing Application.....	61
	Routing After Sending Auto-response Project Template	62
	High-Level Interaction Flow	63
	Creating a New Project.....	64
	Connecting to Configuration Server	66

Defining Contact Services Preferences	68
Entering IPD Properties	69
Adding a Media Server Block	70
Endpoints	70
Defining an Interaction Queue	73
Connecting the Blocks	77
Creating a Workflow Diagram	78
Adding a Workflow Block	81
Workflow-Generated Blocks	83
Connecting the Blocks	83
Adding the Remaining IPD Blocks	84
Defining Second Interaction Queue	84
Adding a Second Workflow Block	86
Publishing an IPD	87
Generating Code	89
Testing a Routing Application	89
Deploying the Application	93
 Chapter 6	
Working with Blocks and Diagrams	95
Working with Blocks	95
Adding Blocks to the Canvas	95
Connecting Blocks	96
Perspectives	97
Changing Perspectives	98
Customizing a Perspective	99
Viewing More Than One Diagram	102
Viewing Properties for Two Blocks	104
Saving Workflow Diagrams as Templates	105
Custom Blocks	109
Using a Custom Block	109
 Appendix	
Composer Blocks	113
IPD Blocks	113
Workflow Diagram-Building Blocks	114
Flow Control Blocks	114
Routing Blocks	116
Voice Treatment Blocks	117
eServices (Multimedia) Blocks	117
Server-Side Blocks	119
Context Services Blocks	121
Outbound Blocks	122

	Composer Blocks Mapped to IRD Objects	122
	Other Functionality	126
Supplements	Related Documentation Resources	129
	Document Conventions	134
Index	137



List of Procedures

Reviewing sample interaction process diagrams	56
Reviewing sample routing projects.	56
Enabling/Disabling Functionality.	58
Creating a new Project	64
Connecting to Configuration Server	67
Connecting to the Context Services Server	68
Entering IPD properties	70
Adding a Media Server Block	72
Defining an Interaction Queue	74
Defining a view for an interaction queue	75
Creating a workflow diagram	78
Adding a Workflow block	82
Publishing an interaction process diagram.	88
Generating code for an interaction process diagram	89
Using the ORS Debugger.	89
Using the Output Link to connect blocks	96
Viewing Multiple Diagrams	102
Displaying Properties for Two Blocks.	104
Saving a workflow diagram as a template	105
Accessing Saved Templates	105
Exporting a Diagram Template to the File System.	106
Importing a Diagram Saved as a Template	108
Creating a custom block.	109
Changing Existing Custom Blocks	110
Deleting a Custom Block	111
Hiding the Custom category	111
Import/Export of Custom Blocks.	112



Preface

Welcome to the *Composer 8.1 Routing Applications User's Guide*. Composer is an Integrated Development Environment used to develop applications for both Genesys Voice Platform (GVP) and the Genesys Orchestration Platform. This guide focuses on creating applications for the Orchestration Platform.

This document is valid only for the 8.1.3 release of this product, which can be installed on Windows or Mac. All screen shots in this guide reflect Composer installed on Windows.

Note: The 8.0.1 version of this product was known as Composer Voice, as it was used only to develop applications for GVP. For versions of this document created for other releases of this product, visit the Genesys Customer Care website, or request the Documentation Library DVD, which you can order by e-mail from Genesys Order Management at orderman@genesyslab.com.

This preface contains the following sections:

- [About Composer, page 9](#)
- [Intended Audience, page 10](#)
- [Making Comments on This Document, page 11](#)
- [Contacting Genesys Customer Care, page 11](#)
- [Document Change History, page 11](#)

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on [page 134](#).

About Composer

An Eclipse-based (www.eclipse.org) Integrated Development Environment, Composer provides for drag-and-drop environment for developing:

- Voice applications for Genesys Voice Platform (GVP)—a software suite that unifies voice and web technologies to provide a complete solution for customer self-service or assisted service.

- Routing applications for the Genesys Orchestration Platform—Orchestration Server (ORS) is responsible for executing orchestration logic (SCXML) that is provided by an Application Server. Composer routing applications are deployed on an Application Server. The responsibility of the Universal Routing Server (URS) within the Orchestration Platform is to provide a necessary service to Orchestration Server, to support routing.

Note: This guide focuses on the creation of applications for the Genesys Orchestration Platform.

Intended Audience

This document is intended for routing application developers, both technical and non-technical. Its primary goal is to introduce you to the Composer interface and the process of building routing applications. The information presented here is at a high level to facilitate a conceptual understanding. The *Composer 8.1 Help* provides the necessary detail.

This document has been written with the assumption that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications
- Network design and operation
- Your own network configurations
- Genesys Framework architecture
- Genesys Universal Routing

Familiarity with State Chart Extensible Markup Language (SCXML) is a plus.

Note: Composer provides a wide range of tools to satisfy the needs of a diverse end user population. This guide assumes you are a non-technical user who does not write SCXML code who wishes to build routing applications by using Composer's diagram editor.

If you have already used Universal Routing's Interaction Routing Designer (IRD) to build routing strategies, you will find similarities with Composer. While IRD uses a proprietary language to build routing strategies, Composer is an open system that uses SCXML to build routing strategies, which it calls *routing workflows*.

Making Comments on This Document

If you especially like or dislike anything about this document, feel free to e-mail your comments to Techpubs.webadmin@genesyslab.com.

You can comment on what you regard as specific errors or omissions, and on the accuracy, organization, subject matter, or completeness of this document. Please limit your comments to the scope of this document only and to the way in which the information is presented. Contact your Genesys Account Representative or Genesys Customer Care if you have suggestions about the product itself.

When you send us comments, you grant Genesys a nonexclusive right to use or distribute your comments in any way it believes appropriate, without incurring any obligation to you.

Contacting Genesys Customer Care

If you have purchased support directly from Genesys, please contact [Genesys Customer Care](#).

Before contacting Customer Care, please refer to the [Genesys Care Program Guide](#) for complete contact information and procedures.

Document Change History

This section lists topics that are new or have changed significantly since the 8.0 release of this document.

New in Document Version 8.1.301.00

The following changes have been made since the 8.1.200.00 edition of this guide:

Updated Figure 21 on [page 41](#) to reflect toolbar changes when installing Composer as a plugin into an Eclipse environment.

Added TLib block to Table 7 on [page 120](#).

Added Attach, Detach, Raise Event, Cancel Event blocks to Table 3 on [page 114](#).

Added “IRD to Composer Migration Guide Wiki” on [page 129](#).

New in Document Version 8.1.201.00

The following changes have been made since the 8.1.100.00 edition of this guide:

- Replaced screenshots to reflect 8.12.
- Added folders to “Project Folders” on [page 20](#).
- Changed “Genesys Functional Modules” to “Orchestration Server Functions” throughout.
- Added references to the [Composer Help Wiki 8.1.3](#) and the [Orchestration Server wiki](#) in place of Composer Help.
- Added Begin Parallel, End Parallel and Wait blocks to Table 3, “Flow Control Blocks,” on [page 114](#).
- Added OPM block to Table 7, “Server-Side Blocks,” on [page 120](#).
- Added IRD Outbound Category to Table 17 on [page 126](#).

New in This Release

The following new section was added since the 8.1.101.00 edition of this guide:

- Added Table 9, “Outbound Blocks,” on [page 122](#).

New in Document Version 8.1.101.00

The following changes have been made since the 8.1.001.00 edition of this guide:

- Replaced screenshots to reflect 8.11.
- Added Flow Control blocks to “IPD Blocks” on [page 17](#).
- Added debugging-results folder to “Project Folders” on [page 20](#).
- Changed screenshot to show ORS Debugger on Figure 22 on [page 42](#).
- Added Flow Control blocks to Table 3 on [page 114](#).

New in This Release

The following new section was added since the 8.1.001.00 edition of this guide:

- Added “Testing a Routing Application” on [page 89](#).

New in Document Version 8.1.001.00

The following has changes have been made since the 8.0.401.00 edition of this guide:

- The “About Composer” on [page 9](#) now refers to the Orchestration Platform instead of the Universal Routing Platform.
- Replaced Application Variables dialog box in Figure 14 on [page 34](#).
- Replaced Expression Builder screenshot in Figure 15 on [page 35](#).
- Replaced *Genesys 8.0 SCXML Technical Reference* with link to Orchestration Server wiki on [page 37](#).
- Replaced Composer Preferences dialog box in Figure 22 on [page 42](#).
- Changed Genesys software component version numbers under “Installing the Required Software” on [page 52](#).
- Replaced the Project Templates list in Figure 24 on [page 57](#).
- Removed Appendix B, Genesys Functional Modules, as more up-to-date information can be found in the [Orchestration Server wiki](#).

New in This Release

The following new sections were added since the 8.0.401.00 edition of this guide:

- Added Workflow-generated blocks under “IPD Blocks” on [page 17](#).
- Updated section “Workflow Diagram-Building Blocks” on [page 114](#) with new blocks for routing applications added for 8.1: Email Response, Chat Transcript, Email Forward, Screen Interaction, Classify Interaction, Update Contact, Identify Contact, Create Interaction, Render Message, SCXML State, and User Data.



Chapter

1

SCXML-Based Routing Applications

This chapter introduces SCXML-based routing applications created in Composer. It introduces the concept of routing customer interactions, such as phone calls or e-mails, to targets, such as agents or agent groups, with the skills to handle those interactions. It also introduces interaction processing diagrams, routing workflows, the workflow diagram-building blocks, Composer Projects, and the SCXML code editor.

This chapter contains the following sections:

- [What is Routing?, page 15](#)
- [Routing Application Structure, page 16](#)
- [Moving Interactions Between IPDs, page 18](#)
- [Projects, page 19](#)
- [SCML Code Editor, page 22](#)

What is Routing?

From a Genesys standpoint, routing is the process of sending an interaction to a target. For example, routing an incoming telephone call from a customer requesting information on Product A to an agent knowledgeable about Product A. In order for such an interaction to reach the appropriate target, the interaction must undergo various types of processing between the time it arrives at the contact center and the selection and use of the appropriate target. You specify the various types of processing that must occur through a *routing application*, which, in the case of Composer, is deployed on an web application server.

Routing Applications

A Composer routing application is comprised of one or more *routing workflows*, which are similar to routing strategies created in Universal Routing's Interaction Routing Designer (IRD).

Using Composer, you can create routing applications using the following methods:

- By writing SCXML code in Composer's code editor or more easily, as described in this guide:
- By creating workflow diagrams, which require you to place, connect, and configure *blocks*.

When loaded on and triggered from a routing point, a routing workflow tells Universal Routing Server how to handle and where to direct interactions under different circumstances (a process which may involve other servers).

Routing Application Structure

At the top level of a Composer routing application is an interaction process diagram (IPD). Figure 6 on [page 26](#) shows an example IPD.

Interaction Process Diagrams

In summary, an interaction process diagram or IPD:

- Functions as the starting SCXML page for a routing application.
- Is automatically created when you start a new Project.
- Provides a high-level view of interaction processing flow (see “High-Level Interaction Flow” on [page 63](#) for an example).
- References media servers for incoming interactions.
- Processes those interactions by moving them through interaction queues, workflow strategies for specialized processing, and (optionally) workbins.

Similar to an IRD business process, an IPD defines what happens to customer interactions from the point of arrival at your contact center to the point of completion (usually in the form of a response to the customer).

IPD Blocks

An IPD can contain the following types of blocks:

- Interaction Queue blocks (as described in “Defining an Interaction Queue” on [page 73](#)).
- Media Server blocks, (as described in “Adding a Media Server Block” on [page 70](#)).
- Workflow blocks (as described in “Adding a Workflow Block” on [page 81](#)) referencing a workflow strategy.
- Workbin blocks referencing a temporary storage place for interactions on the agent’s desktop.
- Flow Control blocks (as described in Table 3 on [page 114](#)): Branching, ECMAScript, and Log to support multiple views (see [page 75](#)) per interaction queue.
- Workflow-generated blocks Outgoing connections automatically appearing from a Workflow block that represent objects specified inside the workflow.

When processing multimedia interactions, an IPD typically starts with a Media Server block and continues with one or more Interaction Queue, Workflow, and/or Workbin blocks. These IPD-building blocks are available on Composer’s **Palette** tab when an IPD is in focus (see Figure 6 on [page 26](#)).

Note: When routing voice interactions only, IPDs do not use Media Server and Interaction Queue blocks.

Workflow Diagram-Building Blocks

The workflow-diagram building blocks are the next level of blocks in an IPD. A Workflow block points to a workflow resource (see Figure 8 on [page 29](#)), which is either:

- a workflow diagram comprised of the diagram building blocks described in Appendix , “Composer Blocks,” on [page 113](#) or
- an SCXML file, which can be created in Composer’s code editor (see Figure 4 on [page 22](#)).

This document describes how to create routing applications comprised of IPDs that contain Workflow blocks pointing to workflow diagrams. The workflow diagram-building blocks are available on Composer’s **Palette** tab when a workflow diagram is in focus (see Figure 8 on [page 29](#)).

Workflow Block Categories

The **Palette** tab groups the workflow diagram-building blocks into the following categories:

- Flow Control
- Routing
- Voice Treatment
- Server-Side
- Context Services
- eServices
- Outbound

Appendix , “Composer Blocks,” on [page 113](#) summarizes the workflow diagram-building blocks in each of the above categories. The *Composer 8.1 Help* details the properties of each block, which are what you configure.

Application Element Relationships

[Figure 1](#) depicts the relationship of the various routing application elements discussed so far.

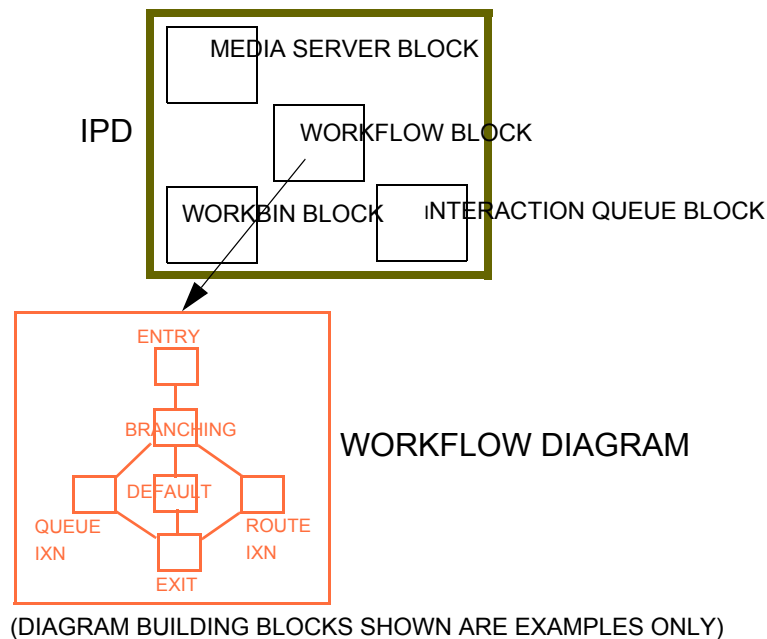


Figure 1: Workflow Block in IPD Pointing to Workflow Diagram

Moving Interactions Between IPDs

You cannot explicitly link IPDs together as there is no higher-level diagram in Composer that shows IPDs as single blocks and then allows you to interconnect them. You can, however, cause interactions to move from one IPD to another IPD through the use of *interaction queues*. This is similar to IRD’s method of linking business processes through interaction queues.

For example, assume IPD1 references Workflow1 and this workflow uses an Queue Interaction block. The block can be set to route the interaction to a Queue2 and Queue2 could exist in IPD2. Therefore IPD1 gets connected to IPD2.

Figure 2 illustrates this graphically.

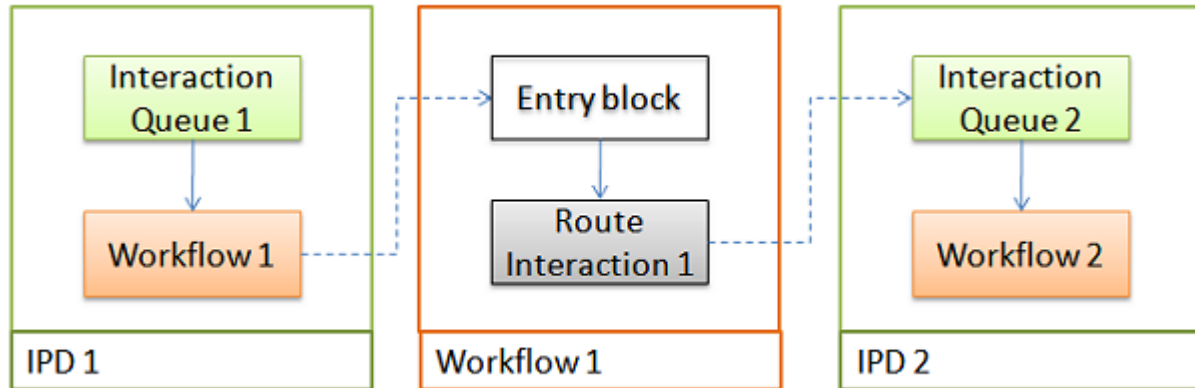


Figure 2: Moving Interactions Between IPDs

For more information, see the “Linking IPDs with Workflows” topic in the *Composer 8.1 Help*.

Projects

To organize all the routing application elements just discussed, Composer uses a *Project* to contain everything related to a single routing application. A *Project Explorer* on the upper left of the Composer window contains all the Projects in your workspace (see Figure 3).

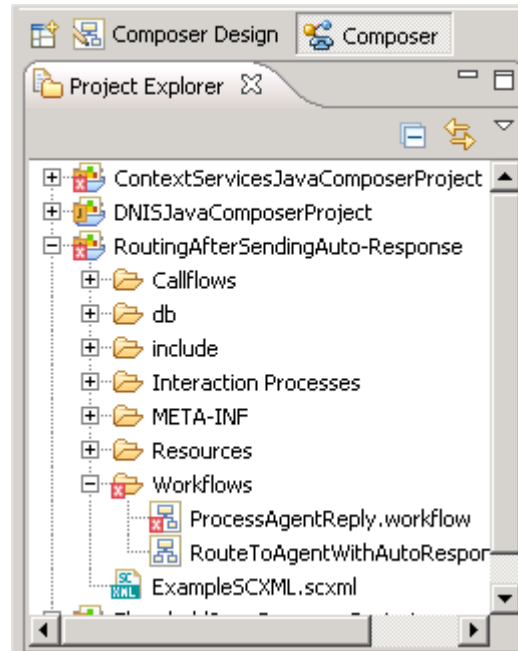


Figure 3: Project Explorer

Note: As described in [Procedure: Creating a new Project](#), on [page 64](#), when you create a new Project, Composer automatically creates a new IPD with the name `default.ixprocess`, which you can change (see [Figure 30](#) on [page 66](#)).

Project Workspace

Within Composer, the term *workspace* refers to a location (folder) for your Projects and files in addition to any special folders that Eclipse needs to maintain for its internal bookkeeping. The dialog box that appears when you first start Composer gives the option of changing the workspace to a different location. New Projects created in Composer will be created under this workspace as subfolders.

Project Folders

A Route or Integrated Voice and Route type Java Composer Project (see [Figure 27](#) on [page 64](#)) will contain some or all of these sub-folders depending on the type of Project:

- **callflows** – Folder for storing all the callflow diagrams (`.callflow` files) as Composer allows you to create integrated Voice and Route Projects.

Note: Callflow diagrams are associated with VXML-based applications created for Genesys Voice Platform (GVP), as described in the *Composer 8.0.x Help*.

- **db** – Database connection properties and `.sql` files are stored here.
- **debugging-results** – Folder for ORS debugging information when debugging routing applications.
- **include** – Composer-provided standard include files used by Backend logic blocks.
- **lib** – Folder for external dependency libraries such as JAR files.
- **META-INF** – Created when you create a new Java Composer Project. It is needed for Java and is included when a `.war` file is exported from Composer. Do not make changes to this directory.
- **WEB-INF/lib** – Java Composer Projects only. Folder for external dependency libraries such as JAR files.
- **Interaction Processes** – Folder for storing all the interaction process diagrams (`.ixnprocess` files).
- **upgradeReports** – When migrating IRD strategies into Composer, folder for migration reports. Also used for reports as result of upgrading Projects and diagrams.
- **Resources** – Folder for the audio and grammar resources. Used only for VXML voice applications as described below.
- **Workflows** – Folder for storing all the workflow diagrams (`.workflow` files).
- **Scripts** – Folder for user-written ECMAScript.
- **src-gen** – Folder for the code generated SCXML or VXML files. This folder is also used to store any hand-coded VXML or SCXML files that may be part of a Project.
- **src** – Folder for custom code such as backend logic pages written by the user.

SCML Code Editor

For those who prefer to create routing applications by writing their own SCXML code, Composer provides a rich editor with use case templates.

Figure 4 shows example SCXML code in the Source tab of the code editor.

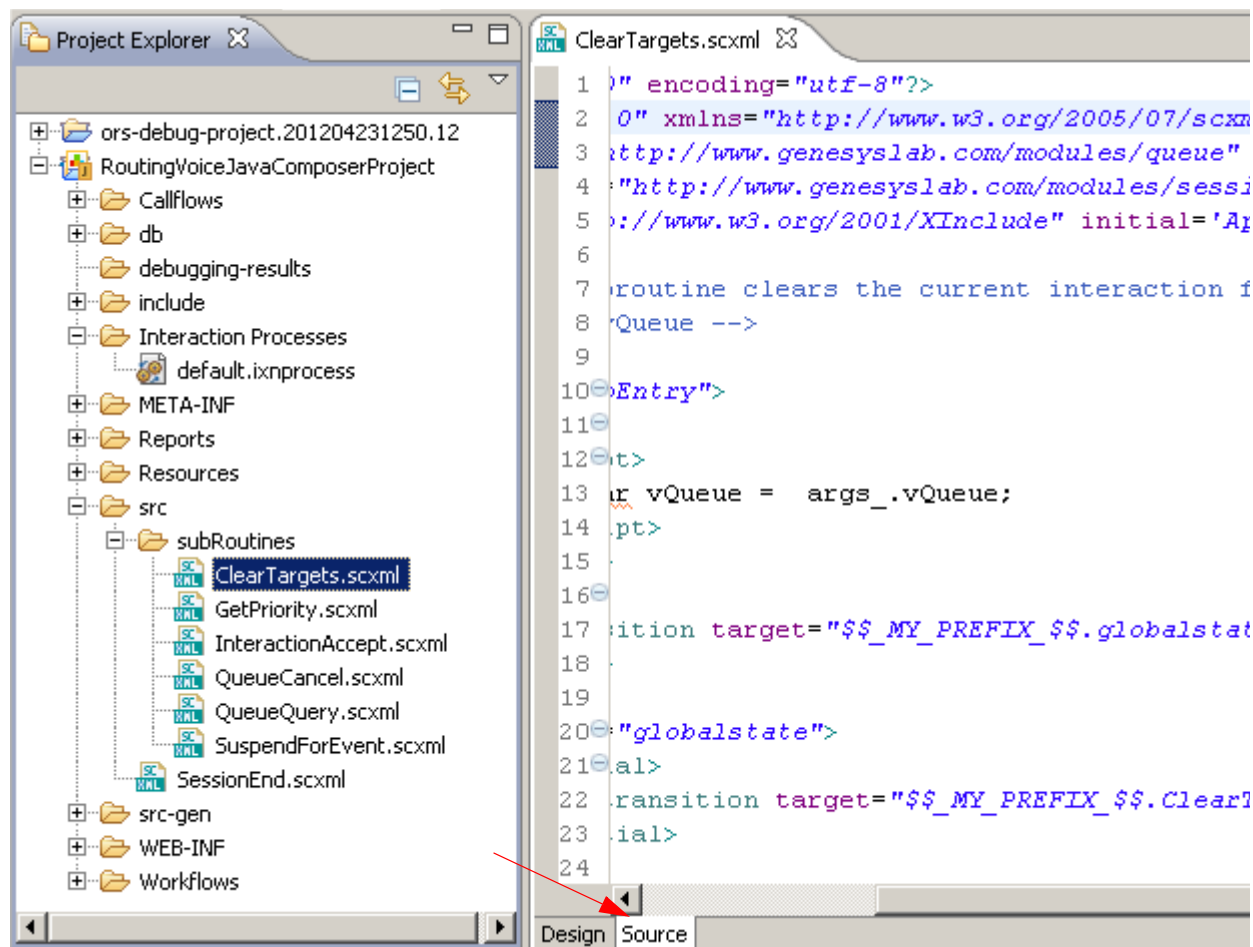


Figure 4: Composer SCXML Code Editor, Source View

Figure 5 shows the Design tab of the code editor.

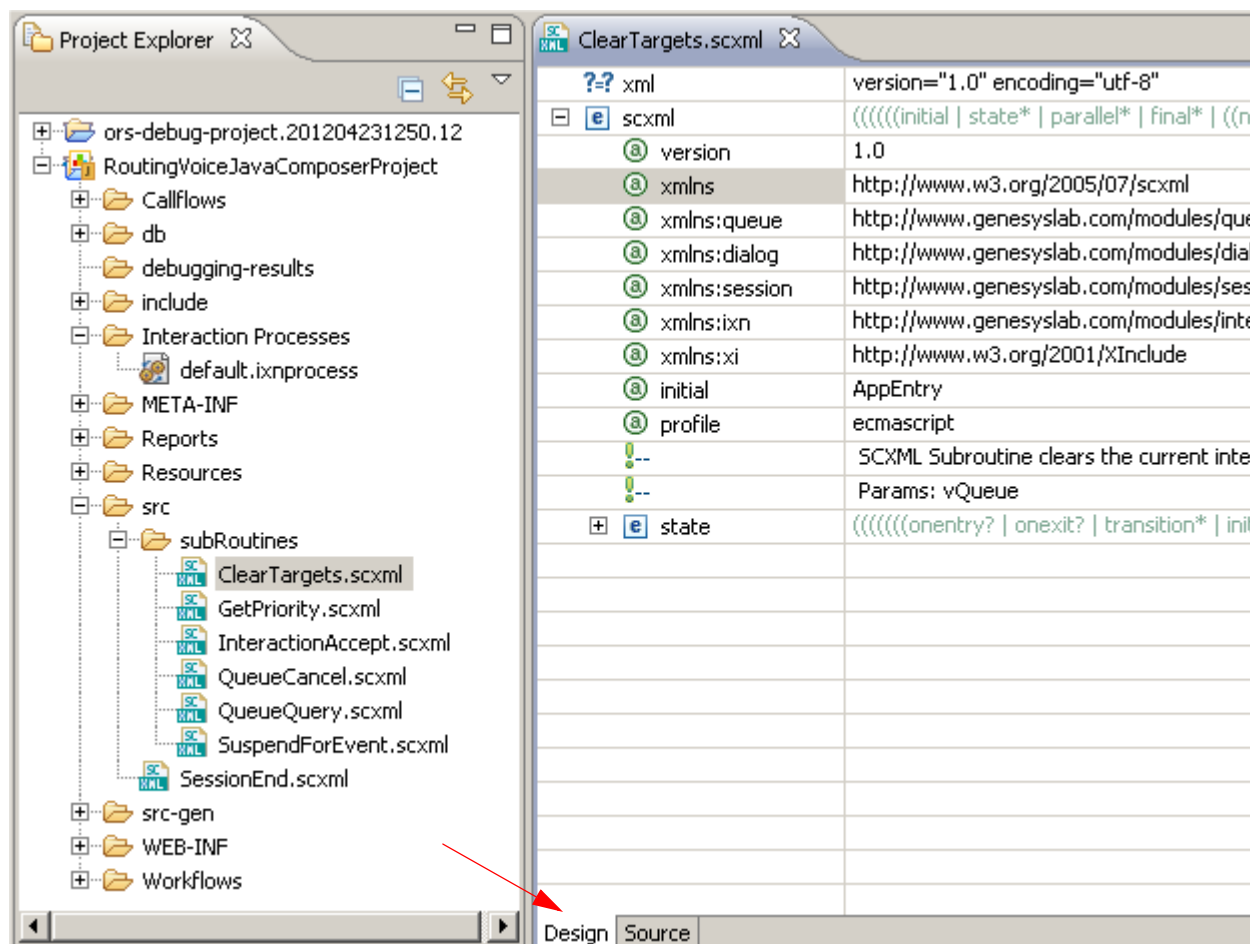


Figure 5: Composer SCXML Code Editor, Design View

You can view and work directly with source code using standard Eclipse text editing features. Features include:

- Smart double-clicking behavior.
- Context-assisted help when typing tags. Also context-assisted help for attributes of a tag upon pressing Space inside a tag.
- New SCXML documents are created with <scxml> as the top level element with the corresponding schema and namespace specifications.
- Ability to edit tag attribute values from the Properties view.
- Basic editor actions are supported: Cut, Copy, Paste, Save, Save as, Undo, Redo, Search and Replace.
- Syntax highlighting.
- Show and hide Line numbers.
- Add/Remove Bookmark and To-Do markers.
- Task tag feature to auto scan To-Do comments in the code.
- Comparing and reverting to local file history.
- Spell checking by showing yellow squiggly line markers.



Chapter

2

Composer GUI

This chapter introduces the main parts of the Composer GUI used for building routing applications. This contains the following sections:

- [Diagram Editor, page 26](#)
- [Exception/Error Handling, page 31](#)
- [Variables: Project and Application, page 34](#)
- [Expression Builder, page 35](#)
- [Statistics Manager/Builder, page 38](#)
- [List Objects Manager/Builder, page 39](#)
- [Skill Expression Builder, page 40](#)
- [Toolbar Buttons for Routing, page 41](#)
- [Menu Bar, page 41](#)
- [Preferences, page 42](#)

Diagram Editor

This section introduces the Composer’s diagram editor, which you use when creating workflow diagrams.

IPD in Canvas Area

Figure 6 shows an example interaction process diagram (as described on page 16) in Composer’s central *canvas* area. Note the Media Server, Workflow, and Interaction Queue blocks that comprise the IPD,

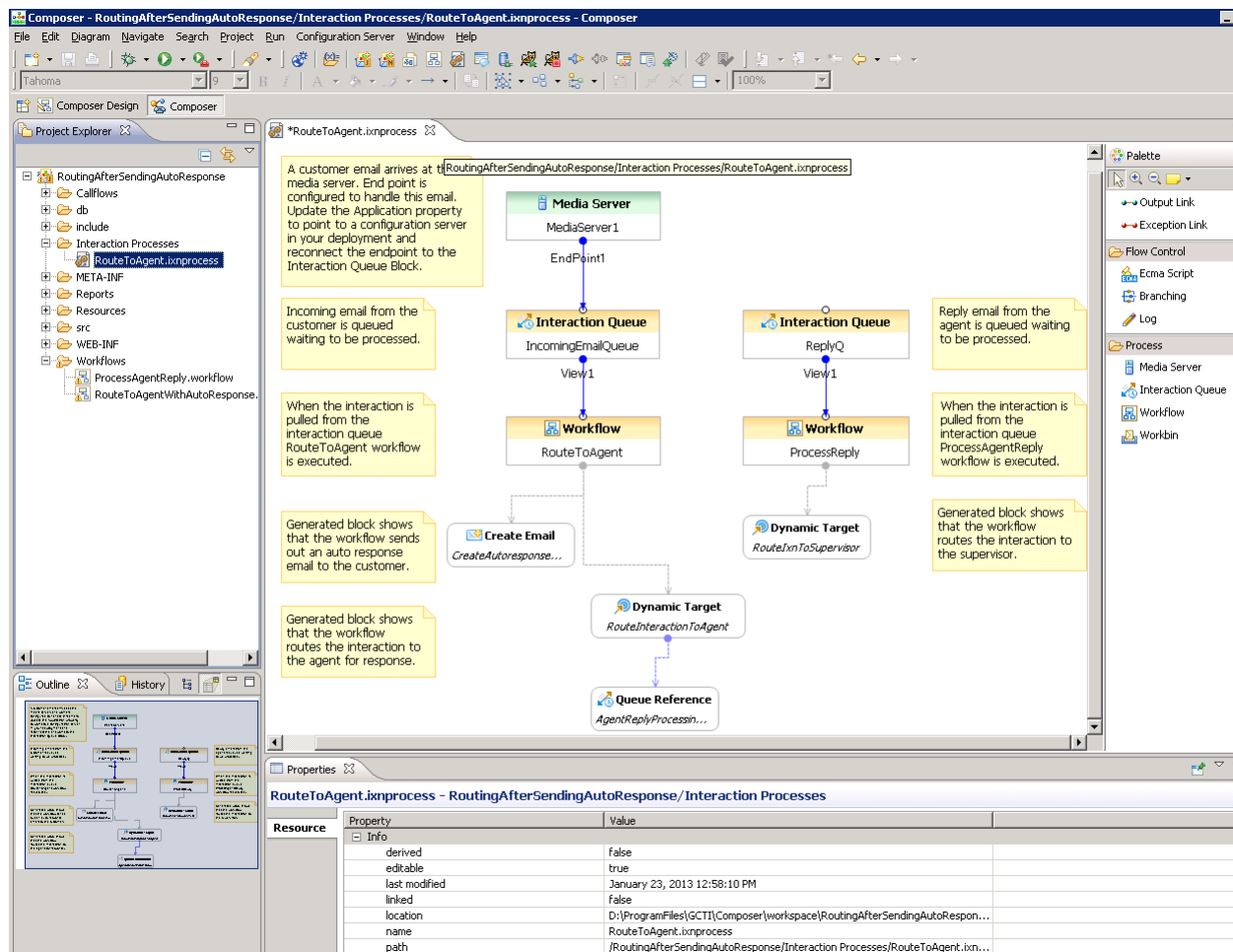


Figure 6: Example Interaction Process Diagram

The white blocks are “Workflow-Generated Blocks” as described on page 83.

Note: *Canvas* is part of the Eclipse standard terminology (see “About Composer” on page 9).

Note the following in [Figure 6](#):

- The name of the IPD appears in the tab above the canvas. In this example, the name is `RouteToAgent_Autoresponse.iprocess`.
- On the left, a `Project Explorer` contains all the files associated with the routing application.

Note: Genesys recommends grouping a single application under a single Project as the best practice and not grouping multiple applications. When deploying a Composer Project to an application server, the whole Project is deployed as a single `*.war` file.

- Underneath the `Project Explorer` is the `Outline` view, which is useful as a point of reference when working with large workflows.
- On the right is a *palette* of blocks.
 - When an IPD is in focus, the palette contains the IPD blocks described on [page 17](#): `Media Server`, `Interaction Queue`, `Workflow` and `Workbin`. The IPD may also show workflow-generated blocks.
 - When a workflow diagram is in focus, the palette contains the workflow diagram-building blocks described in Appendix , “Composer Blocks,” on [page 113](#). [Figure 8](#) on [page 29](#) shows an example.
- Underneath the canvas is the `Properties` view. It shows and gives access to the fields that can be configured for the selected block or diagram.
 - In [Figure 6](#) on [page 26](#), a block is not selected so the `Properties` view shows general fields for the diagram; in this case an interaction process diagram instead of a workflow diagram.

Properties View

Assume you click the `RouteToAgent Workflow` block in [Figure 6](#) on [page 26](#) so it is selected. The `Properties` view at the bottom now shows the fields for the selected Workflow block. (see [Figure 7](#)).

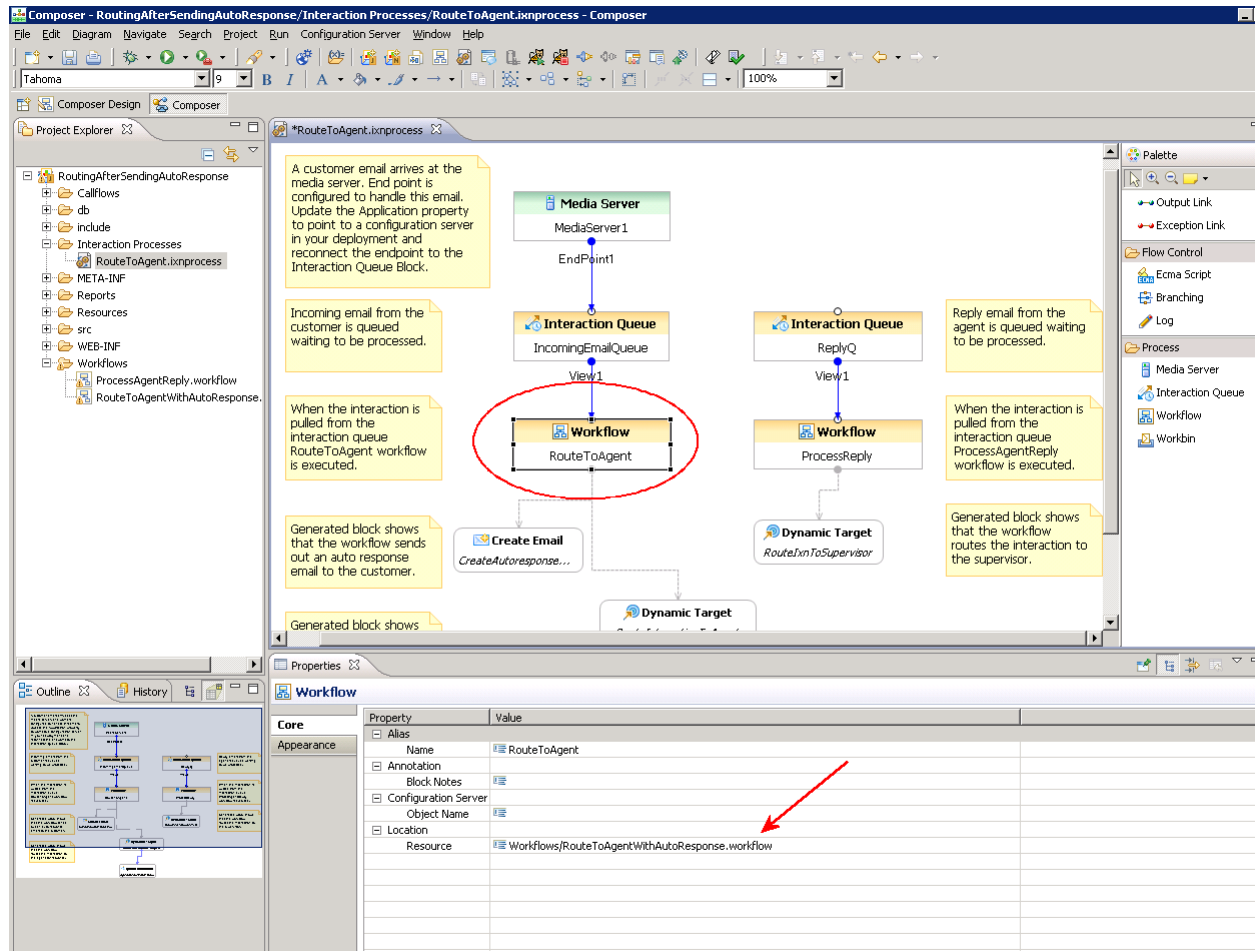


Figure 7: Workflow Block Selected in IPD

Resource Property Points to Workflow Diagram

The Properties view at the bottom of Figure 7 contains a Resource property, which points to a workflow diagram in the Workflows folder in the Project Explorer on the left. The name of the workflow diagram is `RouteToAgentWithAutoResponse.workflow`.

To view this workflow diagram on the canvas, double-click its name in the Project Explorer. The canvas now appears as shown in Figure 8 on page 29.

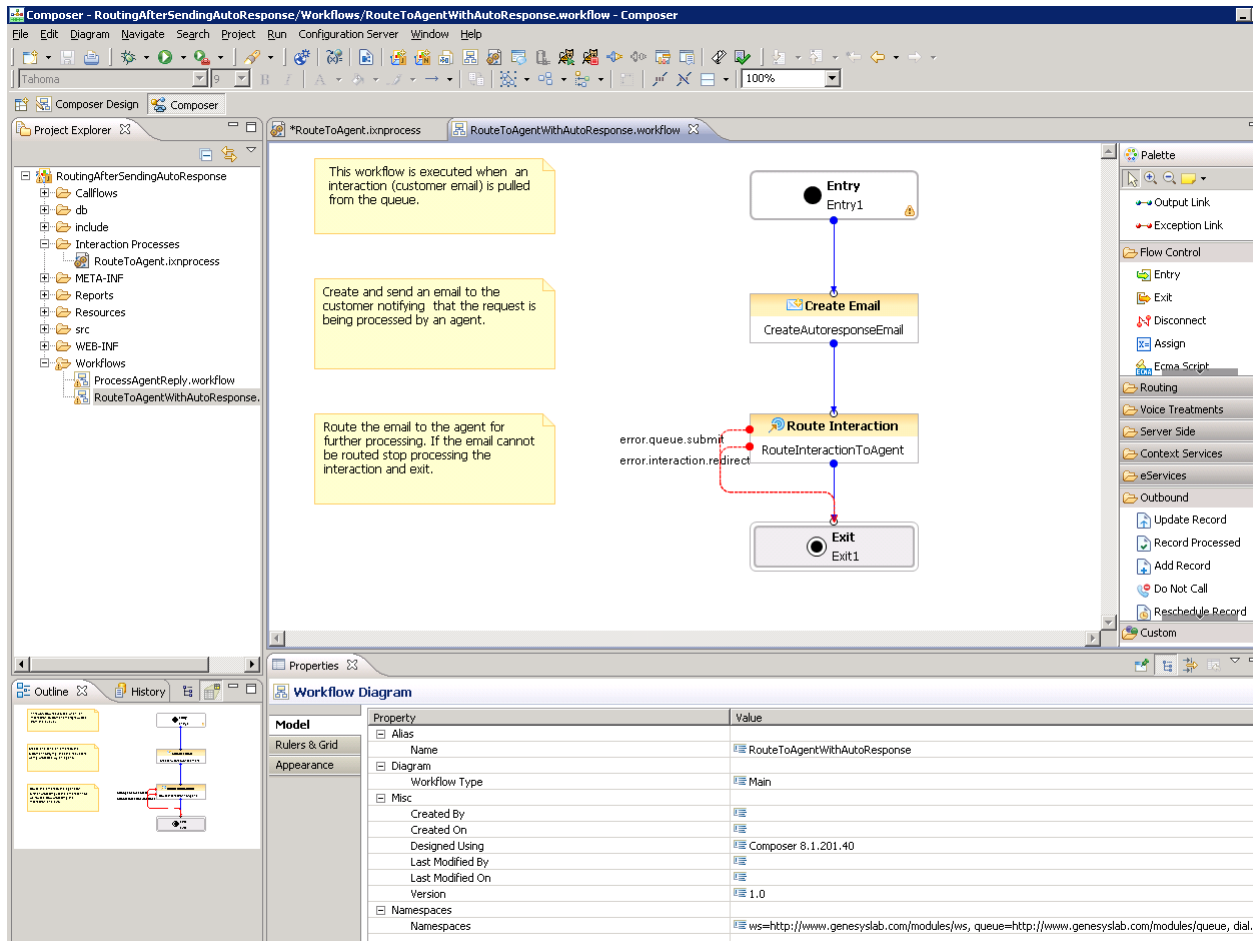


Figure 8: Workflow Diagram Selected in Project Explorer

Note in [Figure 8](#):

- The center canvas area adds a second tab for the workflow diagram: `RouteToAgentWithAutoResponse.workflow`.
- The palette on the right contains workflow diagram blocks grouped under the categories **Flow Control**, **Routing**, **Voice Treatment**, **Server Side**, **Context Services**, and **eServices**. Appendix , “Composer Blocks,” on [page 113](#) summarizes each block.
- Since no particular block is selected, the **Properties** view shows general properties for the workflow diagram.

Viewing/Defining Block Properties

Assume you double-click the `RouteInteractionToAgent` block in [Figure 8](#). The **Properties** view shows the properties (fields) that can be configured for the selected block (see [Figure 9](#)).

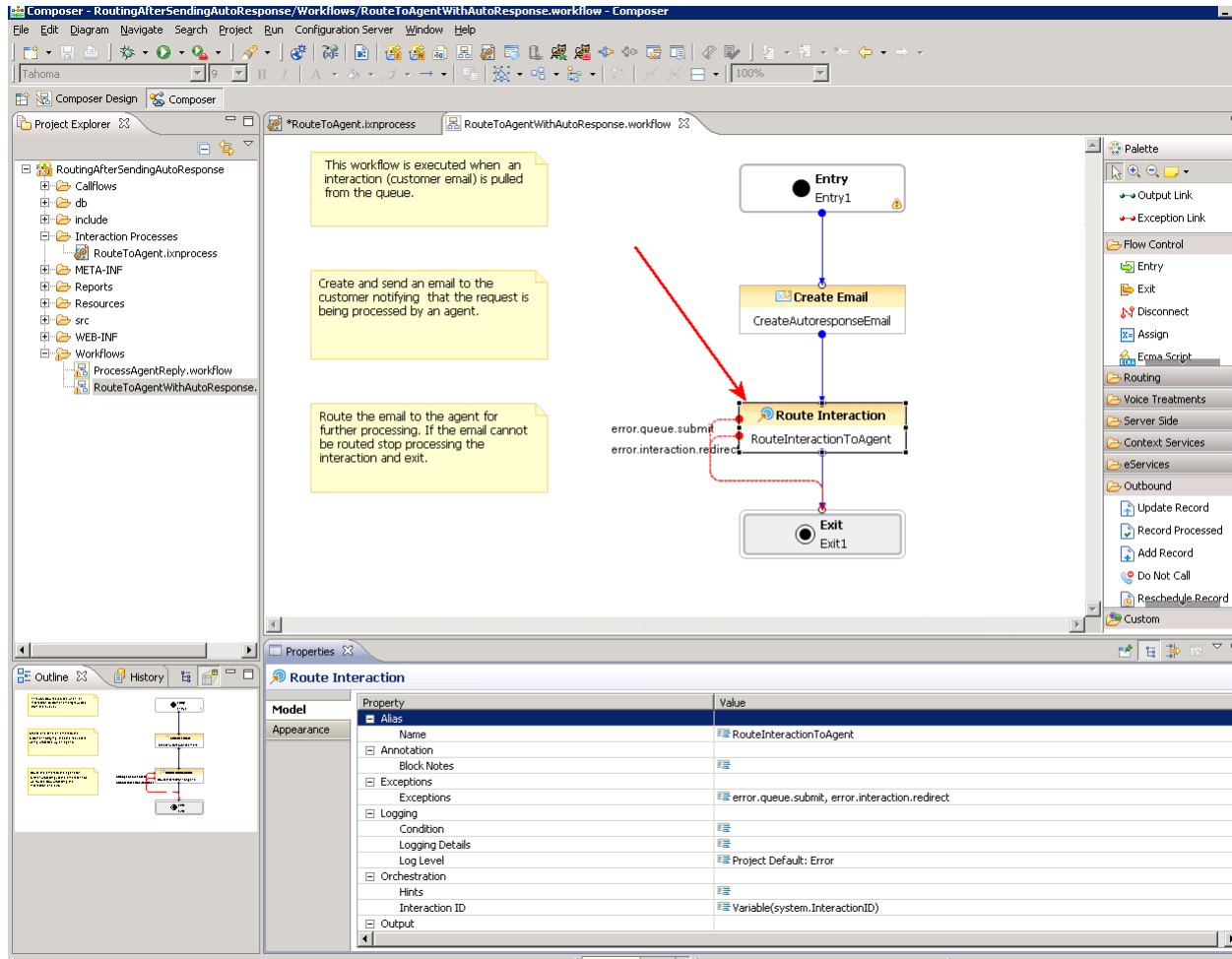



Figure 9: Properties View for Selected Workflow Diagram Block

When configuring/editing fields in the Properties view:

- Some fields allow you type directly in them.
- Other fields display a  button when you click under Value in the Properties view. You then click that button to open a dialog box.

For example, clicking opposite Exceptions under Value causes this button to appear (see [Figure 10](#)).

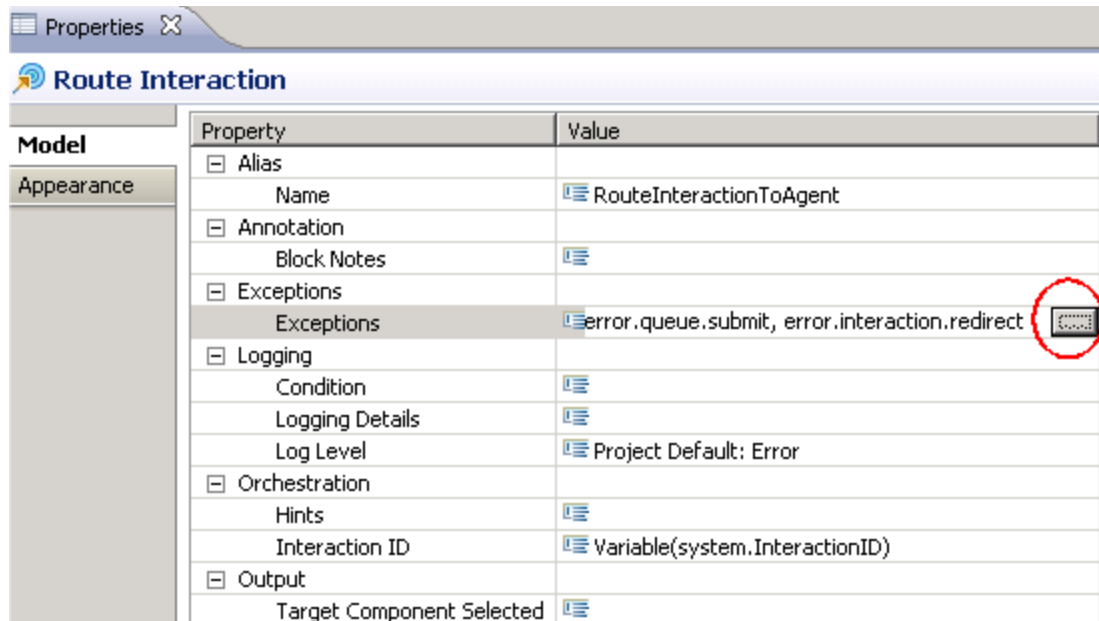



Figure 10: Button for Accessing Properties Dialog Box

Click this button to bring up a dialog box (see [Figure 11](#)) where you configure the handling of events this block may possibly encounter.

Exception/Error Handling

In the case of the Exceptions property, clicking the  button brings up the Exceptions dialog box (see [Figure 11](#)):

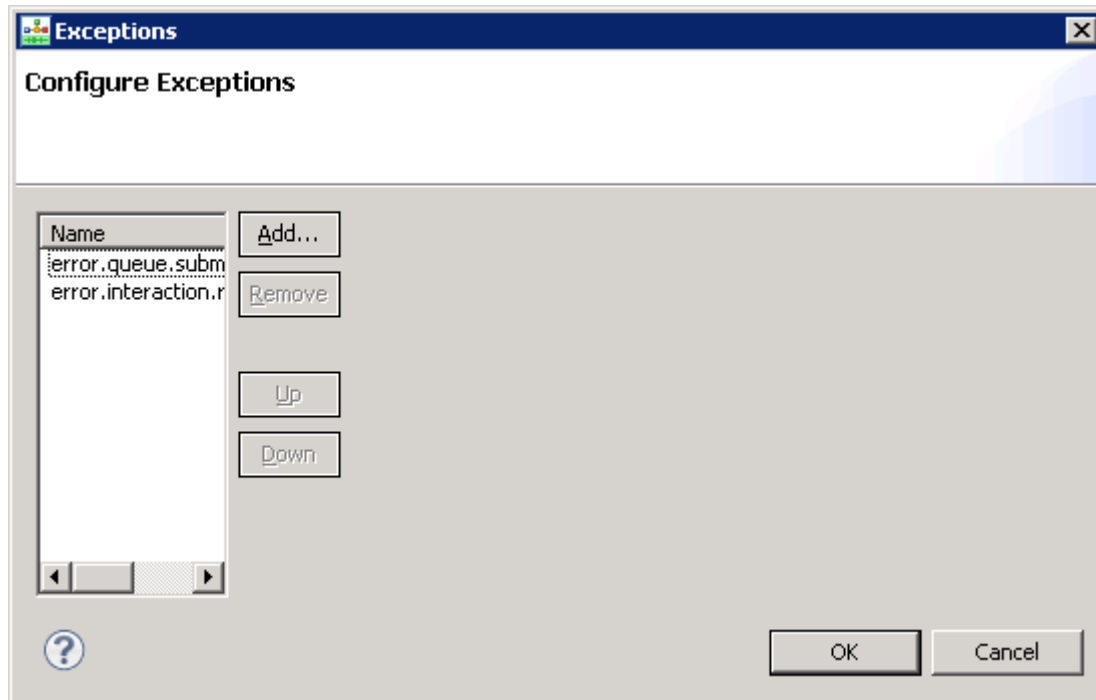


Figure 11: Exceptions Dialog Box

Here you click Add and select supported and non-supported exception events for the block (see [Figure 12](#)).

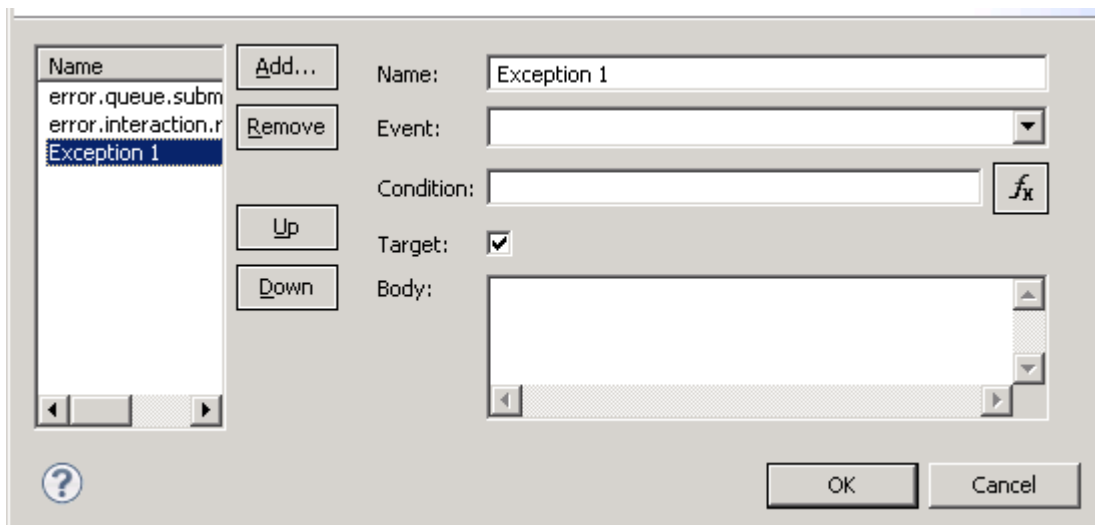


Figure 12: Adding Exceptions

After you click OK to close the dialog box, each supported exception event causes a red error port to appear on the block, which can be connected to another block for error handling.

In the example in [Figure 13](#), the Route Interaction block defines two exception event so two red error ports appear, which connects to a block for handling the exception event. In this example, the block is an Exit block.

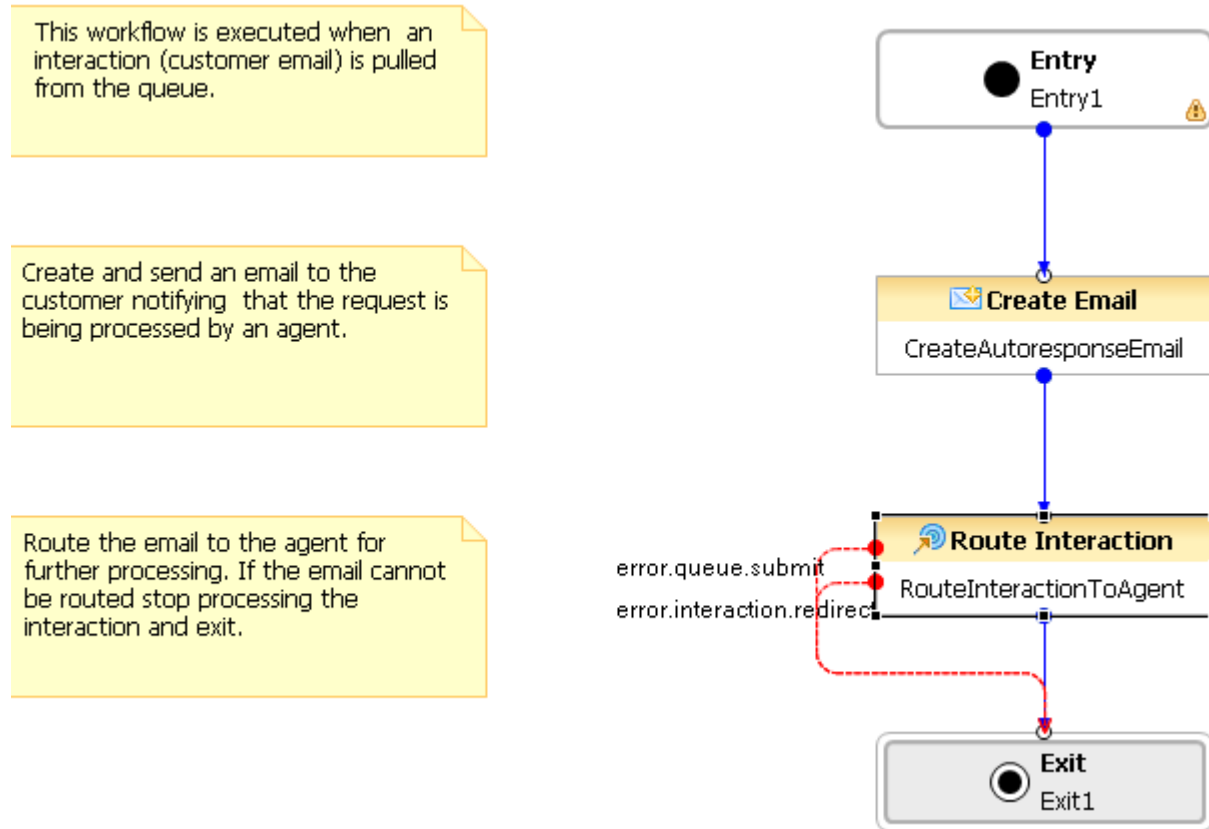


Figure 13: Error Ports Connected to Block for Error Handling

Exceptions can be configured at two levels:

1. At the individual block-level for local exception handling using the `Exceptions` property.

Note: These block-level exceptions may not be present in all blocks.

2. At the top level in the Entry block using the `Exceptions` property.


Variables: Project and Application

You have the option of defining two types of variables in a routing application:

1. Application-level
2. Project-level

Workflow (Application) Variables

You define workflow variables in the Entry block.

Clicking the  button opposite the Variables property in the Properties view opens the Application Variables dialog box. Figure 14 shows an example dialog box.

Variable Name	Category	Value	Description	
system.BaseURL	System	getBaseUrl()	Base URL	Add
system.RelativePathURL	System	getRelativePathURL()	Relative path	Delete
system.Language	System	'en-US'	Application Language	Up
system.InteractionID	System	_event.data.interactionid	The current interaction ID.	Down
system.CallID	System	_genesys.ixn.interactions[system.InteractionID].voi...	callid created by the switch.	
system.DNIS	System	_genesys.ixn.interactions[system.InteractionID].voi...	DNIS associated with Called phone numl	
system.ANI	System	_genesys.ixn.interactions[system.InteractionID].voi...	ANI associated with the calling party.	
system.StartEvent	System	undefined	The content of the specified start event	
system.LastErrorEvent	System	'undefined'	Last error	
system.LastErrorEventName	System	'undefined'	Last error event name	
system.LastErrorDescription	System	'undefined'	Last error description	
system.WebServiceStubbing	System	'0'	Flag to control WebServices Stubbing. '	
system.TerminateIxOnExit	System	1	Flag to control if Exit block should termin	
system.TenantID	System	parseInt(_genesys.ixn.interactions[system.Interacti...	The current Tenant ID.	
system.TenantName	System	_genesys.session.tenant	The current Tenant name.	
system.LastTargetComponentSel...	System	'undefined'	Target to which the Interaction was rou	
system.LastTargetObjectSelected	System	'undefined'	High-level Target to which the Interacti	
system.LastTargetSelected	System	'undefined'	DN and the Switch name of the Target t	
system.LastVirtualQueueSelected	System	'undefined'	The Alias of the Virtual Queue specified	
system.LastSubmitRequestId	System	'undefined'	Requestid value of the Last queue:sub	
system.OPM	System	getOPMParameters()	Operational Parameters Data Variable	
system.OCS_RecordURI	System	getWorkflowRecordURI()	OCS Record URI	
system.OCS_URI	System	getWorkflowOCSURI()	OCS URI	
system.OCS_Record	System	getWorkflowOCSRecord()	OCS Record	

Figure 14: Application Variables Dialog Box

Use application variables when you need to share information across different blocks in the same workflow. For example, the Assign block allows you to assign entered values or values created in Expression Builder to variables.

Composer predefines a number of default routing application variables as shown above in Figure 14 including DNIS and ANI.

Project Variables

You define the second type of variable in the Project Variables dialog box, which opens when you click the access project variables button on the toolbar when an IPD is in focus (see Figure 21 on page 41).

Use Project variables when you need to share information across different workflows.

Expression Builder

Composer's Expression Builder lets you build expressions for branching and conditional routing decisions in a workflow. [Figure 15](#) shows an example simple expression:

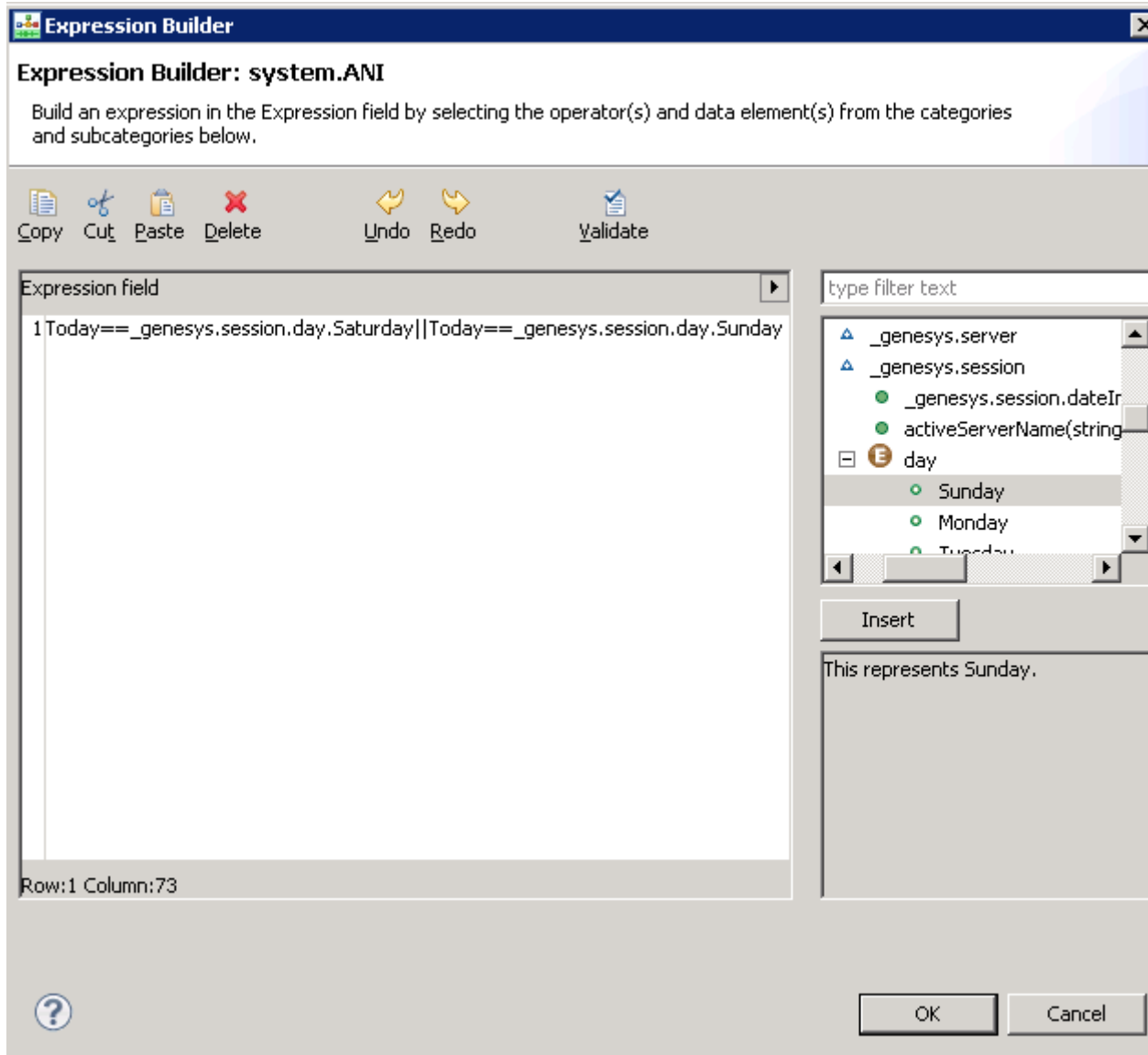


Figure 15: Expression Builder

In summary, you create expressions by:

- Expanding the categories shown above and selecting values.

- Using the buttons under Operators.
- Typing part of the expression directly when necessary.

Expression Builder opens from many blocks including:

- Assign—Assign Data Property
- Branching—Conditions Property
- ECMAScript (for workflows)—Script Property
- Entry—Variables Property
- Log—Logging Details Property
- Looping—Exit Expression Property

ECMAScript Expressions

Universal Routing Server 8.0+ supports SCXML plus ECMAScript as a routing language. While the core SCXML provides State Chart functionality, you can specify URS-specific instructions, such as conditions that can be used for routing decisions, in the form of ECMAScript. The Script property in the ECMAScript block brings up Expression Builder for creating those conditions. [Figure 16](#) shows an example ECMAScript expression.

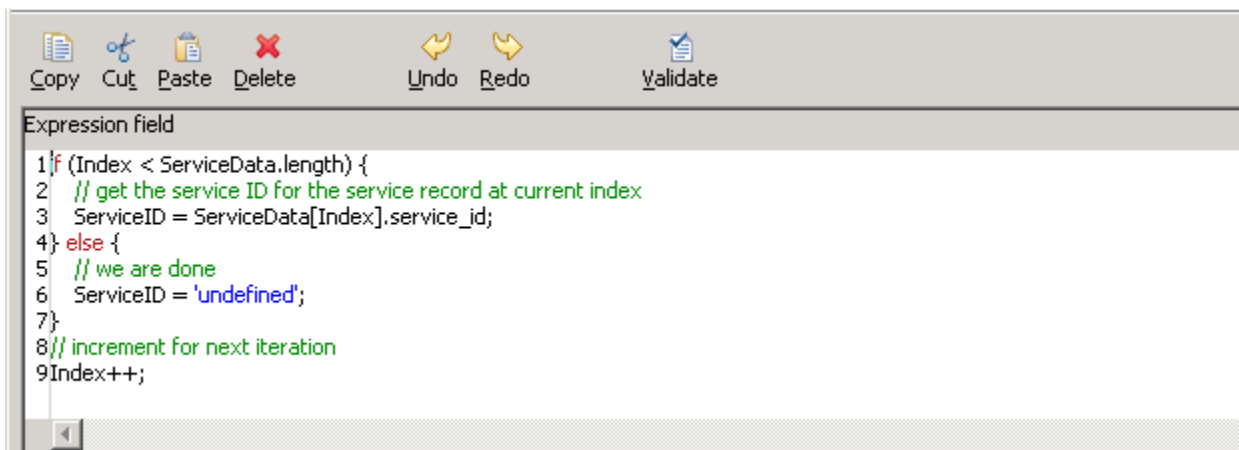


Figure 16: ECMAScript Expression in Expression Builder

Orchestration Server Functions

You can also build expressions that use the Orchestration Server functions. (previously “Genesys-supplied Functional Modules”). For example, in Expression Builder:

1. Select the Orchestration Server Functions data category (see Figure 15 on page 35) to display the various categories.
2. For purposes of this example, select `_genesys` followed by `_genesys.queue`.
3. Double-click a function to insert (see Figure 17).

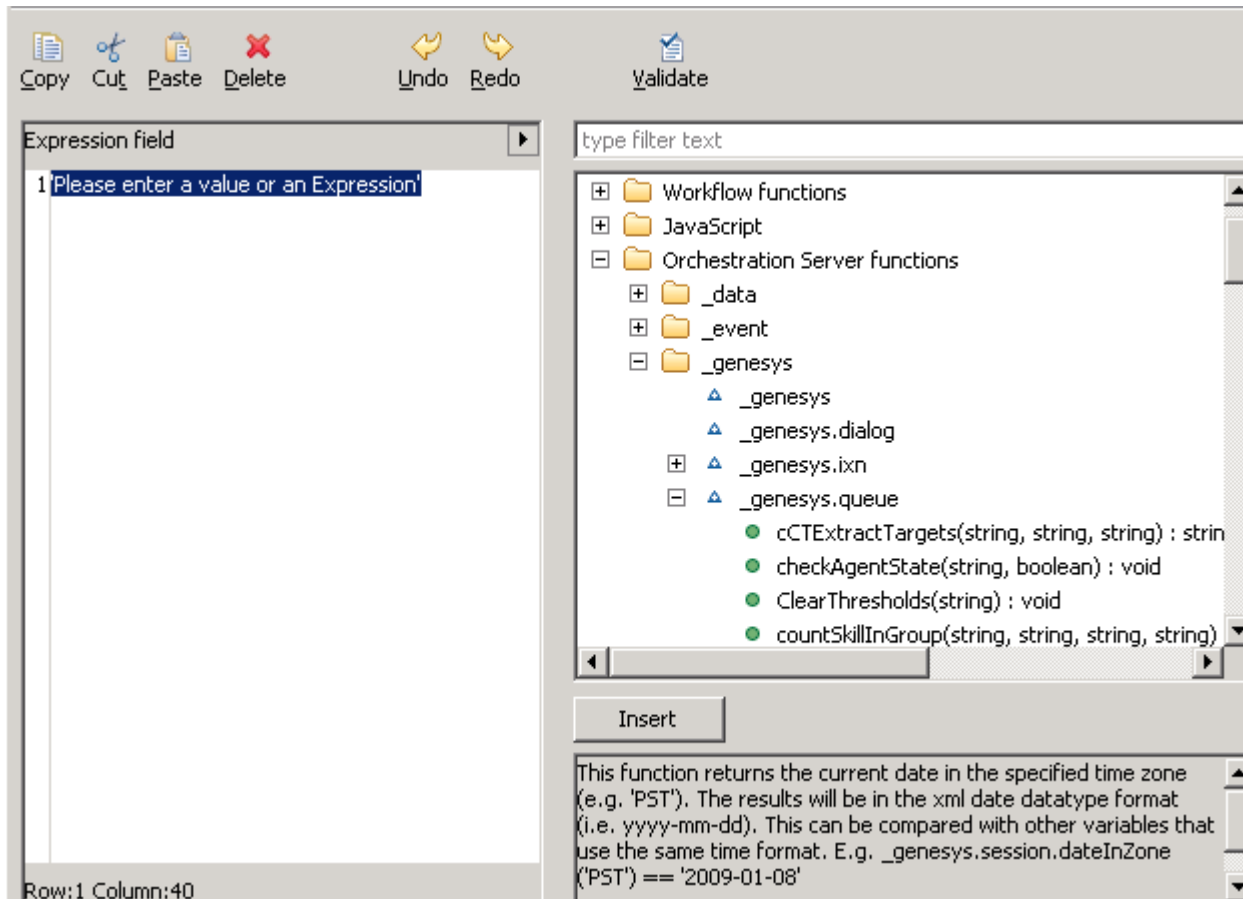


Figure 17: Orchestration Server Functions in Expression Builder

The `_genesys.queue` modules implement the target selection functionality of URS (finding resources for interactions and delivering interactions to the resource). The selected module, `genesys.queue.checkAgentState` corresponds to the URS `CheckAgentState` function as described in the *Universal Routing 8.1 Reference Manual*, `CheckAgentState` function.

In Figure 17, note that a description of any selected Functional Module appears on the right. For more information:

- The SCXML Language Reference on the [Orchestration Server wiki](#), is the Genesys language specification for the following interfaces: (1) SCXML — What we support from the standard, both from an interface and behavioral standpoint; (2) Domain-specific languages (model modules) for all the Genesys-specific functional modules; (3) External interfaces to platform and SCXML sessions.
- Also see:
<http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>

Statistics Manager/Builder

Composer has a Statistics Manager and Builder, which you can use to instruct Universal Routing Server to use the value of a statistic during target selection, such as StatTimeInReadyState (see [Figure 18](#)).

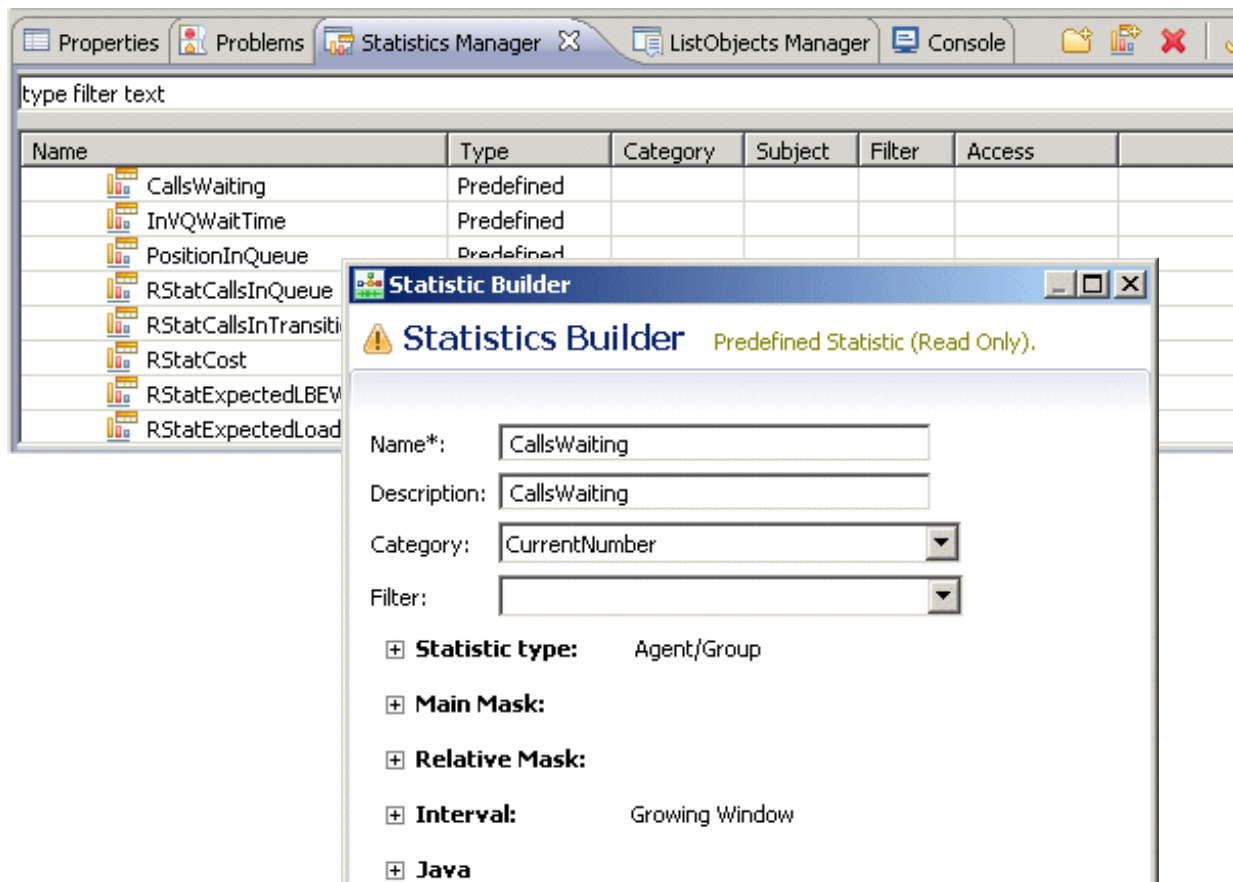


Figure 18: Statistics Manager/Builder

The statistic can be a URS predefined statistic (as described in the *Universal Routing 8.1 Reference Manual*) or a statistic that you create yourself with

Statistics Builder. Once you create a statistic, that statistic becomes available for selection via the Statistics property in Composer's Target block.

Opens by clicking the button for Statistics Manager on the main toolbar (see Figure 21 on page 41).

List Objects Manager/Builder

Composer has a List Objects Manager/Builder, which you can use to store/access strings of any nature; for example, DNIS or ANI strings.

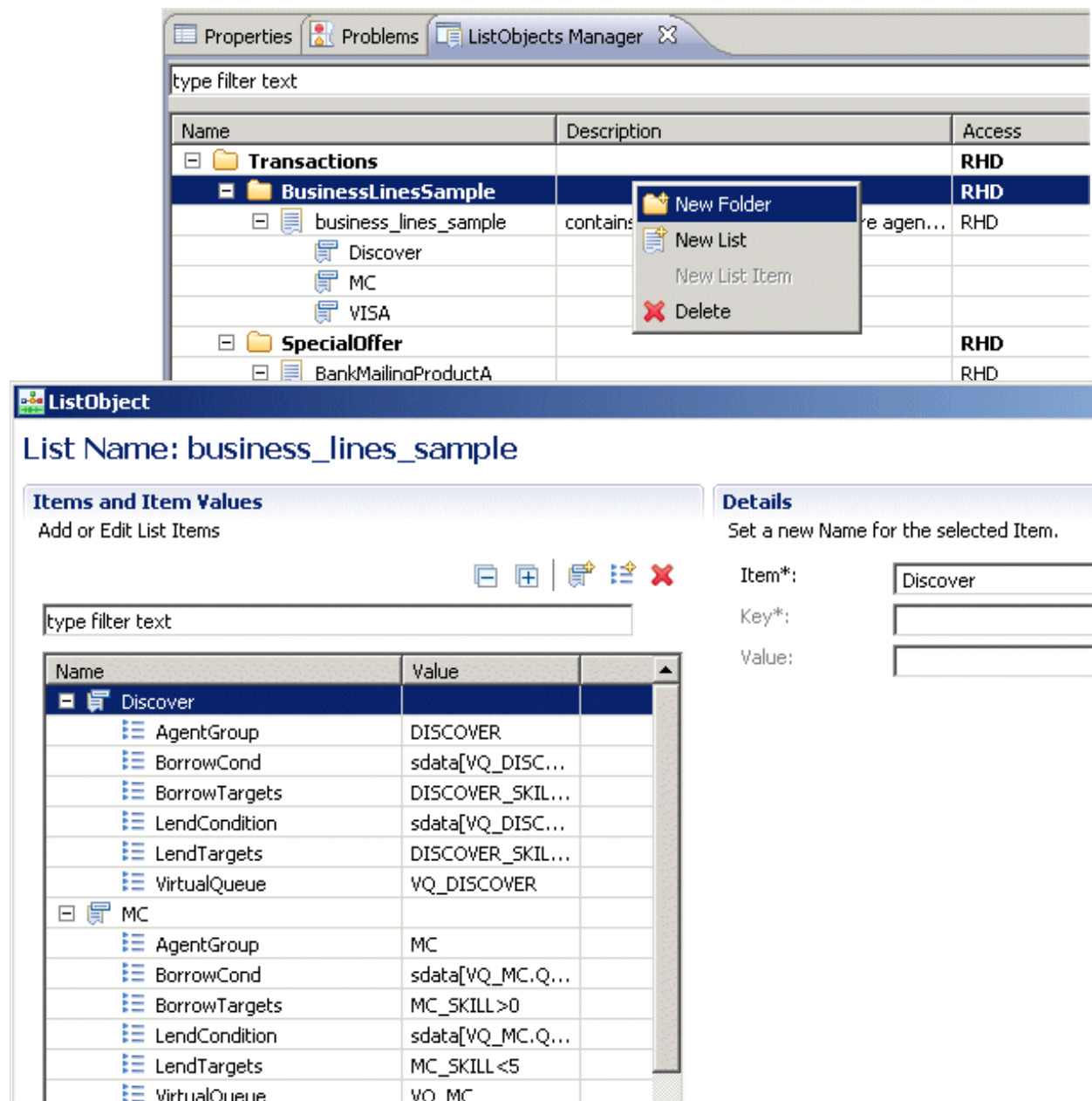


Figure 19: List Objects Manager

The strings can be as simple as 800 numbers or as complex as routing conditions. In Expression Builder, two URS Functions (see Figure 17 on page 37) can be used to access List Objects:

- `_genesys.session.listLookupValue` and
- `_genesys.session.getListItemValue`

List Objects Manager/Builder opens by clicking its button on the main toolbar (see Figure 21 on page 41).

Note: For detailed information on these builders/managers, consult the *Composer 8.1 Help*.

Skill Expression Builder

Composer also has a Skill Expression Builder for creating skill expressions used for routing decisions (see Figure 20).

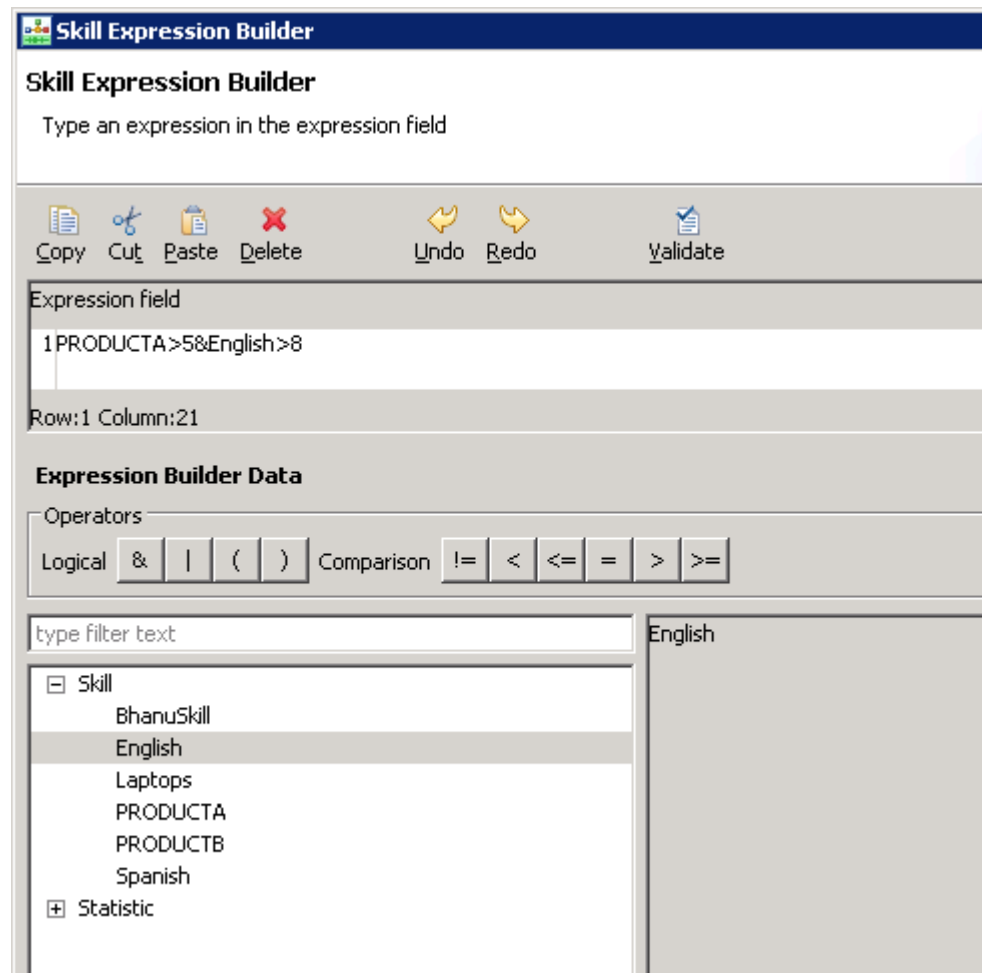


Figure 20: Skill Expression Builder

Skill Expression Builder opens from the Targets property in the Target block after selecting the Skill as the target type.

Toolbar Buttons for Routing

The main toolbar is displayed at the top of the Composer window directly beneath the menu bar.

Note: Buttons on the main toolbar change based on the active perspective (see “Perspectives” on [page 97](#)). Items in the toolbar might also be enabled or disabled based on the state of either the active view or editor.

Sections of the main toolbar can be rearranged using the mouse.

[Figure 21](#) identifies frequently used buttons in Composer Design perspective when creating routing applications

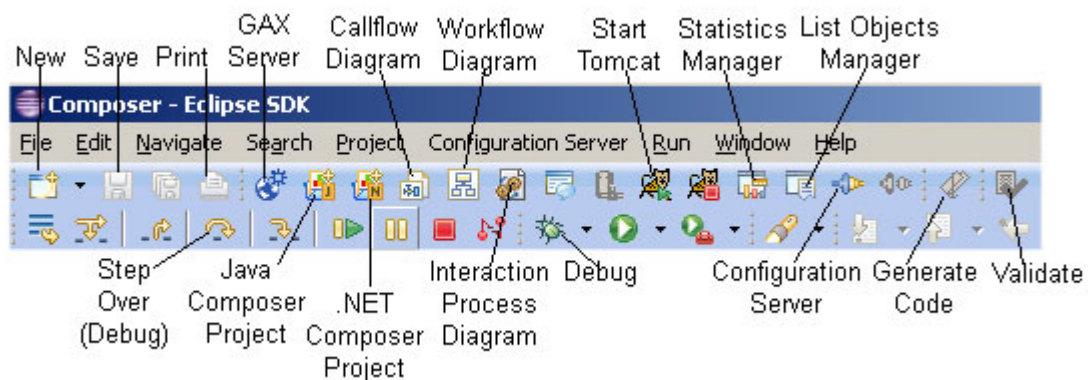


Figure 21: Frequently Used Buttons When Creating Routing Applications

Menu Bar

[Figure 21](#) also shows the Composer menu bar. When working with routing applications, some commonly used menu selections are listed below:

- File > New > Java Composer Project or NET Composer Project when creating a new routing application. You may also frequently use File > New > Workflow Diagram or File > New > SCXML File. The wizard will prompt you to select the Project. In addition, Export and Import commands are available.
- The Diagram menu contains many commands related to lines, colors, arrows, arranging, aligning, sizing, viewing, filtering, and zooming. There are also additional items in Composer Diagram preferences (see [Figure 22](#) on [page 42](#)).

- You will want to connect to Configuration Server (see [page 66](#)) via that menu in order to view objects in the Configuration Database and to update that database when creating new IPDs, diagrams, interaction queues, views, and other objects.
- The Window menu contains many perspective commands as well as giving access to the Preferences dialog box (see [Figure 22 on page 42](#)).
- The Help menu access both the Composer and Universal Routing SCXML help systems.

Preferences

Composer Preferences apply to all Projects within the workspace (see [page 20](#)). You can set preferences for the following: Business Rules, Composer Diagram, ORS Debugger, and SCXML Files. To open the Preferences dialog box, select Window > Preferences (see [Figure 22](#)).

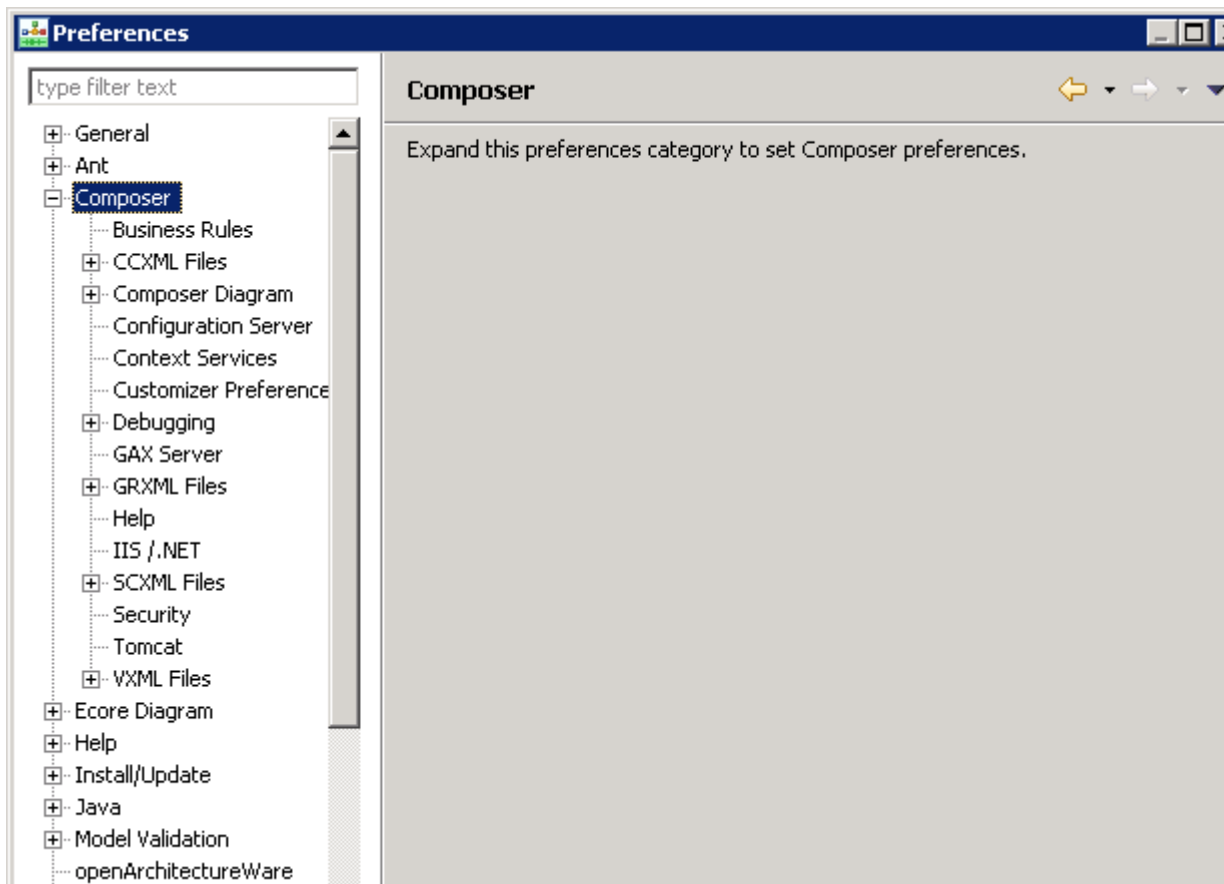


Figure 22: Composer Preferences Used for Routing Applications

For details on the routing-related preferences, see the Preferences for Routing Applications topic on the [Composer Help Wiki](#).



Chapter

3

Summary of Tasks

This chapter summarizes the tasks that are required to create an SCXML-based routing application in Composer. This chapter includes the following sections:

- [Tasks: Planning & Preparation, page 43](#)
- [Task: Creating a New Project, page 44](#)
- [Tasks: Creating the Workflow Diagram\(s\), page 45](#)
- [Tasks: Creating the IPD, page 45](#)
- [Tasks: Code Generation, Testing, and Deployment, page 47](#)

Tasks: Planning & Preparation

After installing Composer as described in the *Composer 8.1 Deployment Guide*, including the post-installation configuration described in that guide, the planning and preparation tasks are provided in [Table Task Summary](#):

Task Summary: Planning & Preparation

Task Objective	Related Procedures and Actions
Plan the design.	<ul style="list-style-type: none">• See “Planning the Design” on page 49
Install the required software.	<p>This task involves installing the required Genesys software components, such as Universal Routing Server, Orchestration Server, Universal Contact Server, and eServices components, if processing multimedia interactions.</p> <ul style="list-style-type: none">• See “Installing the Required Software” on page 52.

Task Summary: Planning & Preparation (Continued)

Task Objective	Related Procedures and Actions
Pre-configure database objects.	<p>This task involves creating objects in the Configuration Database either through Genesys Administrator or Configuration Manager. These objects include agents, agent groups, business attributes, standard responses, and so on.</p> <ul style="list-style-type: none"> See “Preconfiguring Database Objects” on page 52. <p>If processing multimedia interactions, this task also includes defining media servers and media server points.</p> <ul style="list-style-type: none"> See “Endpoints” on page 70.
Set up Context Services (optional).	<p>Optional. This task is required if managing conversations/personalizing services/offers using the Context Services option of the Universal Contact Server database.</p> <ul style="list-style-type: none"> See “Using Context Services” on page 53.
Review the samples.	<ul style="list-style-type: none"> See Procedure: Reviewing sample interaction process diagrams, on page 56. See Procedure: Reviewing sample routing projects, on page 56.

Task: Creating a New Project

As described in “Projects” on [page 19](#), a Project organizes all the elements associated with a routing application. Creating a new Project automatically creates a default interaction process diagram (`default.ixnprocess`) and a default workflow diagram (`default.workflow`), which you can edit.

The tasks are provided in [Table Task Summary](#).

Task Summary: Creating a New Project

Task Objective	Related Procedures and Actions
Create a new Project.	<ul style="list-style-type: none"> Decide whether the Project will be created “from scratch” or based on a predefined Project template (see Figure 24 on page 57). See Procedure: Creating a new Project, on page 64.

Tasks: Creating the Workflow Diagram(s)

Each Workflow block in an IPD references a workflow diagram (see Figure 7 on [page 28](#)). You have the option of creating these diagrams before creating the IPD or having the IPD reference empty placeholder diagrams to be completed later on. If you decide to create the workflow diagrams before the IPD, the required tasks are provided in [Table Task Summary](#):

Task Summary: Creating the Workflow Diagram(s)

Task Objective	Related Procedures and Actions
Connect to Configuration Server.	As described on page 66 , you have the option of developing routing applications in online or offline mode. To work in online mode and have Configuration Database objects viewable/selectable from Composer blocks: <ul style="list-style-type: none"> • See Procedure: Connecting to Configuration Server, on page 67.
Connect to the Context Services Server.	This task is used if managing conversations/personalizing services/offers using the Context Services option of the Universal Contact Server database. To work in online mode and have database objects viewable/selectable from Composer blocks: <ul style="list-style-type: none"> • See Procedure: Connecting to the Context Services Server, on page 68.
Create the workflow diagram(s) that will be referenced in the IPD.	<ul style="list-style-type: none"> • See Procedure: Creating a workflow diagram, on page 78.

Tasks: Creating the IPD

The tasks that are required to create the routing application's interaction process diagram (see [page 16](#)) are provided in [Table Task Summary](#):

Notes: While you can create an IPD on-demand from the toolbar (see Figure 21 on [page 41](#)), the Project creation task previously described automatically created an IPD.

The task summary below applies to creating an IPD for processing multimedia interactions. An IPD for voice only interactions contains a single Workflow block.

Task Summary: Creating the Interaction Process Diagram

Task Objective	Related Procedures and Actions
Enter IPD properties.	<p>In order to enter IPD properties, the IPD must be in focus. To get the IPD in focus, you can select the *.ixnprocess tab or select the IPD in the Interaction Process folder in the Project Explorer.</p> <ul style="list-style-type: none"> See Procedure: Entering IPD properties, on page 70
Add the Media Server block(s).	<p>This task is required if processing multimedia interactions.</p> <ul style="list-style-type: none"> See Procedure: Adding a Media Server Block, on page 72.
Define the interaction queue(s).	<p>An IPD moves interactions from media servers to interaction queues to workflows. The design of your IPD determines how many times this process is repeated.</p> <ul style="list-style-type: none"> See Procedure: Defining an Interaction Queue, on page 74.
Define the views to extract interactions from the interaction queue. You can define multiple views for a single Interaction Queue where each view can have its own set of conditions used for pulling interactions from the queue	<p>You can define the time intervals that Interaction Server uses for checking queues, conditions for extracting interactions, the order for extracting interactions, scheduling, and hints to optimize performance for Oracle databases.</p> <ul style="list-style-type: none"> See Procedure: Defining a view for an interaction queue, on page 75.
If applicable, use the Flow Control blocks.	<p>The Branching, ECMAScript, and Log blocks (as described in Table 3 on page 114) support defining multiple views (see page 75) per interaction queue.</p>
Add the Workflow block(s).	<p>If the workflow diagram has been previously created, the Workflow block Resource property can point to it.</p> <ul style="list-style-type: none"> See Procedure: Adding a Workflow block, on page 82.
Connect the IPD blocks.	<ul style="list-style-type: none"> See Procedure: Using the Output Link to connect blocks, on page 96.
Publish the IPD.	<p>Publishing an IPD validates Project configuration information and pushes the information out to Configuration Server.</p> <ul style="list-style-type: none"> See Procedure: Publishing an interaction process diagram, on page 88.

Tasks: Code Generation, Testing, and Deployment

The tasks that are required for code generation, testing, and deployment are provided in [Table Task Summary](#):

Task Summary: IPD Code Generation and Deployment

Task Objective	Related Procedures and Actions
Generate the code.	<ul style="list-style-type: none"> See Procedure: Generating code for an interaction process diagram, on page 89.
Test the application.	<ul style="list-style-type: none"> See Procedure: Using the ORS Debugger, on page 89.
Deploy the application.	<p>Once your application has been unit tested you will need to deploy it to a web application server. The deployment process involves:</p> <ol style="list-style-type: none"> 1. Exporting your Project 2. Transferring the files to your web/application server. 3. Executing any necessary configuration steps required to make your application work. <p>For information on this step:</p> <ul style="list-style-type: none"> Consult the book <i>Validation, Debugging, and Deployment</i> in the <i>Composer 8.1 Help</i>. See the Deployment sub-book, <i>Deploying a Routing Application</i> topic. Also see the section on Application Server requirements in the <i>Composer 8.1 Deployment Guide</i>.

4

Planning & Preparation

This chapter summarizes recommended planning and preparation processes to undertake prior to working in Composer to create routing applications. It contains the following sections:

- [Planning the Design, page 49](#)
- [Installing the Required Software, page 52](#)
- [Preconfiguring Database Objects, page 52](#)
- [Using Context Services, page 53](#)
- [Reviewing the Samples, page 56](#)
- [Enabling/Disabling Functionality, page 58](#)

Planning the Design

Prior to working in Composer, plan the design of your routing application by considering the various stages of interaction processing that occur (or should occur) at your site. The structure of the IPD will reflect this design.

- If you are routing only voice interactions, the design will focus on the workflow diagram referenced by the IPD. To plan this design, start by studying the sample Projects for routing voice interactions (see Figure 24 on [page 57](#)).
- If you are routing multimedia interactions, the design will initially focus on what blocks to include in the IPD as described below followed by what blocks to include in each workflow diagram.

Stages in Multimedia Processing

When planning the design of an IPD for multimedia processing, start by considering the basic stages in the interaction life-cycle at your site. You can then design an IPD that encompasses all stages, or just one stage (see “Moving Interactions Between IPDs” on [page 18](#)). The next section presents four basic

stages, which are especially applicable to e-mail processing, but could apply to other media types as well. The stages are:

1. Pre-Route
2. Route-to-Agent
3. Review
4. Pre-Send

Each stage is summarized below.

Note: The Genesys Multimedia/eService product contains an *Interaction Workflow Samples* software component. The samples include various business processes (similar to IPDs) and associated queues, views, submitters, workbins, and routing strategies. The design behind the samples could also be applied to routing applications created in Composer. The samples are documented in the appendix of the *Universal Routing 8.1 Business Process User's Guide*. You may wish to study them for design ideas prior to creating routing applications in Composer.

Pre-Routing Stage

The main activities in the pre-routing stage of e-mail handling can potentially include:

- Causing incoming interactions to take different paths in the workflow based on criteria such as time of day, user data contained in the interaction, and so on.
- Determining whether an e-mail has already been processed by Genesys. This can be accomplished via the absence or presence of an *Interaction Subtype Business Attribute* assigned by *Interaction Server*. The samples described in the above note demonstrate how to do this.
- Sending an acknowledgement and/or automatic standard response to the customer who originated the e-mail.
- Determining the agent (if any) who previously handled the interactions that are related to this service.

Route-to-Target Stage

The target may or may not be an agent. For example, the e-mail may be:

- Sent to a queue for submittal to other routing strategies and further processing.
- Sent to a queue for failed interactions.
- Forwarded outside the contact center to an expert with the expectation of getting a response back.

- Redirected to another agent without the expectation of getting a response back.
- Routed to an agent target for construction of a response.

Review Stage

The reviewer could be a manager, supervisor, or QA Person. You may want to have two different types of quality assurance review:

- A supervisor review that checks the skills of the agent who constructed the response.
- An analysis that performs a “sanity check;” for example, to prevent sending out a bank account password in an interaction or to check interactions for inappropriate language.

Pre-Send Stage

The cycle of going from queue to routing workflow to queue can continue until the interaction reaches some final outbound queue. The pre-send stage performs last-minute quality checking and allows for attaching additional information to interactions when needed.

One or Multiple IPDs?

When designing an IPD for multimedia processing, you have two choices:

1. You can create one complex IPD with multiple workflows that encompass all stages of interaction processing
2. You can create multiple IPDs and link them via interaction queues. For example, you could create one IPD for each stage and link them. See “Moving Interactions Between IPDs” on [page 18](#) for more information.

Summary of Planning/Preparation Process

This section summarizes the entire planning/preparation process prior to actually creating, configuring, and placing blocks in Composer.

1. Determine the interaction processing stages/life cycle at your contact center.
2. List the specific interaction processing functionality required at your contact center within each stage.
3. Determine which Composer blocks will be used in routing workflows to perform the various processing required at each stage.
4. Determine if any ECMAScript expressions will be required to be created in Expression Builder for conditional routing decisions (see the ECMAScript block in “Flow Control Blocks” on [page 114](#)).

5. Is there any special interaction processing functionality that is not covered by a Composer block, which will require handwritten SCXML code?
6. Determine which Orchestration Server functions will be used (if any) in Composer blocks.
7. Determine the media server that will be used. In order to get interactions of a particular media type into an interaction queue in an IPD, the Media Service Application object must have one or more Endpoints defined as described ahead in “Adding a Media Server Block” on [page 70](#).
8. Decide whether you will use one complex IPD or multiple linked IPDs (see “Moving Interactions Between IPDs” on [page 18](#)).
9. If you plan on having multiple IPDs linked via workflows, name the queues that will connect the workflows contained within each IPD.
10. Determine the selection criteria for extracting interactions from queues (View property in the Interaction Queue block). For example, you may wish to extract certain interaction types earlier than other interaction types.

Installing the Required Software

In order to create routing applications, Composer requires certain software component. For a complete listing of these software requirements, see the chapter on installation in the *Composer 8.1 Deployment Guide*.

Preconfiguring Database Objects

Prior to using Composer, it is convenient to have pre-configured certain Configuration Database objects so they will be selectable from Composer menus. The objects described below are in addition to the Skills, Persons, Agent Groups, Places, Place Groups, and other Resource type objects described in the chapter on manually configuring routing in the *Universal Routing 8.1 Deployment Guide*.

eService Objects

At some point in an application that routes multimedia interactions, you may want to use eService’s Knowledge Management functionality as described in the *eServices/Multimedia 8.1 User’s Guide*. If so, it is convenient to pre-configure the following:

- Category codes. Knowledge Manager uses a system of category codes, organized in a tree structure, as a means of organizing pre-written text, called standard responses.

- Standard responses. If using the Create E-mail block (see Table 15 on [page 125](#)), you may use pre-written standard responses, such as for acknowledgement or auto response e-mails. Note: Composer currently does not support the use of field codes in standard responses.

You use eServices's Knowledge Manager to create category trees, and to create and edit the standard responses they can contain.

Note: Any Category structures (and associated standard responses) defined in Knowledge Manager (written to the Universal Contact Server database) are automatically carried over to the Configuration Database. The information is therefore viewable in Configuration Manager or Genesys Administrator.

Business Attributes

At some point in a routing application, you may wish to specify *Business Attributes*, which are interaction attributes used in different ways within Genesys. The *Business Attributes* folder (accessible via Configuration Manager or Genesys Administrator) contains a number of sub-folders containing both customer-defined and Genesys-defined *Business Attributes*.

For example, at some point in a routing application, you may wish to define:

- *E-mail Accounts* if using the Send E-mail or Create E-mail blocks
- *Stop Processing Reason* if using the Stop Interaction block
- *Interaction Subtype* if using the Create SMS block
- *Service Type* if using the Associate Service or Query Services block
- *Disposition Code* if using the Complete Service or Complete State block

If you will be creating expressions in Expression Builder (see Figure 16 on [page 36](#)), you may also need to define some new *Business Attributes*.

Using Context Services

Context Services refers to an optional capability of Universal Contact Server and its Universal Contact Server (UCS) Database, a repository of customer-related, service, and interaction-centric data (current and historical). You can use the Context Services capability for:

- **Service personalization.** You can create routing workflows that alter the customer experience based on information known about the customer.
- **Offer personalization.** Workflows can use the results of previous offers made to the customer to decide whether a new offer should be presented.

- **Service resumption.** Workflows can leverage service state/task information to continue a customer service that was not completed in an earlier interaction.

If the Context Services capability is enabled at your site, you can use the Context Services blocks (see Table 8 on [page 121](#)) to create routing applications that extract customer data elements from the UCS Database and apply this knowledge during the routing of interactions.

The Context Services category in Expression Builder gives access to customer profile attributes, both core and extension (see [Figure 23](#)).

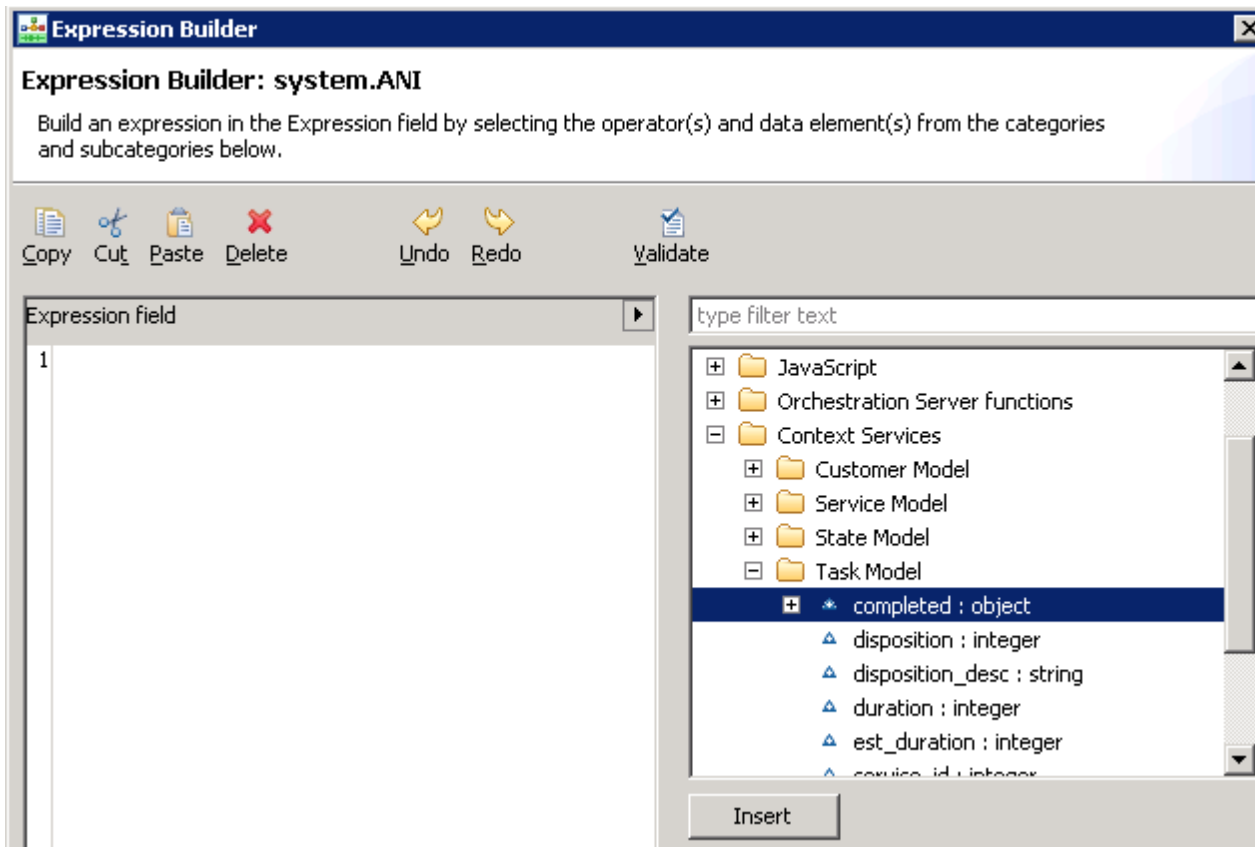


Figure 23: Context Services Data in Expression Builder

Mapping Context Services Attributes

If using the Context Services blocks, you may wish to map Context Services attributes to Configuration Server Business Attributes before configuring those blocks in Composer.

- For more information, consult the *Universal Contact Server 8.1 Context Services User's Guide*, Chapter 2, Configuration section. See the description of the <BusinessAttribute> option, in the *view* section.

Managing/Orchestrating Conversations

If you plan to use routing applications to manage/orchestrate conversations with customers, you will need to define certain objects in the Universal Contact Server Database related to services.

Service Model

Genesys Conversation Manager takes the Genesys core capability of routing and extends it. Rather than taking the call (T-Server) or the interaction (eServices/Multimedia) as the basic entity, Conversation Manager takes the service as the basic entity. It provides the ability to “orchestrate” the service process across channels and over time, using dynamic data and business rules to make decisions about operations.

As described in the *Universal Contact Server 8.1 Context Services User's Guide*, a service may be defined as follows:

- It represents a business process, which in turn may be seen as a communication or series of communications between a customer and an enterprise, and possibly also between various parts of the enterprise.
- It can span multiple interactions.
- It may include interactions in various media.
- It has a temporal beginning and end.

Universal Contact Server/Context Services makes use of a model in which customers are associated with any number of services. Services are composed of any number of states, and states can in turn be composed of any number of tasks. This three-level structure provides a flexible vocabulary by which organizations store the history of the services that they provide to customers.

Defining Services/States/Tasks

This Service model can also be used by any component that can access UCS/CMS's HTTP interface. You can find more information on this interface in the following documents:

- [Context Services Developer's Guide](#), available on the [Genesys Documentation Wiki](#), covers the writing and the optimization of your applications on top of the Context Services.
- *Context Services API Reference*, available on the [Genesys Documentation Wiki](#), covers all the representations and methods available throughout the Context Services.

Reviewing the Samples

Within Composer, you have access to various sample Projects, with sample IPDs and workflows.

Procedure: Reviewing sample interaction process diagrams

Start of procedure

To review sample IPDs:

1. Select **File > New > Interaction Process Diagram**.
2. Select an IPD template and click **Next**.
3. Select an existing Project.
4. Click **Finish**. A simple IPD appears in the **Interaction Processes** folder of the selected Project.

End of procedure

Procedure: Reviewing sample routing projects

To review sample Projects:

Start of procedure

1. Select **File > New > Java Composer Project**.
2. In the resulting dialog box, name the Project, select **Route**, and click **Next**. The dialog box shown in [Figure 24](#) opens.

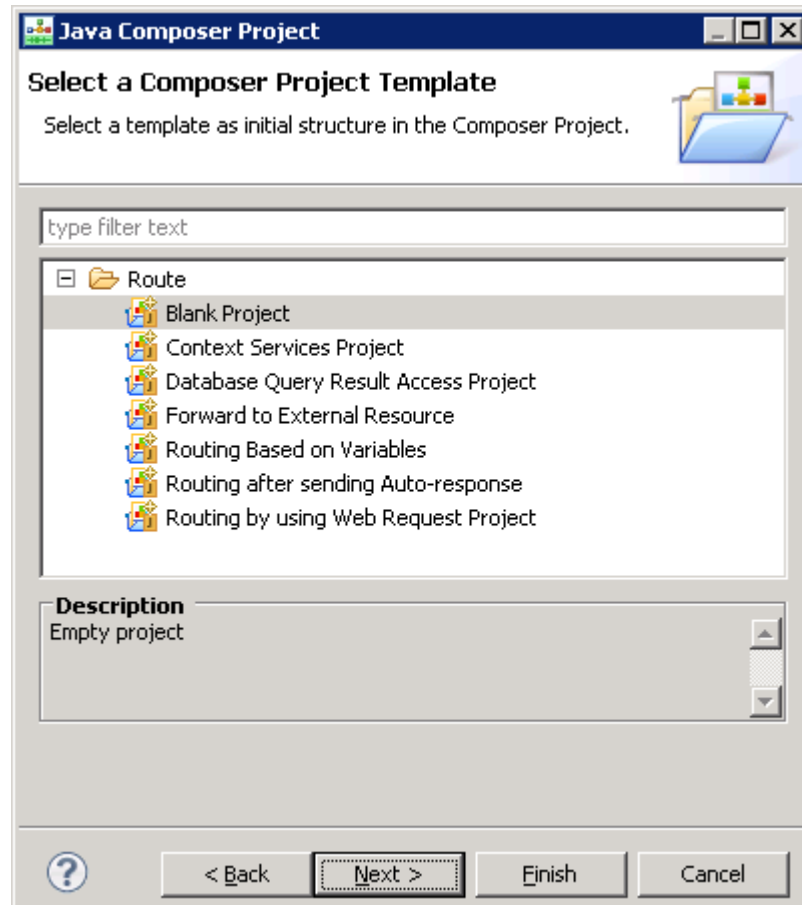


Figure 24: Composer-Supplied Project Templates

Each Project shown above contains sample IPDs.

- The Routing after sending Auto-response template shows an IPD for routing multimedia interactions.
 - The remaining Projects are for voice interactions, which use only Workflow blocks in an IPD.
3. Select a Project template and click **Finish**. The Project appears in the Project Explorer in the upper left of the Composer window.

End of procedure

The next chapter explains how to view the various IPD and Workflow elements.

Enabling/Disabling Functionality

You may hide voice application or routing application development capabilities through a Composer preference setting.

Note: *Voice application* as used above refers to VXML applications that get executed by the Genesys Voice Platform (GVP). Composer Voice applications use voice callflows (as opposed to routing workflows) to generate VXML (as opposed to SCXML).

The ability to enable or disable certain functionality is useful for developers who are only developing applications for a specific Genesys platform.

Procedure: Enabling/Disabling Functionality

Start of procedure

To enable or disable routing or voice application development capability:

1. On the main Composer menu, select Window > Preferences.
2. Expand General (see Figure 22 on [page 42](#)) and select Capabilities.
3. Click the Advanced button.
4. In the Advanced Capabilities dialog box, expand Composer (see [Figure 25](#)).

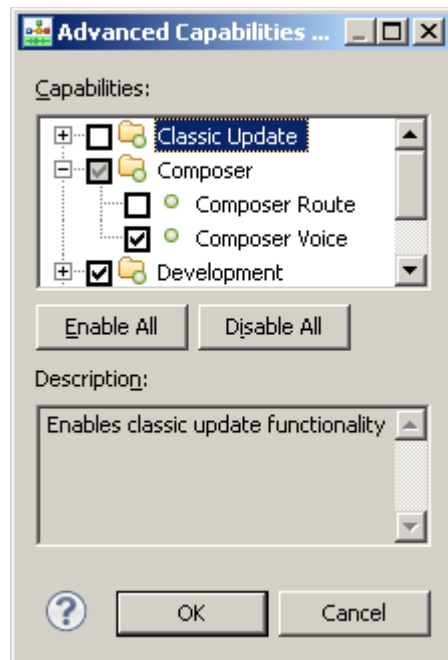


Figure 25: Advanced Capabilities Dialog Box

5. Check/uncheck Composer Route or Composer Voice as desired.
 - By default, both Composer Route and Composer Voice are checked. If you uncheck Composer Route, the ability to create Projects and diagrams with workflows is not available. Also, perspectives and views exclusive to workflows are not available. This means you temporarily won't be able to design routing applications for Universal Routing 8.0 until you enable Composer Route capability.
6. Click OK in both dialog boxes.

End of procedure



Chapter

5

Creating a Routing Application

This chapter leads you through the process of creating a routing application. It contains step-by-step procedures for creating a Project, an interaction process diagram (see [page 16](#)), and a routing workflow diagram (see [page 17](#)). The procedures re-trace the steps used to create one of the Genesys-supplied Project templates.

This chapter contains the following sections:

- [Routing After Sending Auto-response Project Template, page 62](#)
- [Creating a New Project, page 64](#)
- [Connecting to Configuration Server, page 66](#)
- [Defining Contact Services Preferences, page 68](#)
- [Entering IPD Properties, page 69](#)
- [Adding a Media Server Block, page 70](#)
- [Defining an Interaction Queue, page 73](#)
- [Creating a Workflow Diagram, page 78](#)
- [Adding a Workflow Block, page 81](#)
- [Adding the Remaining IPD Blocks, page 84](#)
- [Publishing an IPD, page 87](#)
- [Generating Code, page 89](#)
- [Testing a Routing Application, page 89](#)
- [Deploying the Application, page 93](#)

To avoid getting lost in detail, the procedures in this chapter are highly summarized. For more detailed information on each procedure, consult the *Composer 8.1 Help*.

Routing After Sending Auto-response Project Template

Composer provides a set of predefined Project templates containing sample applications (see Figure 24 on [page 57](#)). You can start off with the Blank Project template and create a routing application from scratch. Or you can use one of the pre-configured templates as a starting point. This chapter starts out with the Blank Project template and re-traces the steps used to create the Routing after sending Auto-response Project shown in Figure 24 on [page 57](#).

This template demonstrates how to create a routing application that:

- Takes an incoming e-mail from a customer and sends an automatic e-mail acknowledgement.
- Routes the customer's e-mail to an agent for reply.
- Has the agent's reply reviewed by a supervisor.
- Sends out the e-mail reply to the customer.

[Figure 26](#) shows the RouteToAgent_Autoresponse IPD associated with the Routing after sending Auto-response Project, which we will be re-creating.

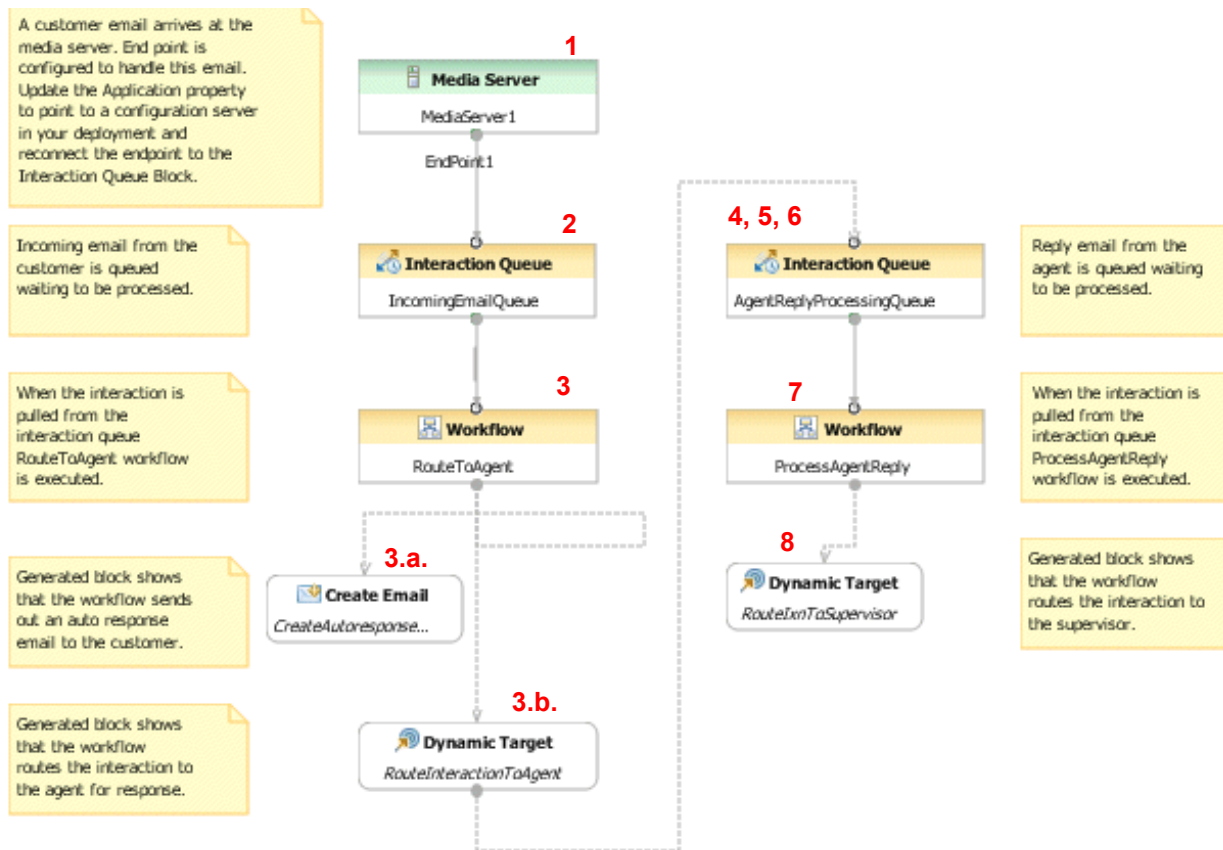


Figure 26: RouteToAgent_Autoresponse IPD

High-Level Interaction Flow

While the Properties view for any Composer block provides processing detail, an IPD for multimedia processing provides a high-level view of interaction flow. As an example, the numbers below are keyed to the IPD blocks in Figure 26 on [page 62](#).

1. A Media Server IPD block (Media Server1) gets incoming interactions (e-mails, in this case) into the IPD for processing. EndPoint1 is associated with a queue for incoming interactions in the Configuration Database.
2. An Interaction Queue IPD block (IncomingEmailQueue) defines the queue for interactions arriving from the media server endpoint.
3. A Workflow IPD block (RouteToAgent) directs Orchestration Server to pull interactions from the queue and process them based on instructions contained in a workflow diagram.
 - a. The workflow diagram pointed to by the RouteToAgent Workflow block uses a Create E-mail diagram-building block to send an auto response e-mail to the customer notifying that their e-mail has received their e-mail and an agent will be contacting them shortly. The CreateAutoResponse block is a “workflow-generated block” as discussed on [page 83](#).
 - b. The same workflow diagram uses a Route Interaction diagram-building block to route the original e-mail from the customer to a target (in this case, an agent group) for a response.

Note: The next two steps (4 and 5) occur outside of the IPD, but could be documented in the IPD with Notes (although this has not been done in the Project template shown in Figure 26 on [page 62](#)).

4. After the agent responds (using the Reply feature in the Genesys desktop application), the response is treated as a new interaction and is placed into the specified queue.
5. The agent then closes the current interaction (using Done feature in the Genesys desktop application).
6. Another Interaction Queue IPD block (AgentReplyProcessingQueue) instructs Orchestration Server to pull the new interaction from the queue and send it to a workflow for processing.
7. A second Workflow IPD block (ProcessAgentReply) points to a workflow diagram that routes the agent response e-mail to a supervisor for review.
8. Once the supervisor is satisfied with the response, the e-mail is placed in the system queue and sent out to the customer.

Creating a New Project

As described on [page 19](#), a Composer *Project* contains everything related to a single routing application. Creating a new Project automatically creates an IPD with the name of `default.ixnprocess` (which can be changed).

Procedure: Creating a new Project

Start of procedure

To create a new Project for a routing application:

1. Select **File > New > Java Composer Project**. This brings up the Java Composer Project dialog box.
2. In this dialog box, type a name for your Project. For this example, type `RoutingAfterSendingAutoResponse`.
3. Select the **Use default location** check box.
4. Select the **Route Project type**, which creates a Project associated with the URS 8.0 SCXML Engine/Interpreter. The Java Composer Project dialog box now appears as shown in [Figure 27](#).

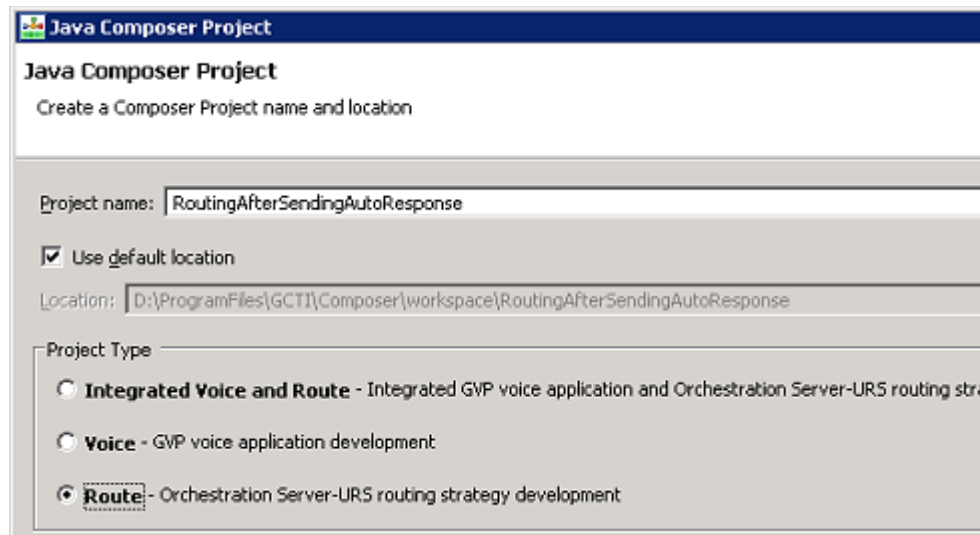


Figure 27: Java Composer Project Dialog Box

5. Click **Next** (button not shown above). In the dialog box for selecting a Project template, select **Blank Project** (see [Figure 28](#)).

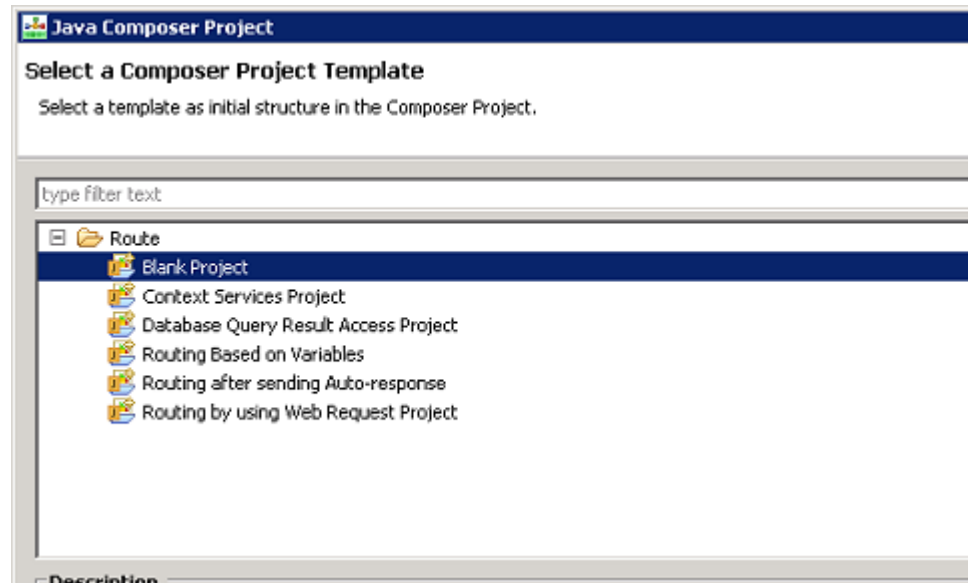


Figure 28: Select a Composer Project Template Dialog Box

6. Click Next (button not shown above).
7. Select the *locale*. As described in the *Composer 8.1 Help*, a locale defines a language and region identifier that you want to work with.

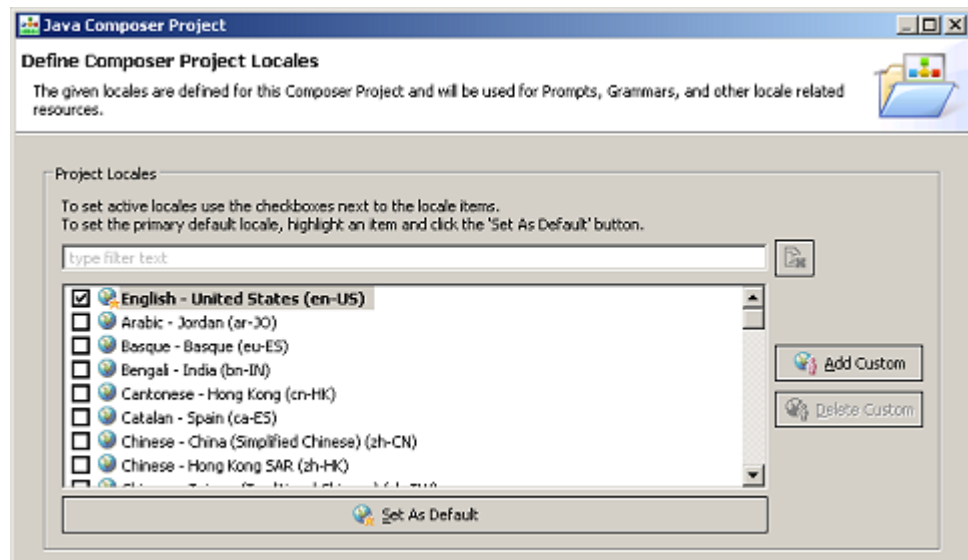


Figure 29: Define Composer Project Locales Dialog Box

8. Click Finish. (button not shown).

End of procedure

Composer now starts your new Project based on the **Blank** template, which includes creating a default interaction process diagram (`default.inprocess`).

The Project Explorer on the left contains all the files and folders associated with the application so far (see [Figure 30](#)).

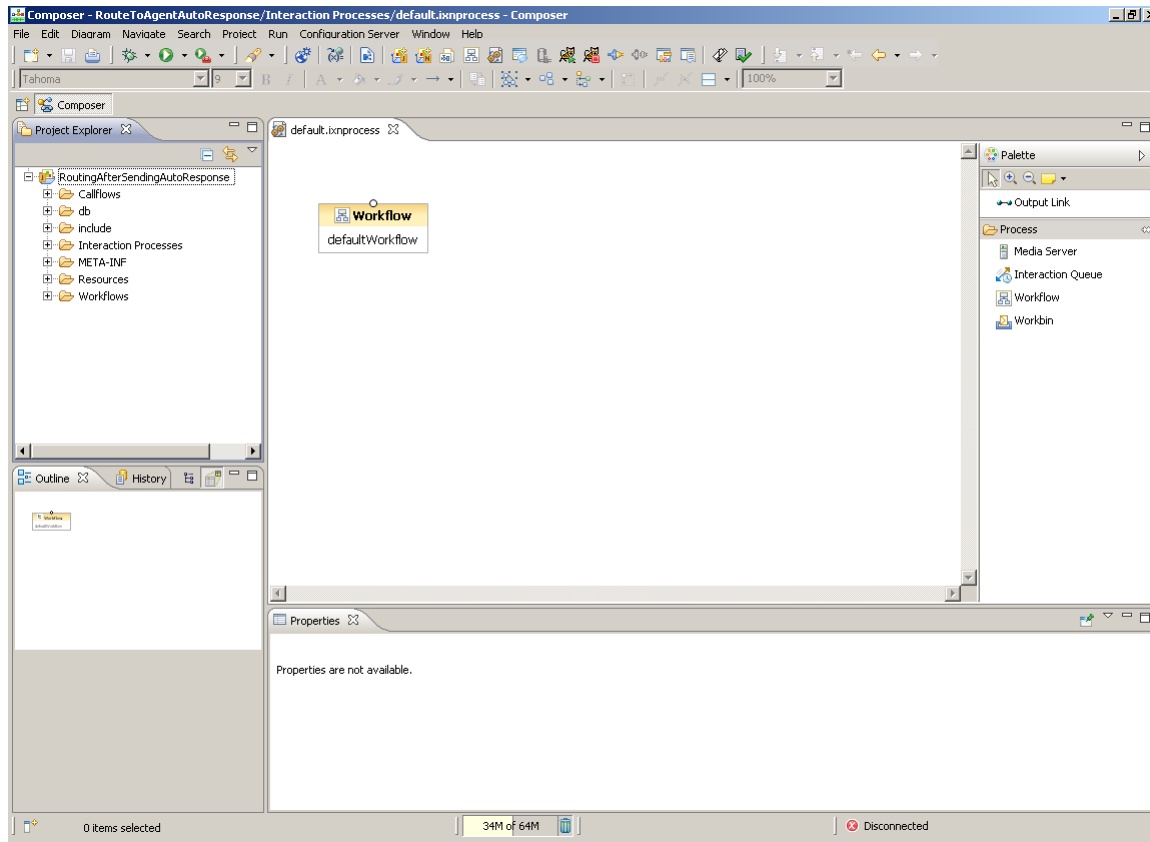


Figure 30: IPD in Composer Perspective, default.ixnprocess Tab

Connecting to Configuration Server

You may develop routing applications:

- With a connection to Configuration Server in “online” mode or
- In an “offline” mode, without connecting to Configuration Server

Whether to connect depends on what you wish to do.

- For example, you would need to connect to Configuration Server in order to view and be able to select Configuration Database objects.
- Otherwise, when working in an offline mode, you can manually type the names of Configuration Database objects. Once you connect to Configuration Server, Composer can then validate that these objects actually exist in your Configuration Database, and warn if there are mismatches.

Procedure: Connecting to Configuration Server

Start of procedure

1. From the Composer main menu, select Configuration Server > Connect. The dialog box in [Figure 31](#) opens.
2. Enter the user name, password, application name, host, and port information for the Configuration Server used in your environment.

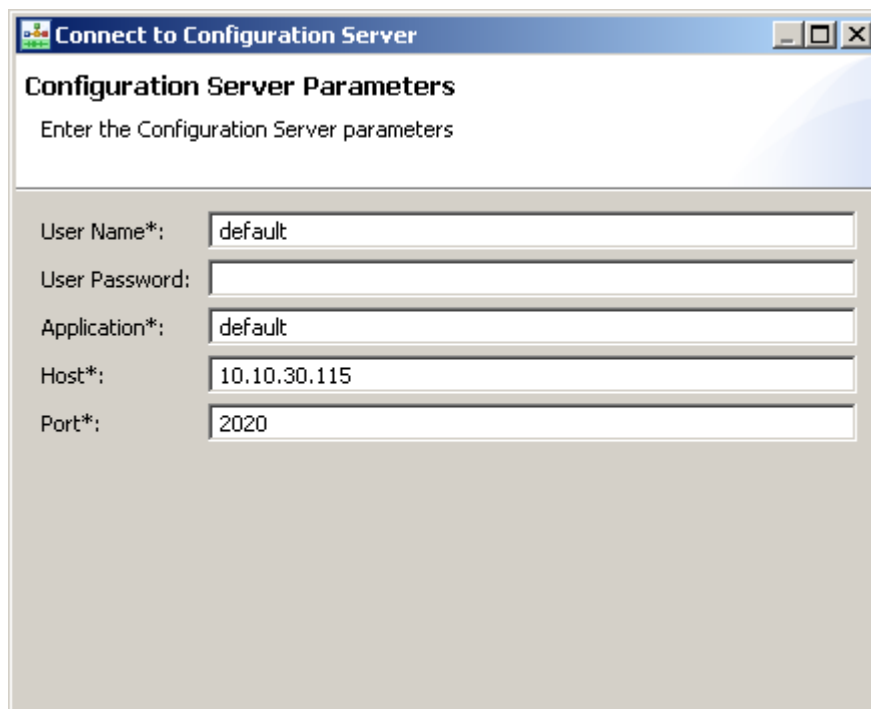


Figure 31: Connect to Configuration Server Dialog Box

3. Click Next (button not shown).
4. Select the tenant. For a single-tenant environment, select Environment.
5. Click Finish. Composer can now access Configuration Server data during validation (if configured to do so) and other operations.

End of procedure

Defining Contact Services Preferences

The Routing after sending Auto-response Project shown in Figure 24 on [page 57](#) does not use Context Services (see [page 53](#)). Information on setting Context Services is included here because many readers will be using those services in routing applications. In this case, you will want to set Contact Services preferences because:

- Customer attributes defined in the Universal Contact Server Database will then be viewable in Expression Builder (see Figure 23 on [page 54](#)).
- Many Context Services blocks (see Table 8 on [page 121](#)) will display customer profile core and extension attributes for selection. While you can work in offline mode, if connected to the Context Services server, you can select these attributes instead of manually entering them in dialog boxes.

Procedure: Connecting to the Context Services Server

Start of procedure

If the Context Services capability is enabled, set preferences as follows:

1. Go to Window > Preferences > Composer > Context Services (see [Figure 32](#)).

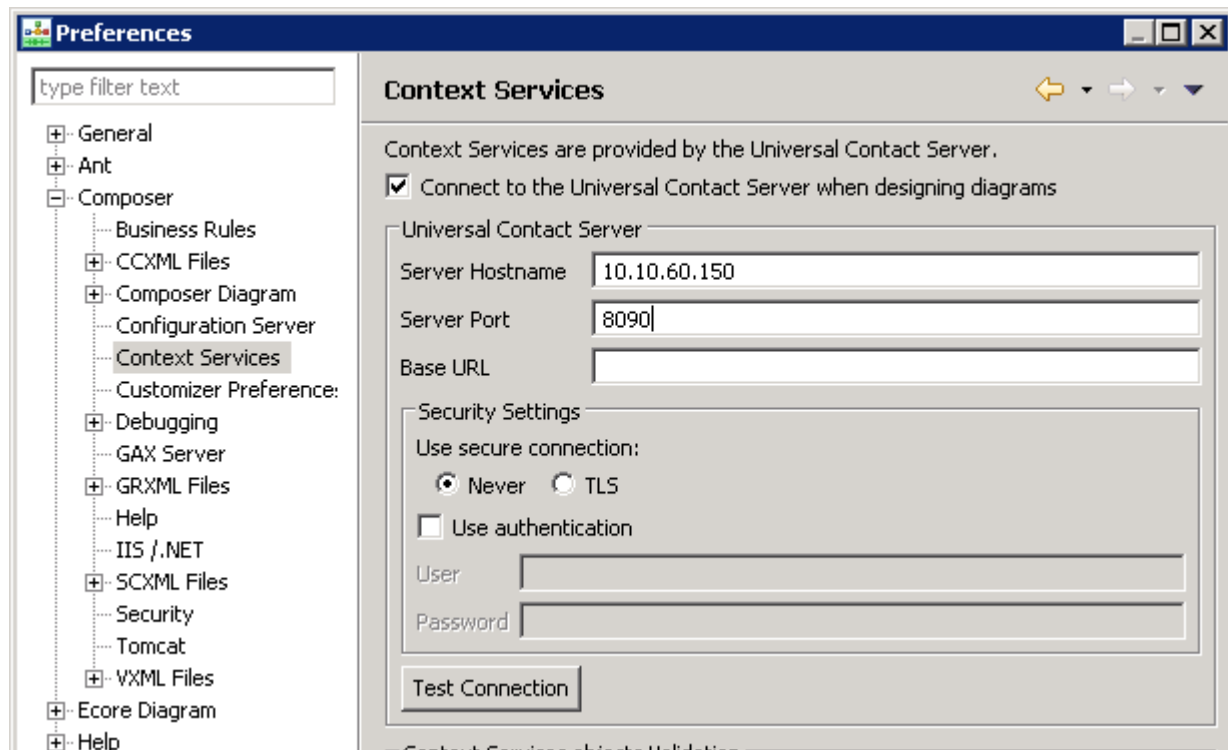


Figure 32: Context Services Preferences

2. Check the following box to specify online or offline mode: Connect to the Universal Contact Server when designing diagrams. This enables the fields below.
3. Under Universal Contact Server, enter the server host name in your Configuration Database, which is the name (or IP address) of the Universal Contact Server.
4. Enter the server port number for Universal Contact Server.

Note: For the port number, open the Universal Contact Server Application object in your Configuration Database, go to the Options tab, select the cview section, and the port option.

5. Enter the base URL for the Context Services server (UCS).
6. Click the Test Connection button. Clicking should cause connection successful to appear. If not, check that Universal Contact Server is running and that the entered host/port values are correct.

Note: If the Context Services Service (CMS) is in Maintenance mode, as described in the *Universal Contact Server 8.0 Context Services User's Guide*, testing the connection will show a failure status message: Connection failure (maintenance mode). This indicates that Composer is able to talk to CMS but CMS must be switched to Production mode before Composer blocks can successfully work with it.

7. Under Context Services object Validation, select one of the following:
 - No validation
 - Validate if connected
 - Validate
8. Click OK.

End of procedure

Entering IPD Properties

When an IPD is in focus, the Properties contains fields associated with the IPD (see Figure 6 on [page 26](#)). Use the procedure below to define general information about the IPD.

Procedure: Entering IPD properties

Start of procedure

1. If not selected, click the `default.ixnprocess` tab to bring the IPD properties into view. Or expand the `Interaction Processes` folder in the `Project Explorer` and double-click `default.ixnprocess`.
2. Enter the fields as follows:
 - a. `Created By`—Enter your name as the author of the Project.
 - b. `Created On`—Auto-populated by Composer to indicate the timestamp when the diagram was created.
 - c. `Designed Using`—Auto-populated by Composer to indicate version of Composer used to create this diagram.
 - d. `Last Modified By`—Provided by you to indicate who updated the diagram last.
 - e. `Last Modified On`—Filled in by Composer when the diagram is modified.
 - f. `Version`—Provided by you for versioning purposes during development.
3. Select `File > Save` from the menu to save the IPD as it exists so far.

End of procedure

Note: Composer can be used with Eclipse Plug-ins, such as those that provide integration to source code management systems, such as ClearCase and Subversion. For more information, see the Integrating with Source Control Systems topic in the *Composer 8.1 Help*.

Adding a Media Server Block

The first block in the IPD shown in Figure 26 on [page 62](#) is a Media Server block. Use the Media Server block to direct interactions of a particular media type (other than voice) into an IPD.

Endpoints

A media server is associated with one or more *endpoints*, with each endpoint connecting the media server to an interaction queue. In order for a Media Server block to show endpoints, those endpoints must first exist in the Configuration Database. Then, after you select the media server application via

the Application property, the endpoint ports appear on the Media Server block as well as being listed in the Properties view. The figure below illustrates this.

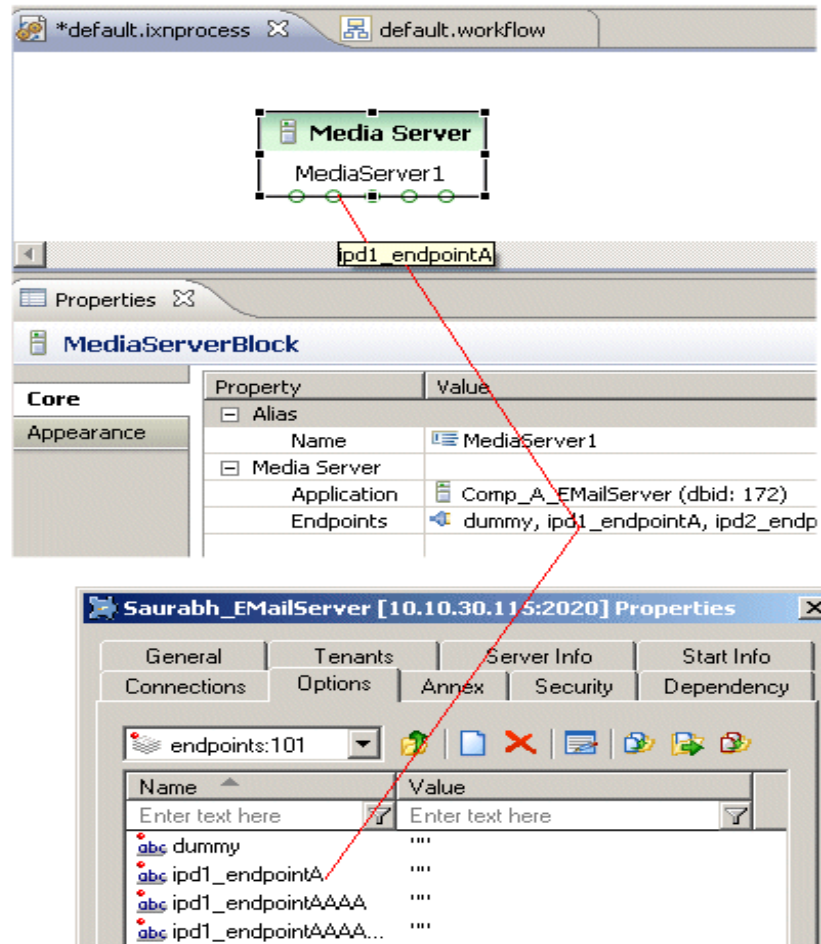


Figure 33: Media Server Endpoints

Procedure: Adding a Media Server Block

Note: Adding a Media Server block in an IPD is a convenience, but is not technically necessary. In Genesys Configuration Manager or Genesys Administrator, you can go into the `Application` object for the media server that will be handling your interactions, and create and configure the endpoints there via the `Options` tab. This will allow you to configure how the interactions get placed into interaction queues.

Start of procedure

1. Outside of Composer, in Genesys Administrator or Configuration Manager, use the instructions in the *eServices (Multimedia) 8.1 Deployment Guide* to create the media server `Application` object and its endpoints.
2. Drag and drop the Media Server block onto the canvas. Or use one of the methods covered in “Adding Blocks to the Canvas” on [page 95](#) to add the Media Server block to the canvas.
3. Configure the following fields in the `Properties` view as shown in [Figure 34](#):
 - a. `Name`—Name the Media Server block in the IPD.
 - b. `Application`—Select a media server to specify the `CfgApplication` object in Configuration Server that this block represents.
 - c. `Endpoints`—Click under `Value` to open a dialog box where you can select one or more endpoints, which can then be connected to interaction queues. When you connect an endpoint to an Interaction Queue block in the IPD diagram, this will cause interactions coming out of this endpoint to go into the named interaction queue.
4. Click the `File` menu and select `Save`.

End of procedure

IPD Block Properties

The `Properties` view for the `MediaServer1` block is shown in [Figure 26](#) on [page 62](#) is shown below in [Figure 34](#).

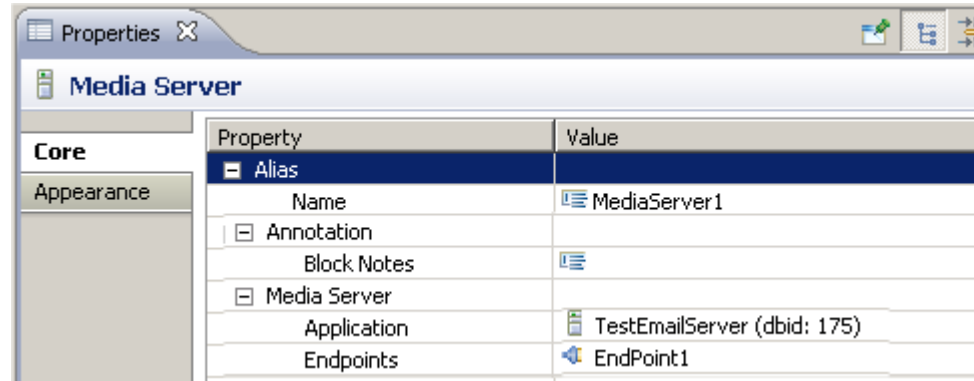


Figure 34: Properties View for MediaServer1 Block

Defining an Interaction Queue

The second block in the IPD shown in Figure 26 on [page 62](#) is an Interaction Queue block. Use it to define a multimedia (non-voice) interaction queue in an IPD and to create a *view*, which can define the conditions for pulling interactions out of the queue for submittal to a workflow. Publishing (see [page 87](#)) pushes this information into the Configuration Database as a CfgScript object of type `InteractionQueue`.

About Interaction Queues:

- Queues that you defined with the Interaction Queue block appear for selection in the Queue Interaction block.
- By design, each Interaction Queue block has only one green output port, which can only be connected to a Workflow block in the IPD.
- No updates to Configuration Server are created until you invoke the Publish operation described on [page 87](#).
- You cannot reuse an existing interaction queue in the same IPD, but you can use the same interaction queue in different IPDs. For more information, see the following topic in *Composer 8.1 Help*: Linking IPDs with Workflows.

Procedure: Defining an Interaction Queue

Start of procedure

1. Drag and drop the Interaction Queue block onto the canvas. Or use one of the methods covered on “Adding Blocks to the Canvas” on [page 95](#) to add an Interaction Queue block.
2. Configure the following fields in the `Properties` view as shown in [Figure 35](#):
 - a. `Name`—Use this property to name the Interaction Queue block in the IPD.
 - b. `Queue Enabled`—Select `true` or `false` to enable or disable this queue in Configuration Server.
 - c. `Queue Description`—Enter a description for the interaction queue. This property will map to the `Description` key in the `Annex` section `Queue` of the `CfgScript` object for the interaction queue.
 - d. `Queue Name`—Mandatory. You must enter a name for the interaction queue, which will appear on the block under the value for the `Name` property. No updates to the Configuration Database are created until you invoke the `Publish` operation.
 - e. `Views`—Use this property to define one or more views for an interaction queue. Each view represents an exit channel from the queue into a workflow.
3. Click the `File` menu and select `Save`.

End of procedure

IPD Block Properties

The `Properties` view for the `IncomingEmailQueue` Interaction Queue block shown in [Figure 26](#) on [page 62](#), is shown in [Figure 35](#).

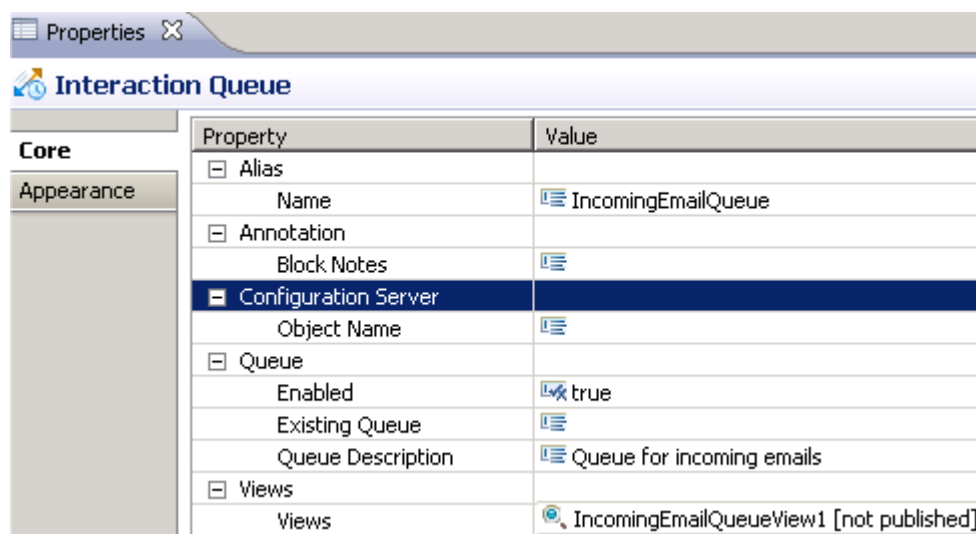




Figure 35: Properties View for IncomingEmailQueue Block

The next procedure describes configuring the Views property shown in [Figure 35](#).

Procedure: Defining a view for an interaction queue

Purpose: To define an exit channel from an interaction queue into a workflow. The view defines the criteria that must be met before an interaction can exit out from the interaction queue. A view can also enable certain types of interactions to be processed earlier than others. It is mandatory to define at least one view for an interaction queue in an IPD although you don't have to define specific conditions.

Start of procedure

1. In an Interaction Queue block, click under Value to display a  button.
2. Click the  button to open the View Properties dialog box.
3. Click Add to display Main, Parameterized Conditions, and Segmentation tabs (see [Figure 36](#) on [page 77](#)).
4. Complete the Main tab. Continuing with this re-creation, use the entries in [Figure 36](#). Summary information on each field is presented below.
 - a. Enabled—Check the box to make the view ready to extract interactions.
 - b. Name—Enter a name for the view to be used when saving as a Configuration Database Script object of type InteractionQueueView.
 - c. Description—Enter text describing the view.

- d. **Check Interval**—Enter the number of seconds to specify the frequency (time interval) that Interaction Server will use to check the queue and, if necessary, adjust the number of interactions that can be submitted to the workflow based on the `Scheduling` property.
- e. **Condition**—You have the option of creating an expression to be used as the basis for extracting interactions from the queue.
- f. **Order**—You have the option of defining the order for extracting interactions from the queue.
`order := [property_order[, order]]`
`property_order := property_name [asc|desc]`
 Example: "priority desc, ReceivedAt"
 Extracts according to priority and the received at timestamp if priority is the same. For more information on the interaction attributes that can be used for ordering, see the section on System Properties in the chapter on Interaction Properties in the *eServices (Multimedia) 8.1 User's Guide*.
- g. **Scheduling**—Use to specify the scheduling condition that Interaction Server should use, based upon the scheduled time contained in interactions. The interaction scheduling functionality uses a database field called `scheduled_at`, which is mapped to an interaction property called `ScheduledAt`. For information on this field, see the chapter on interaction properties in the *eService/Multimedia 8.1 User's Guide*.
- h. **Database Hints**—This field is only applicable to an Oracle database. You can apply a Hint, which will cause Oracle to use a specific index to optimize performance.

Note: For detailed information on all tabs, consult the *Composer 8.1 Help*, Interaction Queue Views topic.

5. After you complete the applicable fields in these tabs, click OK to close the View Properties dialog box. After publishing, each view will be created as a separate `CfgScript` object of type `InteractionQueueView`.
6. Select `File > Save` from the menu to save the IPD as it exists so far.

End of procedure

[Figure 36](#) shows the View Properties dialog box for the `IncomingEmailQueue` defined in the first Interaction Queue block in the IPD.

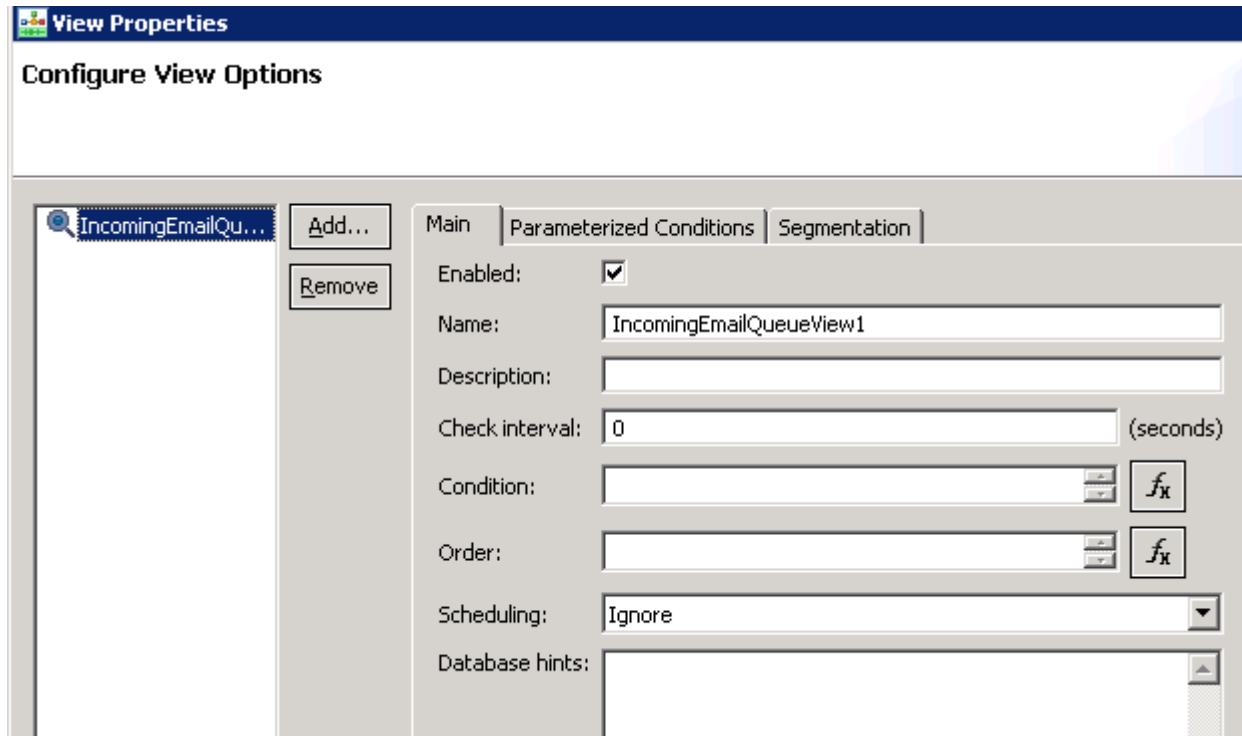


Figure 36: View Properties Dialog Box for IncomingEmailQueue

The Main tab indicates no special condition for pulling interactions from the queue. The Parameterized Conditions and Segmentation tabs are not used in the Project.

Note: Important! While you can define multiple views for an interaction queue, the Interaction Queue block will have only one output port and therefore will feed interactions to only one workflow strategy.

Connecting the Blocks

So far, we have created the Media Server block and the first Interaction Queue block. At this point, you can connect the Media Server block Endpoint to the top input port of the Interaction Queue block. If necessary, use the information in “Connecting Blocks” on [page 96](#).

IPD Flow So Far

When an e-mail (new multimedia interaction) arrives at the Media Server Endpoint1, it gets queued to IncomingEmailQueue (see Figure 35 on [page 75](#)).

Creating a Workflow Diagram

Prior to adding a Workflow block, you may wish to configure the workflow diagram or SCXML file to which interactions should be sent for processing. You can then select it in the Workflow block properties.

Note: The procedure below is a very high-level summary of the steps to create a workflow diagram. A detailed example is available in the topic “Your First Application: DNIS Routing” in the Building SCXML-Based Workflows book of the *Composer 8.1 Help*.

Procedure: Creating a workflow diagram

Start of procedure

1. Click the button on the main toolbar to create a new workflow (see Figure 21 on [page 41](#)) and continue with step 2. Alternatives:
 - Select **File > New > Workflow Diagram** or select **File > New > Other**. In the New dialog box, expand **Composer > Diagrams**. Select **Workflow Diagram** and click **Next**. Continue with step 2.
 - Right-click the **Workflows** folder in the Project Explorer and select **New > Other > Workflow Diagram**. Continue with step 2.
 - Or use the keyboard shortcut: **Ctrl+Alt+R** and continue with step 2.
2. In the Main workflow tab, select **Empty Diagram** and click **Next**.
3. Select the parent Project. In this example, it is `RoutingAfterSendingAutoResponse`.
4. Name the diagram (must have an extension of `.workflow`). In this example, the name is `RouteToAgentWithAutoResponse.workflow` (see Figure 40 on [page 83](#)).
5. Click **Finish**. The **Workflows** folder in the Project Explorer shows the name of your diagram under your Project.
6. Since each workflow diagram starts with an Entry block and ends with an Exit block, you can place those blocks now. The Entry block is where you define variables (see “Variables: Project and Application” on [page 34](#)).
7. Build the workflow diagram.
8. Validate the code by selecting **Diagram > Validate**. You can also click the Validate icon (see Figure 21 on [page 41](#)) on the upper-right of the Composer main window when the workflow canvas is selected. The Problems tab shows the results of validation for this particular Resource. Fix any problems before continuing.

9. Generate the code (see [page 89](#)).

End of procedure

Continuing with our routing application re-creation, build the routing strategy shown in [Figure 37](#). Use the Properties view entries shown in [Figure 38](#) on [page 80](#) and [Figure 39](#) on [page 81](#).

RouteToAgentWithAutoResponse.workflow

The workflow diagram referenced by the RouteToAgent Workflow block in [Figure 26](#) on [page 62](#) is shown in [Figure 37](#).

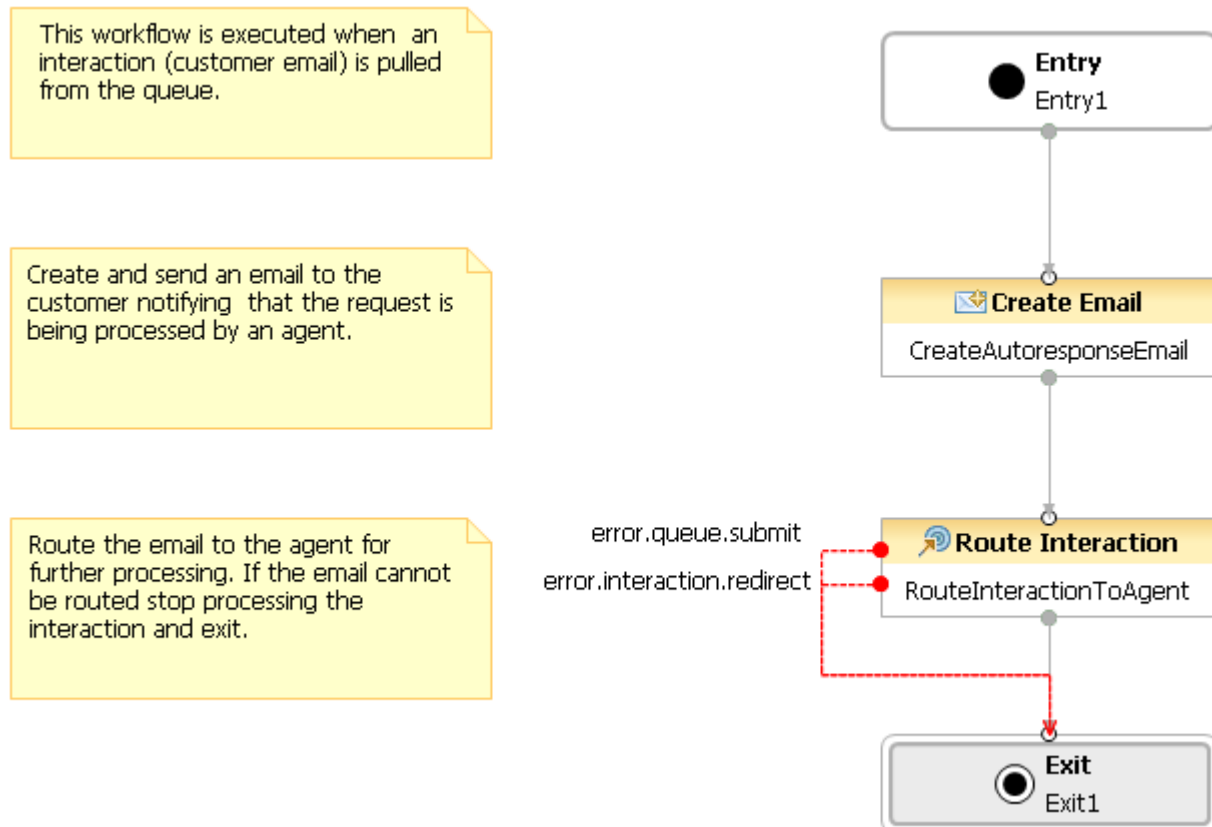


Figure 37: RouteToAgentWithAutoResponse Workflow

The RouteToAgentWithAutoResponse workflow:

- Uses the Create E-mail block to send out an auto-response e-mail to the customer notifying that the system has received their e-mail and someone will be contacting them shortly.
- Uses the Route Interaction block to route the original e-mail from the customer to an agent group for a response. This block also specifies a suggested queue for the new interaction (agent's response e-mail).

Workflow Diagram Block Properties

The Properties view for the `CreateAutoresponseEmail` block in [Figure 37](#) is shown in [Figure 38](#).

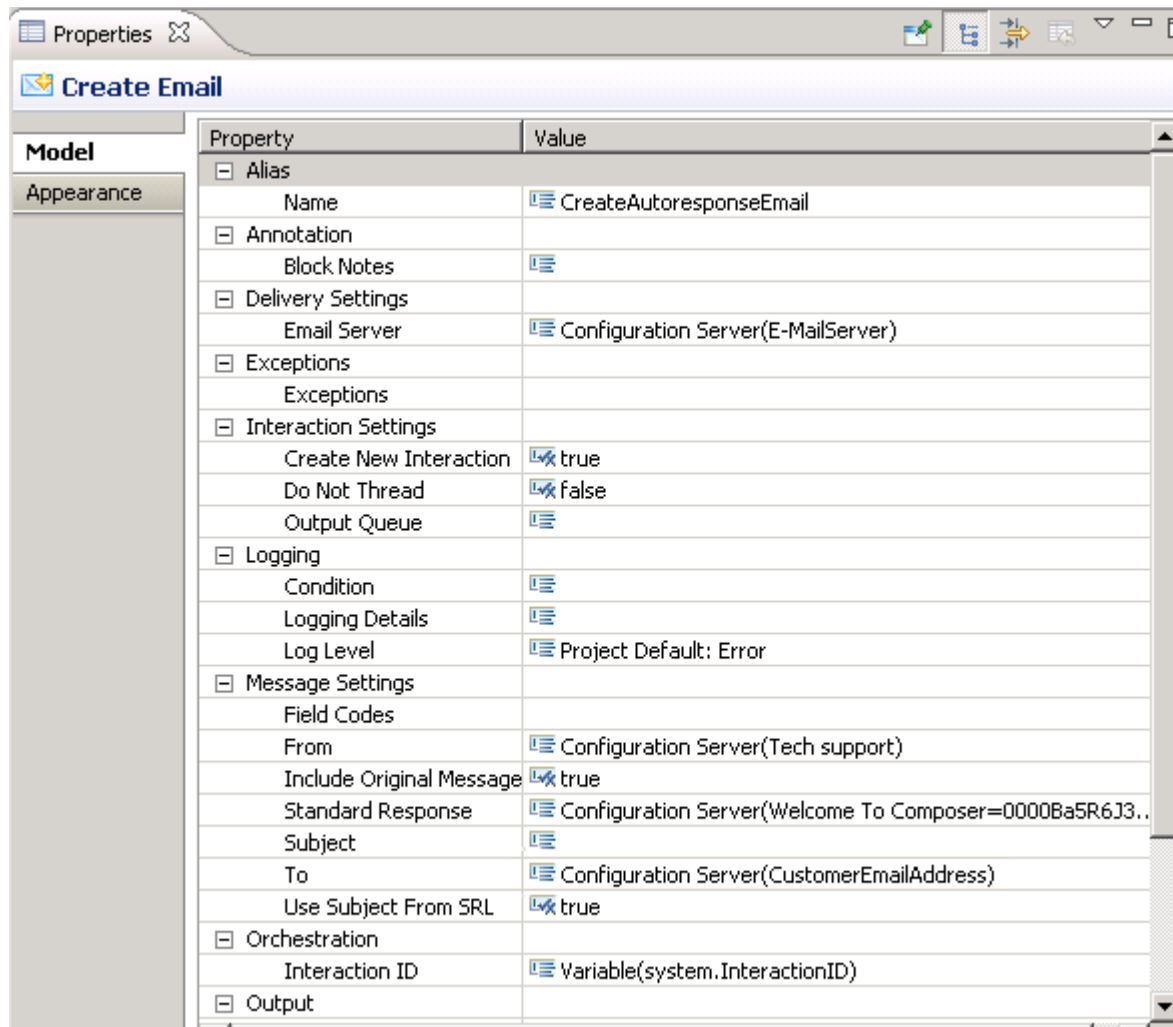


Figure 38: Create AutoresponseEmail Block Properties

The Properties view for the `RouteInteractionToAgent` block [Figure 37](#) is shown in [Figure 39](#).

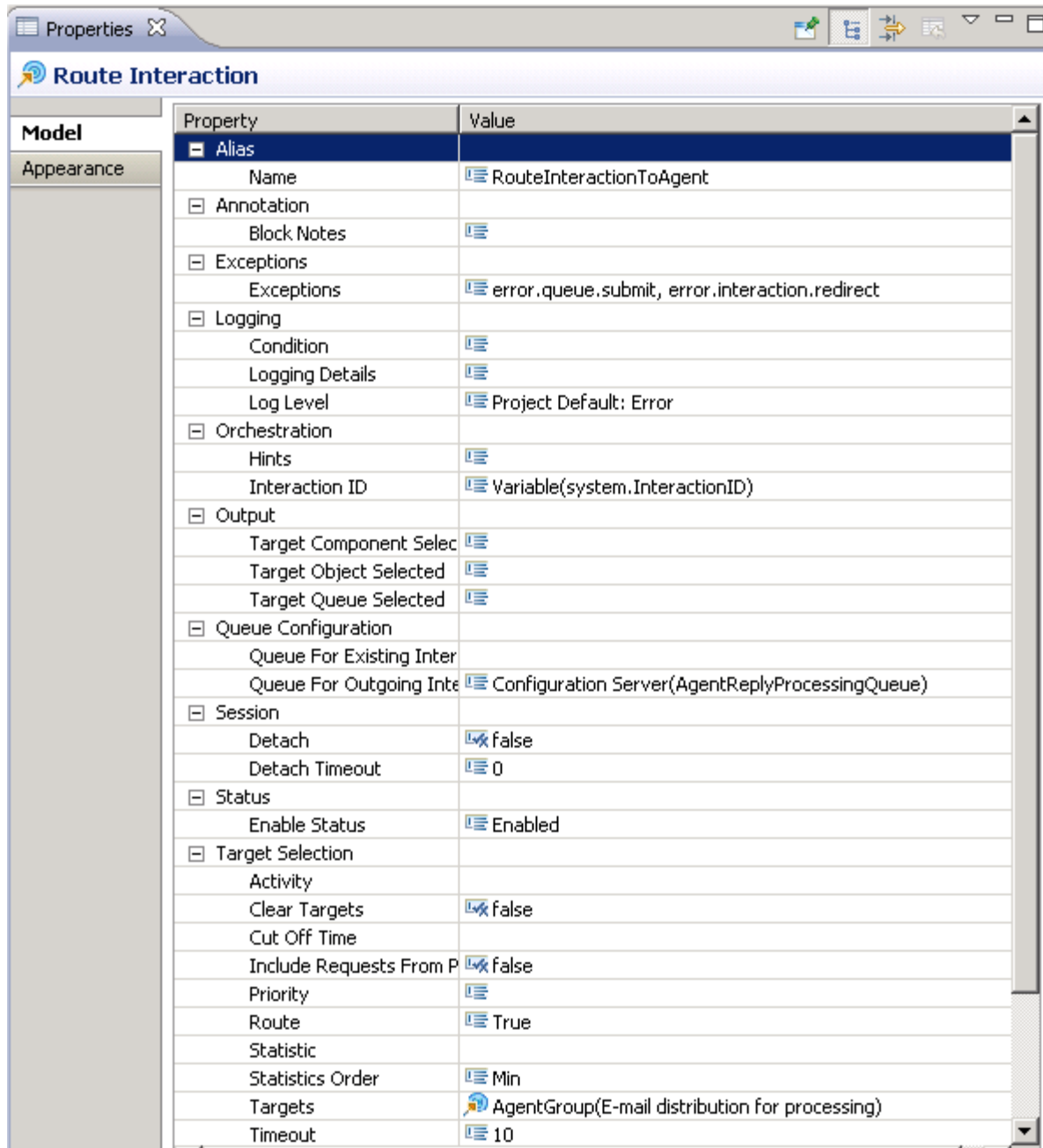


Figure 39: RouteInteractionToAgent Block Properties

Adding a Workflow Block



The third block in the IPD we are re-creating (see Figure 26 on [page 62](#)) is a Workflow block. Use this block in an IPD to point to a workflow resource

(workflow diagram or SCXML file) to which interactions should be sent for processing.

Note: The `RouteToAgentWithAutoResponse` workflow diagram is already configured (see Figure 37 on [page 79](#)) in this example, so it can be selected from the Workflow block.

Procedure: Adding a Workflow block

Start of procedure

1. Drag and drop a Workflow block onto the canvas. Or use one of the methods covered on “Adding Blocks to the Canvas” on [page 95](#) to add the Workflow block to the canvas.
2. Configure the following properties using the entries in [Figure 40](#).
 - a. **Name**—Use this property to name the Workflow block in the IPD. Continuing with the routing application re-creation, name the block `RouteToAgent` as shown in Figure 26 on [page 62](#).
 - b. **Resource**—Use this property to point to a workflow diagram or SCXML file.
 - i. Click under **Value** to display the  button.
 - ii. Click the  button to open the **Select Resource** dialog box.
 - iii. Select the workflow resource, which can be a workflow (strategy) diagram that exists in any of the Projects in the Composer workspace or an SCXML file created in Composer's SCXML Editor (see [page 22](#)). In this case, select the `RouteToAgentWithAutoResponse` workflow (see Figure 37 on [page 79](#)).
 - iv. Click **OK**.

End of procedure

IPD Block Properties

The **Properties** view for the `RouteToAgent` Workflow block in Figure 26 on [page 62](#) is shown in [Figure 40](#).

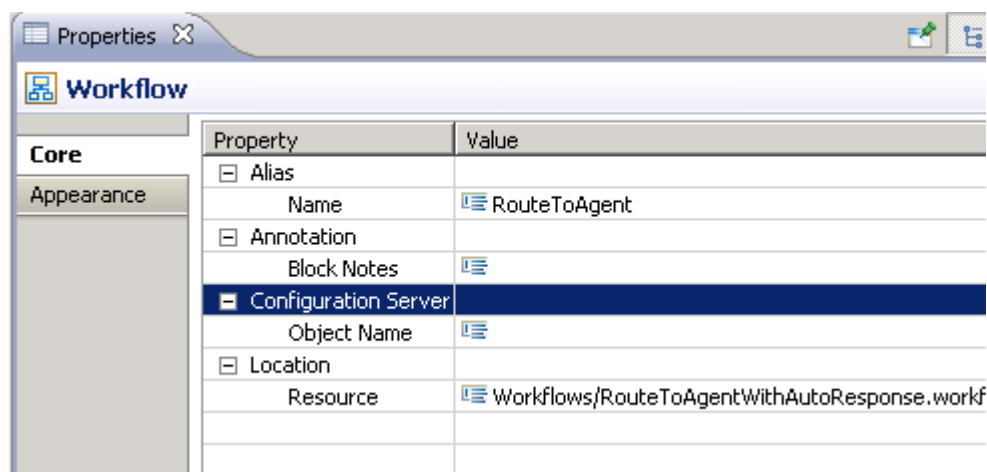


Figure 40: Properties View for RouteToAgent Workflow Block

Workflow-Generated Blocks

Workflow-generated blocks help you visually interpret an IPD. They take the form of outgoing connections and plain white blocks, that automatically appear from a Workflow block. They represent objects specified inside the Workflow block, such as queues, routing targets, and stop processing instructions. Accordingly, Composer generates the following types of workflow-generated blocks:

- Queue Reference
- Dynamic Target
- Stop

Since these blocks are automatically generated by Composer, they are not available on the IPD palette. Composer uses a different color, appearance, and structure for them allowing you to easily differentiate them from other blocks.

In the IPD we are re-creating (see Figure 26 on [page 62](#)), there are three workflow-generated blocks:

- CreateAutoResponse Queue Reference
- RouteInteractionToAgent Dynamic Target
- RouteIxnToSupervisor Dynamic Target

Connecting the Blocks

Previously we created and connected the Media Server block and the first Interaction Queue block in Figure 26 on [page 62](#). You can now connect the first Interaction Queue block to the first Workflow block in the flow. If necessary, use the information in “Connecting Blocks” on [page 96](#).

IPD Flow So Far

1. When a new multimedia interaction (an e-mail in this case) arrives at the Media Server Endpoint1, it gets queued to IncomingEmailQueue (see Figure 35 on [page 75](#)).
2. Orchestration Server then pulls the interaction from the queue and starts processing the workflow referenced by the RouteToAgent workflow block (see Figure 40 on [page 83](#)).
 - a. The RouteToAgentWithAutoreponse workflow uses the Create E-mail block (see Figure 37 on [page 79](#)) to send out an auto-response e-mail (see Standard Response property in Figure 38 on [page 80](#)) to the customer, notifying that the system has received their e-mail and someone be contacting them shortly.
 - b. After sending the auto-response e-mail, the original e-mail from the customer is then routed to the E-mail distribution for processing agent group for a response (see the Targets property in Figure 39 on [page 81](#)). The Route Interaction block is used for this purpose.
 - c. The Route Interaction block also specifies a suggested queue for the new interaction (see the Queue for Outgoing interaction property in Figure 39 on [page 81](#)).

Note: The next two steps (3 and 4) occur outside of the IPD, but could be documented in the IPD with Notes (although this has not been done in the Project template shown in Figure 26 on [page 62](#)).

3. After the agent responds (using Reply feature in Genesys Agent Desktop), the response is now treated as a new interaction and is then placed into the suggested queue.
4. The agent then closes the current interaction (using Done feature in Genesys Agent Desktop).

Adding the Remaining IPD Blocks

To finish creating the IPD shown in Figure 26 on [page 62](#), we must add another Interaction Queue block and another Workflow block.

Defining Second Interaction Queue

The sixth block in the IPD shown in Figure 26 on [page 62](#) is an Interaction Queue block as previously described on [page 73](#). The function of this Interaction Queue block is to define the interaction queue for the agent reply e-mail. As shown in Figure 39 on [page 81](#), the queue for the outgoing interaction is AgentReplyProcessingQueue. This name is reflected in both the

name of the Interaction Queue block and in the Properties view for the block (see [Figure 41](#)).

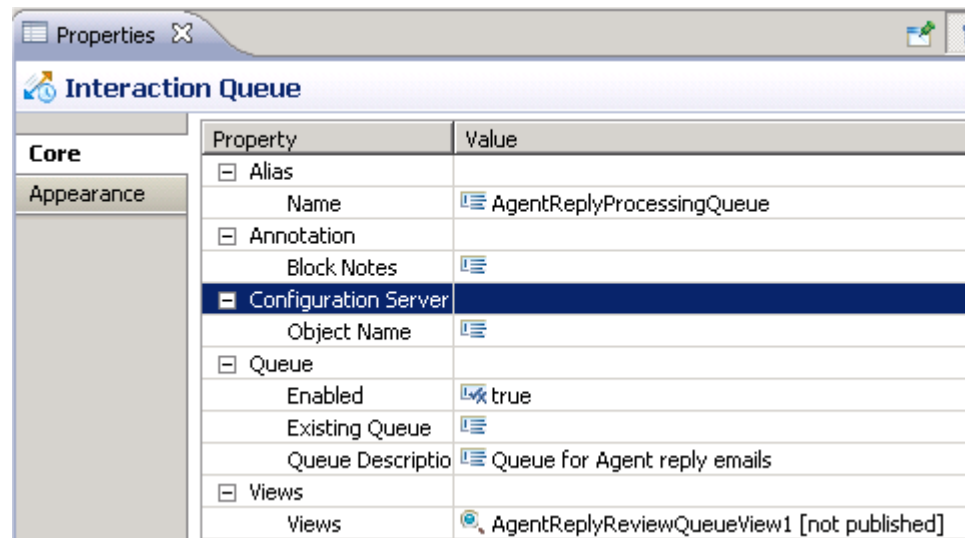


Figure 41: Properties View for AgentReplyProcessingQueue Block

As described on [page 73](#), a view is responsible for pulling interactions out of queues for submittal to a workflow. [Figure 42](#) shows the view configuration for the AgentReplyProcessingQueue.

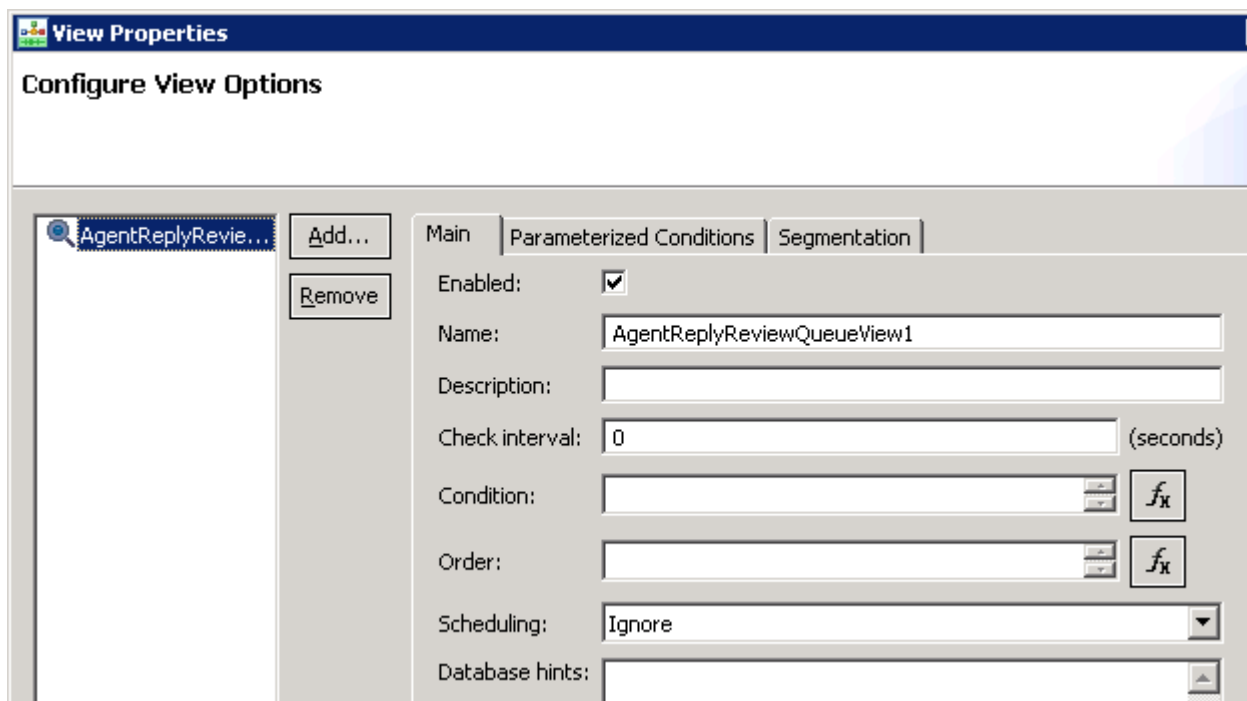


Figure 42: View Properties Dialog Box for AgentReplyProcessing Queue

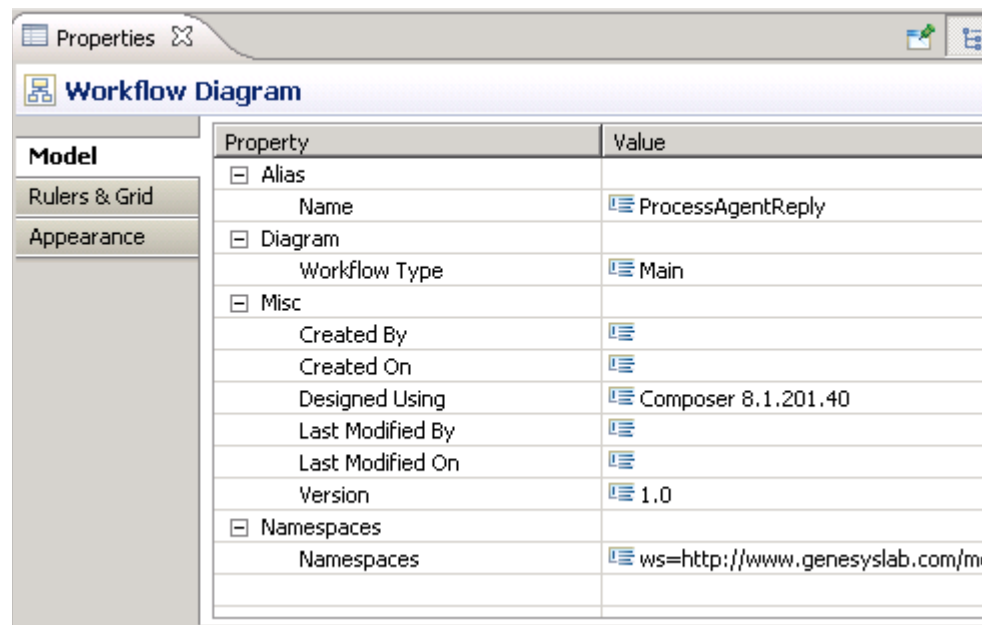
The Main tab indicates no special condition for pulling interactions from the queue. The Parameterized Conditions and Segmentation tabs are not used in the Project. Fields in this dialog box were previously summarized on [page 75](#).

Adding a Second Workflow Block

The seventh block in the IPD shown in Figure 26 on [page 62](#) is another Workflow block as previously described on [page 81](#). The function of ProcessAgentReply Workflow block is to specify the next workflow resource to which interactions should be sent for processing.

Note: The process for adding a Workflow block was previously described in [Procedure: Adding a Workflow block](#), on [page 82](#).

As can be seen in the Properties view for this second Workflow block, the name of the workflow resource is ProcessAgentReply.workflow (see [Figure 43](#)).



Property	Value
Alias	
Name	ProcessAgentReply
Diagram	
Workflow Type	Main
Misc	
Created By	
Created On	
Designed Using	Composer 8.1.201.40
Last Modified By	
Last Modified On	
Version	1.0
Namespaces	
Namespaces	ws=http://www.genesyslab.com/m

Figure 43: Properties View for ProcessAgentReply Workflow Block

ProcessAgentReply.workflow

The workflow diagram referenced by the ProcessAgentReply Workflow block in Figure 26 on [page 62](#) is shown in [Figure 44](#).

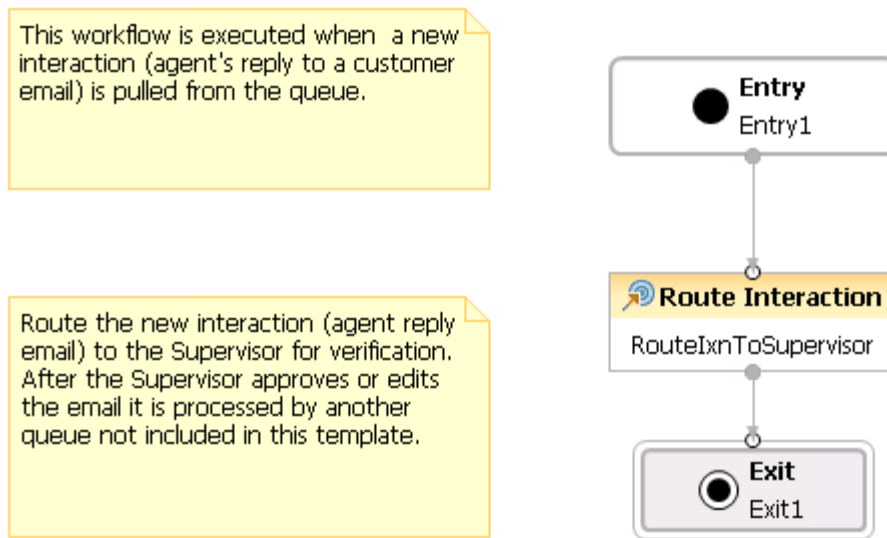


Figure 44: ProcessAgentReply Workflow

When Orchestration Server pulls the new interaction from the AgentReplyProcessing queue, the ProcessAgentReply workflow gets executed.

- ProcessAgentReply is a simple workflow which routes the agent response e-mail to a supervisor for review.
- Once the supervisor is satisfied with the response, the e-mail is placed in the system queue and sent out to the customer.

At this point, there are no further interactions that need to be processed and the application (session) exits.

Publishing an IPD

Publishing an IPD validates Project configuration information and pushes the information out to Configuration Server. For example, when you configure an Interaction Queue block or a Workflow block, Composer does not send the information to Configuration Server until you invoke the Publish operation. This gives you complete control of the update process. Or you can set preferences to automatically publish upon saving.

Procedure: Publishing an interaction process diagram

Start of procedure

1. Before publishing, you must connect to Configuration Server as described on [page 66](#).
2. Right-click an IPD in the Interaction Processes folder in the Project Explorer.
3. Select **Publish to Configuration Server**.
4. As an alternative to the above steps, click the tab on the canvas that contains the IPD and click the toolbar button to publish the active IPD to Configuration Server (see Figure 21 on [page 41](#)).

End of procedure

Once these objects are created successfully in Configuration Server, some manual configuration is still required before interactions can work. For example, to redirect e-mails to an endpoint (see Figure 33 on [page 71](#)), you must set “endpoint” point key in the `pop-clientX` section of the e-mail server Application to the correct end-point. For more details, see the Deploying a Routing Application topic in the Validation, Debugging, & Deployment book of the *Composer 8.1 Help*.

Generating Code

You must generate source code for both the IPD and associated workflow diagrams. All generated files go into the `src-gen` folder of the Project.

Procedure: Generating code for an interaction process diagram

Start of procedure

1. Click the tab on the canvas that contains the IPD.
2. Click the toolbar button to generate code (see Figure 21 on [page 41](#)). The generated file name in the `src-gen` folder will follow the format:
`<ipd_diagram_name>_<workflow_block_name>.scxml`

End of procedure

One file will be generated for each interaction queue. Functionally, these files will be equivalent. Generating one file per queue makes it easier to configure the application URL manually. As the developer, you will know the interaction queue you have submitted the interaction to and therefore can easily identify the correct SCXML page.

Testing a Routing Application

Composer provides real-time debugging capabilities for SCXML-based Orchestration Server (ORS) applications. Debugging can be started on an existing session or it can wait for the next session that runs the application at a given URL. You can debug:

- A workflow diagram built with Composer.
- Any SCXML application or set of SCXML pages regardless of whether they were created with Composer.
- Any combination of workflow diagrams and SCXML pages.

Procedure: Using the ORS Debugger

For details on each step summarized below and ORS Debugger Limitations, see the Debugging Routing Applications topic on the [Composer Help Wiki](#).

Start of procedure


1. Set preferences (see Figure 22 on [page 42](#)) for the ORS Debugger: Window > Preferences > Composer > Debugging > ORS Debugger.
2. If using Context Services, set Context Services preferences: Window > Preferences > Composer > Context Services (see “Defining Contact Services Preferences” on [page 68](#))
3. Create a launch configuration (in order to debug, a launch configuration must exist). One method is to right-click on the diagram/SCXML file in the Project Explorer. Select either Run As > Run Configurations or Debug As > Debug Configurations.
 - In Run mode, metrics (call traces) are displayed and the application continues without stopping at any breakpoints. When the application executes, these metrics can describe, for example, state transitions, ECMAScript executions, and execution warnings or errors.
 - In Debug mode, debugging pauses at breakpoints. You can step over the code, inspect variable and property values, and execute any ECMAScript from the query console.

In either mode, a dialog box opens for creating the launch configuration.

4. Expand Composer - ORS Debugger.
5. Click the button for a new launch configuration or right-click and select New.
 - If using Run mode, the Run Configurations dialog box opens.
 - If using Debug mode, the Debug Configurations dialog box opens.
6. Use Table 1 on [page 91](#) to create a launch configuration.
7. When finished creating the launch configuration, click Apply and Debug (Debug mode) or Apply and Run (Run mode).

Note: After you launch, debugging doesn’t start until ORS starts the session. You can start the session with a SIP call or multimedia interaction or using the ORS REST API (you need to send a POST request to ORS). If the debugging session can’t be started, a dialog box appears with an error message.

In both modes, metrics received from ORS (call traces) display in the Call Trace view.

8. If in Debug mode, click the Step Over  button to single-step over the blocks. View Application state values in the Variables tab. You can also:
 - Input breakpoints from the Breakpoints view or use the context menu on a block and select Toggle Breakpoint. When breakpoints are set, you can press F5 or click Resume to resume the call to the next breakpoint, instead of single stepping block-by-block.

- Change values of variables in the middle of a workflow. This could be used to quickly change the execution path as the call is progressing. Right-click in the Expressions tab and select **Add Watch Expression**. In the **Add Watch Expression** window, add a new expression to watch during debugging.
To change the value, expand the variable, right-click on the child item and select **Change value**. A pop-up window will open, and you can specify the new value. Click **OK**. Proceed with debugging of the application and see the changed value.
9. Review the `debugging-results` folder in the Project Explorer. Clean up the debugging results by deleting the `ors-debug.<timestamp>` folders from the Project Explorer.

End of procedure

Creating a Launch Configuration

Use [Table 1](#) to complete the Launch Configuration dialog box. Consult the [Composer 8.1 Help](#) if you need additional detail.

Table 1: Debug and Run Launch Configuration

Property	Debug Launch Configuration	Run Launch Configuration
Workspace Storage Location tab		
Workspace Location	Specify the Project name and location for saving SCXML pages executed by ORS. Specify the Project name and location.	Same as Debug
Create Automatically	As an alternative, click Create Automatically to have the Debugger create a new Project folder.	Same as Debug
ORS Debugger Launch tab		
ORS Connection Address	Enter the IP address or hostname of the ORS server.	Same as Debug
ORS Connection Port	Enter the debugger port of the ORS server. This is defined in ORS configuration as <code>[scxml]:debug-port</code> , and defaults to 7999. Make sure that ORS has <code>debug-enabled</code> set to true as well.	Same as Debug. The Address and Port fields reflect the ORS Server Host Name and ORS Server Port previously entered as ORS Debugger Preferences, but can be changed.

Table 1: Debug and Run Launch Configuration (Continued)

Property	Debug Launch Configuration	Run Launch Configuration
Application is a: Workflow SCXML	Select Workflow to step through the diagram or SCXML if code. If unchecked, it will step through the SCXML code.	Select SCXML. Leave unchecked to step over the SCXML code. Select Workflow to step over a diagram.
Path	Enter the workspace-relative path of the workflow diagram.	Specify the workspace-relative path of the SCXML file. For example, /MyProject/src-gen/IPD_default_defaultWorkflow.scxml.
Associated IPD	Optional. Enter the name of the interaction process diagram (IPD) associated with the SCXML file to be debugged. This field is optional because it is possible to run a stand-alone SCXML. Most of the time, you will use launch shortcuts (right-click on workflow or SCXML and select Run/Debug As). The fields in the launch configurations are filled in automatically.	Same as Debug

Table 1: Debug and Run Launch Configuration (Continued)

Property	Debug Launch Configuration	Run Launch Configuration
Step Through IPD	If enabled and debugging in code mode (as opposed to workflow mode), then the Debugger steps through the SCXML code that is generated from the IPD. Otherwise, it will "skip" through that code. The SCXML code generated from an IPD is generally setting up global variables and functions, so you might not want to go through that every time.	Same as Debug
Attach to Existing Session	If enabled, ORS will start debugging on an existing session. When you launch the debugging session, Composer will prompt for a session ID in a dialog box. Once you enter the session ID, it will enter debugging mode for that session. If not enabled, ORS will wait for the next session that runs the application at a given URL. Note: The URL will point to the SCXML page that should be debugged. ORS will enter debug mode for the next session that is started for this URL.	Same as Debug

Deploying the Application

Deploying a routing application on an application server is beyond the scope of this guide. For information on this subject, see the Deploying Composer Applications topic on the [Composer Help Wiki](#).

6

Working with Blocks and Diagrams

This chapter describes how to work with Composer diagram editor. It contains the following sections:

- [Working with Blocks, page 95](#)
- [Perspectives, page 97](#)
- [Viewing More Than One Diagram, page 102](#)
- [Viewing Properties for Two Blocks, page 104](#)
- [Saving Workflow Diagrams as Templates, page 105](#)
- [Custom Blocks, page 109](#)

Working with Blocks

In Figure 47 on [page 99](#), note the various categories of workflow diagram blocks: Flow Control, Routing, Voice Treatments, Server Side, Context Services, and eServices. See Appendix , “Composer Blocks,” on [page 113](#) for information on blocks in each category and deciding which blocks to use.

To display blocks for a given category, simply click the category name; for example, click Flow Control.

Adding Blocks to the Canvas

There are a few ways to add blocks from the palette to the canvas. The most common methods are as follows:

- Click on the block icon on the palette, release the mouse and click on the target location on the canvas area.
- Double-click a block icon on the palette.

- Click on the block icon on the palette, and while holding down the mouse button, drag and drop the block to the canvas.

Any of these methods will add the new block and you can then type the name of the block on the canvas itself.

Connecting Blocks

Blocks are connected to each other using connection links (see `Output Link` and `Exception Link` under `Palette` in Figure 47 on [page 99](#)). There are two types of connection links:

- Use `Output Link` to connect one block's output port to another block's input port, and
- Use `Exception Link` to indicate error or exception conditions by connecting from a block's exception port to another block's input port.

Find the connection links at the top of the palette on the right side of the Composer window.

Method #1

Procedure:

Using the Output Link to connect blocks

Start of procedure

1. Click the `Output Link` (or `Exception Link`) icon on the palette.
2. Move the mouse over to the source block. The cursor will change to an upward arrow.
3. Click once on the source block and keep the mouse button pressed. Then drag the mouse onto the target block and release the mouse button. This will add the connection link between the two blocks.

End of procedure

To use an `Exception Link`, the source block must have an exception port defined. This is done by selecting at least one supported exception within the block's `Exceptions` property.

Method #2

Another method for adding an `Output Link` or `Exception Link` between two blocks is as follows:

1. Click once on the source block to select it.

2. Hold the **Ctrl** key and click once on the target block to select it as well.
3. Double-click **Output Link** (or **Exception Link**) in the palette to create a connection between the two blocks.

Method #3

Some users may find this method convenient:

1. Start dragging from a connection port of the originating block.
2. Drop the block into a blank portion on a canvas (any place in the canvas other than a block will also do). A menu representation of the palette will pop up showing a list of all block types.
3. Click the correct block type. A new block of that type will be added and the originating block will be connected to it.

Note: When connecting to a Treatment block, it is your responsibility to verify that your Switch/T-Server combination supports the treatment you select. Composer does not perform a compatibility check.

Perspectives

Within Composer, a perspective is an arrangement of different sections of the GUI in a manner that facilitates easy use of a particular feature, such as design or debugging. All of the figures shown so far in this chapter show Composer perspective as indicated by the Composer button being selected underneath the toolbar (see [Figure 45](#)).

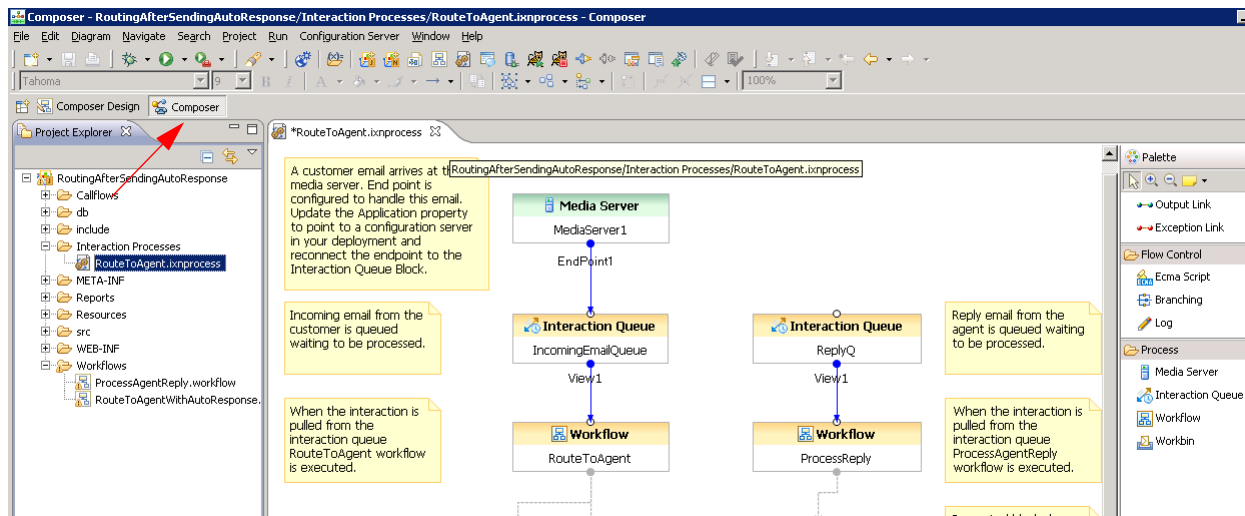


Figure 45: Bar for Perspective Buttons, Composer Perspective Selected

Changing Perspectives

To change perspectives, click the button to open a perspective shown below to drop down a menu of available perspectives (see [Figure 46](#)):

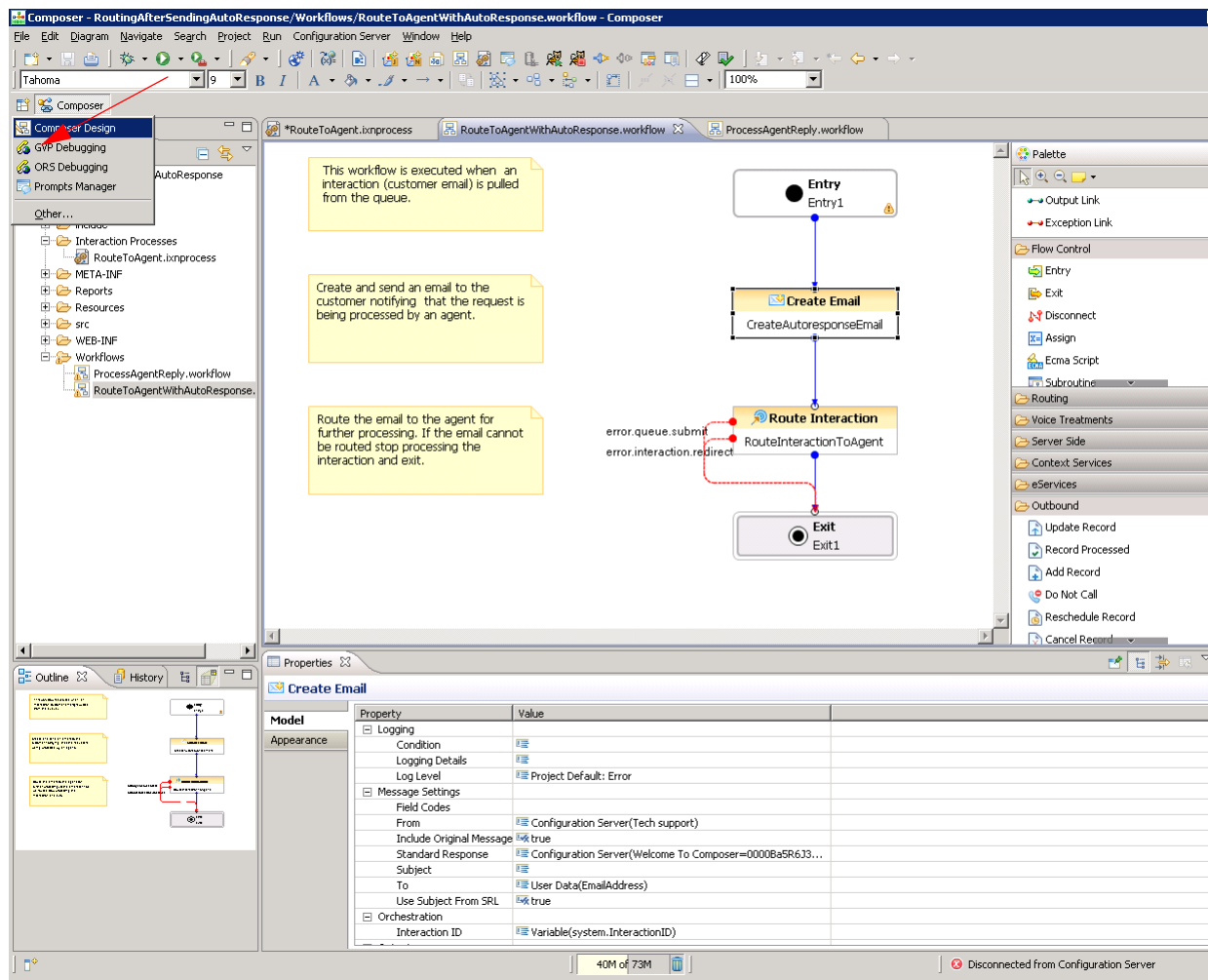


Figure 46: Open Perspective Button

When creating routing applications, you will use either Composer perspective or Composer Design perspective. When debugging, you will use ORS Debugging perspective. The other perspectives on the menu are used for Composer voice applications (not covered in this guide) or are listed there by default since Composer is based on Eclipse (see [page 9](#)).

Assume you select Composer Design perspective as shown in [Figure 46](#). The GUI changes to be more streamlined for placing, connecting and configuring blocks in the canvas as shown in [Figure 47](#):

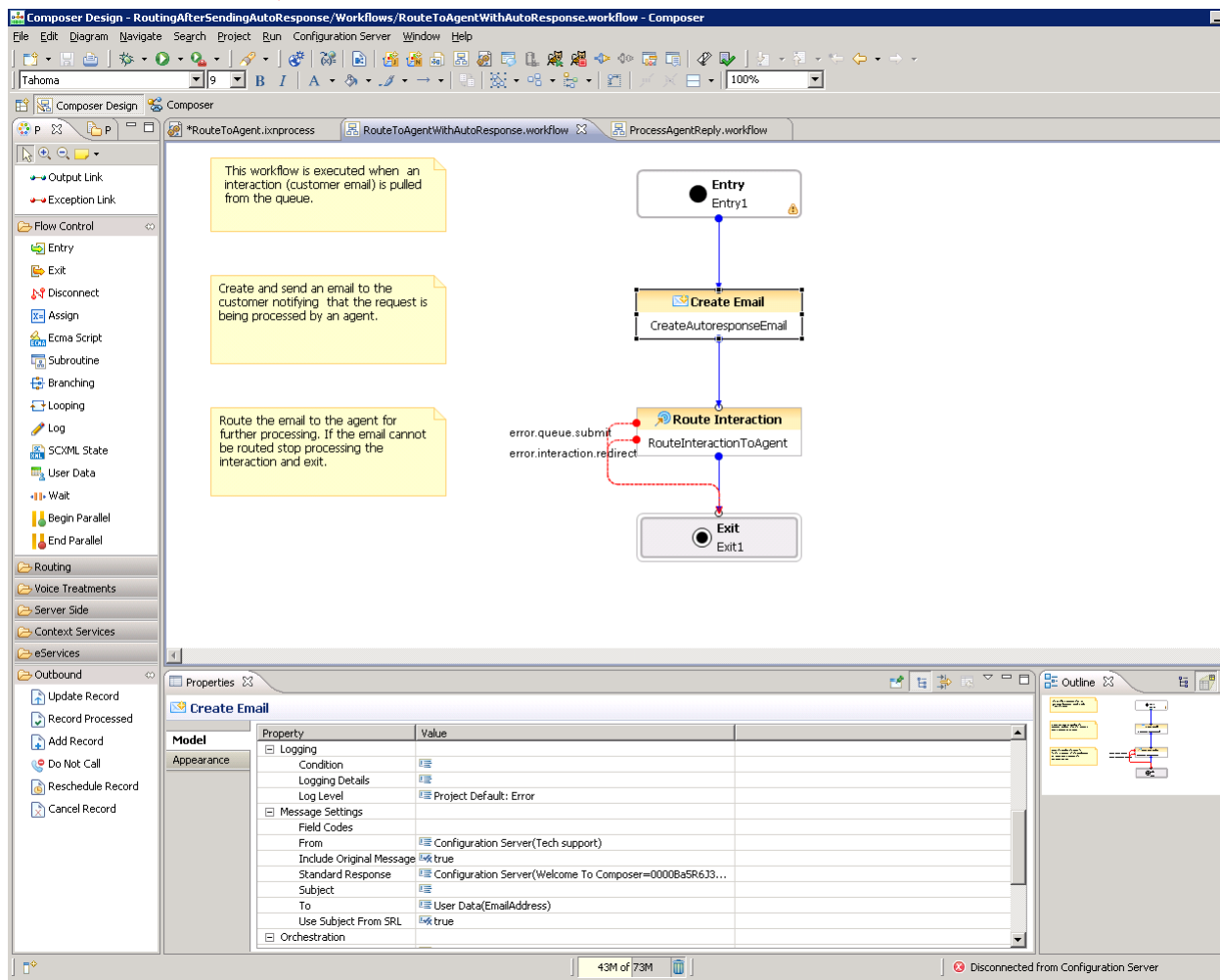


Figure 47: Composer Design Perspective

- You can also change perspectives by selecting **Window > Open Perspective** from the menu bar.

Note: For purposes of simplicity, this guide does not detail all ways to accomplish a task. For information on all methods, always consult the *Composer 8.1 Help*.

Customizing a Perspective

You can customize a perspective by maximizing/minimizing views as well as adding views. Buttons for this purpose appear on the right side of all views.

[Figure 48](#) shows these buttons on the upper right for the **Outline** view.

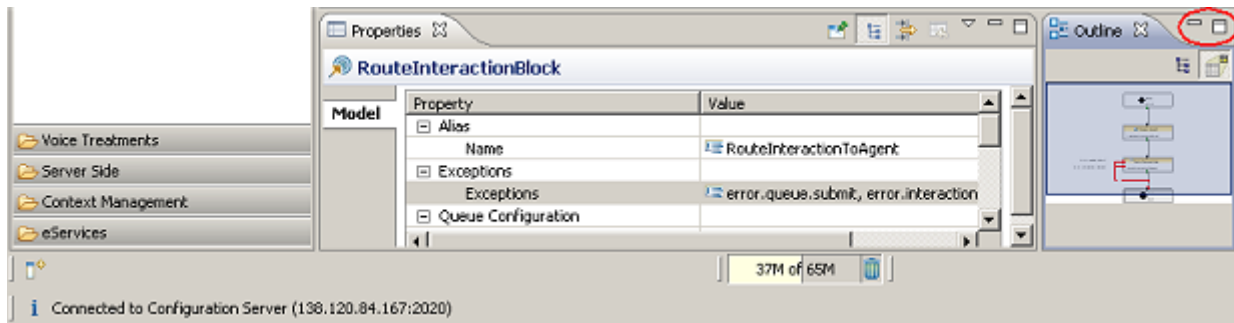


Figure 48: Maximize and Minimize Buttons for a View

Minimizing

Assume that you click the minimize button. The area now appears as shown in [Figure 49](#):

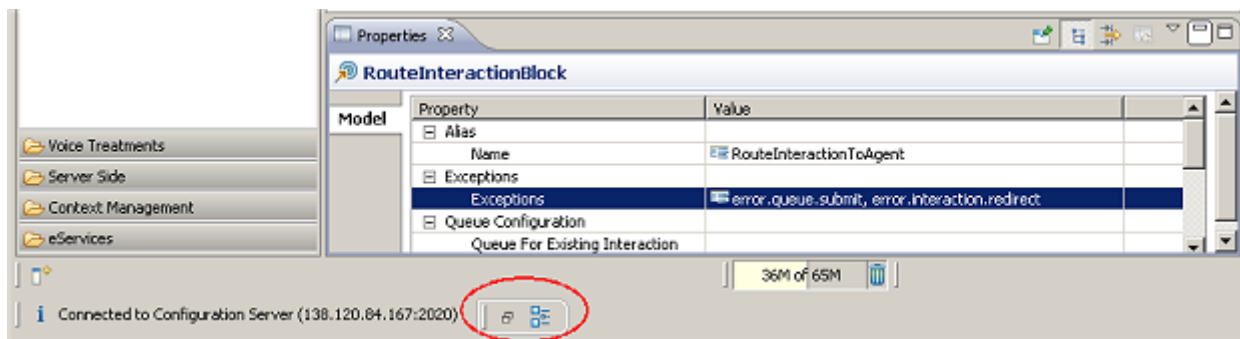


Figure 49: Toolbar to Restore a View

As shown above, the minimization causes a small toolbar to appear at the bottom of Composer. Click the first button (restore) to replace the view next to the Properties tab.

Maximizing

Maximization works similarly. Assume you click the maximize button. The Composer GUI shows only the Outline view. As a result, three small toolbars representing views that no longer appear are in close proximity to where they were originally located (see [Figure 50](#)):

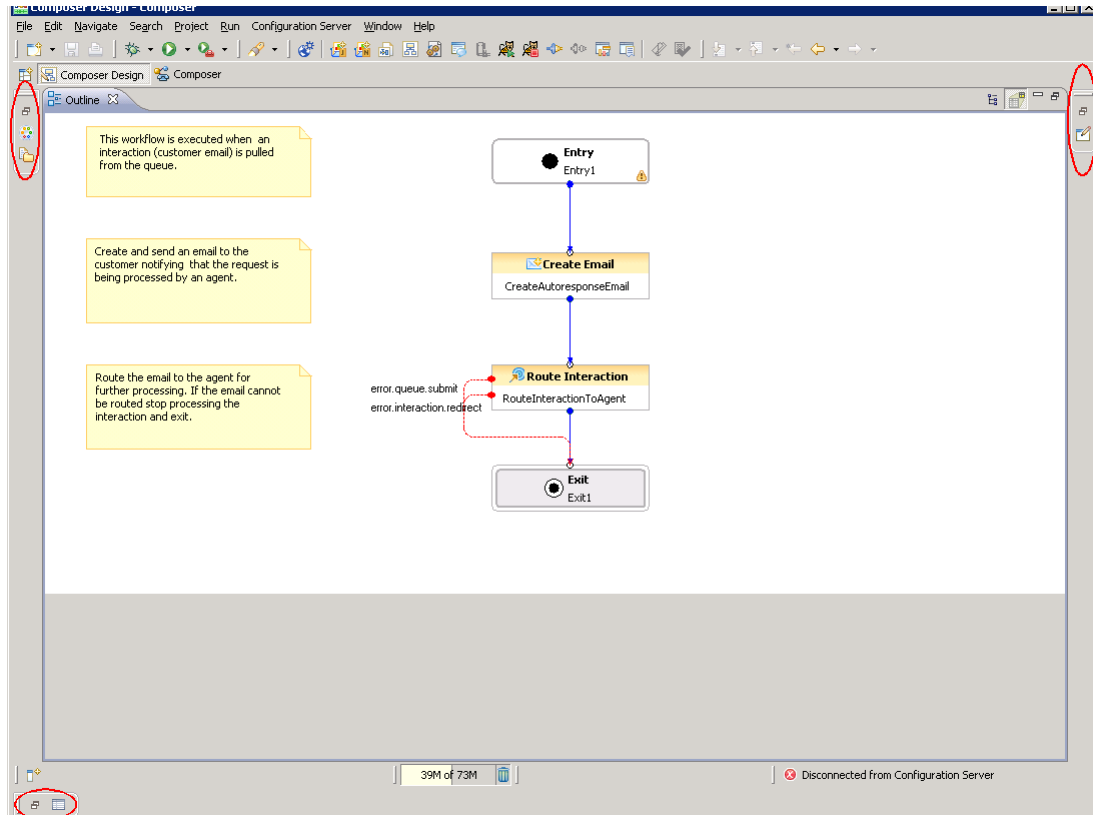


Figure 50: Composer Outline View Maximized

- Click the first button (restore) on the top left toolbar to cause Composer to appear as it did before the maximization (restore all views).
- Click the first button on the bottom toolbar to restore only the Properties view.
- Click the first button on the top right toolbar to restore only the editing area.

Different views can be displayed by selecting **Window > Show View** from the menu and then selecting a view.

Saving a Customized Perspective

The Windows menu has a **Save Perspective As** command. It also has various other perspective commands.

Viewing More Than One Diagram

The procedure below describes how to view more than one diagram side-by-side in Composer:

Procedure: Viewing Multiple Diagrams

Start of procedure

1. Select the diagram by double-clicking it in the **Workflows** folder of the Project Explorer. The diagram appears in the canvas area.
2. Select the next diagram. The canvas now contains two tabs as shown in [Figure 51](#).

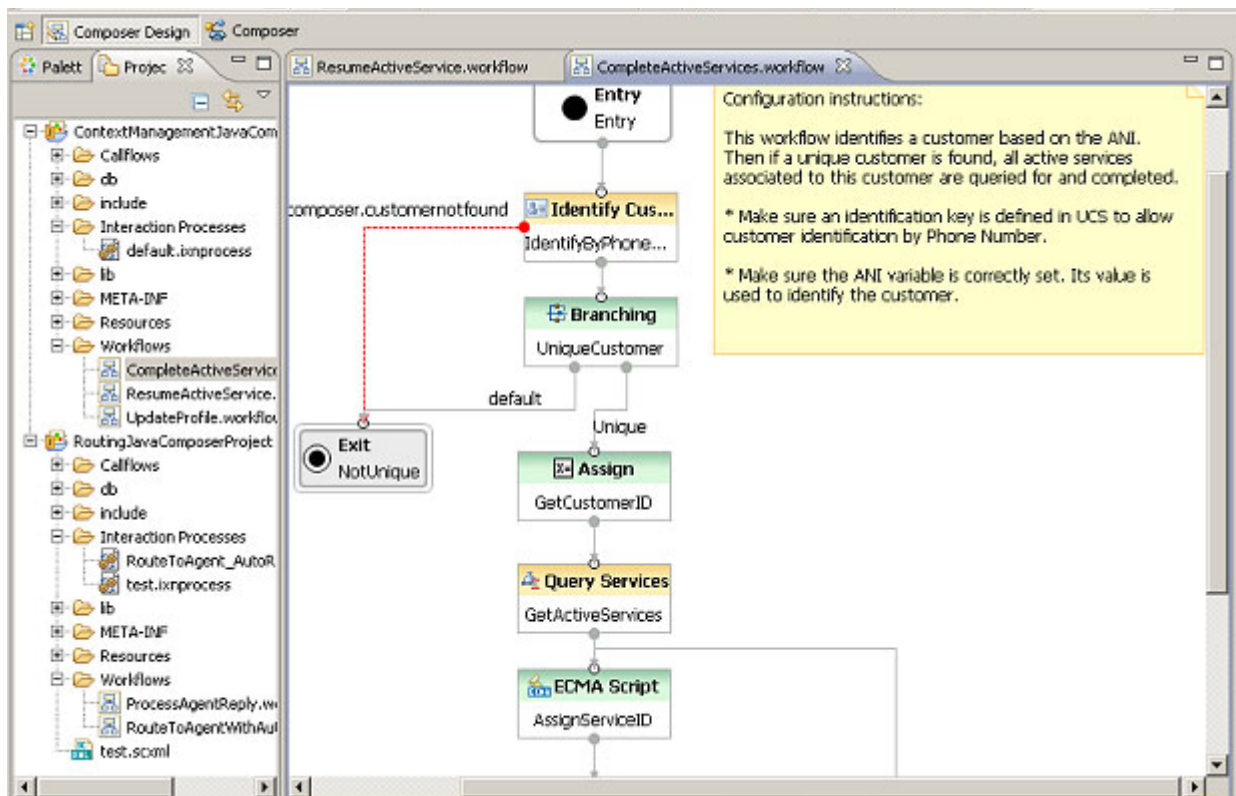


Figure 51: Two Workflow Diagrams Selected from Project Explorer

3. To view diagrams, one on top of the other:
 - a. Place the cursor on a tab, hold down the mouse button, and drag up. As you drag, you will notice a rectangular outline.
 - b. Release the cursor. The dragged diagram appears on top of the other diagram (see [Figure 52](#)).

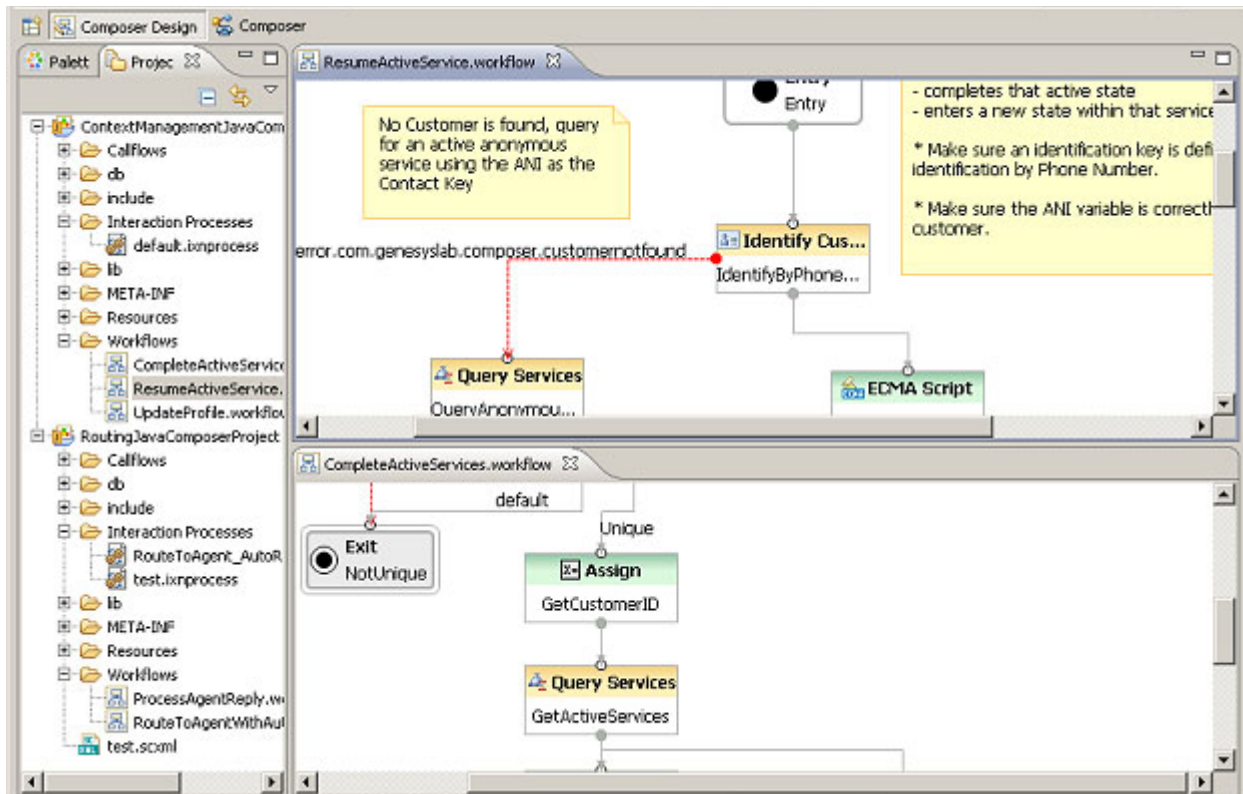


Figure 52: Two Workflow Diagrams Visible in Canvas Area

4. To view diagrams side-by-side:
 - a. Place the cursor on a tab and drag it to the side until the rectangular outline appears.
 - b. Release the cursor and the dragged diagram appears on top of the other diagram.
5. You can always decrease the magnification or close the Outline view if necessary.

End of procedure

Viewing Properties for Two Blocks

The procedure below describes how to view properties of two blocks at the same time.

Procedure: Displaying Properties for Two Blocks

Start of procedure

1. Select the first block.
2. In the Properties tab, click the button for pinning the Properties view to the current selection (see [Figure 53](#)).

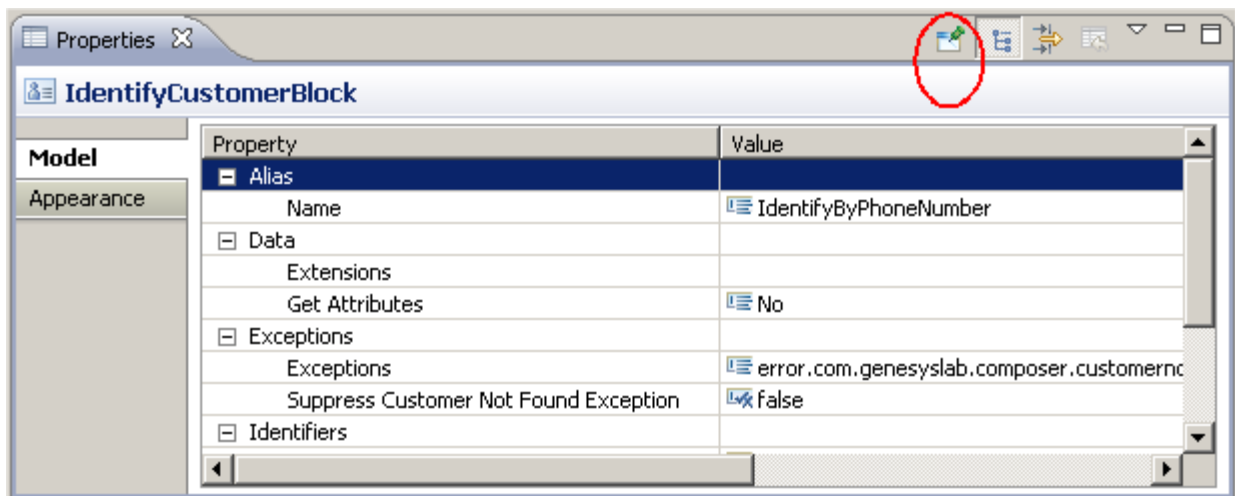


Figure 53: Button for Pinning a Properties View to the Selected Block

Until you un-click the toggle, Composer will always show the properties of the currently selected object even if a different object is selected.

3. Create a new Properties view. Drag out the tab to another row. They will now show up side-by-side.
4. Select the second block. The second properties view will show properties of the second block.

End of procedure

Saving Workflow Diagrams as Templates

You can save a diagram as a template and have it appear on the list of available templates when creating a new workflow diagram. Diagrams saved as templates can be exported to/imported from the file system or to other Composer installations.

Procedure:

Saving a workflow diagram as a template

Start of procedure

To save a diagram as a template:

1. In the Project Explorer, right-click the diagram in the `Workflows` folder.
2. Select `Save Workflow as Template`. The `Add Template` dialog box opens.
3. Name and describe the template.
4. Click `OK`. Upon a successful save the following message appears: `Custom template added to your configuration`.
5. Click `OK` to close the dialog box.

End of procedure

Procedure:

Accessing Saved Templates

Start of procedure

To view a diagram previously saved as template:

1. From the `File` menu, select `New > Workflow Diagram`. The `New Workflow` dialog box opens. The template appears in the `Main Workflow` tab under `Custom Templates` (see [Figure 54](#)).

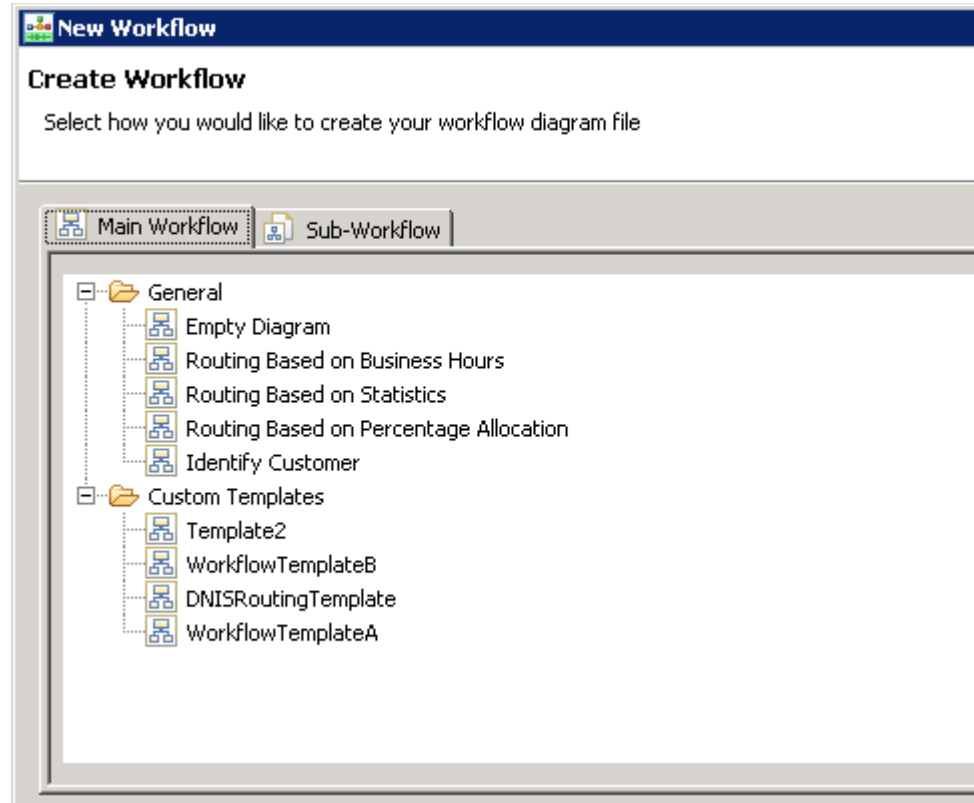


Figure 54: New Workflow Dialog Box

2. Select the template and click one of the following:
 - Next to name the diagram, select the Project, and then click Finish.
 - Finish to keep the template name and save in the Workflows folder under the current Project.

End of procedure

Procedure: Exporting a Diagram Template to the File System

To export a diagram to the file system or another user's Composer:

Start of procedure

1. In the Project Explorer, right-click the diagram in the Workflows folder.
2. Select Export. The Export dialog box opens.
3. Under General, select File System and click Next. The File System dialog box opens (see [Figure 55](#)).

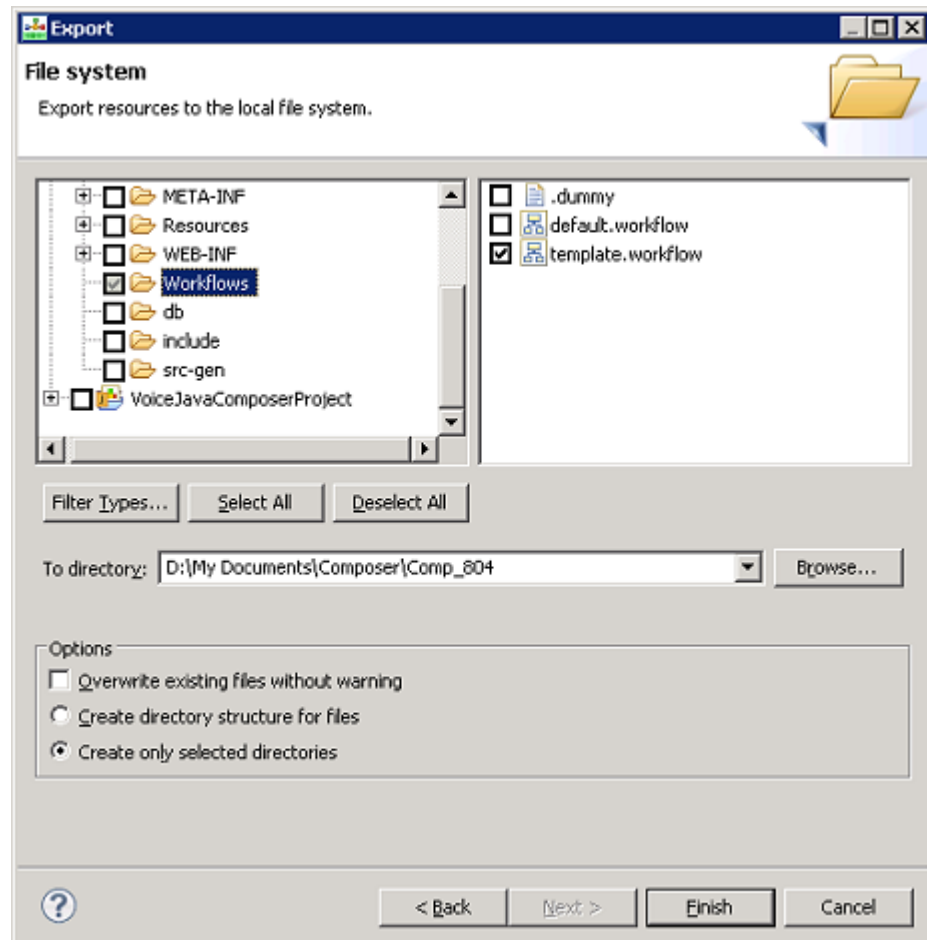


Figure 55: File System Dialog Box for Export

4. In the File System dialog box, select the folder containing the diagram(s).
5. On the right, click check boxes to indicate the diagram(s) to export.
6. Opposite To directory, select the Composer installation to export to or browse for the destination directory.
7. At the bottom of the dialog box, select one of the following:
 - Overwrite existing files without warning
 - Create directory structure for files
 - Create only selected directories (default).
8. Click Finish.

End of procedure

Procedure: Importing a Diagram Saved as a Template

Start of procedure

To import a diagram previously saved as a template:

1. In the Project Explorer, right-click the diagram in the **Workflows** folder.
2. Select **Import**. The **Import** dialog box opens.
3. Under **General**, select **File System** and click **Next**.
4. In the **File System** dialog box, opposite **From directory**, click **Browse**.
5. Open the **workspace** directory followed by the **Project** folder.
6. Within the **Project** folder, select the **Workflows** folder that contains the template to import and click **OK**.
7. In the **File System** dialog box on the right, click check boxes to indicate the template(s) to import. [Figure 56](#) shows an example.

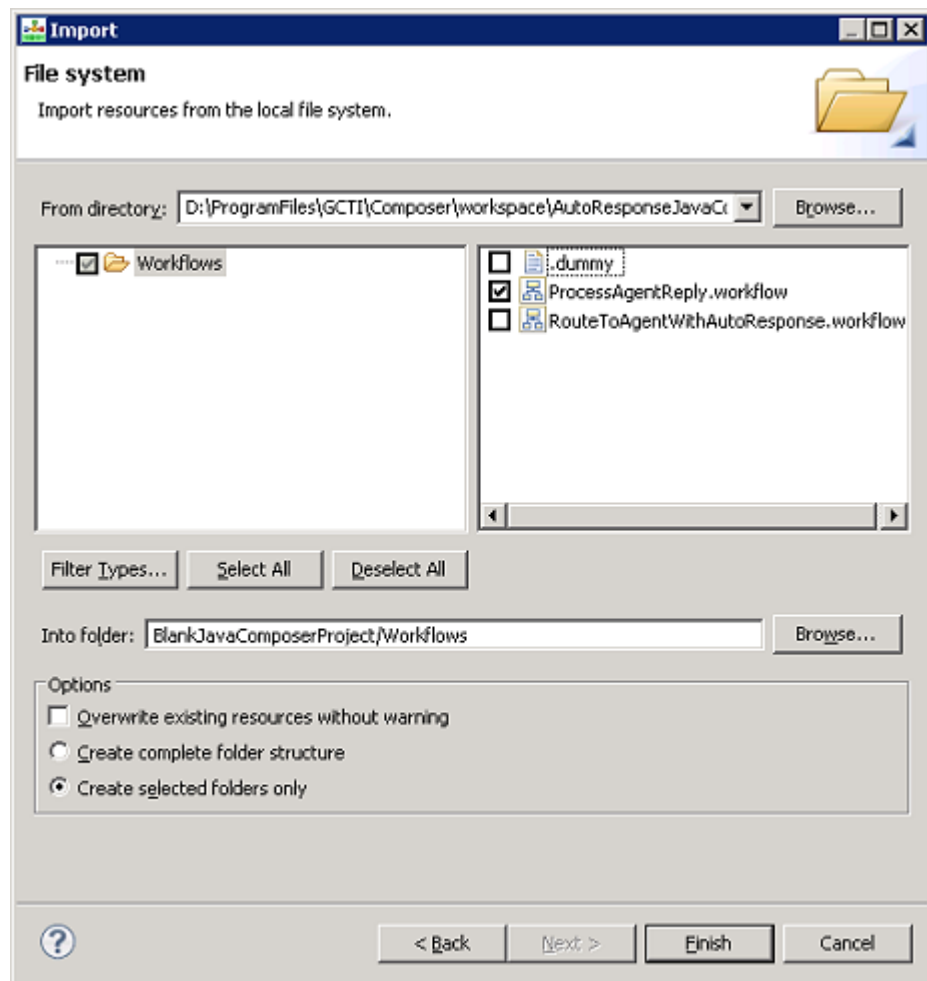


Figure 56: File System Dialog Box for Import

8. Opposite Into folder, browse for and select the folder to import into.
9. Under Options, select one of the following:
 - Overwrite existing files without warning
 - Create directory structure for files
 - Create only selected directories (default).
10. Click Finish.

End of procedure

Custom Blocks

Think of a custom block as a pre-filled template for a block.

Procedure: Creating a custom block

Start of procedure

1. Add the block to use as the pre-filled template.
2. Fill out the properties to be the basis of the template.
3. Right-click the block to bring up a shortcut menu.
4. From the shortcut menu, select Add as custom tool. Note this option is not available for the IPD blocks.
5. In the Custom Tooling dialog box, name and describe the custom block and click OK. Composer adds the block to the palette in the Custom category.

End of procedure

Using a Custom Block

Use the block as you would any others. Fields defined as relevant in the Block Data Map are automatically populated in the Properties tab based on the state of the original block when you made the tooling.

Note: When you create a new block, you must add a mapping to define which fields should be template-aware.

Procedure: Changing Existing Custom Blocks

Start of procedure

To change an existing custom block:

1. Within the palette, right-click a custom block and select **Customize** from the context menu. This opens the **Customize Palette** dialog box.
2. Navigate to/select the block you want to change (see [Figure 57](#)).

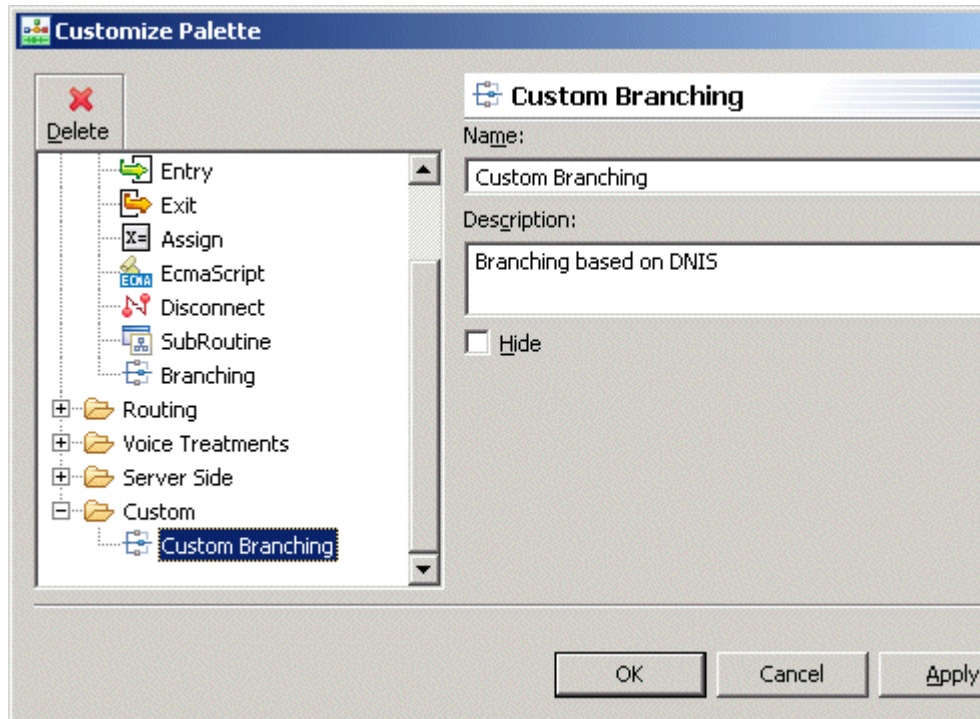


Figure 57: Customize Palette Dialog Box

3. You can:
 - Change the label.
 - Edit the description.
 - Delete the block entirely.

End of procedure

Procedure: Deleting a Custom Block

Start of procedure

To delete a custom block:

1. Within the palette, right-click a custom block and select **Customize** from the context menu. This opens the **Customize Palette** dialog box shown above.
2. Navigate to/select the block you want to delete.
3. Click the **Delete** button at the top of the **Customize Palette** dialog box.
4. Click **OK**.

End of procedure

Procedure: Hiding the Custom category

Start of procedure

To hide the Custom category on the palette:

1. In the **Customize Palette** dialog box (see above steps), select the **Custom** folder.
2. Click the **Hide** checkbox. Later, if you need the Custom category back on the palette, select the **Custom** folder and un-click **Hide**.
3. Click **OK**.

End of procedure

Procedure: Import/Export of Custom Blocks

Start of procedure

To import or export a custom block so that it can be shared across multiple users/Composer installations:

1. Select the custom block in the palette.
2. From the Diagram menu, select one of the following:
 - a. Import Custom Blocks· to open the Select Custom Tooling Definition dialog box.
 - Browse to the location for the previously exported custom block file, which will have a *.ctooling extension.
 - Select the file, and click OK.
 - b. Export Custom Blocks... to open the Create Custom Tooling Definition dialog box.
 - Name the file.
 - Keep the file type as custom tooling, and click OK. The custom block is saved as file with a *.ctooling extension.

End of procedure



Appendix

Composer Blocks

This appendix introduces you to Composer’s IPD and diagram-building blocks. It also lists the Interaction Routing Designer block equivalent for each Composer block, where one exists. See the *Composer 8.1 Help* for the properties associated with each Composer block.

This appendix contains the following sections:

- [IPD Blocks, page 113](#)
- [Workflow Diagram-Building Blocks, page 114](#)
- [Composer Blocks Mapped to IRD Objects, page 122](#)
- [Other Functionality, page 126](#)

IPD Blocks

As described in “Interaction Process Diagrams” on [page 16](#), an interaction process diagram is comprised of various IPD-building blocks. [Table 2](#) summarizes Composer’s IPD blocks.

Table 2: Composer interaction process diagram Blocks

IPD Block	Description
Interaction Queue	Use this block to define a multimedia (non-voice) interaction queue in an interaction process diagram and to create views, which define conditions for pulling interactions out of queues for submittal to workflows. See “Defining an Interaction Queue” on page 73 for more information.
Media Server	Represents an existing media server, such as an e-mail server. Use to direct interactions from a media server into an interaction process diagram. See “Adding a Media Server Block” on page 70 for more information.
Workflow	Points to an existing Workflow resource (either a workflow diagram or SCXML file) to which an interaction can be sent for specialized processing. See “Adding a Workflow Block” on page 81 for more information.

Table 2: Composer interaction process diagram Blocks (Continued)

IPD Block	Description
Workbin	Use to define a temporary storage area for interactions accessible from the agent desktop.
Flow Control Blocks	In an IPD, when an interaction is submitted from an interaction queue to a routing workflow, you can create multiple views per queue, with each view having its own set of conditions and managing submission of an interaction to a separate routing workflow. To support multiple views per interaction queue, the following Flow Control blocks are available when creating an IPD: Branching, ECMAScript, Log. (definitions in Table 3).

Workflow Diagram-Building Blocks

A workflow diagram is comprised of various blocks, which can be thought of as “diagram-building” blocks. Composer’s **Palette** tab (see [Figure 51 on page 102](#)) groups these blocks into different categories. This section summarizes the blocks in each category.

Flow Control Blocks

[Table 3](#) summarizes the workflow diagram-building blocks used for flow control.

Table 3: Flow Control Blocks

Block Name	Description
Assign	Use to assign a computed value/expression or a literal value to a variable.
Attach	Use the Attach block for attaching a specific interaction to the current Orchestration Server session.
Begin Parallel	Use this block to enable the design of multiple threads, such as running busy treatments in parallel files. A thread is a list of blocks that run one after another. Use the End Parallel block to mark the end of the threads that were started by a matching Begin Parallel block.
Branching	Use the Branching block as a decision point in a callflow or workflow. It enables you to specify multiple application routes based on a branching condition. Depending on which condition is satisfied, the call follows the corresponding application route.
Cancel Event	Use this block to cancel custom events. The event name and message can be specified. You can select a dynamic variable as the message. ORS 8.1.2+ versions are required for Raise and Cancel Event blocks.

Table 3: Flow Control Blocks (Continued)

Block Name	Description
Detach	Use the Detach block for detaching a specific interaction from the current Orchestration Server session.
Disconnect	Use to disconnect the caller and end the call. The Disconnect block invokes the Cancel Call treatment, which ends the workflow and deletes the interaction from URS memory.
ECMAScript	Use to build an ECMAScript expression for routing decisions. Universal Routing Server 8.0 supports SCXML plus ECMAScript as a routing language. While the core SCXML provides State Chart functionality, you can specify URS-specific instructions, such as conditions that can be used for routing decisions, in the form of ECMAScript. The Script property brings up Composer's Expression Builder for creating those conditions in the form of expressions.
End Parallel	Use the End Parallel block to mark the end of the threads that were started by a matching Begin Parallel block.
Entry	All workflow diagrams must start with an Entry block. Defines variables that can be shared across different blocks in the same workflow.
Exit	Use to terminate the workflow or to return control back to the calling workflow in case of a sub-workflow (subroutine).
Log	Use to record information about the application; for example, caller-recorded input that is collected while the application is running, or error messages.
Looping	Use this block to iterate over a sequence of blocks multiple times in the following scenarios: <ul style="list-style-type: none"> • Iterate over a sequence of blocks based on a self-incrementing counter (FOR). • Iterate indefinitely until an exit condition is met (WHILE). • Iterate over records/data returned by the DB Data block (CURSOR/FOREACH). Also, populate variables if variables mapping is defined as described in the <i>Composer 8.0 Help</i>. • Iterate over data returned by Context Services blocks (FOREACH). Also, populate variables if variables mapping is defined. • Iterate over a JSON array defined in the application.
Raise Event	Use this block to throw custom events. The event name and message can be specified. You can select a dynamic variable as the message. ORS 8.1.2+ versions are required for Raise and Cancel Event blocks.

Table 3: Flow Control Blocks (Continued)

Block Name	Description
SCXML State	Use to write custom SCXML code for Composer to include in the SCXML document that it generates based on the workflow diagram.
Subroutine	Use to invoke external SCXML documents or a sub-workflow created using Composer.
User Data	Use in a routing application to update an interaction's User Data. Corresponds to the Universal Routing Server function <code>_genesys.ixn.setUserData(input, xn)</code> available in Expression Builder. This block generates ECMAScript inside an SCXML state and does not rely on External Service Protocol via <code><session:fetch></code> .
Wait	Use to have ORS transition out when one of the defined events is received and the associated condition is true.

Routing Blocks

[Table 4](#) summarizes the workflow diagram-building blocks used for routing.

Table 4: Routing Blocks

Block Name	Description
Default Route	Instructs URS to route a voice interaction to the default destination.
Force Route	Forces Universal Routing Server to route the interaction to the first target type without any other operations.
Queue Interaction	Places a non-voice interaction into an existing queue.
Route Interaction	Routes a non-voice interaction to one or more target objects.
Routing Rule	Select routing rules that currently exist in the Configuration Database, such as those created with Interaction Routing Designer: You can select load balancing, percentage, or statistics routing rules.
Stop Interaction	Requests Interaction Server to stop processing an interaction and allows assignment of a reason code.
Target	Routes a voice interaction to a target. Can be used for percentage and/or conditional routing using threshold expressions, such as those used for share agent by service level agreement routing.

Voice Treatment Blocks

Busy treatments can be played to callers when all the targets selected by URS are busy and the interaction is waiting for an available target. [Table 5](#) summarizes the workflow diagram blocks for voice treatments.

Table 5: Voice Treatment Blocks

Block Name	Description
Cancel Call	Use to stop a currently running dialog.
Create User Announcement	Use to record a caller announcement.
Delete User Announcement	Use to delete an announcement created by a caller using the Create User Announcement block.
IVR	Use to invoke an interactive voice response (IVR) unit and connect the interaction to the IVR.
Pause	Use to suspend treatment processing for a specified duration.
Play Application	Use to execute an application (such as a Composer voice application) or a script on a device, such as an IVR.
Play Sound	Use to play audio resources of the following type: Music, BusyTone, FastBusyTone, RingBack, RecordedAnnouncement (on Stream Manager), Silence.
Play Message	Use to invoke/play audio or text-to-speech Announcement treatments.
Set Default Route	Use to set/change the default destination.
User Input	Use to play a text-to-speech announcement, to play an announcement and collect digits, and for collecting digits.

eServices (Multimedia) Blocks

The eServices blocks perform specialized processing of multimedia interactions. [Table 6](#) summarizes these blocks.

Table 6: eServices Blocks

Block Name	Description
Chat Transcript	Use to create (but not send) an e-mail message that is generated from your site's Standard Response Library and which has the customer's chat transcript attached. Use the Send Email block to send the message out.
Classify Interaction	Use to have Universal Routing Server instruction Classification Server to assign one more category codes (configured in Knowledge Manager as described in eServices 8.1 User Guide) to a text-based interaction. Once a category code is assigned, other types of processing can occur based on the category code.
Create E-mail	Use to create an e-mail message to be sent out to a customer or to another agent and to specify the interaction queue where the outbound e-mail should be placed.
Create Interaction	Use this block to create an interaction record in the Universal Contact Server Database, for a customer contact. This saves the current interaction being processed in the database.
Create SMS	Use to create an outbound message, which can be sent out as a Short Message Service (SMS) message to an external SMS Server. SMS refers to the common text messaging service available on cell phones and other handheld devices.
E-mail Forward	Use to send an incoming e-mail to an external address, such as for agent collaboration. This block combines the functionality of IRD's Forward E-mail, Redirect E-mail, and Reply E-mail from External Resource objects. The Forward Type property specifies the type of functionality by allowing you to select Forward, Reply to Customer, or Redirect.
E-mail Response	Use to send an e-mail in response to incoming interaction resulting from inbound e-mail or an open media request. This block combines the functionality of IRD's Acknowledgement, Autoresponse, and Create Notification objects.
Identify Contact	<p>This block can be used for various purposes. You can:</p> <ul style="list-style-type: none"> Identify a contact based on the User Data of the current interaction. Return a list of matching Contact IDs based on the User Data. This occurs only if a single matching contact record is found or if the Return Unique property is set to false. Contact attribute values (first name, last name, email address, and so on) are returned only when a single matching contact is found (no matter what is the value of the Return Unique property). Create a contact record in the Universal Contact Server (UCS) Database with information in the User Data if a matching contact is not found. Update the interaction's User Data with data returned by UCS.

Table 6: eServices Blocks (Continued)

Block Name	Description
Render Message	Use the Render Message block to request Universal Contact Server to create message content. You can create message content using text from either the Message Text to Render property, the Result property, or User Data. This block causes Universal Routing Server to generate a request to Universal Contact Server for the method RenderMessageContent. The primary reason for this block is to create message content for use in the Create SMS block,
Screen Interaction	Use to filter a text-based interaction for specific content (specific words or patterns) based on evaluation of one or more screening rules by Classification Server. You then have the option of segmenting incoming interactions to different logical branches based on the result of the screening query. Screening rules are created in Knowledge Manager as described in the <i>eServices 8.1 User's Guide</i> .
Send E-mail	Use to send an e-mail waiting in a queue that was previously created using the Create E-mail block.
Send SMS	Use this block to send an Short Message Service (SMS) message created with the Create SMS block.
Update Contact	Use this block to update customer profile information in the Universal Contact Server Database, based on data attached to an interaction.

Server-Side Blocks

Server-Side blocks provide the ability to interact with internal and external custom server-side pages, Web Services, and URLs. [Table 7](#) summarizes Server-Side blocks.

Table 7: Server-Side Blocks

Block Name	Description
Backend	Use to invoke custom backend Java Server Pages (JSP). You have the option to pass back all the application session state data to the backend logic page on the server.
Business Rule Block	<p>Once the Rule Packages (created from Rule Templates) that you want to work with are deployed to the Genesys Rules Engine, you can use the Business Rule block on the Server Side palette to create voice and routing applications that use business rules.</p> <p>Use this block to have Composer query the Genesys Rules Authoring Tool (GRAT) for deployed packages. For the Rule Package that you specify, Composer will query the GRAT for the Facts associated with the Rule Package. You can then set values for the Facts, call the Genesys Rules Engine for evaluation, and save the results in a variable.</p>
DB Data	Use for connecting to a database and retrieving/manipulating information from/in a database.
External Service	Use to exchange data with third party (non-Genesys) servers that use the Genesys Interaction SDK or any other server or application that complies with the GIS communication protocol.
OPM	Enables VXML and SCXML applications to use Operational Parameters (OPM), which allow a business user to control the behavior of these applications externally. Operational Parameters are defined and managed in the Operational Parameter Management (OPM) feature of Genesys Administrator Extension (GAX).
TLib	Use in workflows and sub-workflows that will use <session:fetch> method="tlib". The block exposes properties to form a TLib request to set agent status not ready equivalent to TAgentSetNotReady.
Web Request	Use to invoke any supported HTTP web request or REST-style web Service.
Web Service	Use to invoke Web Services. Data returned by the Web Service is converted to JSON format and made available in the application.

Context Services Blocks

[Table 8](#) summarizes the workflow diagram-building blocks used for Context Services as described on [page 53](#).

Table 8: Context Services Blocks

Block Name	Description
Associate Service	Use the Associate Service block to associate an anonymous service record with a customer whose profile exists in the database used for Context Services.
Complete Service	Use to mark an active service as completed in the Universal Contact Server Database used for Context Services.
Complete State	Use to mark the completion of a specified state in the context of a service in the database used for Context Services.
Complete Task	Use to mark the application as completing a specified task within a service/state.
Create Customer	Use to create a voice callflow/routing workflow that includes the capability to create a customer profile through Context Services.
Enter State	Use the Enter State block to mark the entry of the application into a specified state in the context of a service.
Identify Customer	Use to identify a customer in the database based on search criteria, which can be customer profile core data or customer profile extension data.
Query Customer	Use to look up a customer's core profile and profile extension attributes.
Query Services	Use to query the database for a list of services associated with a particular Customer ID or, in case of unassociated services, the Contact Key.
Query States	Use to query the database used for Context Services for active and completed states data for a specified service.
Query Tasks	Use to query the database used for Context Services for active and completed tasks within a state for a specified service.
Start Service	Use to communicate the creation or start of a service in the UCS Database.
Start Task	Use to mark the application as entering a specified task within a service/state.
Update Customer	Use to update the customer profile in the database used for Context Services.

Outbound Blocks

The Outbound blocks support Genesys Outbound Contact, an automated product for creating, modifying, running, and reporting on outbound campaigns for proactive customer contact. Outbound Contact Solution (OCS) provides automated dialing and call-progress detection, so that an Agent is required only when a customer is connected.

Table 9: Outbound Blocks

Block Name	Description
Add Record	Use to automate building of Calling Lists by adding a new record to a specified Calling List.
Cancel Record	Use to cancel a customer record in a calling list.
Do Not Call	Use to add a contact record, such as a phone number or an e-mail address, to a specified Do Not Call List and marks the corresponding record as Do Not Call.
Record Processed	Use to mark a record as requiring no further handling.
Reschedule Record	Use to Reschedule a customer interaction from the specified Calling List.
Update Record	Use this block to update a Calling List record that you specify via a RecordHandle parameter.

Composer Blocks Mapped to IRD Objects

The tables below list IRD objects based on their IRD toolbar category name and point to the corresponding Composer diagram building block.

Note: Composer refers to the fundamental element of a workflow diagram as a *block* whereas in IRD documentation, this element is referred to as an *object*.

Table 10: IRD Data & Services Category

IRD Object Name	Composer Block Name
Database Wizard object	DB Data block
Web Service object	Web Service block.
External Service object	External Service block

Table 11: IRD Miscellaneous Category

IRD Object Name	Composer Block Name
Assign object	Assign block
Call Subroutine object	Subroutine block
Entry object	Entry block
Exit object	Exit block
Function object	ECMAScript block
If object	Assign, Branching, ECMAScript, Log, Entry, and Looping blocks all open Expression Builder

Table 12: IRD Routing Category

IRD Object Name	Composer Block Name
Selection object	Target block
Percentage object	Target block
Default object	Default Route block
Force Routing object	Force Route block
Percentage, Load Balancing, Statistics Routing Rule objects	Routing Rule block
Statistics object	Target block

Table 13: IRD Segmentation Category

IRD Object Name	Composer Block Name
ANI object	Branching. Expression Builder contains a DNIS variable under Workflow Variables > System. Also see ANI predefined application variable in Figure 14 on page 34 .
DNIS object	Branching. Expression Builder contains a DNIS variable under Workflow Variables > System. Also see ANI predefined application variable in Figure 14 on page 34
Date object	Branching. You can also use the <code>_genesys.IsSpecialday</code> Functional Module.

Table 13: IRD Segmentation Category (Continued)

IRD Object Name	Composer Block Name
Day of Week object	Branching. You can also use the <code>_genesys.session.day</code> Functional Module.
Time object	Branching. You can also use the <code>_genesys.timeinZone</code> Functional Module.
Generic object	Branching. See “Expression Builder” on page 35 .
Classification Segmentation	An ECMAScript function allows you to manually attach Classification categories to interactions. You can then segment interactions to different logical branches based on the different categories.

Table 14: IRD Treatment Category

IRD Treatment	Composer Equivalent	Composer Busy Treatment?
Record user announcement	Create User Announcement	no
Delete user announcement	Delete User Announcement	no
Cancel call	Disconnect	no
IVR	IVR	yes
Pause	Pause	yes
Play announcement	Play Message	yes
Text to speech	Play Message	yes
Play announcement and collect digits	User Input	yes
Text to speech and collect digits	User Input	yes
Verify digits	Digit verification in User Input	yes
Collect digits	User Input	yes
Play Application	Play Application	Play Application
Busy	Play Sound	yes
Fast busy	Play Sound	yes
Music	Play Sound	yes

Table 14: IRD Treatment Category (Continued)

IRD Treatment	Composer Equivalent	Composer Busy Treatment?
Ringback	Play Sound	yes
Silence	Play Sound	yes
RAN	Play Sound	yes
Set default destination	Set Default Route	no

Table 15: IRD Multimedia Category

Composer Route Block Name	Equivalent IRD Object
Create E-mail block	Create E-mail object
Send E-mail block	Send E-mail object
Email Response block	Combines the functionality of IRD's Acknowledgement, Autoresponse, and Create Notification objects.
Create SMS block	Create SMS object
Chat Transcript block	Chat Transcript object
Email Forward block	Combines the functionality of IRD's Forward E-mail, Redirect E-mail, and Reply E-mail from External Resource objects.
Screen Interaction block	Screen, Multiscreen object
Classify Interaction block	Classify object
Update Contact block	Update Contact object
Identify Contact block	Identify Contact object
Create Interaction block	Create Interaction object
Render Message block	Render Message object
Queue Interaction block	Queue Interaction object
Route Interaction block	Route Interaction object
Stop Interaction block	Stop Interaction object
External Service (ESP) block	External Service object

Table 16: SMS Category

Composer Route Block Name	Equivalent IRD Object
Create SMS	Create SMS Out object
Send SMS	Send SMS Out object

Table 17: IRD Outbound Category

Composer Route Block Name	Equivalent IRD Object
Add Record	Add Record object
Cancel Record	Cancel Record object
Do Not Call	Do Not Call object
Record Processed	Processed object
Reschedule Record	Reschedule object
Update Record	Update Record object

Other Functionality

As shown in [Table 19](#), Composer's interaction process diagrams supply functionality found in IRD business processes.

Table 18: IRD Business Process Functionality

IRD Business Process	Composer Interaction Process Diagram
Queue object	Interaction Queue block
View object	Interaction Queue view defined in Interaction Queue block
Strategy object	Workflow block
Workbin object	Workbin block
Strategy-linked nodes	Workflow-generated blocks
Endpoint object	Media Server block endpoint

[Table 19](#) lists other IRD functionality that can also be found in Composer.

Table 19: Other Functionality

IRD Functionality	Composer Functionality
List Objects	List Object Manager. See Composer's List Object Manager (see page 39).
Statistics Dialog Box	Statistics Manager and Builder. See Composer's Statistics Builder (see page 38).
Generic Segmentation object	Expression Builder (see page 35)
Generic Segmentation object	Skill Expression Builder (see page 40)

Related Documentation Resources

The following resources provide additional information that is relevant to this software. Consult these additional resources as necessary.

Composer Help Wiki

- The *Composer 8.1 Help*, which is your main source of information for using Composer to develop voice and routing applications, is available on the [Composer 8.1.3 Help Wiki](#). Selecting Help > Help Contents from within Composer directs new and existing users to the wiki.
- Since Composer is based on Eclipse (www.eclipse.org), you should familiarize yourself with basic Eclipse concepts by referring to the *Workbench User Guide* available from within Composer.

IRD to Composer Migration Guide Wiki

- The [IRD to Composer Migration Guide](#) describes how to migrate routing strategies created in Interaction Routing Designer into Composer Projects as SCXML-based workflow diagrams.

Orchestration Server Wiki

- The [Orchestration Server wiki](#) contains the Genesys language specification for the following interfaces: (1) SCXML — What we support from the standard, both from an interface and behavioral standpoint; (2) Domain-specific languages (model modules) for all the Genesys-specific functional modules; (3) External interfaces to platform and SCXML sessions.

Genesys Voice Platform Wiki

- The [Genesys Voice Platform Wiki](#) contains Genesys VoiceXML 2.1 Reference Help, which provides information about developing Voice Extensible Markup Language (VoiceXML) applications. It presents VoiceXML concepts, and provides examples that focus on the GVP Next Generation Interpreter (NGI) implementation of VoiceXML.

Composer Deployment Guide

- This guide assists the first-time Composer user in deploying Composer. It covers software requirements, installing Composer on Windows and Mac, displaying the user interface, post-installation configuration, uninstalling and re-installing.

Cheat Sheets

- Selecting Help > Cheat Sheets opens a dialog box where you can expand Composer and select tutorials to quickly get started with the concepts. This includes tutorials for pre-configurations like for the SIP Phone settings, creating first voice applications, and so on.

Management Framework

- *Framework 8.1 Deployment Guide*, which provides information about configuring, installing, starting, and stopping Framework components.
- *Framework 8.1 Genesys Administrator Help*, which provides information about configuring and provisioning contact center objects by using the Genesys Administrator.
- *Framework 8.1 Configuration Options Reference Manual*, which provides descriptions of the configuration options for Framework components.
- *Stat Server 8.1 User's Guide*, which describes the configuration, installation, and start procedures relevant to deploying Stat Server.

SIP Server

- *Framework 8.1 SIP Server Deployment Guide*, which provides information about configuring and installing SIP Server.

Universal Routing

- *Orchestration Server 8.1 Deployment Guide*. Contains deployment information for Genesys Orchestration Server, which offers an open standards-based platform with an SCXML engine enabling intelligent

distribution of interactions throughout the enterprise. Orchestration Server interprets the top-level SCXML document created as a result of an interaction processing diagram created in Composer.

eServices/Multimedia

The context services user's guide, available on the Context Services Wiki, which provides information on the Universal Contact Server database of customer-related, service, and interaction-centric data (current and historical). Composer's Context Services blocks use this database.

Genesys Voice Platform

- *Genesys Voice Platform 8.1 Deployment Guide*, which provides information about installing and configuring Genesys Voice Platform (GVP).
- *Genesys Voice Platform 8.1 User's Guide*, which provides information about configuring, provisioning, and monitoring GVP and its components.
- *Genesys Voice Platform 8.1 Legacy VoiceXML 2.1 Reference*, which describes the VoiceXML 2.1 language as implemented by the Legacy GVP Interpreter (GVPI) in GVP 7.6 and earlier, and which is now supported in the GVP 8.1 release.
- *Genesys Voice Platform 8.1 CCXML Reference Manual*, which provides information about developing Call Control Extensible Markup Language (CCXML) applications for GVP.
- *Genesys Voice Platform 8.1 Troubleshooting Guide*, which provides information about Simple Network Management Protocol (SNMP) Management Information Bases (MIBs) and traps for GVP, as well as troubleshooting methodology.
- *Genesys Voice Platform 8.1 Configuration Options Reference*, which replicates the metadata available in the Genesys provisioning GUI, to provide information about all the GVP configuration options, including descriptions, syntax, valid values, and default values.
- *Genesys Voice Platform 8.1 Metrics Reference*, which provides information about all the GVP metrics (VoiceXML and CCXML application event logs), including descriptions, format, logging level, source component, and metric ID.
- *Voice Platform Solution 8.1 Integration Guide*, which provides information about integrating GVP, SIP Server, and, if applicable, IVR Server.

Open Standards

- *W3C Voice Extensible Markup Language (VoiceXML) 2.1, W3C Recommendation 19 June 2007*, which is the World Wide Web Consortium (W3C) VoiceXML specification that GVP NGI supports.
- *W3C Voice Extensible Markup Language (VoiceXML) 2.0, W3C Recommendation 16 March 2004*, which is the W3C VoiceXML specification that GVP supports.
- *W3C Speech Synthesis Markup Language (SSML) Version 1.0, Recommendation 7 September 2004*, which is the W3C SSML specification that GVP supports.
- *W3C Voice Browser Call Control: CCXML Version 1.0, W3C Working Draft 29 June 2005*, which is the W3C CCXML specification that GVP supports.
- *W3C Semantic Interpretation for Speech Recognition (SISR) Version 1.0, W3C Recommendation 5 April 2007*, which is the W3C SISR specification that GVP supports.
- *W3C Speech Recognition Grammar Specification (SRGS) Version 1.0, W3C Recommendation 16 March 2004*, which is the W3C SRGS specification that GVP supports.

Genesys

- *Genesys Technical Publications Glossary*, which ships on the Genesys Documentation Library DVD and which provides a comprehensive list of the Genesys and computer-telephony integration (CTI) terminology and acronyms used in this document.
- *Genesys Migration Guide*, which ships on the Genesys Documentation Library DVD, and which provides documented migration strategies for Genesys product releases. Contact Genesys Customer Care for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Customer Care website at <http://genesyslab.com/support>.

Information about supported hardware and third-party software is available on the Genesys Customer Care website in the following documents:

- [*Genesys Supported Operating Environment Reference Guide*](#).
- [*Genesys Supported Media Interfaces Reference Manual*](#).

Consult these additional resources as necessary:

- [*Genesys Hardware Sizing Guide*](#), which provides information about Genesys hardware sizing guidelines for the Genesys 7.x and 8.x releases.

- [*Genesys Interoperability Guide*](#), which provides information on the compatibility of Genesys products with various Configuration Layer Environments; Interoperability of Reporting Templates and Solutions; and Gplus Adapters Interoperability.
- [*Genesys Licensing Guide*](#), which introduces you to the concepts, terminology, and procedures relevant to the Genesys licensing system.
- [*Genesys Database Sizing Estimator 8.0 Worksheets*](#), which provides a range of expected database sizes for various Genesys products.

Document Conventions

This document uses certain stylistic and typographical conventions—introduced here—that serve as shorthands for particular kinds of information.

Document Version Number

A version number appears at the bottom of the inside front cover of this document. Version numbers change as new information is added to this document. Here is a sample version number:

80fr_ref_06-2008_v8.0.001.00

You will need this number when you are talking with Genesys Technical Support about this product.

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, might sometimes contain minor spelling, capitalization, or grammatical errors. The text accompanying and explaining the screen captures corrects such errors *except* when such a correction would prevent you from installing, configuring, or successfully using the product. For example, if the name of an option contains a usage error, the name would be presented exactly as it appears in the product GUI; the error would not be corrected in any accompanying text.

Type Styles

[Table 20](#) describes and illustrates the type conventions that are used in this document.

Table 20: Type Styles

Type Style	Used For	Examples
Italic	<ul style="list-style-type: none"> Document titles Emphasis Definitions of (or first references to) unfamiliar terms Mathematical variables <p>Also used to indicate placeholder text within code samples or commands, in the special case where angle brackets are a required part of the syntax (see the note about angle brackets on page 135).</p>	<p>Please consult the <i>Genesys Migration Guide</i> for more information.</p> <p>Do <i>not</i> use this value for this option.</p> <p>A <i>customary and usual</i> practice is one that is widely accepted and used within a particular industry or profession.</p> <p>The formula, $x + 1 = 7$ where x stands for . . .</p>

Table 20: Type Styles (Continued)

Type Style	Used For	Examples
Monospace font (Looks like teletype or typewriter text)	<p>All programming identifiers and GUI elements. This convention includes:</p> <ul style="list-style-type: none"> The <i>names</i> of directories, files, folders, configuration objects, paths, scripts, dialog boxes, options, fields, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages. The values of options. Logical arguments and command syntax. Code samples. <p>Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.</p>	<p>Select the Show variables on screen check box.</p> <p>In the Operand text box, enter your formula.</p> <p>Click OK to exit the Properties dialog box.</p> <p>T-Server distributes the error messages in EventError events.</p> <p>If you select true for the inbound-bsns-calls option, all established inbound calls on a local agent are considered business calls.</p> <p>Enter exit on the command line.</p>
Square brackets ([])	A particular parameter or value that is optional within a logical argument, a command, or some programming syntax. That is, the presence of the parameter or value is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information.	smcp_server -host [/flags]
Angle brackets (< >)	<p>A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise.</p> <p>Note: In some cases, angle brackets are required characters in code syntax (for example, in XML schemas). In these cases, italic text is used for placeholder values.</p>	smcp_server -host <confighost>



Index

Symbols

[] (square brackets)	135
< > (angle brackets)	135

A

acknowledgement e-mail	50
Add Record block	122
agent last handling	50
angle brackets	135
ANI	34, 39, 123
application server	15
application variables	34
Assign block	114
Associate Service block	121
Attach Block	114
audience, for document	10
auto-response e-mail	79

B

Backend block	120
Begin Parallel block	114
blocks	
adding to the canvas	95
connecting	96
Context Services	121
descriptions of	113
eServices	117
flow control	114
IRD equivalent	122
routing	116
server-side	119
voice treatment	117
brackets	
angle	135
square	135
Branching block	114
branching in Expression Builder	35

Business Attributes	53, 54
business processes	126
Business Rule block	120
buttons on main toolbar	41

C

Cancel Call block	117
Cancel Event block	114
Cancel Record block	122
canvas area	26
Category codes	52
Chat Transcript block	118
Check Interval	76
CheckAgentState function	37
Classification Segmentation	124
Classify Interaction block	118
ClearCase	70
code validation	89
commenting on this document	11
Complete Service block	121
Complete State block	121
Complete Task block	121
Composer block listing	113
Composer Voice	9
conditional routing decisions	35
Configuration Database objects	52, 66
Configuration Server	
connecting to	66
menu	42
preferences	42
connecting blocks	96
Contact Server	53
Context Services	53
blocks	121
preferences	42
conventions	
in document	134
type styles	134
conversation management	55
Create Customer block	121

Create E-mail block 118
 Create Interaction block 118
 Create SMS block 118
 Create User Announcement block 117
 creating
 new project 41
 new routing application 61
 customizing perspectives 99

D

Database Hints 76
 Date segmentation functionality 123
 Day of Week segmetation functionality 124
 DB Data block 115, 120
 default destination 116
 Default Route block 116
 default.ixnprocess 65
 Delete User Announcement block 117
 Detach block 115
 diagram editor 26
 Diagram menu 41
 Disconnect block 115
 Disposition Code 53
 DNIS 34, 39, 78, 123
 Do Not Call block 122
 document
 audience 10
 change history 11
 conventions 134
 errors, commenting on 11
 version number 134
 Dynamic Target block 83

E

Eclipse 9
 Eclipse Plug-ins 70
 ECMAScript block 115
 ECMAScript in Expression Builder 36
 editor for diagrams 26
 editor for SCXML code 22
 E-mail Accounts Business Attribute 53
 E-mail Forward block 118
 E-mail Response block 118
 enabling routing functionality 58
 End Parallel block 115
 endpoints 70
 Enter State block 121
 Entry block 33, 115
 error handling 31
 error or exception conditions 96
 eServices blocks 117
 event handling 31
 Exception Link 96

Exceptions dialog box 31
 Exit block 115
 Expression Builder 35, 53
 External Service block 120

F

Field codes 53
 flow control blocks 114
 Flow Control Blocks in an IPD 114
 folders in Project Explorer 20
 font styles
 italic 134
 monospace 135
 Force Route block 116
 Functional Modules 37

G

Generic segmentation object 124
 Genesys Functional Modules 37
 Genesys Voice Platform 9
 genesys.queue.modules 37
 GVP 9
 GVPI 131

I

Identify Contact block 118
 Identify Customer block 121
 intended audience 10
 interaction processing diagram
 blocks 17
 definition 16
 example 26
 linking IPDs 18
 properties 69
 publishing 87
 samples 56
 Interaction Queue block 17, 73, 113
 interaction queue definition 73
 Interaction Routing Designer 16
 Interaction Subtype 53
 Interaction Subtype Business Attribute 50
 IRD and Composer 16
 IRD business processes 126
 IRD objects mapped to Composer blocks 122
 italics 134
 IVR block 117

J

Java Composer Project 41
 JSON Array 115

K

Knowledge Management. 52

L

language for routing application 65
 List Objects Manager/Builder, 39
 load balancing routing rule 116
 locale 65
 Log block 115
 Looping block 115

M

main toolbar 41
 managing conversations 55
 maximizing/minimizing views 99
 media server 52
 Media Server block 17
 adding 70
 definition 113
 endpoints 70
 menu bar 41
 META-INF folder 21
 minimize button 100
 monospace font 135
 multimedia processing blocks 117
 multimedia processing stages 49

N

NET Composer Project. 41
 new project creation 41

O

Offer personalization 53
 offline mode 66
 online mode 66
 Operator buttons in Expression Builder 36
 OPM block 120
 Oracle database 76
 orchestration 55
 Orchestration Platform 10
 order for extracting interactions 76
 Outbound blocks 122
 Outline view 27
 Output Link 96

P

palette

 block categories 17
 location 27
 workflow block categories 29
 Pause block 117
 percentage routing rule 116
 perspectives
 changing 98
 Composer perspective 97
 customizing 99
 planning and design 49
 Play Application block 117
 Play Message block 117
 Play Sound block 117
 predefined statistics 38
 Preferences
 enabling routing functionality 58
 routing preferences 42
 preparation 49
 Project
 creating new 64
 defined 19
 folders 20
 templates 56
 type 64
 variables 34
 workspace 20
 Project Explorer 19, 27
 Project templates 62
 Properties view 27, 30
 publishing an IPD 87

Q

Query Customer block 121
 Query Services block 121
 Query States block 121
 Query Tasks block 121
 Queue Interaction block 116
 Queue Reference block 83
 queue.modules 37

R

Raise Event block 115
 reason code 116
 Record Processed block 122
 required software 52
 Reschedule Record block 122
 Resource property 28
 Route Interaction block 116
 routing application defined 16
 routing blocks 116
 routing defined 15
 routing functionality
 enabling 58

routing point	16
routing rules	116

S

sample interaction processing diagrams	56
Scheduling	76
Screen Interaction block	119
Script property	36
SCXML code	16, 17, 41
SCXML editor	22
SCXML State block	116
SCXML-based routing applications	15
Send E-mail block	119
Send SMS block	119
server-side blocks	119
service level agreement routing	116
Service Model	55
Service personalization	53
Service Type	53
Set Default Route block	117
single-tenant environment	67
Skill Expression Builder	40
software required	52
square brackets	135
Standard responses	53
Start Service block	121
Start Task block	121
state	54
Statistics Manager and Builder	38
statistics routing rule	116
Stop block	83
Stop Interaction block	116
Stop Processing Reason	53
Subroutine block	116
Subversion	70

T

Target block	116
tasks	54, 55
templates for IPDs	56
tenant	67
Time segmentation functionality	124
Timezone preferences	42
toolbar buttons	41
type styles	
conventions	134
italic	134
monospace	135
typographical styles	134

U

Universal Contact Server	53
Universal Contact Server Database	53
Update Contact block	119
Update Customer block	121
Update Record block	122
URS functions	37
URS predefined statistic	38
User Data block	116
User Input block	117, 119

V

validating code	78, 89
variables definition	34
version numbering, document	134
view	
Condition	76
defining	75
Order	76
scheduling	76
view for pulling interactions	73
viewing multiple diagrams	102
voice treatment blocks	117

W

Wait block	116
Web Request block	120
Web Service block	120
white blocks	83
Window menu	42
Workbin block	17, 114
Workflow block	17, 81, 113
workflow cannot create	59
workflow diagram	
creation	78
general properties	29
name in Workflow block	28
preferences	42
viewing multiple	102
workflow variables	34
Workflow-generated blocks	17
workflow-generated blocks	83