Orchestration Server 8.1.2

Deployment Guide

For releases 8.1.0 through 8.1.2

About Genesys

Alcatel-Lucent's Genesys solutions feature leading software that manages customer interactions over phone, Web, and mobile devices. The Genesys software suite handles customer conversations across multiple channels and resources—self-service, assisted-service, and proactive outreach—fulfilling customer requests and optimizing customer care goals while efficiently using resources. Genesys software directs more than 100 million customer interactions every day for 4000 companies and government agencies in 80 countries. These companies and agencies leverage their entire organization, from the contact center to the back office, while dynamically engaging their customers. Go to www.genesyslab.com for more information.

Each product has its own documentation for online viewing at the Genesys Technical Support web site 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, the Genesys logo, and T-Server are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other trademarks and trade names referred to in this document are the property of other companies. 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.

Technical Support from Genesys

If you have purchased support directly from Genesys, please contact Genesys Technical Support at the regional numbers provided on page 15. For complete contact information and procedures, refer to the ***Genesys Technical Support Guide***.

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: 81ors_dep_02-2012_v8.1.201.00

Table of Contents

List of Procedures 9

Preface 11

About Orchestration Server 11
CIM Platform 12
Intended Audience 14
Making Comments on This Document 15
Contacting Genesys Technical Support 15
Document Change History 16
Release 8.1.2 16
Release 8.1.1 16

Chapter 1 **Orchestration Server Overview** 19

What is Orchestration? 19
What is SCXML? 20
Why SCXML? 20
SCXML-Based Applications 21
Application Servers 24
Orchestration Platform Capabilities 24
Modeling/Managing the Customer Service Process 25
Coordinating Customer Interactions and Dialogues 25
Orchestrating Resources and Product Services 26
Service History Binding 26
New Features 27
Release 8.1.2 27
Release 8.1.1 27
Release 8.1.0 28
Security 28
Client-Side Port Definition 29

Chapter 2 **Orchestration Server Architecture and Interaction Flows** 31

Orchestration Server Architecture 31
Voice Interactions 32

eServices Interactions 34
Enhanced Orchestration Multi-Site Support 37

Chapter 3 **Persistence, High Availability, and Load Balancing 45**

Persistence 45
Persistence Design Features 45
Persistent Storage Operation 46
Deploying Persistent Storage 48
Configuring Persistent Storage 49
High Availability and Load Balancing 52
Clusters 52
Load Balancing Operation 54

Chapter 4 **Deployment Process 57**

Before You Begin 57
How to Configure 58
Packaging 58
Management Framework CD 58
eServices CDs 58
Real-Time Metrics Engine CD 59
Genesys Info Mart CD 59
Orchestration Component Installation Order 59
Order for Individual Components 60
What Each Component Does 61

Chapter 5 **Configuring Orchestration Server 63**

How to Configure 63
Importing the ORS Application Template and Creating the ORS Application Object 68
Defining Client-Side Port Information 75
Configuring Options in the Orchestration Server Application Object 75
Configuring the Template Optons in the orchestration Section 76
Adding/Setting Non-Template Options in the orchestration Section 77
Configuring the Template Options in the persistence Section 77
Adding/Setting Non-Template Options in the persistence Section 78
Configuring the Template Options in the scxml Section 79
Adding/Setting Non-Template Options in the scxml Section 80
Configuring the Options in the log Section 82
Configuring ORS Memory Optimization 83
Configuring ORS Clustering 85
Configuring Other Options That Affect Orchestration Server 89

Configuring the application Option on a DN, RoutePoint, or eServices Queue Object 90
Configuring Options Specified in Script Objects of Type CfgEnhancedRouting (Enhanced Routing Script) 91
Configuring ORS to Operate with eServices Interactions 94
TMakeCall Parameter Support in Switch Configurations 100
Other Manual Configuration Operations 101
Creating Business Attributes 101
Manually Configuring Stat Server 102
Premise T-Server for Network Routing 102
Orchestration Server Configuration Options 102
Setting Options 102
Orchestration Server Options 103
Summary of Options 103
List of Orchestration Server Options 104
Orchestration Server Option Descriptions 109
mcr-pull-interval 109
mcr-pull-limit 109
session-hung-timeout 109
cassandra-listenport 110
cassandra-nodes 110
fips_enable 110
http-client-side-port-range 111
http-enable-continue-header 111
http-max-age-local-file 111
http-max-cache-entry-count 112
http-max-cache-entry-size 112
http-max-cache-size 112
http-max-redirections 113
http-no-cache-urls 113
http-proxy 113
http-ssl-ca-info 114
http-ssl-ca-path 114
http-ssl-cert 114
http-ssl-cert-type 115
http-ssl-cipher-list 115
http-ssl-key 115
http-ssl-key-type 116
http-ssl-random-file 116
http-ssl-verify-host 116
http-ssl-verify-peer 117
http-ssl-version 117
https-proxy 117
max-compiler-cache-size 118

max-compiler-cached-docs 118
max-compiler-cached-doc-size 118
max-includes 119
max-preprocessor-cache-size 119
max-preprocessor-cached-docs 119
max-preprocessor-cached-doc-size 119
password 120
persistence-default 120
persistence-max-active 120
session-processing-threads 121
system-id 121
x-server-trace-level 121
x-server-gcti-trace-level 121
x-server-config-trace-level 122
x-print-attached-data 122
om-memory-optimization 122
om-max-in-memory 123
om-delete-from-memory 123
name 123
super_node 124
hostname 124
port 124
application 125
alternate-url 126
fetch-timeout 128
http-useragent 128
http-version 128
max-age 129
max-duration 129
max-loop-count 130
max-stale 130
url 131
{Parameter Name} 132
Operational Reporting 133
Configuring Operational Reporting 134
Accessing and Using the Operational Reporting Interface 136

Chapter 6 **Business Logic for Advice of Charge 137**

Implementation of Business Logic for Advice of Charge 137

Chapter 7 **SCXML Application Support 141**

Creating SCXML-Based Applications 141
Genesys Composer 142

Deploying 143
Samples 144

Chapter 8 **Installing Orchestration Server 145**

Installation Package Location 146
Installing on Windows Operating Systems 146
Installing on UNIX-Based Platforms 150
Installing Orchestration Server on a UNIX-Based Platform 151

Chapter 9 **Starting and Stopping Procedures 153**

Prestart Information 153
Starting Orchestration Server 154
Starting Orchestration Server Manually 155
Stopping 156
Non-Stop Operation 157
Version Identification 158

Chapter 10 **Uninstalling Orchestration Server 159**

Removing Orchestration Server with Genesys Administrator 160
Removing Orchestration Server Manually 160

Appendix **Installing and Configuring Apache Cassandra Server 163**

Installing, Configuring, Starting, and Testing a Cassandra Node 164
Downloading and Extracting the Install Image 164
Configuring Cassandra Server 165
Starting Cassandra Server and Loading the Schema 169
Testing Cassandra Server 171
Sample Cassandra Output 173

Supplements **Related Documentation Resources 177**

Document Conventions 180

Index **..... 183**

List of Procedures

Configuring the Cassandra Persistence Type in ORS Using Configuration Manager 50
Configuring the Cassandra Persistence Type in ORS Using Genesys Administrator 51
Logging into Configuration Manager 68
Importing the application template for Orchestration Server 71
Creating/Configuring the Orchestration Server Application object 72
Viewing/changing template options in the orchestration section 76
Adding and setting non-template options in the orchestration section . . 77
Viewing/changing template options in the persistence section 78
Adding and setting non-template options in the persistence section . . . 78
Viewing/changing template options in the scxml section 80
Adding and setting non-template options in the scxml section 81
Adding and setting non-template options in the log section 82
Adding an mcr section, then adding and setting memory optimization options for an ORS instance 84
Adding a cluster section, then adding and setting clustering options for an ORS node 86
Adding a web_services section, then adding and setting web services options 87
Configuring a default port for each node of a cluster 89
Adding and setting the application option in the Orchestration section of a DN or Queue object 90
Adding and setting options specified in the Application section of CfgEnhancedRouting Script objects 92
Adding and setting the {Parameter Name} option in the ApplicationParms section of CfgEnhancedRouting Script objects 93
Configuring ORS to operate with eServices Interactions 95
Configuring a Super node for Operational Reporting Using Genesys Administrator 134
Configuring a Super node for Operational Reporting Using Configuration Manager 135

Manual deployment to an Application Server (IIS) 143
Installing Orchestration Server on Windows using the Installation Wizard. 147
Installing Orchestration Server on a UNIX platform 151
Starting Orchestration Server with Solution Control Interface 155
Starting Orchestration Server on UNIX-Based Platforms. 155
Stopping Orchestration Server using Solution Control Interface 157
Removing the Orchestration Server component with Genesys Administrator 160
Manually removing Orchestration Server on Windows. 160
Manually removing Orchestration Server on UNIX-Based Platforms. . 161
Downloading and extracting the install image 164
Configuring Cassandra Server 165
Configuring a multi-node Cassandra cluster 168
Starting Cassandra Server 169
Loading the Cassandra Schema 170
Testing Cassandra Server 172

Preface

Welcome to the *Orchestration Server 8.1 Deployment Guide.* This guide familiarizes you with Genesys Orchestration Server features, functions, and architecture; provides deployment-planning guidance, and explains how to configure, install, uninstall, start, and stop the Orchestration Server.

This document is valid for the 8.1.0 through 8.1.2 release(s) of this product.

This preface contains the following sections:

- About Orchestration Server, page 11
- Intended Audience, page 14
- Making Comments on This Document, page 15
- Contacting Genesys Technical Support, page 15
- Document Change History, page 16

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on page 177.

About Orchestration Server

Genesys Orchestration takes Genesys' core capability of routing and extends it, generalizes it, and integrates it more tightly with other Genesys products. Orchestration provides dynamic management of interactions through the use of business rule tools, dynamic data, and applications that are based on open standards.

For example, a bank customer calls an 800 number inquiring about mortgage preapproval. An IVR prompts him to enter his account number, then transfers him to an agent, who fills in an application form for him and asks him to fax some supporting documents. After he faxes the documents, he receives an automated SMS message thanking him and informing him that he will receive a response within 48 hours. The next day he receives an e-mail congratulating him on the approval of his application.

This example involves voice, IVR, fax, SMS, and e-mail channels. Orchestration is able to treat the entire transaction as a single service.

The Genesys Orchestration Server offers an open standards-based platform with a State Chart eXtensible Markup Language (SCXML) engine that enables

intelligent distribution of interactions throughout the enterprise, whether you have a single-tenant or a multi-tenant environment. The Orchestration Server in conjunction with the Universal Routing Server (URS) can direct interactions from a wide variety of platforms, such as toll-free carrier switches, premise PBXs or CDs, IVRs, IP PBXs, e-mail servers, web servers, and workflow servers. It can handle pure-voice, non-voice, and multimedia environments, enabling routing of each media type based on appropriate criteria. Routing strategies and business processes automate interaction routing to the most appropriate agent/resource based on factors such as the type of inquiry, the business value of the customer interaction, context and customer profile, and the media channel.

CIM Platform

Universal Routing Server and Orchestration Server are a part of the Genesys Customer Interaction Management (CIM) Platform that provides the core interaction management functionality.

Universal Routing Server allows the processing of interactions of any type. Together with Orchestration Server, you now have the possibility to coordinate processing of multiple interactions that are involved in a single service.

CIM Components

Genesys CIM is the collection of core servers that enable the rest of your Genesys environment to process the thousands of interactions that represent the needs of your customers. The CIM Platform consists of the following Genesys products:

- Genesys Framework (including Genesys Administrator)
- Interaction Management, which in turn consists of:
  - Universal Routing Server
  - Orchestration Server
  - Real-Time Metrics Engine (Stat Server)
- Composer
- Reporting (CCPulse+, CCA, and for non-voice - Reporting Toolkit (restricted)
- Supervisor Desktop

Figure 1 depicts CIM graphically.

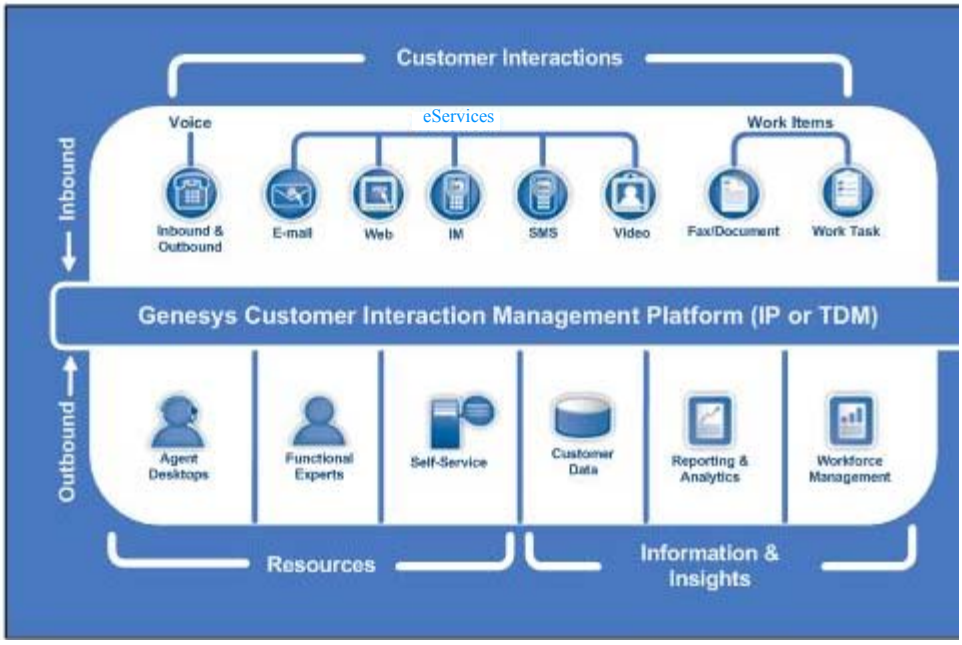

Figure 1: Customer Interaction Management Platform

Orchestration Server

The CIM Platform can handle various media channels. Orchestration Server in conjunction with Universal Routing Server and Interaction Server provides a platform for different Genesys solutions to work together managing interactions regardless of media type.

This multimedia capability includes some parts of the Genesys Customer Interaction Management (CIM) Platform, plus certain of the media channels that run on top of the Platform as follows:

- From the CIM Platform, Orchestration Server provides centralized handling of interactions regardless of media type.
- From the media channels, at least one of the following:
  - Genesys E-mail
  - Genesys Chat
  - Genesys Open Media—The ability to add customized support for other media (fax, for example)
  - Optionally, Web Collaboration—The ability for agents and customers to cobrowse (simultaneously navigate) shared web pages. This is an option that you can add to either Genesys Chat or Inbound Voice.

An Orchestration solution can consist of many interactive and integrated components. An example is shown in Figure 2.

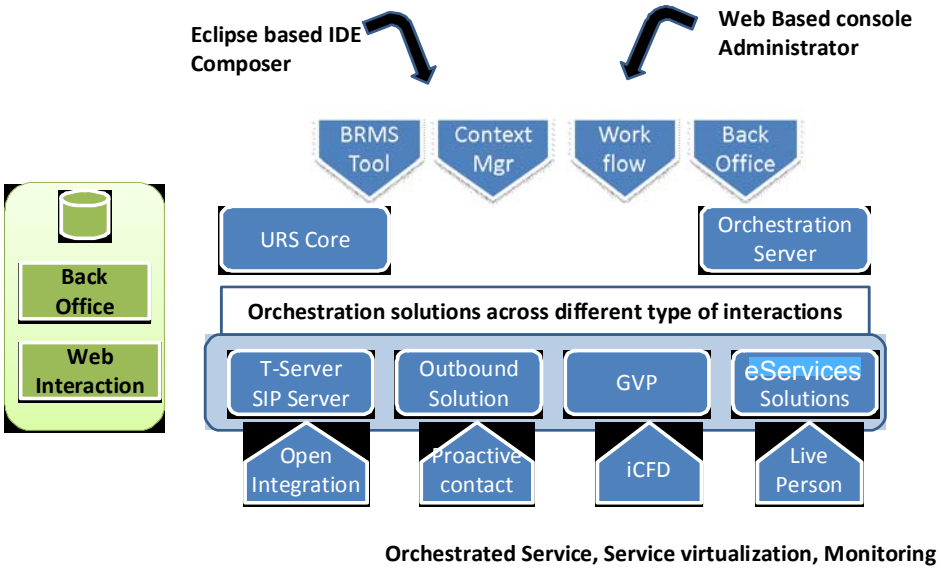

Figure 2: An Example of an Orchestration Solution

Note: Any functioning solution (platform plus channels) that includes any part of the Interaction Management sector requires Universal Routing Server.

Orchestration is a new suite capability for Genesys platforms. It is a logic integration platform that provides the ability to create customer service applications:

- Across multiple interactions with a customer.
- Across multiple channels for interacting with a customer.
- That leverage the spectrum of both Genesys and third-party capabilities
- That are integrated and consistent with an organization's business processes.

Orchestration provides the ability to create multiple application types, such as routing strategies, session logic (Genesys Business Processes), service logic, combinations, and more.

Intended Audience

This document is primarily intended for those involved in deploying the Orchestration Server with Genesys Universal Routing 8.1. It has been written with the assumption that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications
- Web concepts such as “web services”

- Network design and operation
- Your own network configurations

You should also be familiar with Genesys Framework architecture and functions and Genesys eServices (if installed).

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 Technical Support 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 Technical Support

If you have purchased support directly from Genesys, contact Genesys Technical Support at the following regional numbers:

Region	Telephone	E-Mail
North America and Latin America	+888-369-5555 (toll-free) +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-127-645-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Malaysia	1-800-814-472 (toll-free) +61-7-3368-6868	support@genesyslab.com.au
India	000-800-100-7136 (toll-free) +61-7-3368-6868 (International)	support@genesyslab.com.au
Japan	+81-3-6361-8950 +61-7-3368-6868 (International)	support@genesyslab.co.jp
Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

Document Change History

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

Release 8.1.2

- Chapter 1, "Orchestration Server Overview":
  - The section, "New Features" on page 27 has been updated with the new features in this release.
- Chapter 2, "Orchestration Server Architecture and Interaction Flows"
  - Persistence Storage has been added to the existing descriptions in "Orchestration Server Architecture" on page 31. See Figure 3 and Figure 5.
  - Enhanced Orchestration Multi-Site Support section added. See page 37.
- Chapter 3, "Persistence, High Availability, and Load Balancing"
  - New method for configuring persistence storage. See "Persistence Design Features" on page 45.
  - The `<scxml>` tag's attribute `_persist` default value has been changed to `false`. See "Enabling Persistence in the SCXML Application" on page 51.
- Chapter 5, "Configuring Orchestration Server":

The following new options have been added in the option section: `scxml`:

- `fips_enable`
- `http-client-side-port-range`
- `max-compiler-cache-size`
- `max-compiler-cached-docs`
- `max-compiler-cached-doc-size`
- `max-preprocessor-cached-doc-size`
- `persistence-default`

The following new options have bee added in the option section- `Application`:

- `alternate-url`

Release 8.1.1

- Chapter 1, "Orchestration Server Overview":
  - The section, "New Features" on page 27 has been updated with the new features in this release.
- Chapter 5, "Configuring Orchestration Server":
  - In Table 5 on page 104, the new `scxml.password` option has been added to replace the `scxml.http-ssl-key-password` option.

- In Procedure: Configuring a Super node for Operational Reporting Using Genesys Administrator, on page 134 and Procedure: Configuring a Super node for Operational Reporting Using Configuration Manager, on page 135 the references to “Reporting node” have been changed to “Master Super node”.
- A new section, “Configuring ORS to Operate with eServices Interactions” on page 94 has been added, which includes new graphics and a new procedure.

- Chapter 6, “Business Logic for Advice of Charge”
  - This is a new chapter, added to describe how ORS implements the SIP Server Advice of Charge (AoC) feature.
- Chapter 7, “Installing Orchestration Server”:
  - In the Procedure: Installing Orchestration Server on Windows using the Installation Wizard, on page 147 and Procedure: Installing Orchestration Server on a UNIX platform, on page 151, the steps to install licensing was removed.

Chapter

1 Orchestration Server Overview

The Orchestration Server (ORS) is responsible for executing orchestration logic (SCXML) that is provided by 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 functions.

Please refer to the *Genesys Universal Routing 8.1 Deployment Guide* for a thorough overview of Universal Routing capabilities, features, licensing, security, and migration issues.

This chapter includes the following sections:

- What is Orchestration?, page 19
- Orchestration Platform Capabilities, page 24
- New Features, page 27
- Security, page 28

What is Orchestration?

Orchestration is a new suite capability for Genesys platforms. It is a logic integration platform that provides the ability to create customer service applications:

- Across multiple interactions with a customer.
- Across multiple channels for interacting with a customer
- That leverage the spectrum of Genesys capabilities and third-party tools.
- That are integrated and consistent with an organization's business processes.

Orchestration provides the ability to create multiple application types, such as routing strategies, session logic, service logic, and Service State logic of the interaction.

Universal Routing (Orchestration Server plus Universal Routing Server) takes a more open approach to routing strategies. While the Universal Routing Server executes routing strategies that are created by using the Genesys Interaction Routing Language (IRL)[1], the Orchestration Server executes routing applications that are written in SCXML (State Chart EXtensible Markup Language). SCXML-based applications are hosted on an Application Server/Web Server and provisioned through Genesys Administrator by defining the HTTP definition of the path and parameters.

What is SCXML?

SCXML is an XML-based markup language that is based on Harel State charts.

The language supports parallel states, sub-state, entry/exit functions, transition conditions, and other state machine mechanisms.

It is well-suited to event-driven solutions, but it can also be used for sequential workflow solutions.

SCXML provides the backbone logic for Orchestration applications. The SCXML application utilizes Functional Modules that exist in URS, Interaction Server and other components, which are exposed to be used within the application that is driving the business logic

Why SCXML?

Customer service occurs in a highly event-driven, asynchronous environment. From a routing-application standpoint, SCXML is ideal for this type of environment because:

- SCXML can accommodate various customer service states and the transitions between them. While relatively new as a notation/language, SCXML is well-proven for building state-based models and facilitates the process of orchestrating customer service solutions.
- When fully implemented, this new SCXML-support feature will allow integration of your existing Genesys routing with other operational systems in your enterprise.
- While IRD will continue to provide its rich, graphically oriented approach, customers will benefit when routing logic is expressed in SCXML. SCXML applications can be realized in many ways:

1.IRL scripts are the output of the current Interaction Routing Designer (IRD) application.

- Built using the Genesys Composer Eclipse-based environment (refer to the *Genesys Composer 8.1 Deployment Guide* and *Genesys Composer 8.1 Help*)
- Built using a simple text editor or XML editor
- Generated by an XML-based Application Server framework with which you are already comfortable
- Created by a third-party integrated-development environment

SCXML-Based Applications

Orchestration Server 8.1 with Universal Routing 8.1 supports SCXML plus ECMAScript as a routing language. ECMAScript is functionally equivalent to JavaScript and helps customers build applications in a natural-language construct. Orchestration Server supports SCXML in accordance with the World Wide Web Consortium (W3C) Working Draft (see Table 1).

The core SCXML provides state chart functionality, while Orchestration-specific instructions are specified in the executable content of SCXML in the form of SCXML extensions (action elements) and/or ECMA script extensions (properties of special ECMA script objects).

URS 8.1 extensions to the SCXML executable content are grouped by Functional Module as identified in the *Genesys 8.1 SCXML Technical Reference*. (see Table 2 on page 22).

Reference Documents

Table 1 lists publicly available reference documents for the SCXML-based routing-application language:

Table 1: External Reference Documents

Document Title	File Name
State Chart XML (SCXML): State Machine Notation for Control Abstraction, May 2009	`http://www.w3.org/TR/2009/WD-scxml-20090507/`
ECMAScript Language Specification	`http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf`
Standard ECMA-327	`http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-327.pdf`

Table 2 lists Genesys-supplied resource documents for the SCXML-based routing-application language:

Table 2: Genesys Reference Documents

Document Title
Genesys 8.1 SCXML Technical Reference
Genesys 8.1 SCXML Samples

Simple Sample of SCXML-Based Applications

The following SCXML code sample for a simple voice routing application provides a general introduction to SCXML-based applications:

```
<scxml version="1.0" xmlns="http://www.w3.org/2005/07/scxml"
xmlns:queue="www.genesyslab.com/modules/queue"
initial="initial">
   <state id="initial" >
      <transition event="interaction.added" target="queued"/>
   </state>
   <state id="queued">
      <onentry>
         <queue:submit queue="'vq1'" priority="5" timeout="100">
            <queue:targets type="agent">
               <queue:target name="'Agent1'"/>
               <queue:target name="'Agent2'"/>
               <queue:target name="'Agent3'"/>
            </queue:targets>
         </queue:submit>
      </onentry>
      <transition event="queue.submit.done" target="exit" />
      <transition event="error.queue.submit" target="error" />
      <transition event="interaction.deleted" target="exit" />
   </state>
   <final id="exit" />
   <final id="error" />
</scxml>
```

Functional Modules

All Orchestration-related functionality is categorized by Functional Module as described in the *Genesys 8.1 SCXML Technical Reference* document that is listed in Table 2. The preceding example uses the Queue Functional Module. The namespace definition of the Queue Functional Module that is used in the preceding application is the following:

```
xmlns:queue="www.genesyslab.com/modules/queue"
```

The preceding functional module can be called/addressed within the logic of the application as follows:

```
<queue:submit queue="'vq1'" priority="5" timeout="100">
```

Universal Routing 8.1 provides the following Functional Modules:

- **Classification** module. This module element uses the namespace label `classification` which stands for `www.genesyslab.com/modules/classification`. It implements the ability to classify and screen interaction content to help orchestration logic determine what the customer wants.
- **Queue** module. This module element uses the namespace label `queue` which stands for `www.genesyslab.com/modules/queue`. It implements the target selection functionality of URS (finding resources for interactions and delivering interactions to the resource).
- **Dialog** module. This module element uses the namespace label `dialog` which stands for `www.genesyslab.com/modules/dialog`. It implements the call treatment functionality of URS.
- **Web Services** module. This module provides web services support in Orchestration, and covers both Web 2.0 RESTful web services interface and legacy SOAP Web Services interface.
- **Statistics** module. This module element uses the namespace label `metric` which stands for `www.genesyslab.com/modules/statistic`. It implements the statistics retrieving functionality of URS.
- **Interaction** module. This module element uses the namespace label `interaction` which stands for `www.genesyslab.com/modules/interaction`. It implements getting and changing interaction related data, such as attached data.
- **Session** module. This module element uses the namespace label `session` which stands for `www.genesyslab.com/modules/session`. It encapsulates all other URS functionality.
- **Resource** module. This module element uses the namespace label `resource` which stands for `www.genesyslab.com/modules/resource`. A resource can be either a human (such as an agent) or a device (such as an IVR).

Developers specify Functional Module action items as custom action elements inside SCXML executable content. All Functional Modules are prefixed with the corresponding namespace label; for example:

`queue:submit` (in this case, the queue prefix stands for `www.genesyslab.com/modules/queue`)

`dialog:playandcollect` (in this case, the `dialog` prefix stands for `www.genesyslab.com/modules/dialog`)

When Functional Modules are executed, events return back from the platform to the instance of logic that is running the SCXML document that requested the action.

Functional Module functions and data are exposed (and used) as properties of ECMA Script objects. ORS provides a built-in ECMA Script object for every Functional Module listed above.

Application Servers

You deploy SCXML-based applications to your production environment by publishing them to an Application Server, which then performs two main functions:

1. Upon HTTP request, the Application Server is responsible for providing the routing strategy logic to ORS in the form of a document (or series of documents).
2. The Application Server may interact with ORS during the strategy processing to perform business-specific tasks, or to provide access to other enterprise systems.

Supported Application Servers

Genesys supports the following types of Application Server software:

- Microsoft Internet Information Services (IIS), which was formerly called Internet Information Server). Genesys supports IIS 6.0–7.0.
- JBoss Application Server (or JBoss AS). This free software/open source Java EE-based Application Server is usable on any operating system that Java supports. Genesys supports version JBoss 4.0–6.0.
- IBM's Websphere Application Server (WAS). This software Application Server, built using open standards (such as Java EE, XML, and Web Services) works with a number of Web servers. Genesys supports IBM Websphere Application Server 5.1–7.0.
- See the following note on Tomcat.

Servlet Engines

Composer bundles Tomcat and deploys it as a Windows Service. Apache 2.0 and Tomcat 6.0 are supported.

Orchestration Platform Capabilities

Orchestration enables an enterprise to model customer service processes[1] and dialogues in order to:

- Deliver a consistent and tailored customer experience across interaction channels, no matter when and how the customer chooses to transact with the enterprise, and regardless of whether the transaction cycle spans multiple interactions over a period of time.

- Empower the enterprise to create differentiated services and products using flexible and adaptive business rules and process flows.

Orchestration provides four main areas of capability:

- Modeling and managing the customer service process with agility
- Coordinating customer interactions and dialog in the service process
- "Orchestrating" resources and product services spanning the Genesys suite in real time
- Service history binding of customer-enterprise dialogues to service-process work steps

Each capability is briefly discussed in the following sections.

Modeling/Managing the Customer Service Process

The Orchestration platform:

- Provides an integrated development tool (Composer) that is used to define a cohesive service process. Composer is available separately from the Orchestration Server component and is part of the Genesys CIM component.
  - The service process spans contact center and back-office touch points (IVR, e-mail, SMS, web, fax, agents, and knowledge experts in branch office) as a customer transacts with the enterprise.
  - The service process provides openness for the customer to use a commercially available language to describe the service process.
- Interacts and collaborates with an ecosystem of business processes (such as CRM or BPM) in the enterprise. Data exchange and events notification trigger the next-best action to serve the customer.
- Authors and embeds business rules in the model to promote business agility and adaptability in the service process flow.
- Provides users with access to a real-time view of sessions within an orchestrated environment. Users can employ existing reporting functionality with different media.

Coordinating Customer Interactions and Dialogues

The Orchestration platform tailors customer-enterprise dialogues and corresponding message content based on contextual knowledge of the

1.Service process is the flow of events, procedures and tasks to achieve the customer service activities that combine to form a dialogue between customer-facing resources (agent or device) and the cus tomer. The terms "customer service process" and "customer service application" are used to describe this.

customer's service history that binds cross-channel interactions and service process work steps:

- Picks up and continues previous dialogues when the customer transacts on the same channel or switches channels in the middle of a long, live transaction.
- Proactively invokes timely and relevant notification to the customer in addition to invitations for assistance on any channel.
- Presents relevant up-sell engagement conditioning based on the customer's web behavior, qualification for a marketing and customer retention programs, and recency of acceptance/rejection of a previous up-sell engagement.

Orchestrating Resources and Product Services

The Orchestration layer executes the service process flow; and coordinates customer interactions and dialogues (such as response, proactive engagement for upsell, notifications, and agent assistance) by invoking Genesys product services such as the following:

- Genesys Voice Platform (GVP) services
- Proactive contact notification services (Outbound solution)
- eServices (e-mail, SMS, and web)
- Media channel services (TDM and IP voice)

There are two kinds of proactive engagements: transition of self service to assisted service (and vice versa) and up-sell engagement.

Service History Binding

Orchestration provides Context Management Services (through Universal Contact Server) for the customer-enterprise's dialogues.

A new service history (as an operational data store) contextually links interactions and dialogues to service process work steps:

- In order to communicate to the customer regarding its service progress
- In order to enhance the agent's insights into the customer's history in the context of service progress, enabling intelligent interaction with the customer
- So that service logic can assess the service state of the customer and use that information to determine the next best steps in the service process for the customer.

Next best steps might include:

- Sending a notification to the customer.
- Scheduling an appointment to follow up.

- Offering the customer a personalized interaction dialogue on the Voice self-service channel.

New Features

This section contains a brief description of the new features in Genesys Orchestration Server (ORS) 8.1.x releases. For features that were introduced in earlier versions of Orchestration Server, see the "Overview" chapter in the applicable deployment guide.

Release 8.1.2

- Orchestration platform now supports call treatment which just includes ORS and media server only. In the previous versions, this functionality was managed by URS working in tandem with ORS.
- Genesys Orchestration Server (ORS) now has features enhancing multisite routing from an ORS cluster.
- ORS now supports fallback URI, as a failover mechanism to provision SCXML application from a Web Server.
- This release of ORS supports multiple views of an Interaction queue, this provides flexibility to users to branch out earlier based on conditions up-front before kick starting a work flow.

Release 8.1.1

- ORS supports the `PrivateServiceRequest` method of TLib, which enables support for the SIP Server Advice of Charge (AoC) feature. AoC enables the creation of applications that implement the business logic to insert charge messages.
- ORS now encrypts the security password by using the Secure Socket Layer (SSL) protocol.
- ORS supports Cassandra 0.7x and later versions. (Cassandra 0.6x is no longer supported in ORS 8.1.1 and later releases.) Orchestration Server 8.1 and later releases require persistence storage, and the only supported method is Cassandra NoSQL.
- ORS supports Service Level routing rules.
- ORS supports user name, password, and headers in the `session:fetch` action.

Release 8.1.0

- ORS supports interaction persistence through the use of Cassandra. This functionality provides uninterrupted processing of sessions regardless of ORS instance failure. Other applications within the ORS cluster will pick up the pending task as stipulated in persistence database.

Note: Cassandra is a highly scalable second generation distributed database and is part of the Apache project. Cassandra services can also be implemented in an N+1 architecture similar to ORS and is bundled on the Universal Routing CD. For more information on Cassandra, please refer to `http://cassandra.apache.org`.

- ORS now facilitates a real-time view of sessions through Genesys Administrator. Users can see all current sessions listed by cluster, Orchestration service, and so on, as well as activity for up to the previous 24 hours.
- ORS now allows the use of Statistical Functional Module functions without subscribing the required statistics beforehand within the SCXML application.
- ORS is now supported natively on the Red Hat Enterprise Linux 5.5 32-bit and 64-bit operating systems, in addition to Windows Server 2003/2008.
- ORS now supports TCP/IP v6. TCP/IP v6 improves network security and addresses allocation and efficiency in routing traffic (not supported for Windows 2003).

Security

As described in the *Genesys 8.1 Security Deployment Guide*, Genesys uses the Transport Layer Security (TLS) protocol that is based on the Secure Sockets Layer (SSL) 3.0 protocol. TLS uses digital certificates to authenticate the user as well as to authenticate the network (in a wireless network, the user could be logging on to a rogue access point).

You can secure all communications (SSL/TLS) between Genesys components, using authentication and authorization (certification validation). This functionality is configurable so that you can secure all connections, a selected set of connections, or none of the connections.

Note: Summary information on Orchestration Server 8.1 security is presented in the following subsection. For detailed information on how to implement security within Genesys, see the *Genesys 8.1 Security Deployment Guide*. For information about how to deploy a third-party authentication system in order to control access to Genesys applications, see the *Framework 8.0 External Authentication Reference Manual*.

Client-Side Port Definition

The client-side port definition feature of Genesys security enables a client application (of server type) to define its connection parameters before it connects to the server application. This enables the server application to control the number of client connections. In addition, if the client application is located behind a firewall, the server application is able to accept the client connection by verifying its predefined connection parameters.

Table 3 indicates where client-side port configuration is supported for other servers.

Table 3: Component Support for Client-Side Port Security

Clients	Config. Server/Config. Server Proxy	T-Server	URS
ORS	No	Yes	No

Client-side port configuration is not supported for Interaction Server and Message Server.

For detailed information on client-side port configuration, see the "Client-Side Port Definition" chapter of the *Genesys 8.1 Security Deployment Guide*.

Chapter

2 Orchestration Server Architecture and Interaction Flows

This chapter describes the architecture and interaction flows of the Orchestration Server.

This chapter includes the following section:

- Orchestration Server Architecture, page 31

Orchestration Server Architecture

The following figures (Figure 3 and Figure 4) illustrate, respectively, the high-level architecture and interaction flow of Orchestration Server 8.1.

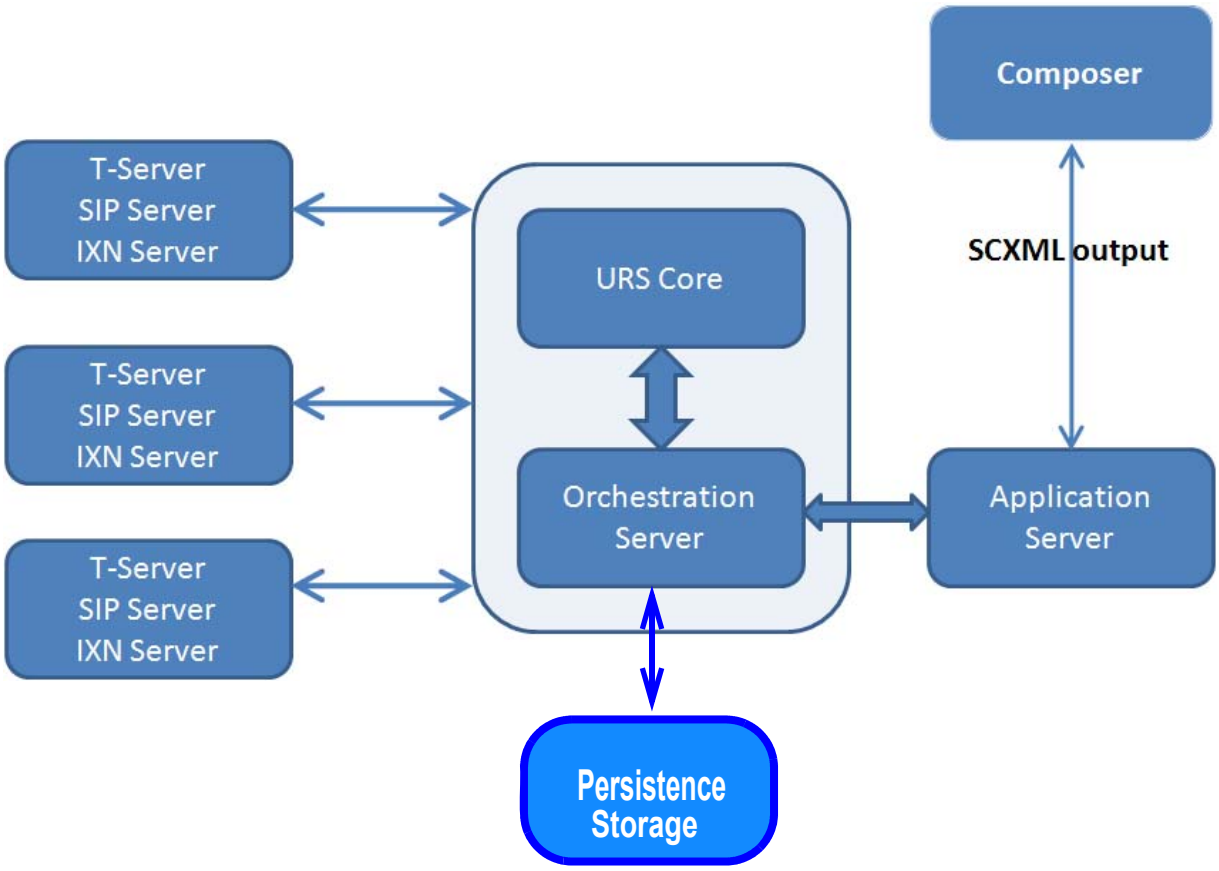

Figure 3: Orchestration Server Architecture

As the diagram in Figure 3 shows, Universal Routing consists of the Universal Routing Server core connected to the Orchestration Server (ORS). T-Servers, SIP Servers, and/or Interaction (IXN) Servers communicate with Universal Routing Server/Orchestration Server. Composer is available at a solution level to create and test SCXML-based routing applications and an Application Server provisions SCXML applications to ORS. Persistence Storage is used to store SCXML sessions and documents, as well as scheduled activities (such as start and stop).

Voice Interactions

Figure 4 shows a sample voice interaction flow that is based on the preceding architecture diagram.

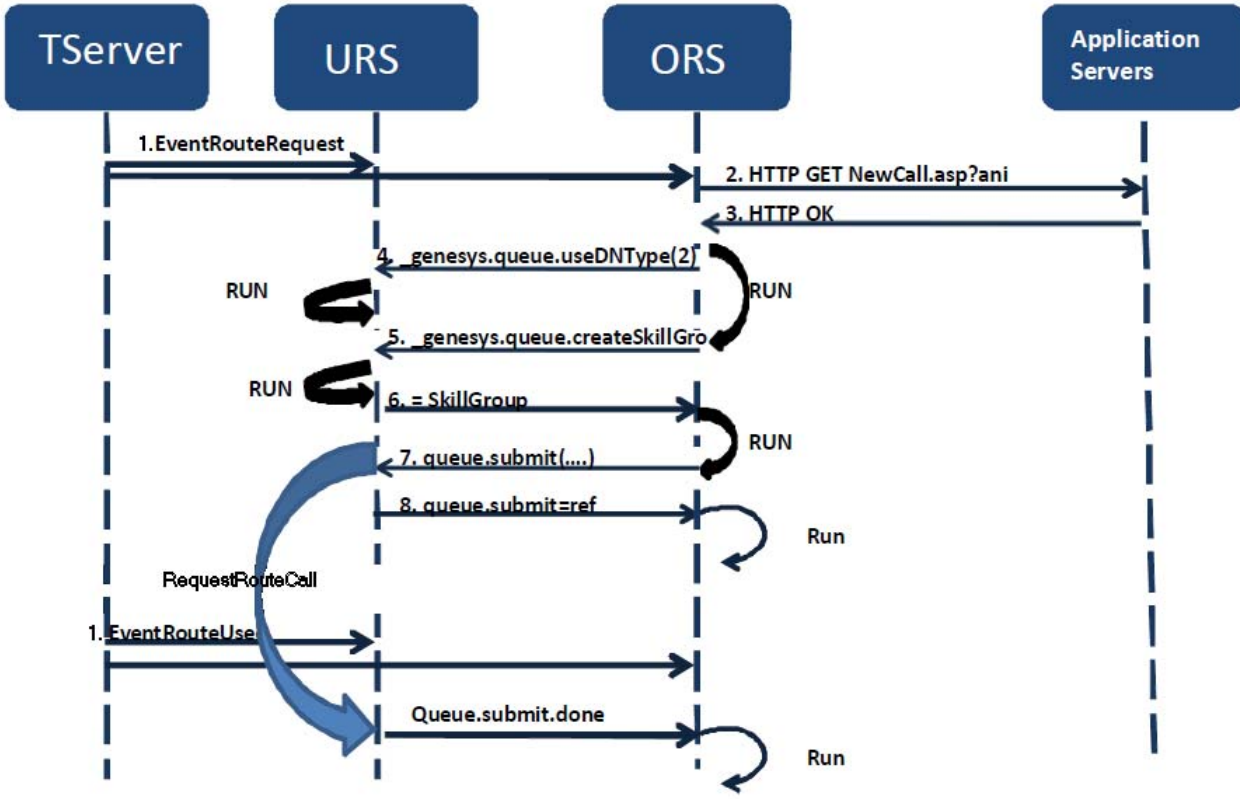

Figure 4: Sample Interaction Flow

The numbered sequences in the preceding figure are identified as follows:

Item 1. A new interaction arrives. T-server notifies both ORS and URS.

Item 2. If the configuration is enabled to use an ORS application, ORS asks the Application Server for an SCXML application session.

Item 3. The Application Server provides an application to ORS. ORS starts execution of an SCXML application.

Items 4,5,and 7. From time to time during SCXML application execution, ORS asks URS for assistance with tasks such as routing, invoking functional modules via SCXML action events

Items 6, 8, and 11 URS performs the required action(s) and reports the results to ORS.

Items 9 and 10. If needed, URS sends a request to T-Server. In this example, URS sends a request to T-Server that corresponds with the routing request that ORS sent to URS in step 7 (`queue.submit`) of the preceding figure.

eServices Interactions

This section provides information on supporting eServices interactions in Orchestration. Specifically, key use cases and key functional areas are described.

Figure 5 shows an eServices-specific architecture diagram for an Orchestration Server (ORS) deployment.

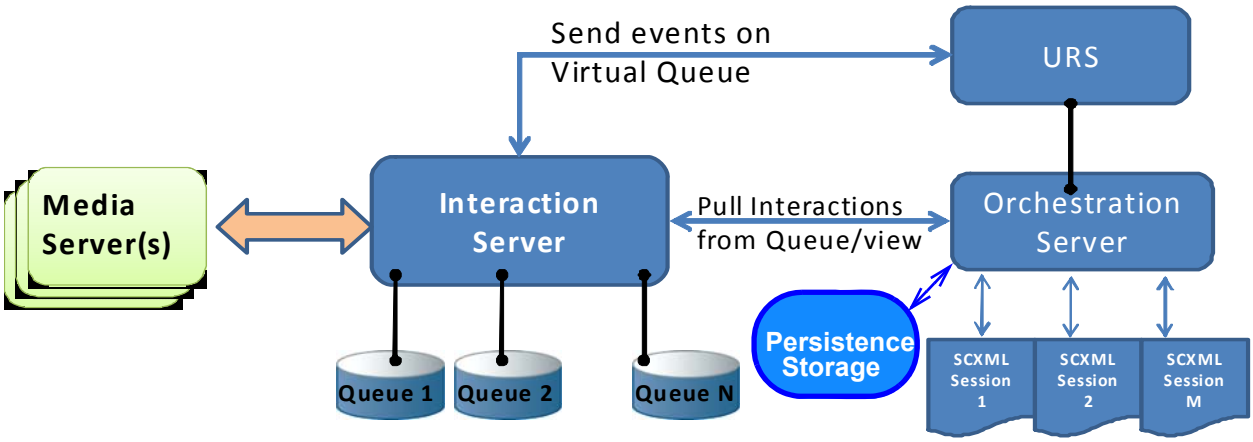

Figure 5: eServices-Specific ORS Architecture

The following are key design principles to consider when supporting eServices in Orchestration:

- Orchestration Server (ORS) connects to the Interaction Server. ORS registers as a Routing client so that ORS can use a subset of requests and events suitable for the Routing client.

Note: Depending on the Interaction Server application type, ORS may require a connection to an additional Interaction Server. (Interaction Server can be configured based either on an Interaction Server template or on a T-Server template). If the Interaction Server is configured based on an Interaction Server template, ORS needs to be connected as a *client* to one more Interaction Server(s) that was (were) configured based on a T-Server template, since communication between ORS and Interaction Server uses T-Server protocol. Both types of Interaction Servers must share the same configuration port.

- ORS processes interactions by "pulling" them from the Interaction Server. Request `RequestPull` is used to retrieve a subset of interactions from the specified queue/view combination. The ORS log shows `EventTakenFromQueue` as evidence that interactions were pulled from the queue/view.

- Only when interactions are "pulled" from the Interaction Server does ORS process them. Interactions may be created and placed into the queue by the Interaction Server, but ORS will only process an interaction after ORS has pulled it from this queue.

  This allows:
  - Interactions to be processed in the order in which they arrive, and at the proper rate.
  - The "startup" case to be addressed: when ORS starts, many interactions could be queued, and ORS would start pulling and processing them.
  - Specific ordering and sequencing functionalities to be applied to interactions, as provided by Interaction Server's Queues and Views mechanisms.
- Queues from which ORS pulls interactions must be explicitly associated with the ORS. This association is provided by provisioning. That is, ORS looks for the queue with the specific parameters in the object's Annex tab, and pulls interactions from these queues only.
- No other Interaction Server clients of type "routing client" will process interactions from queues that are associated with ORS. Media Server(s), as part of open media may process interactions in these queues. The desktop client may also process interactions.
- To achieve load sharing, multiple ORS instances might pull and process interactions from the same queue.
- ORS utilizes Universal Routing Server (URS) to select the target for the eServices interaction when routing is necessary. ORS uses the connection with URS to inform URS that a new eServices interaction is going to be processed. ORS then calls functions (if specified in SCXML) and `queue:submit` action event to select the target. URS responds with the selected target, and ORS routes interactions to this target using `RequestDeliver`.
- It is acceptable for the SCXML application to redirect (or place) eServices interactions into another queue in the Interaction Server. In this case, processing of this interaction is continued when ORS pulls it again, this time from another queue. When it is pulled, ORS has information about which SCXML session is associated with this interaction, and ORS sends the corresponding event to this SCXML session. Note that the queue where the interaction is placed must be associated with ORS so that ORS knows to pull interactions from it. A queue is associated with ORS by creating the `Orchestration` section in the queue's `Annex` tab. The ORS only monitors associated queues.

Note: All ORS requests with their attributes, including `RequestPull` can be seen in the Interaction Server log.

- Persistence Storage is used to store SCXML sessions and documents, as well as scheduled activities (such as start and stop).

Basic eServices Interaction Routing

The following sequence diagram (Figure 6) illustrates a scenario for basic eServices interaction routing in Orchestration.

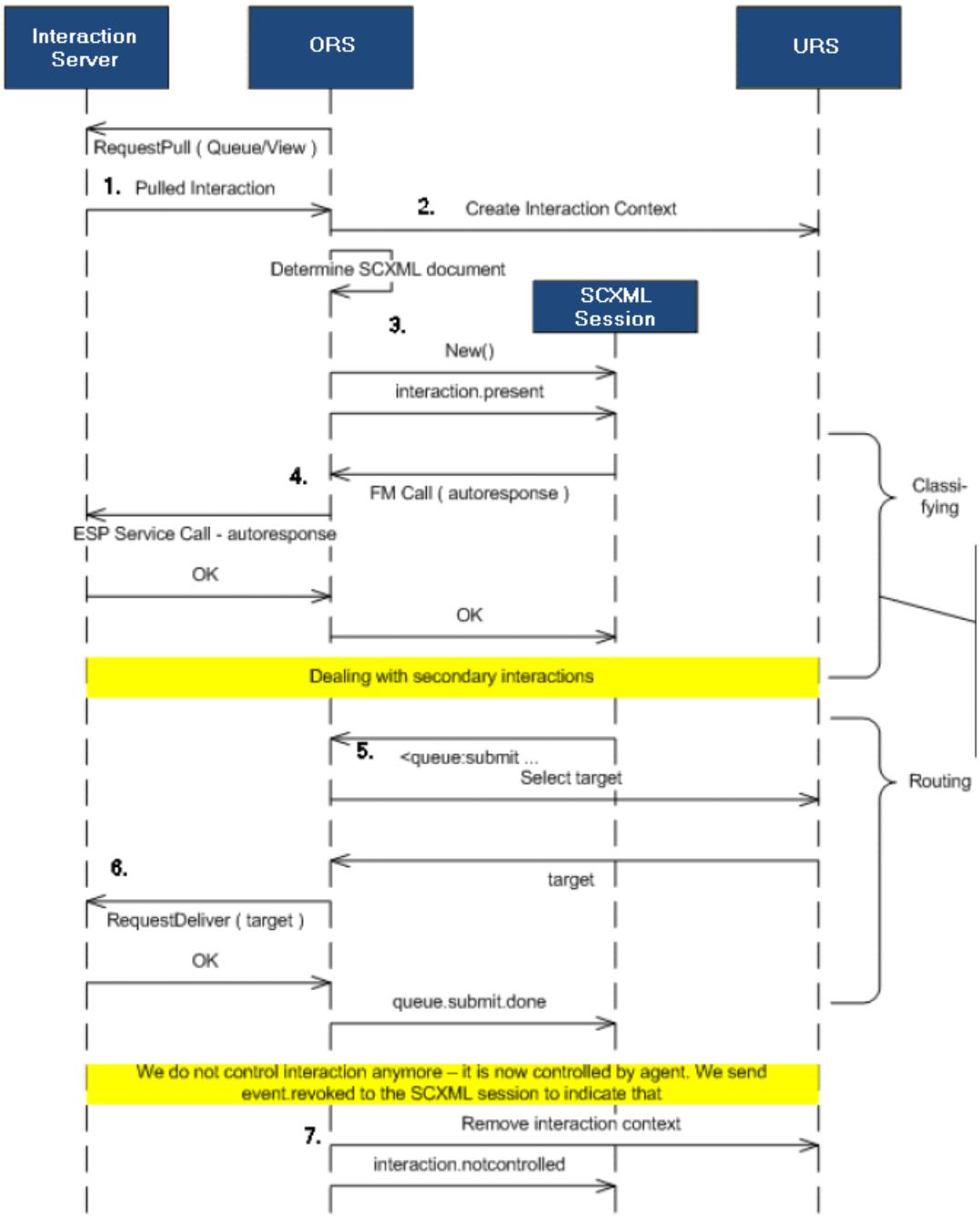

Figure 6: Basic eServices Interaction Flow Scenario

The numbered steps in the preceding figure are identified as follows:

Item 1. Orchestration Server pulls the next eServices interaction from a queue.

Item 2. When the interaction is successfully pulled:

- Application Server receives a strategy request.
- URS is notified about the new interaction (and prepares to begin processing).

Item 3. A new session starts, and an `interaction.present` event is fired into the session, which allows the interaction to be processed.

Item 4. The SCXML application submits a request to Interaction Server to auto respond to the interaction.

Item 5. After this, a `queue:submit` action is invoked which locates the appropriate resource to process the interaction.

Item 6. When an available resource is found, the interaction is delivered to this resource.

Item 7. When the interaction is redirected to the resource:

- `interaction.notcontrolled` is fired into the session.
- URS also is notified that the interaction is not controlled.

Enhanced Orchestration Multi-Site Support

Multi Site routing within Orchestration is concerned with the handling of a call across one or more T-Servers (Referred to as a site). In such deployments prior to Orchestration, IRD strategies would typically be provisioned against Routing Points across various T-Servers to support the desired behavior, allowing segmentation of routing logic to be controlled as the call is redirected from strategy to strategy or from an agent resource back to a Routing Point with an associated strategy within the various switches.

Within Orchestration, segmentation of the routing logic may be accomplished by combining SCXML documents and controlling the flow programmatically rather than requiring the movement of interaction to dictate what SCXML document needs to be executed. Orchestration is able to undertake this by associating an interaction with a controlling SCXML session. Such an association between an Interaction and an Orchestration Session is created when a call first enters the site and executes its first Orchestration SCXML Session. The association is maintained until one of the following is true:

- The SCXML session exits.

- The SCXML explicitly transfers the association to another session using the <associate> action.
- The Interaction is no longer valid due to the interaction being deleted from the system, in other words, the customer hangs up

While such an associated session is active, all interaction events would be directed to this owning or controlling session for handling, allowing for the creation of a single SCXML session that can control the complete interaction handling encompassing pre routing and selection of a resource as well as post routing once a resource has redirected it to another Routing Point. This is seen to provide valuable benefits such as streamlining the configuration required, and also allows for the creation of more obvious interaction and conversation processing logic because the interaction can be controlled during periods not currently supported by URS/IRD.

However, the ability to maintain this association can be seen to provide unexpected behavior for customers that wish to maintain the legacy way of breaking up their routing logic based on Routing Points. Typically, this may be seen as SCXML documents not executing new sessions because of a preexisting association with an existing session. It is for this reason to maintain equality with existing structures that Orchestration 8.1.2 has introduced additional SCXML actions that can be used to help control the association between an interaction and an Orchestration Session. This allows customers to recreate the legacy routing logic separated by provisioning rather than centralized SCXML control logic which results in the handling of the interaction across multiple sessions.

The following SCXML actions provide the application developer with increased control of the association.

<attach> - Allows a session to explicitly request an association with an interaction that is currently not associated with any session.

<detach> - Allows a session to explicitly inform Orchestration that it no longer requires an association with the indicated interaction.

These two actions are in addition to the existing action <associate> which allows a session to explicitly associate an interaction it owns to any other session. Details of such actions may be found within the *Orchestration Developer's Guide* on the Genesys Documentation Wiki.

When an interaction is currently not associated with any session, the determination of what SCXML document to execute will be purely based off existing configuration allowing for similar structures to be created as they are presently with URS/IRD strategies.

Example Use Case

The following is a use case to exemplify the behavior that <attach> and <detach> may provide. In most cases, to allow a new session to be created

<detach> should be called before the routing operation, and on failure of the routing operation <attach> should be called.

ORS Routes to ORS controlled Route Point

Figure 7 exemplifies an SCXML session (Session 1) that has been executed as a result of a call entering Routing Point 1 (RP1). Upon entry, Session 1 may determine to continue the logic and call handling that it needs to transfer the call to Routing Point 2 (RP2) on Site 2. To accomplish this, a <queue:submit> is performed with route equal to false. Upon success, the interaction is then detached from Session 1 and is then redirected to the destination returned back from the <queue:submit>, upon success, a new session Session 2 is started and Session 1 exits. On failure to route to RP2, Session 1 will perform an <attach> to reestablish the association with Session 1 and perform some other processing within the same session. The second session will route the call to a local agent.

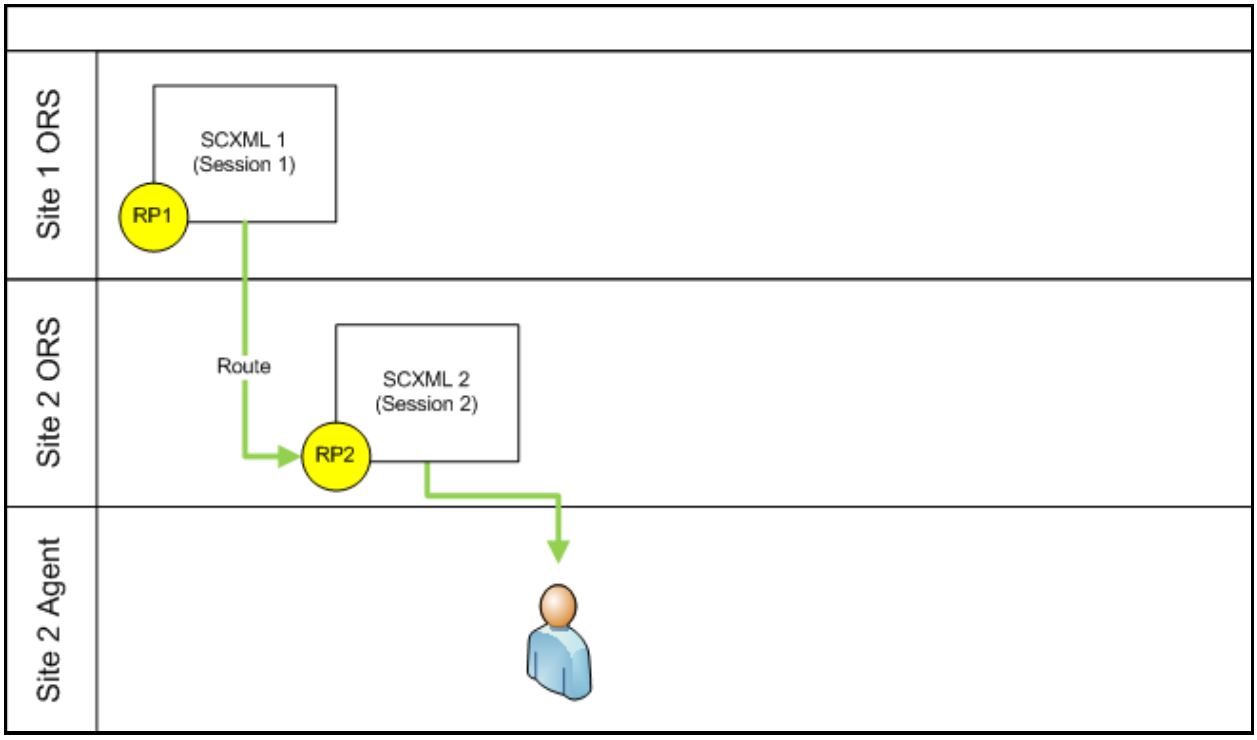

Figure 7: SCXML Session

The SCXML for Session 1 is as follows

```
<scxml version="1.0" xmlns="http://www.w3.org/2005/07/scxml"
                     xmlns:queue="www.genesyslab.com/modules/queue"
                     xmlns:ixn="http://www.genesyslab.com/modules/interaction"
                     xmlns:dialog="www.genesyslab.com/modules/dialog"
        initial="initial">
<datamodel>
```

```
                    <data ID="reqid" expr="''" />
                    <data ID="ixnid" expr="''" />
                    <data ID="resource" expr="''" />
</datamodel>
<!-- Wait for interaction to be added to session -->
<state id="initial">
<transition event="interaction.added" target="selectresource">
 <script>
                     _data.ixnid = _event.data.interactionid;
 </script>
</transition>
 </state>
<!—Select resource -->
<state id="selectresource">
   <onentry>
    <queue:submit priority="5" timeout="20" route=”false”>
     <queue:targets>
      <queue:targetid expr="'SITE2_RP2@.RP'"/>
     </queue:targets>
    </queue:submit>
   </onentry>
<!-- Submit to RP complete, exit session -->
   <transition event="queue.submit.done" target="detach">
 <script>
                     _data.resource = _event.data.resource;
 </script>
   </transition>
<!-- Submit to RP failed perform reattach -->
   <transition event="error.queue.submit" target="exit"/>
</state>
<!-- Perform the detach of the interaction -->
<state id="detach">
                    <onentry>
<ixn:detach interactionid="_data.ixnid"
                    requestid="_data.reqid" />
                    </onentry>
```

```
                    <!-- If Interaction is not in correct state to be detached,
reattempt -->
<transition event="error.interaction.detach" cond="(_event.data.error ==
'invalidstate') &amp;amp;&amp;amp; (_event.data.requestid == _data.reqid)">
                     <ixn:detach interactionid="_data.ixnid"
requestid="_data.reqid" />
                    </transition>
                    <!-- Detach completed route the call -->
                    <transition event="interaction.detach.done"
               cond="_event.data.requestid == _data.reqid" target="routing"/>
</state>
<!—Redirect to selected resource -->
<state id="routing">
<onentry>
                    <ixn:redirect interactionid="_data.ixnid" from="
_genesys.ixn.interactions[_data.ixnid].voice.dnis " to="_data.resource"/>
</onentry>
<transition event="interaction.redirect.done" target="exit"/>
<transition event="error.interaction.redirect" target="reattach"/>
</state>
<!-- Reattach -->
<state id="reattach">
   <onentry>
                    <ixn:attach requestid="_data.reqid"
interactionid="_data.ixid"/>
   </onentry>
<transition event="interaction.attach.done" target="localprocessing"/>
   <transition event="error.interaction.attach" target="exit"/>
</state>
<!--Perform local processing or recovery action here -->
<state id="localprocessing">
                    ..
                    ..
                    ..
</state>
<!—Exit Session -->
<final id="exit"/>
</scxml>
```

The second SCXML session will use a similar structure to the above to handle the routing of the call to a local agent and for this reason is not shown.

For more information and detailed SCXML samples please refer to the *Orchestration Developer's Guide* on the Genesys Documentation Wiki.

Implementation Notes

The following should be taken into account when working with these actions to support multiple site and multi-session controlled routing.

<detach> may result in error.interaction.detach

To be able to detach an interaction, and remove the associate between the Orchestration session and the interaction, the interaction must be confirmed to be owned by the Orchestration session for the `<detach>` action to succeed. This is in part controlled by the addition of User Data that is used to help track such an association as well as internal state within the Orchestration Server. When an interaction is associated to a session, either directly through the creation of a session from a Routing Point, or indirectly by another session calling `<associate>`, there is a small window of time required to allow this information to be successfully propagated by the user data update. Should `<detach>` be called prior to this update succeeding, the `<detach>` call will return the `error.interaction.detach` event.

To help safe guard against this, the user should ensure that they call `<detach>` just prior to the session undertaking its routing of the interaction. The user should also ensure that they correctly handle the `error.interaction.detach` within a transition, and should ensure that they only undertake their routing operation once it has been confirmed that the interaction has been detached from the session. This can be observed by only commencing the routing operation once the `interaction.detach.done` event has been received.

By observing this implementation pattern, it will allow the user to correctly build SCXML documents that operate in a multi-session manner and across multiple sites.

Handling Routing errors

After calling `<detach>` and having confirmation that the interaction is detached, the user should then immediately route the interaction. If the routing fails, to ensure that subsequent interaction related events are provided back to the SCXML session, the interaction should be re-associated with the session if it is desired to provide subsequent processing within the SCXML document. This can be achieved by calling the `<attach>` action. To undertake this, it is expected that the user should store the Interaction ID until the SCXML document has completely handled the detach, routing and or error handling and subsequent routing. By detaching an interaction, the session looses the ability to obtain interaction related events that are not related to interaction related actions, therefore the developer should be aware that to ensure the session has all

related events for the interaction, it must be associated with the interaction. It is therefore recommended that <detach> is not called too far in advance of the action used to route the call.

Returning Call back to Original Site

During routing using separate SCXML sessions, a call may be transferred to multiple sites during its treatment, this may result in multiple related interactions being associated with the SCXML session that represent the call across different T-Servers due to the manner that the T-Server represents calls during ISCC transfers. In scenarios where a call is transferred back to its original site, it is recommended that such SCXML documents not perform long post call routing processing following a <detach> of their main interaction and routing operation. Failure to observe this may result in a new SCXML session failing to start once the call reenters the original site due to the time required to propagate the detach across the related interactions. Should post call routing however be desired in such a case, there are two recommendations for how a user may undertake this from the transferring site.

First, the user may want to perform a <detach> for all interactions currently being handled by the transferring SCXML session. By ensuring this, it will ensure the user data updates is correctly propagated across related calls.

Secondly, the user may wish to perform any post routing operations in a separate session initiated through a <session:start> action. This is the preferred suggested manner to perform post routing operations in a multi-site, multi-session routing environment.

By undertaking either of these, it ensures that no lingering session association is present which would block the creation of a new session on the original site.

<detach> and <queue:submit>

When using <queue:submit>, and there is a need to have multiple sessions running, it is not recommended to call <detach> prior to <queue:submit>. For scenarios that are using <queue:submit>, it is recommended that the resource is targeted first with the route attribute of <queue:submit> set to be false. This will allow the resource selection events (queue.submit.done) to be returned back to the current session. Once a resource has been located successfully you may then proceed to <detach> the interaction from the session and <redirect> the interaction to the selected resource as indicated in the queue.submit.done event.

Chapter

3 Persistence, High Availability, and Load Balancing

Persistence, as the term is used in Orchestration Server (ORS), provides the capability to store and retrieve information relating to potentially millions of SCXML sessions and SCXML documents, including SCXML session *data* and SCXML session *state*, such that when required, sessions or documents that were previously persisted may be fully recovered. Persistence is scalable, and facilitates high availability (HA) (through "clusters") and load balancing.

This chapter contains the following topics:

- Persistence, page 45
- High Availability and Load Balancing, page 52

Persistence

The significant service of persistence is to store SCXML sessions and documents, as well as scheduled activities (such as start and stop).

Persistence Design Features

The following are the key design principles with regards to persistence in the Orchestration 8.1 implementation.

- The persistence layer allows storage and retrieval of data related to active SCXML sessions. The amount of data stored is sufficient to restore the SCXML session fully and continue its execution.
- The persistence layer allows storage and retrieval of additional information needed for Orchestration Server to operate. This includes:
  - A list of actions that are scheduled to be executed at specific time.

    - Information about which SCXML session is currently handled by which Orchestration Server instance
- All Orchestration Server instances within a single cluster *must* be configured to work with the same persistent storage. This allows each Orchestration instance to have access to the complete information about all SCXML sessions, schedules, and which Orchestration Server is currently handling the session.
- For release 8.1.x, persistent storage is configured under the following method:
    - **Apache Cassandra (NoSQL Datastore).** This class of product provides decentralized, fault tolerant, elastic, durable, and highly scalable distributed database systems on commodity hardware. This leads to a simpler deployment/installation and alleviates concerns over the scalability and performance of the back-end datastore. ORS supports Cassandra 0.7x, and later versions. (ORS 8.1.1 and later releases do not support Cassandra 0.6x.) Orchestration Server 8.1 and later releases require persistence storage, and the only supported method is Apache Cassandra.
- If the persistent storage connection is lost, Orchestration Server will exit.
- SCXML sessions are persisted when the application asks for it *and* the last queued event is flushed.
- The most important function of persistence is to support retrieval of SCXML session information to restore the session context, as required. Session context is restored, for example, upon a restart of the Orchestration Server component or the arrival of an event for an SCXML session, which was previously deactivated to reduce the memory required, or respond to an extended idle state for that session.

Persistent Storage Operation

The datastore type is determined by configuration.

The 8.1 release of Orchestration Server supports the following datastore implementation:

- *Apache Cassandra*, a NoSQL distributed database management system

This class of product provides decentralized, fault tolerant, elastic, durable, and highly scalable distributed database systems on commodity hardware. This leads to a simpler deployment/installation and alleviates concerns over the scalability and performance of the back-end datastore.

Document Persistence

SCXML documents are the basis for session construction, and as such many sessions may refer to the same document when persisted. The SCXML engine:

1. Retrieves the requested document from the specified location. For example, in the `Annex` tab of a RoutePoint configuration there would be an Orchestration section with the `application` specified, such as: `http://test52/Queues_1.xml`.
2. Requests the persistence component to store the document information in the persistent storage once the document is verified and compiled.

The compiled document is serialized and a request to store this document is made to the specified Cassandra cluster. This is described in "Configuring Persistent Storage" on page 49, and configuration options are described in Chapter 5, "Configuring Orchestration Server," beginning on page 63.

For the Cassandra persistence type, the serialized document is stored in Cassandra within the `Orchestration` Keyspace, in the `Document` ColumnFamily, and with `DocumentContent` ColumnName, associated with the appropriate Document ID as the key. When the Cassandra operation is complete, no further action is required.

Session Persistence

SCXML sessions are the current SCXML representation of an active session. These are constructed by using the previously mentioned documents. The SCXML engine requests the persistence component to store the session information in the persistent storage.

The compiled session is serialized and a request to store this session is made to the specified Cassandra cluster. This is described in "Configuring Persistent Storage" on page 49, and configuration options are described in Chapter 5, "Configuring Orchestration Server," beginning on page 63.

For the Cassandra persistence type, the serialized session is stored in Cassandra within the `Orchestration` Keyspace, in the `Session` ColumnFamily, and with `SessionContent` ColumnName, associated with the sessionID as the key. When the Cassandra operation is complete, no further action is required.

Note: Sessions are present in persistent storage only if they are active. This means that the sessions as defined in the SCXML document have not reached the `final` state. If this state is reached, the session information is removed from persistent storage.

Session-to-ORS Persistence

For High Availability of a session to be effective, the following information must be maintained:

- The ORS node that is currently responsible for processing the session.
- The sessionID of that session.

The Cassandra data model accomplishes this “persistence.”

When the session is created, the Orchestration Server on which the session is created persists the `sessionID` and its Orchestration `Node ID`, which is the `dbid` of the Orchestration application. If, during the life of the session, another Orchestration node assumes processing of this session, the persistence information is updated by the new node. When the session reaches the state `final`, the entry for the session is removed from persistence. The session-to-server node information is placed in the ColumnFamily `SessionIDServerInfo`, with ColumnName `SessionServerInfo` with keys of the `sessionID`. To facilitate the retrieval of the sessions for a given server node, the ColumnFamily `SessionIDServerInfoRIndex` is employed, with keys that are the string form of the `Node ID` and Columns that are the `sessionids` for that node.

Persistence Scheduling

SCXML sessions can be started in the future, or hung sessions can be terminated, through the *Schedule* component. When Orchestration Server receives a request to schedule a session, if the requested time has passed (defined as the requested time plus the latency in processing the request, [currently five seconds is allowed]) the session action is processed immediately.

For the currently-supported Cassandra persistence type, the scheduled session information is serialized and stored in Cassandra within the `Orchestration` Keyspace, in the `Schedule` SuperColumn. The keys for the `Schedule` SuperColumn are the scheduled `sessionID`. The columns within the SuperColumn are start time for the scheduled action and the column entries are a unique global identifier generated for the action with the action type `concatenated`. The column values are the serialized schedule content. To facilitate the schedule retrieval based on time interval, the SuperColumn `ScheduleRIndex` is employed. The keys for this SuperColumn are the start time for the action, in milliseconds since the epoch, divided by the retrieval interval, currently 60 seconds. The `ScheduleRIndex` Columns are the column entries in the `Schedule` SuperColumn, and the values are the start time for the scheduled action.

Deploying Persistent Storage

In Figure 8, multiple instances of Orchestration Server are running and processing sessions and schedules at the same time. Refer to “High Availability and Load Balancing” on page 52 for a description of multiple ORS

instances configured as clusters. All Orchestration Server instances are active, and all work with persistent storage.

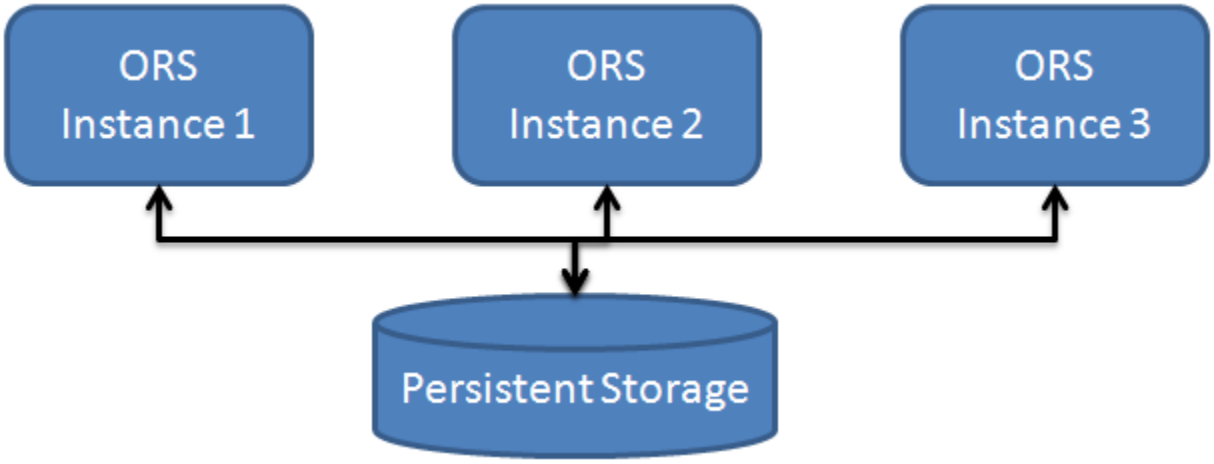

Figure 8: Multiple Orchestration Server Instances Running

As shown in Figure 9 on page 49, if any of the instances fail, the remaining instances will continue processing the sessions or schedules that were previously handled by the failed node. The remaining instances will restore information about these sessions or schedules from the persistent storage.

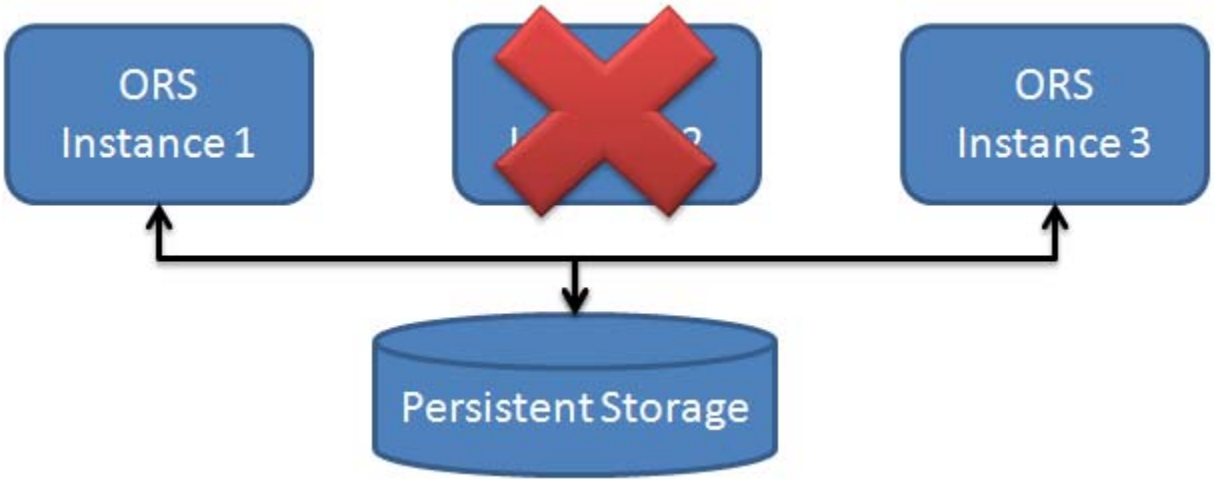

Figure 9: Remaining Orchestration Server Instances Work With Persistent Storage

Configuring Persistent Storage

Orchestration Server 8.1 and later releases require persistence storage, and the only supported method is Cassandra NoSQL.

In addition, there are other configuration options that you must add and set to configure the Cassandra persistence type. These are the general steps:

1. In the `cassandra-listenport` ORS configuration option, retain the default value (`9160`), or modify the value for the Cassandra client port, as required.

2. Add the `cassandra-nodes` ORS configuration option and enter values for the host names (or IP addresses) that will constitute the individual nodes in the Cassandra cluster.

Use one of the following detailed procedures, which describe how to configure persistent storage in ORS by using Configuration Manager or Genesys Administrator.

Procedure: Configuring the Cassandra Persistence Type in ORS Using Configuration Manager

Purpose: To add and configure the Orchestration Server options to enable it to use Cassandra persistent storage.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object.
4. Select the `Options` tab.
5. Select the `persistence` section.
6. Retain the default value (`9160`) or modify the `value` field of the `cassandra-listenport` configuration option, as required.

   This value represents the Cassandra client port.
7. Add the `cassandra-nodes` configuration option and enter values for the host names (or IP addresses) that constitute the individual nodes in the Cassandra cluster.

   For example:

   `DWS;mpswin;test47`

   See also, "High Availability and Load Balancing" on page 52.
8. Click `OK` to save.

End of procedure

Next Steps

- No further steps are required.

Procedure: Configuring the Cassandra Persistence Type in ORS Using Genesys Administrator

Purpose: To add and configure the Orchestration Server options to enable it to use Cassandra persistent storage.

Start of procedure

1. Log on to Genesys Administrator and select `Provisioning > Environment > Applications`.
2. Double-click the ORS `Application` that you want to configure.
3. On the `Options` tab, scroll to the `persistence` section.
4. In the `Value` field of the `cassandra-listenport` option, retain the default value (`9160`) or enter a different port number, as required.

   This value represents the Cassandra client port.

   Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
5. Click `OK` to save.
6. Right-click inside the persistence options window again and select `New` from the shortcut menu.
7. In the resulting `New Option` dialog box, in the `Option Name` field, type `cassandra-nodes`.
8. In the `Option Value` field, type a semi-colon-separated list of hostnames and/or IP addresses to identify the nodes in the Cassandra cluster

   For example:

   `DWS;mpswin;test47`

   Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
9. Click `OK` to save.

End of procedure

Enabling Persistence in the SCXML Application

Enabling persistence for an SCXML application is controlled with the `<scxml>` tag's attribute `_persist`, which has values of `true` and `false`.

Starting with release 8.1.2, the default value for this attribute is `false`, therefore persistence for an SCXML application is disabled by default. A value of true activates persistence for an SCXML application.

Also see "persistence-default" on page 120.

When persistence for an SCXML application is enabled, ORS will regularly try to store the current session state into persistent storage (into the Cassandra cluster).

ORS selects the specific moments in time when it performs persistence steps. You can *force* ORS to persist the SCXML application state from within the SCXML application itself.

This is achieved using the `<state>` tag's attribute `_persist`, which has values of `no`, `may`, or `must`, with the default value `may`. If set to `must` the SCXML session state will be persisted upon entering into the defined state. The value `no` is not currently used.

High Availability and Load Balancing

The Orchestration solution allows for extended sessions (on the order of several months), is scalable, and is highly fault tolerant. The solution achieves this by supporting the ability to run multiple instances of Orchestration Server in a single, logical entity called a *cluster*.

Clusters

In this context, a cluster refers to running instances of Orchestration Server (ORS) or *nodes*, where each additional ORS instance is able to share in the handling of the workload as well as resume the tasks of a failed or removed node.

Cluster Deployment

Each node has similar configurations: connections to T-Servers, Interaction Servers, persistent storage, and Universal Routing Server (URS).

An ORS cluster distributes the load among its *member nodes*. That is, SCXML sessions are distributed and processed across all nodes in the cluster. The more nodes, the more SCXML sessions can be processed simultaneously.

A cluster must be configured to specify at least one *Super Node*. The Super Node maintains connections to all member nodes and helps facilitate workload distribution and failover handling. Each node in the cluster is connected to at least one Super Node (called the *Master Super Node*). This Super Node can manage and notify member nodes as needed when a node fails. At system startup, all assigned Super Nodes work together to select a Master Super Node. All of the other nodes (member nodes) connect to all Super Nodes.

Figure 10 shows an example of a cluster deployment as previously described.

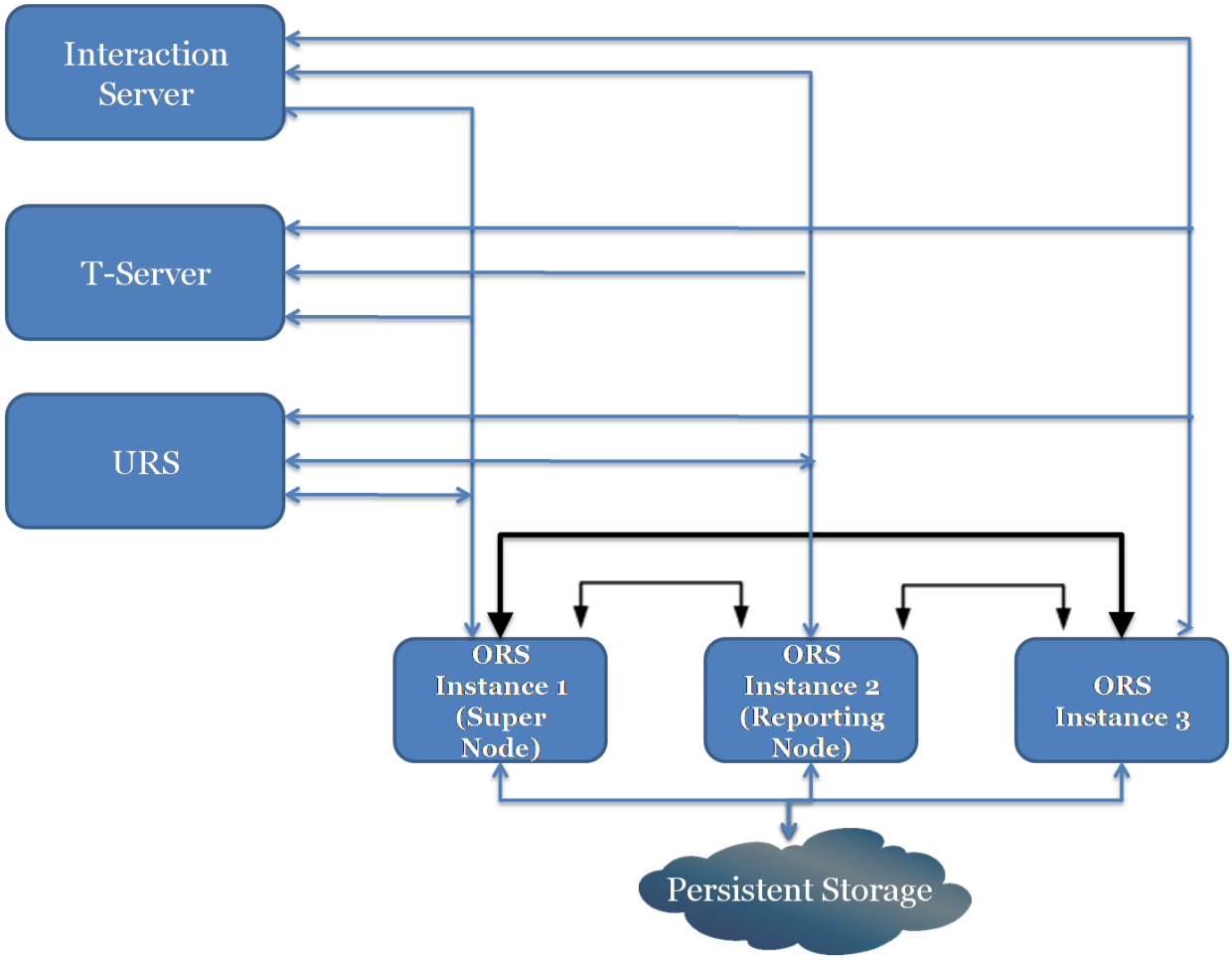

Figure 10: Typical ORS Cluster Deployment

The figure illustrates a typical cluster deployment, in which:

- There are several instances of ORS.
- Each ORS instance is connected to the same T-Server, Interaction Server, and URS.
- Each ORS instance is connected to all Super Node ORS instances.
- Each ORS instance is connected to the same persistent storage.
- At least one ORS instance has been designated as a Super Node.
- One ORS instance has been configured as a Master Super node (refer to the section "Operational Reporting" on page 133 in Chapter 5).

To configure clustering in ORS, refer to the section "Configuring ORS Clustering" on page 85 in Chapter 5.

Failure Recovery

The cluster provides the capability to recover from failure. When a node in the cluster fails or is removed, other nodes pick up processing of the SCXML sessions that were handled by the failed node. The information about such SCXML sessions is taken from the persistent storage, which must be the same for the all nodes in the cluster.

Load Balancing Method

There is no Load Distribution System (LDS) assumed for Orchestration Server deployment. Load is shared between ORS instances and managed by the cluster.

Load Balancing Operation

There are three main interfaces on which an ORS cluster needs to distribute load:

- Voice Interactions interface. This is the connection to the T-Server(s).
- eServices Interactions interface. This is the connection to the Interaction Server(s).
- RESTful interface. This is the HTTP port opened on all ORS cluster nodes.

Load Balancing for Voice Interactions

Orchestration Server receives events about voice interactions via the connection to T-Server(s). Each ORS node must have a connection to the same T-Server(s), so that all nodes are aware of all voice calls.

Load Balancing for eServices Interactions

Load distribution for eServices interactions is based on the principle that ORS is *pulling* interactions for processing.

Each ORS instance pulls interactions from the same set of queues/views. Therefore each ORS node receives a subset of the interactions that are currently in these queues. Each ORS node processes these interactions normally, start the SCXML session, and send events about these interactions to the SCXML sessions.

Load Balancing for RESTful Interface

ORS exposes the RESTful web services interface. This interface is based on HTTP. ORS may accept and execute a set of HTTP requests.

This interface is based on standard HTTP and uses standard mechanisms of load balancing that are typical for web servers. These mechanisms include:

- DNS-based load balancing
- Hardware-based load balancing.

These methods result in distribution of HTTP requests across all ORS nodes in a cluster.

To enable *optimal* load-balancing, ORS supports "stickiness" of HTTP sessions. This is achieved by providing a cookie with the SCXML session ID in the HTTP responses. Hardware load balancers (such as F5) may then use this cookie to ensure that HTTP requests are about the same SCXML session.

There could be a scenario when an HTTP request about a specific SCXML session is delivered to an ORS node that is not handling this SCXML session. In this case, the ORS node replies with the HTTP response "`301 Moved Permanently`" pointing to the ORS node that is processing the needed SCXML session. The responsibility of the HTTP Client is to resend the same request to the given URI, which results in the request arriving at the needed node.

For RESTful interface, be sure to set up the client-side port definition during installation, as described in Procedure: Installing Orchestration Server on Windows using the Installation Wizard, on page 147.

SCXML Session Startup Rules

When an existing SCXML session (Session1) starts another SCXML session (Session2), the *child* SCXML session (Session2) is always started on the same ORS node as the *parent* SCXML session (Session1).

When an existing SCXML session (Session1) invokes another SCXML session (Session2), the invoked SCXML session (Session2) is always started on the same ORS node as the existing SCXML session (Session1).

Chapter

4 Deployment Process

This chapter provides an overview of the planning and deployment activities required when you set up your Orchestration Server Application.

Before proceeding with the deployment of Orchestration Server, be sure that Framework has already been configured and installed. If it has not, refer to the *Framework 8.1 Deployment Guide.*

To facilitate configuration, it is important to configure components in the following order:

1. Framework components
2. Universal Routing components
3. Orchestration Server components

The deployment process involves the configuration and installation of the applications needed for a functional setup of Orchestration Server.

This chapter includes the following topics:

- Before You Begin, page 57
- How to Configure, page 58
- Packaging, page 58
- Orchestration Component Installation Order, page 59

Before You Begin

Before deploying Orchestration Server, investigate the sizing, security, availability, and performance required for the specific environment of your deployment.

How to Configure

Configuration of Orchestration Server is covered in Chapter 5, "Configuring Orchestration Server," on page 63.

You may use Genesys Administrator or Genesys Configuration Manager to carry out Orchestration Server configuration tasks.

Packaging

The Orchestration Server (ORS) component is delivered on the Universal Routing Product CD.

Management Framework CD

Before configuring and installing Orchestration Server, you must install the Framework components. The Framework components include such applications as Management Framework Configuration Wizard, Configuration Server, Configuration Manager, Genesys Administrator, License Manager, Management Layer (Local Control Agent, Solution Control Interface, Message Server, and Solution Control Server). Orchestration Server relies on Framework components to function as a solution.

Media Configuration Wizard for T-Servers is located on the Media CD.

Note: Network T-Server is considered a Network media interface, not a Framework component and is located on a different Network Media Interface CD (see "Orchestration Component Installation Order" on page 59).

For information on installation and configuration of Framework components, see the *Framework 8.1 Deployment Guide* and the T-Server document specific to your T-Server, or SIP Server document.

Chapter 2, "Orchestration Server Architecture and Interaction Flows," on page 31 lists the Framework components, Universal Routing components, and Orchestration Server components.

eServices CDs

The Genesys E-Mail CD contains Genesys E-Mail 8.0, a separately packaged media channel for the CIM Platform. Genesys E-Mail is a highly flexible and unified e-mail management solution with extensive real-time and historical reporting capability.

The Interaction Management CD contains the components for Genesys Interaction Management, the core functionality of eServices on the CIM Platform.

The Knowledge Management CD contains Genesys Knowledge Management. Knowledge Manager, the user interface component of Genesys Knowledge Management, is used to administer content across the CIM Platform for routing decisions and by the CIM Platform knowledge base for self and assisted services.

The Genesys Chat CD contains Genesys Chat 8.0, a separately-packaged media channel for the CIM Platform. Genesys Chat must be deployed in conjunction with the Genesys Interaction Management Platform, which delivers the core capabilities of eServices.

The Genesys Web Collaboration CD contains Genesys Web Collaboration 8.0, a separately-packaged option for the CIM Platform. Web Collaboration enables agents and customers to view and navigate web pages together. This enables agents to provide superior customer service by assisting customers in using capabilities of web sites such as making purchases, completing forms, finding information, and so on.

The Genesys SMS CD contains Genesys SMS Server 8.0, a separately-packaged media channel for the CIM Platform. Genesys SMS Server is a highly flexible and unified SMS management solution.

Real-Time Metrics Engine CD

Stat Server has its own installation CD (Real-Time Metrics Engine), which contains both the Stat Server Wizard and the Resource Capacity Wizard used for configuring agent capacity rules.

Genesys Info Mart CD

Genesys Info Mart produces a data mart containing several star schemas you can use for contact center historical reporting. This includes detailed reporting on Genesys e-mail, chat, and virtual queue interactions, and as well as support for reporting on interactions involved in basic Network Routing call flows.

Orchestration Component Installation Order

To facilitate configuration when running with eServices, it is important to configure component groups in the following order:

1. Framework components (including Stat Server Wizard, Resource Capacity Wizard, and Genesys Administrator required for ORS Operational Reporting and provisioning)

2. Routing Components (URS)
3. eServices components (if the solution requires)

> **Note:** The eServices Configuration Wizard assumes that Universal Routing configuration has been completed.

4. Genesys Voice Platform (GVP) (if the solution requires)
5. Orchestration Server component (ORS)
6. Genesys Agent Desktop (optional)
7. Orchestration Server 8.1 and later releases require persistence storage, and the only supported method is Apache Cassandra.

Order for Individual Components

The individual components that are used to deploy Orchestration Server must be installed in the following order:

1. DB Server (if not already installed with other Framework components).
2. Configuration Server (if not already installed with other Framework components).
3. Configuration Manager or Genesys Administrator (if not already installed with other Framework components).
4. T-Server or Network T-Server (use the Network T-Server CD to install Network T-Servers). This is used only for voice interaction processing.
5. Stat Server (if not already installed with other Framework components).

> **Note:** Stat Server has its own Wizard, which is not part of the Common Wizard Set used by Genesys Wizard Manager. If planning to route based on the agent capacity rules, you must also install the Resource Capacity Wizard component. Both the Stat Server Wizard and the Resource Capacity Wizard are located on the Real-Time Metrics Engine CD.

6. Message Server(s) (if not already installed with other Management Layer components)
7. Universal Routing Server
8. After installation of the above components, install eServices as described in the *eServices 8.0 Deployment Guide*.
9. The eServices Configuration Wizard gives the option of using the Resource Capacity Wizard for setting up agent-capacity rules. For more information on this wizard, see the *Genesys 8.1 Resource Capacity*

Planning Guide. For summary information on agent-capacity rules and how they can affect routing, see “Configuring the Template Optons in the orchestration Section” on page 76.

10. Orchestration Server

What Each Component Does

The following is a summary of what each component does.

- T-Server: Generates events and receives requests.
- URS: Provides routing service (Queue Functional module) for Orchestration.
- Stat Server: Indicates agent availability.
- Configuration Server: Provides contact center objects.
- Message Server: Used with Orchestration Server for logging only.
- DB Server: Enables access to the Genesys configuration database.
- PBX: Routes a call.
- eServices components: See the *eServices 8.0 Deployment Guide*.

Note: Your choice of an Application Server also must be configured and provisioned for this installation.

Chapter

5 Configuring Orchestration Server

Before you can install the Orchestration Server component as described in Chapter 8 on page 145, you must configure it.

This chapter describes the configuration process for the Orchestration Server and includes the following topics:

- How to Configure, page 63
- Importing the ORS Application Template and Creating the ORS Application Object, page 68
- Configuring Options in the Orchestration Server Application Object, page 75
- Configuring Other Options That Affect Orchestration Server, page 89
- Other Manual Configuration Operations, page 101
- Orchestration Server Configuration Options, page 102
- Operational Reporting, page 133

See the *Universal Routing 8.1 Deployment Guide* for information on how to configure Universal Routing Server (URS), Custom Server, and Interaction Routing Designer (IRD).

How to Configure

This document describes how to manually install Orchestration Server.

This procedure involves:

- Creating an `Application` object in Configuration Manager for the Orchestration Server component.

Note: Objects can also be created and configured in Genesys Administrator. Refer to the *Framework 8.1 Genesys Administrator Help* for information.

- Giving the objects the proper settings for options and other attributes.

You can configure Orchestration Server entirely in Configuration Manager within the specific Genesys applications. This involves importing the `Application Template`, creating the `Application` object, setting up properties of the `Application` object in a dialog box, and adding properties to servers to which the application connects. The following is a summary of this process.

Task Summary: Configuring With Configuration Manager

Objective	Related Procedures and Actions
Log in to Configuration Manager.	Procedure: Logging into Configuration Manager, on page 68
Import the appropriate `Application Template` for Orchestration Server.	Procedure: Importing the application template for Orchestration Server, on page 71
Create the Orchestration Server `Application` object.	Procedure: Creating/Configuring the Orchestration Server Application object, on page 72
Create a connection for the Orchestration Server `Application` object to the following servers: • T-server(s) • Interaction Server • Universal Routing Server (to enable resource allocation functionality).	Use the Connections tab of the `Application` object in Configuration Manager to set up connections to other servers. Step 12 on page 74 of Procedure: Creating/Configuring the Orchestration Server Application object
Configure the template options in the `orchestration` section: • `mcr-pull-interval`	Procedure: Viewing/changing template options in the orchestration section, on page 76
Add and set non-template options in the `orchestration` section: • `session-hung-timeout`	Procedure: Adding and setting non-template options in the orchestration section, on page 77
Configure the template option in the `persistence` section: • `cassandra-listenport`	Procedure: Viewing/changing template options in the persistence section, on page 78

Task Summary: Configuring With Configuration Manager

Objective	Related Procedures and Actions
Add and set non-template options in the `persistence` section: • `cassandra-nodes`	Procedure: Adding and setting non-template options in the orchestration section, on page 77
Configure the template options in the `scxml` section: • `fips_enable` • `http-client-side-port-range` • `http-enable-continue-header` • `http-max-age-local-file` • `http-max-cache-entry-count` • `http-max-cache-entry-size` • `http-max-cache-size` • `http-max-redirections` • `http-ssl-cert-type` • `http-ssl-key-type` • `http-ssl-verify-host` • `http-ssl-verify-peer` • `http-ssl-version` • `max-includes` • `max-compiler-cache-size` • `max-compiler-cached-docs` • `max-compiler-cached-doc-size` • `max-preprocessor-cache-size` • `max-preprocessor-cached-docs` • `max-preprocessor-cached-doc-size` • `persistence-default` • `session-processing-threads`	Procedure: Viewing/changing template options in the scxml section, on page 80

Task Summary: Configuring With Configuration Manager

Objective	Related Procedures and Actions
Add and set non-template options in the `scxml` section: • `http-no-cache-urls` • `http-proxy` • `http-ssl-ca-info` • `http-ssl-ca-path` • `http-ssl-cert` • `http-ssl-cipher-list` • `http-ssl-key` • `http-ssl-random-file` • `https-proxy` • `password` • `persistence-max-active` • `system-id`	Procedure: Adding and setting non-template options in the scxml section, on page 81
Add and set non-template options in the `log` section: • `x-server-trace-level` • `x-server-gcti-trace-level` • `x-server-config-trace-level` • `x-print-attached-data`	Procedure: Adding and setting non-template options in the log section, on page 82
(For memory optimization): Add an `mcr` section to the ORS Application object, then add and set non-template options in the `mcr` section. • `om-memory-optimization` • `om-max-in-memory` • `om-delete-from-memory`	Procedure: Adding an mcr section, then adding and setting memory optimization options for an ORS instance, on page 84
(For clustering functionality): Add a `cluster` section to the ORS Application object, then add and set non-template options in the `cluster` section. • `name` • `super_node`	Procedure: Adding a cluster section, then adding and setting clustering options for an ORS node, on page 86

Task Summary: Configuring With Configuration Manager

Objective	Related Procedures and Actions
(For web services functionality): Add a `web_services` section to the ORS Application object, then add and set non-template options in the `web_services` section. • `hostname` • `port`	Procedure: Adding a web_services section, then adding and setting web services options, on page 87
Configure the `application` option in the `Orchestration` section of a `DN` (`Extension` or `RoutePoint`) or `Interaction Queue`.	Procedure: Adding and setting the application option in the Orchestration section of a DN or Queue object, on page 90
Configure options specified in Script objects of type `CfgEnhancedRouting` (Enhanced Routing Script): • `alternate-url` • `fetch-timeout` • `http-useragent` • `http-version` • `max-age` • `max-duration` • `max-loop-count` • `max-stale` • `url` • `{Parameter Name}` (in `ApplicationParms` section)	Procedure: Adding and setting options specified in the Application section of CfgEnhancedRouting Script objects, on page 92 Procedure: Adding and setting the {Parameter Name} option in the ApplicationParms section of CfgEnhancedRouting Script objects, on page 93
Configure Super nodes for use in Operational Reporting.	Procedure: Configuring a Super node for Operational Reporting Using Genesys Administrator, on page 134 or Procedure: Configuring a Super node for Operational Reporting Using Configuration Manager, on page 135

Importing the ORS Application Template and Creating the ORS Application Object

This section describes the following procedures:

- "Logging into Configuration Manager"
- "Importing the application template for Orchestration Server"
- "Creating/Configuring the Orchestration Server Application object"

Procedure: Logging into Configuration Manager

Purpose: To start the Configuration Manager tool, which allows you to create the Orchestration Server `Application` object associated with the Orchestration Server and to configure Orchestration Server options.

Note: Objects can also be created and configured in Genesys Administrator. Refer to the *Framework 8.1 Genesys Administrator Help* for information.

Start of procedure

1. Open Configuration Manager from the `Start` menu on your PC. The default path is `Start > Genesys Solutions > Framework > Configuration Manager > Start Configuration Manager`. The `Configuration Manager` login dialog box opens with the last entries. Figure 11 shows an example.

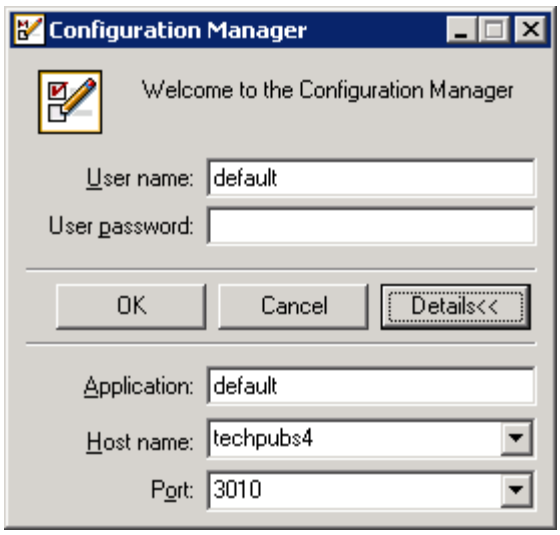

Figure 11: Configuration Manager Login Dialog Box

2. Use the information in Table 4 to complete the login dialog box.

Table 4: Configuration Manager Login Dialog Box

Field	Description
User name:	Name of `Person` object defined in Configuration Manager.
User password:	Password of `Person` object defined in Configuration Manager.
Application:	Enter the name of the Configuration Manager `Application` object or `default`.
Host name:	Name of the computer on which Configuration Server is running.
Port:	Port number used by Configuration Server.

3. To open Configuration Manager, click `OK` in the login dialog box.

   Figure 12 on page 70 shows a sample Configuration Manager for a Multi-Tenant environment.

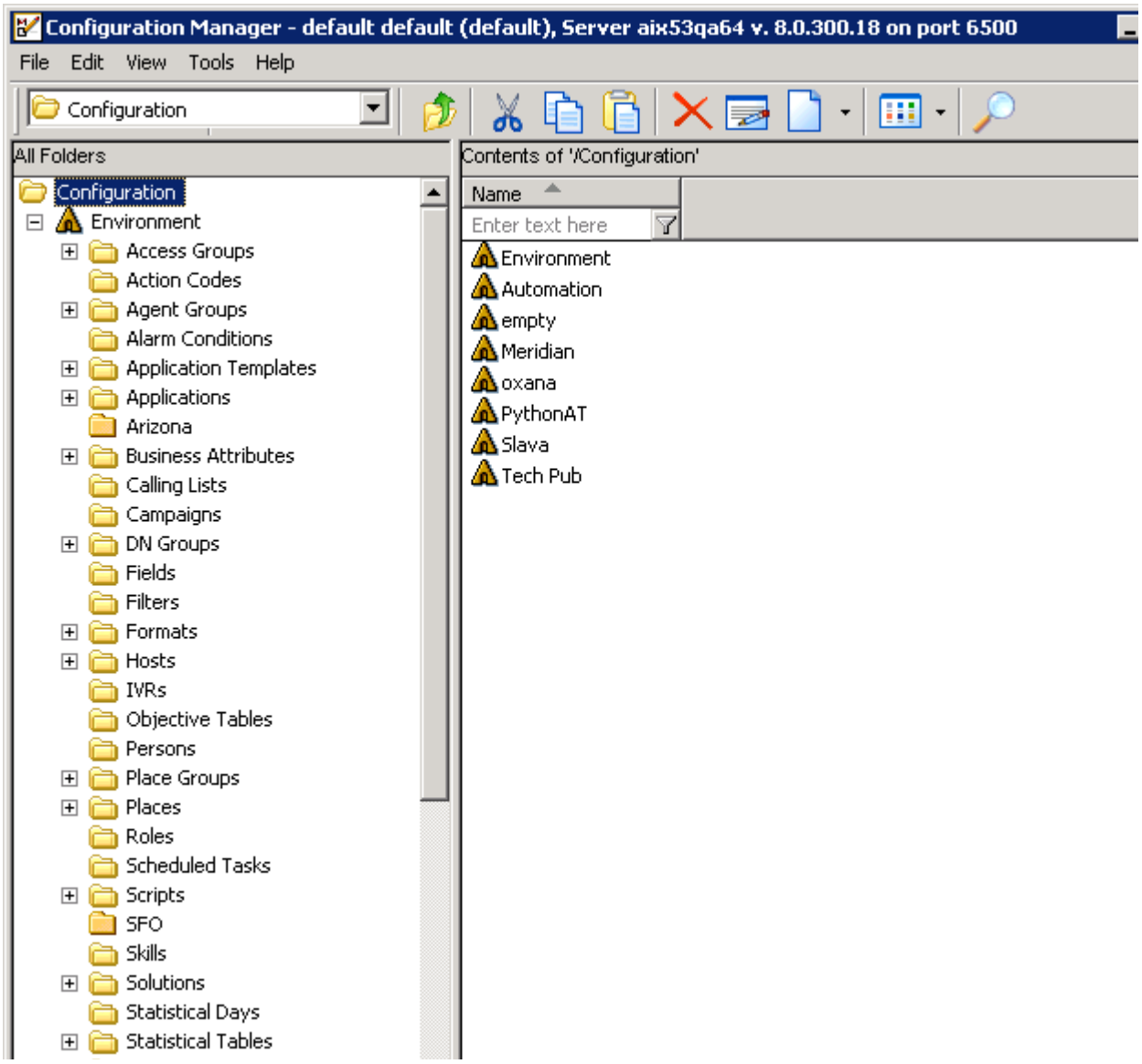

Figure 12: Configuration Manager

Note: If you wish to use the Management Layer and its Solution Control Interface (SCI as shown in Figure 30 on page 154) to stop and start applications, you must install Local Control Agent (LCA) as documented in the *Framework 8.1 Deployment Guide*.

After installing and configuring a solution, any changes made to the properties of an application or a configuration object may delay Orchestration Server processes for a few seconds.

End of procedure

Next Steps

- Continue with Procedure: Importing the application template for Orchestration Server, on page 71

Procedure: Importing the application template for Orchestration Server

Purpose: To import the `Application Template` associated with the Orchestration Server using Configuration Manager.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. In Configuration Manager, select `Environment` > `Application Templates`.
3. Right-click `Application Templates`.
4. From the shortcut menu that opens, select `Import Application Template`.
5. In the `Open` dialog box, navigate to the file for the Orchestration Server's `Application Template`. (Its location on your hard drive or installation media can vary.)

   The file name is `OR_Server_810.apd` if you are using Configuration Server 8.0.3 or later.

   The file name is `OR_Server_Genesys_Server.apd` if you are using a Configuration Server earlier than release 8.0.3.
6. Select this file and click `Open`.
7. In the `Properties` dialog box, click `OK`.

End of procedure

Next Steps

- Continue with Procedure: Creating/Configuring the Orchestration Server Application object

Procedure: Creating/Configuring the Orchestration Server Application object

Start of procedure

1. Log into Configuration Manager as described on page 68.
2. Import the template as described on page 71.
3. In Configuration Manager, select `Environment` > `Applications`.
4. Right-click either the `Applications` folder or the subfolder in which you want to create your `Application` object.
5. From the shortcut menu that opens, select `New` > `Application`.
6. In the `Open` dialog box, locate the template that you just imported, and double-click it to open the Orchestration Server `Application` object. Figure 13 shows an example.

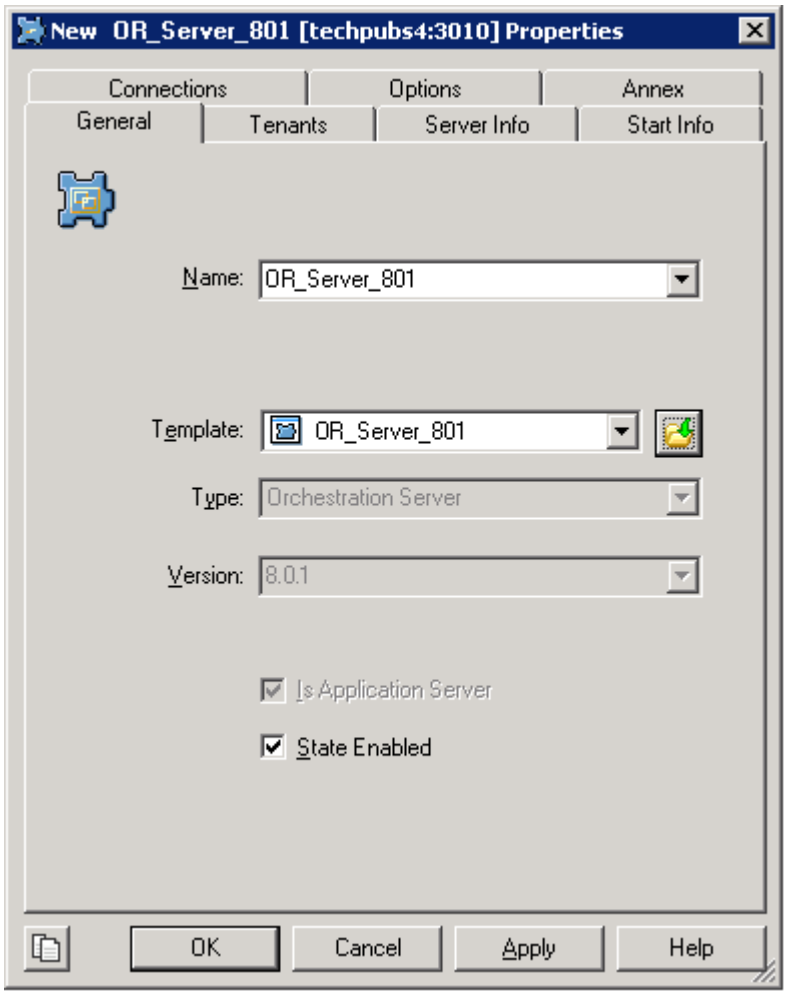

Figure 13: Orchestration Server Application Object

Note: For Configuration Server versions before 8.0.3, the `Type` field will display `Genesys Generic Server`.

7. Select the `General` tab and change the `Application` name (if desired).

Note: The `Application` name should not contain spaces.

8. Make sure that the `State Enabled` check box is selected.
9. In a multi-tenant environment, select the `Tenants` tab and set up the list of tenants that use Orchestration Server.
10. Click the `Server Info` tab and select the following:
    - `Host`—the name of the host on which Orchestration Server resides
    - `Port`—the port through which communication with Orchestration Server can be established. After you select a `Host`, a default port is provided for your convenience. You select the port and click `Edit Port` or you can configure a new port by clicking `Add Port`. Either action brings up the `New Port Info` dialog box (see Figure 14).

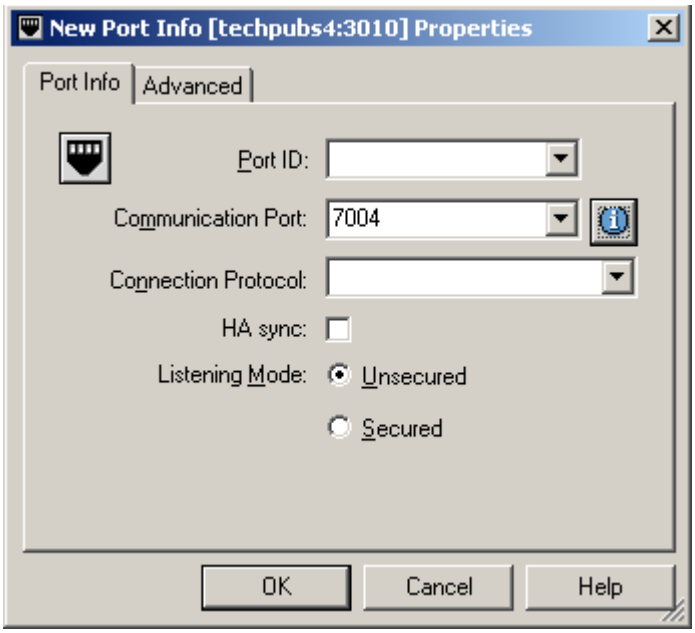

Figure 14: New Port Info Dialog Box

Note: For information on this dialog box, see the Port Info Tab topic in the *Framework 8.1 Configuration Manager Help*.
You can also configure the HTTP port here as well.

11. Select the `Start Info` tab and specify the following:
    - `Working Directory`—the `Application` location (example: `C:/GCTI/or_server`)

- `Command Line`—name of executable file (example: `orchestration.exe`)

Note: If there is a space in the Orchestration Server `Application` name, you must place quotation marks before and after the name of the Orchestration Server `Application`.

- `Command Line Arguments`—list of arguments to start the `Application` (example: `-host <name of Configuration Server host> -port <name of Configuration Server port>-app <name of ORS Application>`

Note: If you are using Configuration Server Proxy and do not use Management Layer, enter the name of Configuration Server proxy for host and the port of the Configuration Server Proxy.

- `Startup time`—the time interval the server waits until restart if the server fails.
- `Shutdown time`—the time interval the server takes to shut down.
- `Auto-Restart setting`—selecting this option causes the server to restart automatically if the server fails.
- `Primary setting`—selecting this option specifies the server as the primary routing server (unavailable).

12. Select the `Connections` tab and specify all the servers to which Orchestration Server must connect
    - T-Server
    - Interaction Server

Note: Depending on the Interaction Server application type, ORS may require connection to an additional Interaction Server. (This Interaction Server can be configured based either on Interaction Server template or on a T-Server template). If the Interaction Server is configured based on an Interaction Server template, ORS needs to be connected as a *client* to one more Interaction Server(s) that was (were) configured based on a T-Server template, since communication between ORS and Interaction Server uses T-Server protocol.

    - Universal Routing Server

> **Note:** To support reconnecting to Configuration Server, you must still create or update the existing connection to Configuration Server in the Orchestration Server `Application` object's `Connections` tab. Follow the standard procedure for configuring connections to other servers. For specific instructions about client-side port connections, see the *Genesys 8.1 Security Deployment Guide*.

End of procedure

Next Steps

- Continue with Procedure: Viewing/changing template options in the orchestration section, on page 76

Defining Client-Side Port Information

To increase security, you can define a fixed port for the connection between an Orchestration Server component and another server that is behind a firewall. The client-side port definition feature allows a server application to control the number of client connections, preventing the server from an excessive number of malicious requests to the same server-side port.

For configuration instructions, see the "Client-Side Port Definition" chapter of the G*enesys 8.1 Security Deployment Guide*. Table 3 on page 29 identifies which Universal Routing-specific components support this type of configuration.

Configuring Options in the Orchestration Server Application Object

This section describes the following procedures:

- "Viewing/changing template options in the orchestration section"
- "Adding and setting non-template options in the orchestration section"
- "Viewing/changing template options in the persistence section"
- "Adding and setting non-template options in the persistence section"
- "Viewing/changing template options in the scxml section"
- "Adding and setting non-template options in the scxml section"
- "Adding and setting non-template options in the log section"
- "Adding a cluster section, then adding and setting clustering options for an ORS node"
- "Adding a web_services section, then adding and setting web services options"

- "Configuring a default port for each node of a cluster"

The Orchestration Server `Application` object that you created in Configuration Manager (or in Genesys Administrator) must be configured, as described in these procedures.

Note: After installing and configuring a solution, any changes made to the properties of an Application or a configuration object may delay Orchestration Server processes for a few seconds.

Configuring the Template Optons in the orchestration Section

The Orchestration Server `Application Template` contains the following configuration option in the `orchestration` section:

Orchestration section template options

- `mcr-pull-interval`

Use the following procedure to view and/or change this option.

Procedure: Viewing/changing template options in the orchestration section

Start of procedure

1. Log in to Configuration Manager as described on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object.
4. Select the `Options` tab.
5. Select the `orchestration` section.
6. In the `orchestration` section, double-click the `mcr-pull-interval` option.
7. In the resulting `Edit Option` dialog box, keep the default `Option Value` or change it to another valid value.

   Refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
8. Click `OK` to save.

End of procedure

Adding/Setting Non-Template Options in the orchestration Section

The following configuration option can be added manually to the Orchestration Server `Application` object in the `orchestration` section:

Orchestration section non-template options

- `session-hung-timeout`

Use the following procedure to add and set values for this option.

Procedure: Adding and setting non-template options in the orchestration section

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the `Environment` tenant and navigate to the `Applications` folder.
   c. Open the Orchestration Server `Application` object.
   d. Select the `Options` tab.
   e. Select the `orchestration` section.
2. Right-click inside the orchestration options window and select `New` from the shortcut menu.
3. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `session-hung-timeout`.
4. In the `Option Value` field, type the default value of `0`, as indicated for this option in the section "Orchestration Server Option Descriptions" on page 109, or type a different valid value.
5. Click `OK` to save.

End of procedure

Configuring the Template Options in the persistence Section

The Orchestration Server `Application Template` contains the following configuration option in the `persistence` section:

Persistence section template options

- `cassandra-listenport`

Use the following procedure to view and/or change this option.

Procedure: Viewing/changing template options in the persistence section

Start of procedure

1. Log in to Configuration Manager as described on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object.
4. Select the `Options` tab.
5. Select the `persistence` section.
6. In the `persistence` section, double-click the `cassandra-listenport` option.
7. In the resulting `Edit Option` dialog box, keep the default `Option Value` `9160`, or change it the port number you want to use.

   Refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
8. Click `OK` to save.

End of procedure

Adding/Setting Non-Template Options in the persistence Section

The following configuration options can be added manually to the Orchestration Server `Application` object in the `persistence` section:

Persistence section non-template options

- `cassandra-nodes`

Use the following procedure to add and set values for these options.

Procedure: Adding and setting non-template options in the persistence section

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the `Environment` tenant and navigate to the `Applications` folder.
   c. Open the Orchestration Server `Application` object.

d. Select the `Options` tab.

e. Select the `persistence` section.

2. Right-click inside the persistence options window and select `New` from the shortcut menu.
3. In the resulting `Edit Option` dialog box, in the `Option Name` field, type the option name listed in "Persistence section non-template options" on page 78 above.
4. In the `Option Value` field, type the default value as indicated for the option in the section "Orchestration Server Option Descriptions" on page 109, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
5. Click `OK` to save.

End of procedure

Configuring the Template Options in the scxml Section

The Orchestration Server `Application Template` contains the following configuration options in the `scxml` section:

- SCXML section template options
- `fips_enable`
- `http-client-side-port-range`
- `http-enable-continue-header`
- `http-max-age-local-file`
- `http-max-cache-entry-count`
- `http-max-cache-entry-size`
- `http-max-cache-size`
- `http-max-redirections`
- `http-ssl-cert-type`
- `http-ssl-key-type`
- `http-ssl-verify-host`
- `http-ssl-verify-peer`
- `http-ssl-version`
- `max-compiler-cache-size`
- `max-compiler-cached-docs`
- `max-includes`
- `max-preprocessor-cache-size`

- max-preprocessor-cached-docs
- max-preprocessor-cached-doc-size
- persistence-default
- session-processing-threads

Use the following procedure to view and/or change these options.

Procedure: Viewing/changing template options in the scxml section

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the Environment tenant and navigate to the Applications folder.
   c. Open the Orchestration Server Application object.
   d. Select the Options tab.
2. Select the scxml section.
3. In the scxml section, double-click one of the options listed in "SCXML section template options" on page 79.
4. In the resulting Edit Option dialog box, keep the default Option Value or change to another valid value.

   Refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
5. Click OK to save.
6. Repeat from Step 3 to view and/or change another option in this section.

End of procedure

Adding/Setting Non-Template Options in the scxml Section

The following configuration options can be added manually to the Orchestration Server Application object in the scxml section:

SCXML section non-template options

- http-no-cache-urls
- http-proxy
- http-ssl-ca-info
- http-ssl-ca-path
- http-ssl-cert

- http-ssl-cipher-list
- http-ssl-key
- http-ssl-random-file
- https-proxy
- password
- persistence-max-active
- system-id

Use the following procedure to add and set values for these options.

Procedure: Adding and setting non-template options in the scxml section

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the Environment tenant and navigate to the Applications folder.
   c. Open the Orchestration Server Application object.
   d. Select the Options tab.
   e. Select the scxml section.
2. Right-click inside the scxml options window, and select New from the shortcut menu.
3. In the resulting Edit Option dialog box, in the Option Name field, type one of the option names listed in "SCXML section non-template options" on page 80.
4. In the Option Value field, type the default value as indicated for the appropriate option in the section "Orchestration Server Option Descriptions" on page 109, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
5. Click OK to save.
6. Repeat from Step 2 to add another option in this section.

End of procedure

Configuring the Options in the log Section

The Orchestration Server Application Template contains the following configuration options in the log section:

Standard Log Options

- all
- buffering
- expire
- segment
- verbose

These are standard Genesys log options. For information on how to set these options, please refer to the appropriate Genesys documentation.

In addition, the following log options can be added to the Application object for Orchestration Server:

Log section non-template options

- x-server-trace-level
- x-server-gcti-trace-level
- x-server-config-trace-level
- x-print-attached-data

Use the following procedure to add these options and set their values.

Procedure: Adding and setting non-template options in the log section

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the Environment tenant and navigate to the Applications folder.
   c. Open the Orchestration Server Application object.
   d. Select the Options tab.
2. Select the log section.
3. Right-click inside the log options window and select New from the shortcut menu.
4. In the resulting Edit Option dialog box, in the Option Name field, type one of the option names listed in "Log section non-template options".

5. In the `Option Value` field, type the default value as indicated in the section "Orchestration Server Option Descriptions" on page 109, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.
6. Click `OK` to save.
7. Repeat from Step 3 to add another option in this section.

End of procedure

Configuring ORS Memory Optimization

An Orchestration Server can be configured to remove passive multimedia interactions from memory cache. ORS will place multimedia interactions in memory cache up to a configurable number. Beyond this maximum value, the oldest multimedia interactions will be removed from the memory cache. You can also set the number of calls that should be deleted when this maximum value is attained.

To configure memory optimization, create an `mcr` section, and add and set options in that section for every ORS `Application` object that will employ memory optimization.

Adding mcr and scxml Sections and Memory Optimization Options for an ORS Instance

To configure memory optimization for an ORS instance, do the following:

1. Add an `mcr` section to the ORS `Application` object.
2. Within the `mcr` section, add and set individual memory optimization options.

Refer to the detailed procedure that follows.

Procedure:
Adding an mcr section, then adding and setting memory optimization options for an ORS instance

Note: When Orchestration is deployed to process multi-media interactions in an environment, in which there may be periods of very few agents available, with a large volume of multi-media interactions waiting to be processed, the following steps should be taken to prevent excessive memory utilization.

Purpose: To configure an ORS instance for memory optimization.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object.
4. Select the `Options` tab.
5. With `Sections` selected in the drop-down list, right-click inside the `Sections` window and select `New` from the shortcut menu.
6. In the `Add Section` dialog box, type `mcr` as the new section name (all lowercase), and click `OK`.
7. Select the `mcr` section.
8. Right-click inside the `mcr` options window and select `New` from the shortcut menu.
9. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `om-memory-optimization`.
10. In the `Option Value` field, type `true` to turn on memory optimization.

    Refer to the section “Orchestration Server Option Descriptions” on page 109 for a more detailed description of this configuration option.
11. Click `OK` to save.
12. Right-click inside the mcr options window and select `New` from the shortcut menu.
13. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `om-max-in-memory`.

14. In the `Option Value` field, type an integer value from `1` to `2000` representing the number of interactions (in thousands) to save in memory cache.

    Note: For high volume loading, Genesys recommends the default value of `100`.

    Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
15. Click `OK` to save.
16. Right-click inside the mcr options window and select `New` from the shortcut menu.
17. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `om-delete-from-memory`.
18. In the `Option Value` field, type an integer value from `1` to `2000000` representing the number of calls to delete when the value that is set for `om-max-in-memory` is reached.

    Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
19. Click `OK` to save.
20. Right-click inside the `scxml` options window and select `New` from the shortcut menu.
21. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `persistence-max-active`.
22. In the `Option Value` field, type an integer value from `100` to `1000000` representing the maximum number of active sessions that the SCXML engine will keep in memory

    Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
23. Click `OK` to save.
24. Click `Apply`.
25. Repeat this procedure for each other ORS `Application` object that will use memory optimization.

End of procedure

Configuring ORS Clustering

Refer to Chapter 3, "Persistence, High Availability, and Load Balancing," on page 45 for a description of clustering and its functionality. To configure clustering, the following general steps are required:

1. Configure persistence (refer to "Configuring Persistent Storage" on page 49 in Chapter 3).
2. Create a `cluster` section and add and set options in that section for every ORS `Application` object (node of the cluster).
3. (Optional). If you will be employing web services, create a `web_services` section and add and set options in that section for each host name or IP address that you will use.
4. Configure ports (on every cluster node); required for nodes to communicate with the cluster.

Adding a cluster Section and Clustering Options for an ORS Instance (Node)

To configure clustering, do the following *for each ORS `Application` object*:

1. Add a `cluster` section to the ORS `Application` object.
2. Within the cluster section, add and set individual clustering options.

Refer to the detailed procedure that follows.

Procedure: Adding a cluster section, then adding and setting clustering options for an ORS node

Purpose: To configure an ORS instance in an ORS cluster.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object (the particular cluster node you are configuring at this time).
4. Select the `Options` tab.
5. With `Sections` selected in the drop-down list, right-click inside the `Sections` window and select `New` from the shortcut menu.
6. In the `Add Section` dialog box, type `cluster` as the new section name (all lowercase), and click `OK`.
7. Select the `cluster` section.
8. Right-click inside the cluster options window and select `New` from the shortcut menu.

9. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `name`.
10. In the `Option Value` field, type the name of the ORS cluster to which this ORS instance belongs.

    Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
11. Click `OK` to save.
12. Right-click inside the cluster options window and select `New` from the shortcut menu.
13. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `super_node`.
14. In the `Option Value` field, type `true` if this node will act as a Super Node, or type `false` if this node will not act as a Super Node.

    Refer to the section "Orchestration Server Option Descriptions" on page 109 for a more detailed description of this configuration option.
15. Click `OK` to save.
16. Click `Apply`.
17. Repeat this procedure for each other ORS Application object that will be part of the cluster.

End of procedure

Adding a web_services Section and Web Services Options

This configuration enables users to invoke SCXML applications from external sources through HTTP calls. This request can be made to ORS at a specific host and port.

To identify a web service (hostname and port):

1. Add a `web_services` section to the ORS `Application` object.
2. Within the web_services section, add and set individual options.

Refer to the detailed procedure that follows.

Procedure: Adding a web_services section, then adding and setting web services options

Purpose: To configure web services in an ORS cluster.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object (the particular cluster node you are configuring at this time).
4. Select the `Options` tab.
5. With `Sections` selected in the drop-down list, right-click inside the `Sections` window and select `New` from the shortcut menu.
6. In the Add Section dialog box, type `web_services` as the new section name (all lowercase), and click `OK`.
7. Select the `web_services` section.
8. Right-click inside the cluster options window and select `New` from the shortcut menu.
9. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `hostname`.
10. In the `Option Value` field, type the IP address or hostname to expose externally for the Web Services functional module.

    Refer to the section “Orchestration Server Option Descriptions” on page 109 for a more detailed description of this configuration option.
11. Click `OK` to save.
12. Right-click inside the cluster options window and select `New` from the shortcut menu.
13. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `port`.
14. In the `Option Value` field, type a port number associated with the IP address or hostname.

    Refer to the section “Orchestration Server Option Descriptions” on page 109 for a more detailed description of this configuration option.
15. Click `OK` to save.
16. Click `Apply`.
17. Repeat this procedure for each other web service IP address or hostname (with associated port) that will be required.

End of procedure

Configuring a port for each node of a cluster

Each ORS instance (node) of a cluster must have a `default` port defined to enable communication among cluster nodes. To configure ports:

- Set a `default` port in the ORS `Application` object's `Server Info` tab in the `Ports` section.

Refer to the detailed procedure that follows.

Procedure:
Configuring a default port for each node of a cluster

Purpose: To configure ports that are required for communication between the nodes of a cluster.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 68.
2. Select the `Environment` tenant and navigate to the `Applications` folder.
3. Open the Orchestration Server `Application` object (the particular cluster node you are configuring at this time).
4. Select the `Server Info` tab.
5. Browse to (using the `Browse` button) the name of a valid host using the `Host` drop-down list; select it and click `OK`.
6. Click `Add Port` to open the `New Port Info Properties` dialog box.
7. To create a `default` port:
   a. Type `default` in the `Port ID` field.
   b. Type or select an unused port number in the `Communication Port` field.
   c. Click `OK`, and add another port, or click `Apply` to apply your settings, or select another tab in the `ORS Application Properties` dialog box to continue.
8. Repeat this procedure for each other ORS instance (cluster node).

End of procedure

Configuring Other Options That Affect Orchestration Server

Configuration options affecting the operation of Orchestration Server are not confined to those available in the Orchestration Server `Application` object. Other options affecting the operation of Orchestration Server include:

- The `application` option in the `Orchestration` section (on the `Annex` tab) of a `DN`, `RoutePoint`, or eServices `Queue` object.

- Various options specified in the Enhanced Routing Script object (`CfgEnhancedRouting`) which represents the SCXML application in Configuration Manager or Genesys Administrator.

Note: In order for Orchestration Server to communicate with Interaction Server, the `service_timeout` option must be set in Universal Routing Server. For information on URS options that affect Orchestration Server/Interaction Server processing including the `service_timeout` option, see the *Universal Routing 8.1 Reference Manual*.

Configuring the application Option on a DN, RoutePoint, or eServices Queue Object

This section describes the following procedure:

- "Adding and setting the application option in the Orchestration section of a DN or Queue object"

The `application` option is specified on the `Annex` Tab of a `DN`, `RoutePoint` or eServices `Queue` object in order to specify the provisioning of the SCXML application associated with this resource.

Procedure: Adding and setting the application option in the Orchestration section of a DN or Queue object

Start of procedure

1. If not already performed previously, log in to Configuration Manager as described on page 68.
2. Select the appropriate Tenant folder, Switch name, and DN folder.
3. Open the appropriate `DN` or eServices `Queue` object. Navigate to the `Scripts` folder to select appropriate eServices `Queue` object.
4. Select the `Annex` tab.
5. Select or add the `Orchestration` section.
6. Right-click inside the options window and select `New` from the shortcut menu.
7. In the resulting `Edit Option` dialog box, in the `Option Name` field, type `application`.

8. In the Option Value field, type the URL of the SCXML document to load.

   Refer to the option description for "application" on page 125 for a full description of this configuration option its valid values. Table 6, "URL Parameter Elements for application option," on page 125 provides useful information about parameters that can be added to the URL.

9. Click OK to save.

End of procedure

Configuring Options Specified in Script Objects of Type CfgEnhancedRouting (Enhanced Routing Script)

This section describes the following procedures:

- "Adding and setting options specified in the Application section of CfgEnhancedRouting Script objects"
- "Adding and setting the {Parameter Name} option in the ApplicationParms section of CfgEnhancedRouting Script objects"

The Annex tab of a DN or Interaction Queue resource can also refer to a Script object of type CfgEnhancedRouting. This allows more complex parameters and options to be configured to locate and control the SCXML application. In addition, multiple DN or Interaction Queue objects can refer to the same Script object, allowing application configuration to be centralized and reused from multiple places.

The following configuration options can be added manually to a CfgEnhancedRouting Script object in the Application section (on the Annex tab):

Application section non-template options

- alternate-url
- fetch-timeout
- http-useragent
- http-version
- max-age
- max-duration
- max-loop-count
- max-stale
- url

Use the following procedure to add and set values for these options.

Procedure: Adding and setting options specified in the Application section of CfgEnhancedRouting Script objects

Start of procedure

1. If not already performed previously, log in to Configuration Manager as described on page 68.
2. Select the appropriate `Tenant` and navigate to the `Scripts` folder.
3. Open the appropriate `Script` object of type `Enhanced Routing Script` (`CfgEnhancedRouting`).
4. Select the `Annex` tab.
5. Select or add the `Application` section.
6. Right-click inside the options window and select `New` from the shortcut menu.
7. In the resulting `Edit Option` dialog box, in the `Option Name` field, type one of the option names listed in "Application section non-template options" on page 91.
8. In the `Option Value` field, type the default value, as indicated for the appropriate option in the section "Orchestration Server Option Descriptions" on page 109, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 109 for a full description of the configuration options in this section and their valid values.

   For the `url` option, refer to the description for "url" on page 131 for a full description of this configuration option its valid values. Table 8, "URL Parameter Elements for url option," on page 131 provides useful information about parameters that can be added to the URL.

   For the `alternate-url` option, refer to the description for "alternate-url" on page 126 for a full description of this configuration option its valid values. Table 7, "URL Parameter Elements for alternate-url option," on page 127 provides useful information about parameters that can be added to the URL.
9. Click `OK` to save.
10. Repeat from Step 3 above to add another option in this section.

End of procedure

In addition, an option can be used to specify a string that represents a parameter value that is to be passed to the application. The `ApplicationParms` section contains the values for data elements that can be referred to within the

SCXML application. The Enhanced Routing Script object is named as such to identify SCXML applications and Routing applications. Existing IRD-based IRL applications are provisioned as script objects.

Procedure:
Adding and setting the {Parameter Name} option in the ApplicationParms section of CfgEnhancedRouting Script objects

Start of procedure

1. If not already performed previously:
   a. Log in to Configuration Manager as described on page 68.
   b. Select the appropriate `Tenant` and navigate to the `Scripts` folder.
   c. Open the appropriate `Script` object of type `Enhanced Routing Script` (`CfgEnhancedRouting`).
   d. Select the `Annex` tab.
2. Select or add the `ApplicationParms` section.
3. Right-click inside the options window and select `New` from the shortcut menu.
4. In the resulting `Edit Option` dialog box, in the `Option Name` field, type a name for the parameter option.
5. In the `Option Value` field, type the value for the option.

   Refer to the option description for "{Parameter Name}" on page 132 for a full description of this configuration option its valid values. Table 9, "Parameter Elements for ApplicationParms," on page 132 provides useful information about parameters that can be added. Figure 15 shows an example of the use of the `ApplicationParms` section.

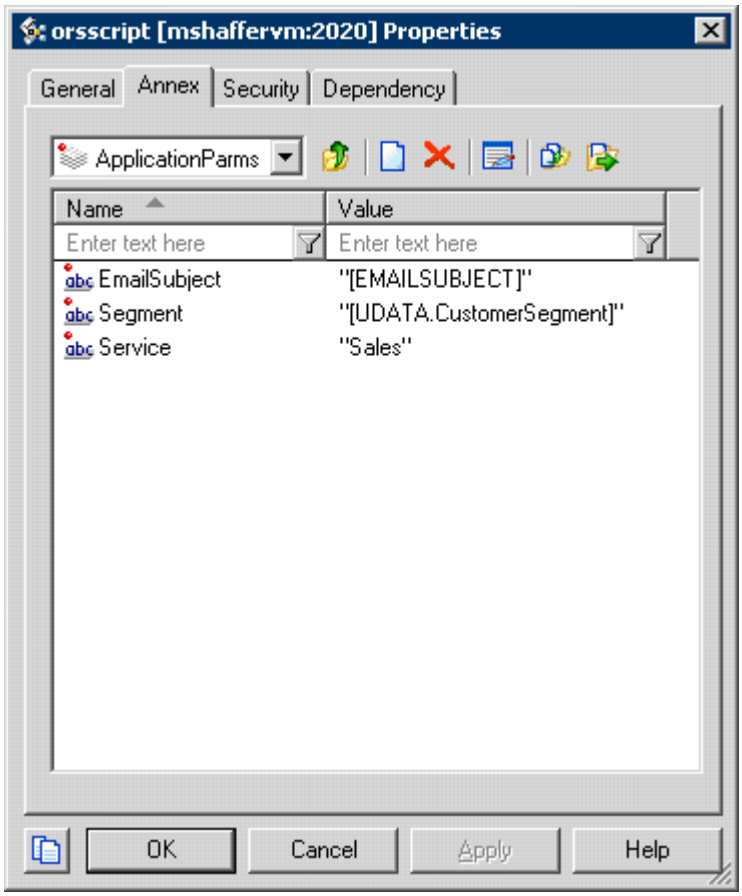

Figure 15: Script Object Annex Tab for ApplicationParms

6. Click `OK` to save.
7. Repeat from Step 3 to add another option in this section.

End of procedure

Configuring ORS to Operate with eServices Interactions

ORS, acting as a client, must have two separate connections to every instance of Interaction Server operating in the environment. Each instance of Interaction Server is represented by two `Application` objects in the ORS `Connections` tab. See Figure 16 on page 95.

You can think of the first Interaction Server `Application` object as a server and and the second object as its proxy. You need only install and operate one instance of Interaction Server—the server instance. It is not necessary to install the proxy instance because it does not run as an application.

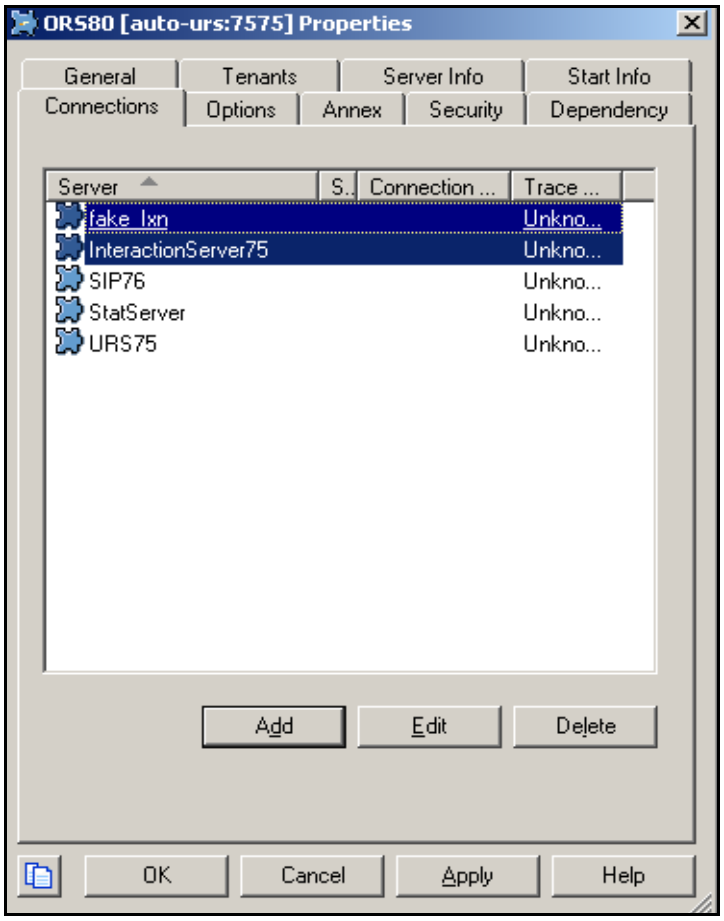

Figure 16: Interaction Server Instances in ORS Application Properties

Procedure: Configuring ORS to operate with eServices Interactions

Purpose: To enable ORS to operate with eServices interactions by configuring two Interaction Server `Application` objects.

Start of procedure

1. Log on to Genesys Administrator.
2. Import two templates to create the Interaction Server `Application` objects.
   a. Use an Interaction Server `Application` template for the `Application` object that you are using for the actual installation.
   b. Use a T-Server `Application` template (with its default configuration) for the second `Application` object. See Figure 17.

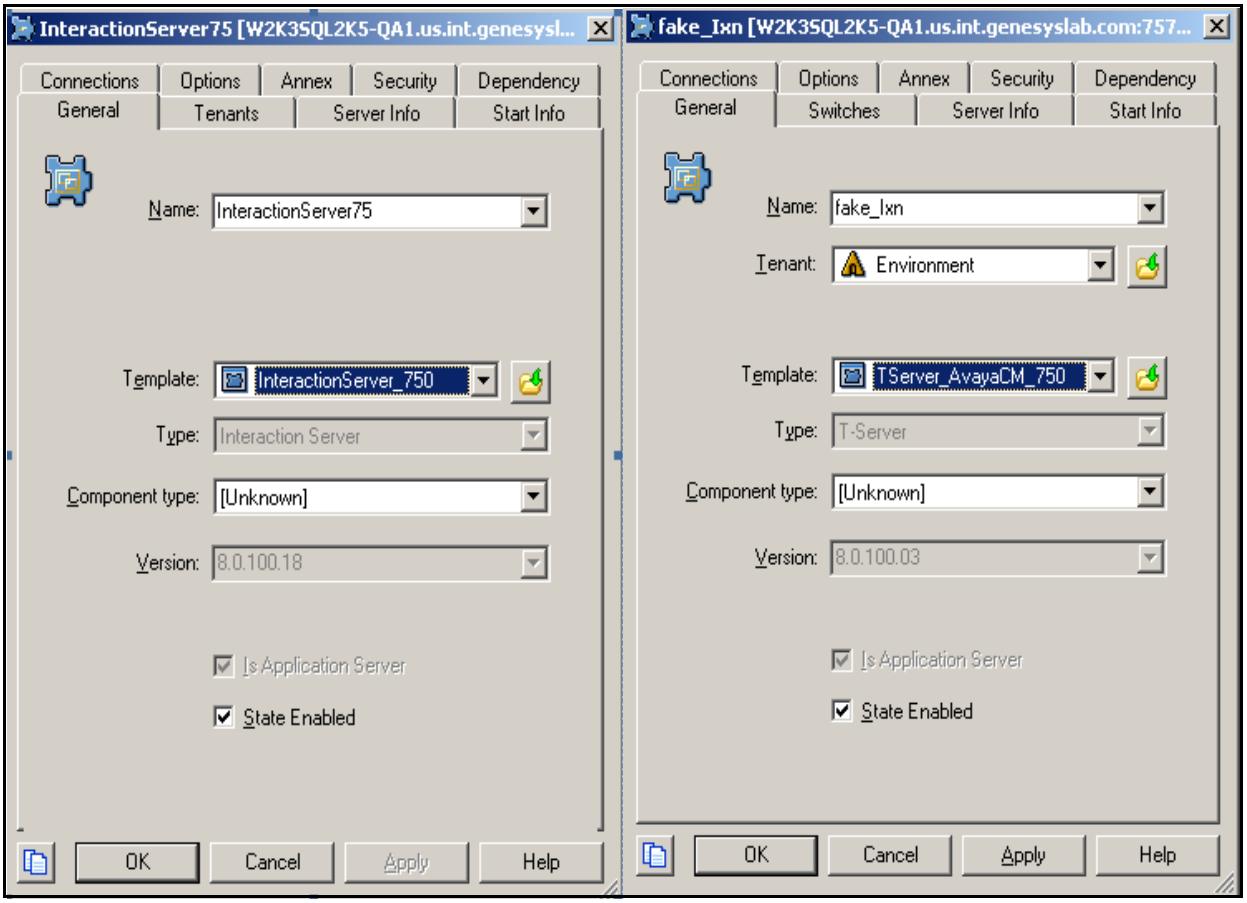

Figure 17: Interaction Server and T-Server Application Templates

3. Create the Interaction Server `Application` objects.

   Ensure that both of the Interaction Server `Application` objects have different `Application` object names. (see Figure 16)

4. In the second `Application` object (based on the T-Server template) configure a connection to the same multimedia switch that is used for the first `Application` object (based on the Interaction Server template).

> **Note:** The first Interaction Server `Application` (created by using the Interaction Server template) has no association in its properties with a multimedia switch. ORS determines this association, based on the tenant that is specified in `General` tab of the ORS `Application`. Alternatively, the second the Interaction Server `Application` (based on T-Server template) must have a multimedia switch configured in its properties, and it must be the same one that is used by the first Interaction Server `Application`.

5. On the `Server Info` tab of each Interaction Server `Application`, configure the same port number and ensure both `Applications` exist on the same host. See Figure 18.

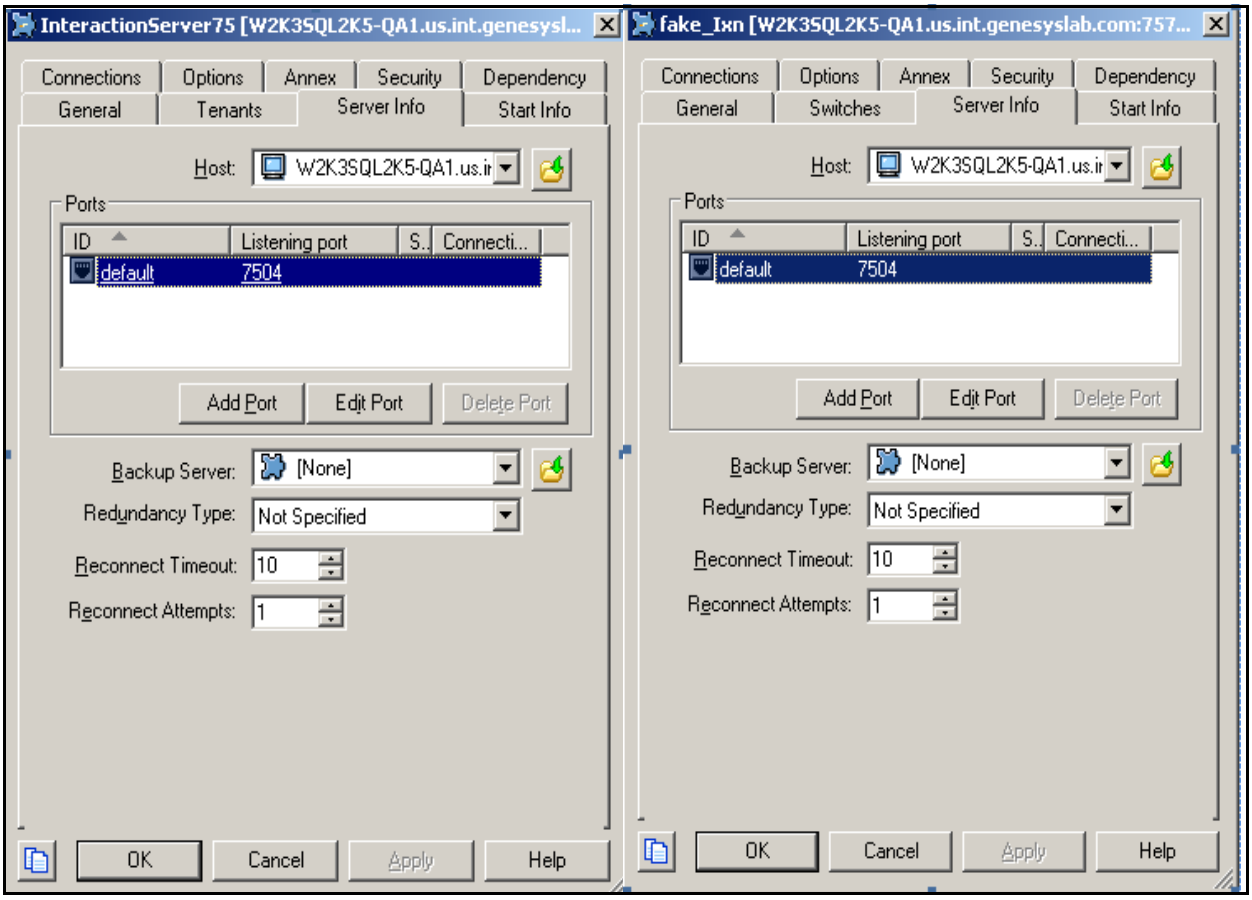

Figure 18: Interaction Server Application Objects on the Same Host

6. Save the configuration.

End of procedure

How ORS Processes eServices Interactions

The routing strategies associated with eServices interactions that are operating with ORS are not loaded to the Virtual Routing Points that are configured in the multimedia switch (unlike those associated with eServices interactions with URS). Instead, all eServices interactions are loaded directly to the Interaction Queues, which are located in the `Scripts` folder of Configuration Manager.

ORS processes the interactions by pulling them from the Interaction Server. A `RequestPull` request is used to retrieve a subset of interactions from the specified Queue/View combination (every `Queue` object must have at least one `View` associated with it). The Queues from which ORS pulls the interactions must be explicitly associated with that specific ORS `Application`. ORS checks

the `Queue` section in the `Annex` tab of the Interaction Queue `Application` for the `Orchestration` section and pulls interactions from these Queues only. See the `Annex` tab in Figure 19.

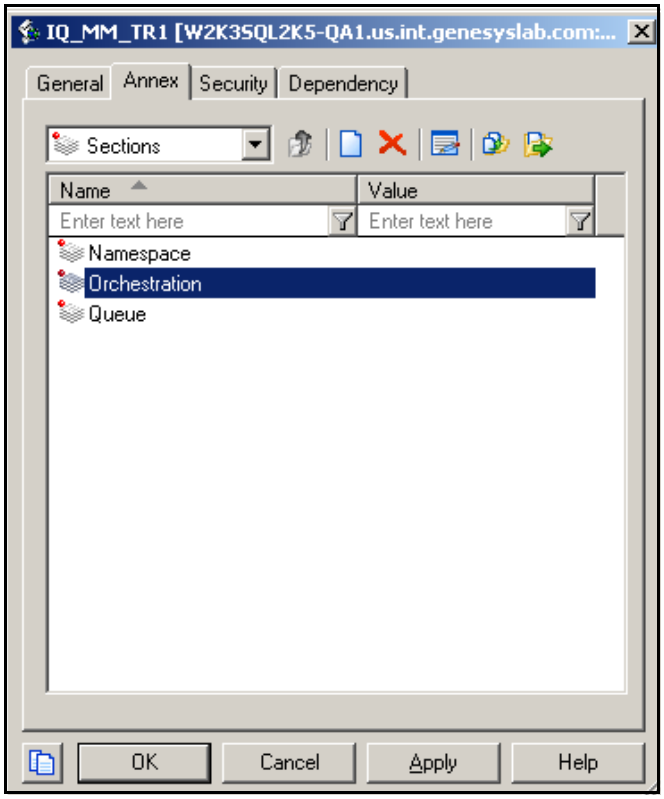

Figure 19: Queue and Orchestration Sections in Interaction Queue

The `Orchestration` configuration section can contain an `application` option with a value that is the path to the strategy (SCXML script) that will be executed. Figure 20 on page 99 shows the manually created Orchestration section and application option on the `Annex` tab of the Interaction Queue in Configuration Manager.

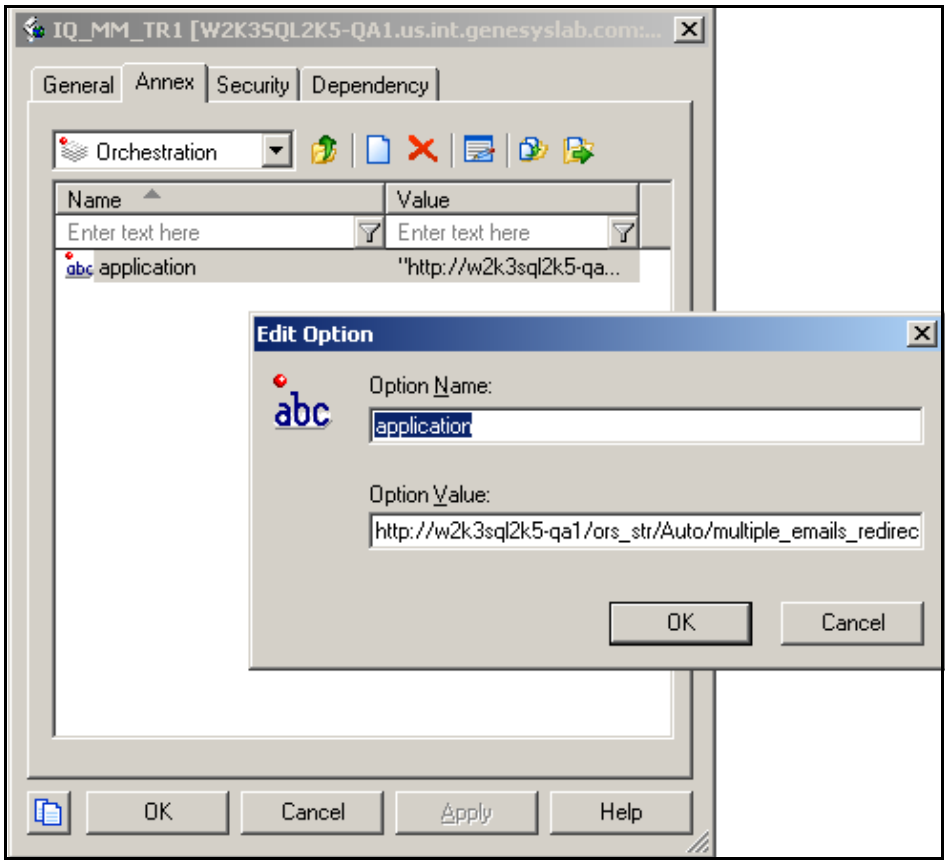

Figure 20: Annex Tab—Orchestration Configuration Option

You can also associate the Interaction `Queue` with the a specific ORS by assigning the strategy (SCXML script) to that `Queue` in Genesys Administrator (see Figure 21). For example, in Genesys Administrator:

1. Go to `Provisioning > Routing/eServices > Interaction Queues`.
2. Navigate to the properties of a particular Interaction Queue.
3. In the `Application` field of `Orchestration` section, select the strategy (SCXML script) that will be assigned to that Queue and save the changes.

Figure 21: Interaction Queue in Genesys Administrator

You can also associate an Interaction Queue with ORS by assigning the strategy to a `Queue` by using Composer. For information about how to assign Queues in Composer, see the *Composer 8.0 Routing Applications User's Guide*.

All properly associated (managed) combinations of the Interaction `Queues/Views` are written to the ORS log at startup. See Figure 22.

```
10:58:30.888 W-NODE[208]: 1 >> 1 [E: 4, T: 4]
10:58:30.888 S-NODE[208]: 1 >> 1 [E: 6, T: 6]
10:58:31.888 Std 04562 Warm Standby (Primary) mode activated
10:58:31.888 [ITXMAN]: 'IQ_MM_TR1/IV_MM_TR1' view managed by Orchestration
10:58:31.888 [ITXMAN]: 'IQ_MM_TR2/IV_MM_TR2' view managed by Orchestration
10:58:31.888 [ITXMAN]: 'IQ_MM_TR3/IV_MM_TR3' view managed by Orchestration
10:58:31.888 [ITXMAN]: 'IQ_MM_TR4/IV_MM_TR4' view managed by Orchestration
10:58:31.888 Std 21027 S-NODE[208](MASTER): This super node is operating as master
10:58:31.888 W-NODE[208]: << Timer: STARTUP <<
```

Figure 22: Interaction Queue Data in ORS Log

TMakeCall Parameter Support in Switch Configurations

The `TMakeCall` parameter supports only numeric values for parameters that do not have additional switch configurations.

By default, the destination address in the `<createcall>` action contains only digits, to a maximum of 25 digits. This default limitation can be modified by explicitly configuring the range of valid symbols and the maximum number of symbols. You can configure in the `Switch` configuration object in the `gts` section of `Annexes`.

The following two options control which symbols are valid and the maximum number of valid symbols that can be dialed:

Option: valid-digits

Value: An explicit set of all acceptable symbols that can be dialed.

Example: `0123456789`

Option: `max-digits`

Value: The maximum number of digits that can be dialed.

Example: `25`.

Other Manual Configuration Operations

This section describes other manual configuration operations that you may wish to perform in Configuration Manager.

Creating Business Attributes

Some `Business Attributes` might already be defined:

- Any screening rule `Business Attributes`, defined in Knowledge Manager, will have carried over into Configuration Manager from the Universal Contact Server database. Their names will be viewable in the `<language_name>` folder associated with the particular Tenant.
- Any classification category Business Attributes, defined in Knowledge Manager, will have also carried over from the Universal Contact Server database. The names will be viewable in the `Category Structure` folder along with the name of the associated standard responses.

You may still need to use Configuration Manager to define the following `Business Attributes`:

- `Service Type` and `Customer Segment` used in the MultiAttach object or for a Cost-based Routing solution as described in the *Universal Routing 7.6 Routing Application Configuration Guide*.
- `Disposition Code`. Contained in `Interactions` table.
- `Language`. The primary motivation for creating is the necessity to use this business attribute in conjunction with some particular `Tenant` in Knowledge Manager. All objects in Knowledge Manager, including classification categories and screening rules, can only be created under a specific combination of `Language` and `Tenant`.
- `Reason Code` and `Stop Processing Reason`.
- You may want to create a custom stop processing reason that will be added to the Genesys predefined set.
- `E-mail Accounts` specifies an external e-mail address.

- `Media Type`. When implementing the Genesys Open Media, another business attribute that you may want to create is a custom `Media Type` that will be added to the Genesys predefined set of `Media Types`. A service performed by some third-party server associated with a custom media type can be used in the External Service object.

Manually Configuring Stat Server

Stat Server tracks the real-time status of resources such as agents. For detailed information about configuring Stat Server, see the *Framework 8.1 Stat Server User's Guide*.

Premise T-Server for Network Routing

Refer to the appropriate T-Server document for configuring a premise T-Server application for Network Routing.

Orchestration Server Configuration Options

This section provides details on all configuration options available in the Orchestration Server Application.

Setting Options

Most options described in this section are specified on the `Options` tab of the Properties window of the `Application` object. Figure 23 shows an example of the properties window of the Orchestration Server `Application` object in Configuration Manager with the `scxml` section visible.

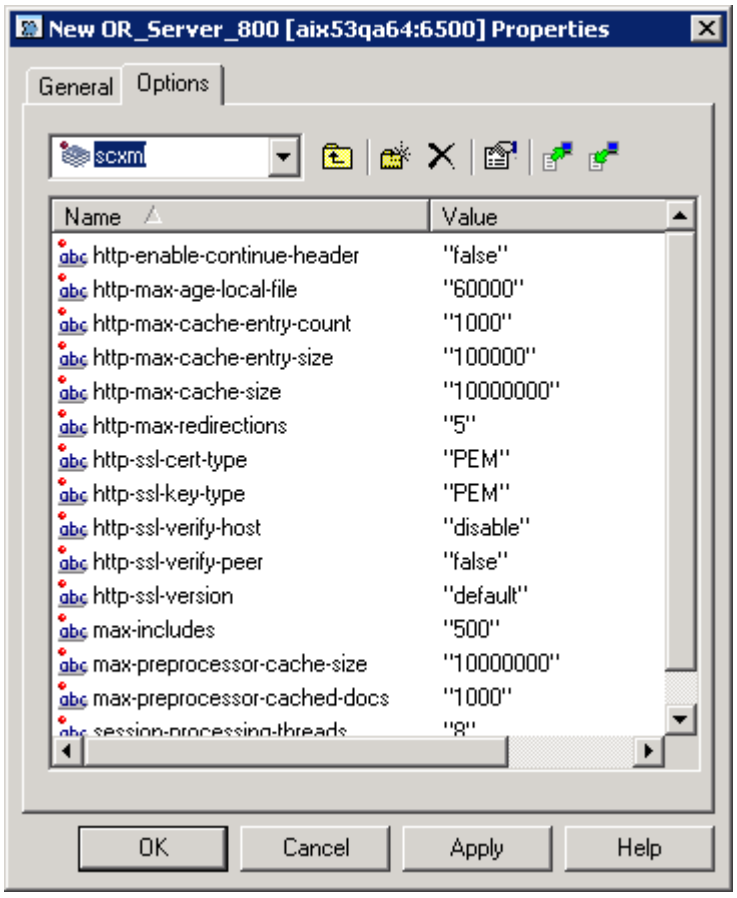

Figure 23: Properties Dialog Box, Options Tab, scxml Section

Orchestration Server Options

Orchestration Server options are placed in the following option folders:

- For the Orchestration Server `Application` object—In the `orchestration`, `persistence`, `scxml`, `log`, `mcr`, and `cluster` sections of the `Options` tab of the Orchestration Server `Properties` dialog box. If you are configuring clustering, you must create a `cluster` section and add options to it.
- For `DN` (`Extension` or `RoutePoint`), or `Interaction Queue` objects—In the `Orchestration` section of the `Annex` tab of the appropriate `Properties` dialog box.
- For Script objects of type `CfgEnhancedRouting` (Enhanced Routing Script objects)—In the `Application` or `ApplicationParms` section of the `Annex` tab of the appropriate `Properties` dialog box.

Summary of Options

Table 5 lists the options that you can configure in the `Application` object for Orchestration Server, or other appropriate components (as specified in the full

description of the option), a short option description, and the page number on which the option information can be found. The table contains brief descriptions of the options, for quick reference.

List of Orchestration Server Options

Table 5 lists Orchestration Server options by *section*, and then in alphabetical order.

Table 5: Orchestration Server Options

Name	Brief Description	Detailed Description Page
orchestration section (ORS `Application` object)		
mcr-pull-interval	Specifies the number of milliseconds between attempts to pull interactions from the pairs of queues/views managed by the Orchestration Server.	109
mcr-pull-limit	Specifies the maximum number of pulled interactions that each node in a cluster (or ORS working in standalone mode) is allowed to have at any given time.	109
session-hung-timeout	Specifies the time (in seconds) that a hung session has to complete processing and exit gracefully before being terminated by the system.	109
persistence section (ORS `Application` object)		
cassandra-listenport	Specifies the Cassandra client connection port when using Cassandra-based persistence method.	110
cassandra-nodes	Semi-colon-separated list of host names or IP addresses when using Cassandra-based persistence method.	110
scxml section (ORS `Application` object)		
fips_enable	Enables FIPS compliance in the SCXML library.	110
http-client-side-port-range	Specifies the port range of the socket used for an HTTP client side connection. An empty value, or lack of option, indicates that the default should be used.	111
http-enable-continue-header	Specifies whether to enable or disable the `100-continue` header in the HTTP 1.1 post request.	111

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
http-max-age-local-file	The maximum age of a local file, in milliseconds	111
http-max-cache-entry-count	The maximum number of entries that can be stored in the cache	112
http-max-cache-entry-size	The maximum size of each cache entry, in bytes	112
http-max-cache-size	The maximum size of the HTTP cache, in bytes.	112
http-max-redirections	The maximum number of times to follow the `Location:header` in the HTTP response	113
http-no-cache-urls	A comma-delimited list of substrings to prevent a response from being cached.	113
http-proxy	The HTTP proxy server (if applicable).	113
http-ssl-ca-info	The file name holding one or more CA certificates with which to verify the peer.	114
http-ssl-ca-path	The path holding one or more CA certificates with which to verify the peer.	114
http-ssl-cert	The file name of the SSL certificate.	114
http-ssl-cert-type	The SSL Certificate type (PEM or DER)	115
http-ssl-cipher-list	The cipher list as defined by OpenSSL.	115
http-ssl-key	The path or file name of the SSL private key.	115
http-ssl-key-type	The SSL Key type (PEM or DER)	116
http-ssl-random-file	The file which is read from in order to see the random engine for SSL.	116
http-ssl-verify-host	Specifies how the common name from the peer certificate should be verified during the SSL handshake	116
http-ssl-verify-peer	Specifies whether or not the system should verify the peer's certificate	117
http-ssl-version	The SSL version to be used	117
https-proxy	The HTTPS proxy server (if applicable).	117

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
max-compiler-cache-size	The maximum amount of memory (in bytes) allowed for the `<xi:include>` compiler cache.	118
max-compiler-cached-docs	The maximum number of items that the `<xi:include>` compiler cache can have.	118
max-compiler-cached-doc-size	The maximum size, in bytes, allowed to cache the results of the compiled `<xi:include>` document.	118
max-includes	The maximum number of documents that may be included using `<xi:include>`	119
max-preprocessor-cache-size	The amount of memory, in bytes, that the `<xi:include>` pre-processor cache can use	119
max-preprocessor-cached-docs	The maximum number of items that the `<xi:include>` pre-processor cache can have	119
max-preprocessor-cached-doc-size	The maximum size, in bytes, of the cached results of the preprocessed `<xi:include>` documents.	119
password	The SSL key password.	120
persistence-default	Allows to suppress all persistence capabilities with the SCXML engine.	120
persistence-max-active	Allows setup of a maximum number of active sessions that the SCXML engine will keep in memory.	120
session-processing-threads	The number of threads in the thread pool	121
system-id	Allows setup of the Orchestration Server (ORS) system ID in order to have unique session IDs across ORS instances.	121
log section (ORS `Application` object)		
all buffering expire segment verbose	These are standard Genesys log options.	n/a

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
x-server-trace-level	The level of tracing to be enabled for the Orchestration Server	121
x-server-gcti-trace-level	The level of GCTI tracing to be enabled for the Orchestration Server. Controls how much detail should be in the logs for GCTI-related events, such as those from T-server.	121
x-server-config-trace-level	The level of configuration tracing to be enabled for the Orchestration Server. Controls how much detail should be in the logs for configuration-related events	122
x-print-attached-data	Specifies whether or not attached data should be formatted and printed in the logs	122
mcr section (ORS `Application` object)		
om-memory-optimization	Specifies whether memory optimization is in effect	122
om-max-in-memory	The maximum number of interactions that will be stored in memory cache before any interactions will be removed (when `om-memory-optimization` is set to `true`)	123
om-delete-from-memory	The number of calls that will be deleted when the value of `om-max-in-memory` has been attained	123
cluster section (ORS `Application` object)		
name	The string name of the Orchestration Server cluster this to which ORS application node belongs to	123
super_node	Specifies whether the node should act as a Super Node in the cluster	124
web_services section (ORS `Application` object)		
hostname	The IP address or hostname to expose externally for the Web Services functional module.	124
port	The port number associated with the IP address or hostname identified for the web services in the `hostname` option.	124
Orchestration section (DN Extension or RoutePoint, or Interaction Queue)		

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
application	Specifies the URL of the SCXML document to load.	125
Application section (Script object `CfgEnhancedRouting`)		
alternate-url	This option specifies the URL of the SCXML document to load.	126
fetch-timeout	Specifies the time (in milliseconds) before a document fetch is abandoned.	128
http-useragent	The string to use in the HTTP header field `User Agent`, which identifies the application to the web server.	128
http-version	The HTTP version to use when fetching documents from the application server	128
max-age	Tells the Orchestration Server how long an application script can be cached.	129
max-duration	The maximum duration, in milliseconds, of a single ECMAScript script	129
max-loop-count	The maximum number of loops that can be performed in a single ECMAScript script	130
max-stale	The number of seconds to extend the life of a cached file.	130
url	Specifies the URL of the SCXML document to load.	131
ApplicationParms section (Script object `CfgEnhancedRouting`)		
{Parameter Name}	Specifies a string that represents a parameter value to be passed to the application.	132

The "Orchestration Server Option Descriptions" section lists all Orchestration Server options.

Orchestration Server Option Descriptions

mcr-pull-interval

Option section: `orchestration`
Configuration object: ORS `Application` object
Default value: `1000`
Valid values: Any positive integer.
Value changes: in setting the next timer interval

This option provides the number of milliseconds between attempts to pull interactions from the queues/views managed by the Orchestration Server.

For example:

```
orchestration/mcr-pull-interval = 5000
```

mcr-pull-limit

Option section: `orchestration`
Configuration object: ORS `Application` object
Default value: `1000`
Valid values: Any positive integer from `0` to `5000`
Value changes: in setting the next timer interval

This option provides the maximum number of pulled interactions that each node in a cluster (or ORS working in standalone mode) is allowed to have at any given time. A value of `0` disables the pulling of multi-media interactions completely for this instance of Orchestration Server.

For example:

```
orchestration/mcr-pull-limit = 5000
```

session-hung-timeout

Option section: `orchestration`
Configuration object: ORS `Application` object
Default value: `0`
Valid values: Any integer greater than or equal to `0`
Value changes: Immediately

This option specifies the time (in seconds) that a hung session has to complete processing and exit gracefully before being terminated by the system. A value of `0` indicates that the session will not be forcibly terminated.

The Orchestration Server can detect the following conditions that represent a hung session:

- a session spends too long executing a script element (see the `max-duration` option)

- a session executes too many iterations of a loop (stuck in a loop) in a single ECMAScript script. This applies to `<script>`, `expr` or `cond` (see the `max-loop-count` option)

For example:

`orchestration/ session-hung-timeout = 0` (no timeout)

`orchestration/ session-hung-timeout = 3600` (1 hour)

`orchestration/ session-hung-timeout = 604800` (1 week)

cassandra-listenport

Option section: `persistence`
Configuration object: ORS `Application` object
Default value: `9160`
Valid values: Any valid socket port.
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option provides the Cassandra client connection port when using the Cassandra-based persistence method.

This option must be supplied with an appropriate value for correct Cassandra-based persistence operation to occur.

For example:

```
persistence/cassandra-listenport = 9160
```

cassandra-nodes

Option section: `persistence`
Configuration object: ORS `Application` object
Default value: <empty string>
Valid values: A list of hostnames for Cassandra nodes in the Cassandra cluster.
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option provides the Cassandra client connection port when using the Cassandra-based persistence method.

This is a semi-colon separated list of host names or IP addresses.

For example:

```
persistence/cassandra-nodes = DWS;mpswin;test47
```

fips_enable

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `false`

Valid values: `true, false`
Value changes: During startup. Changes to this option are not applied dynamically.

Enables FIPS compliance in the SCXML library.

For example:

```
scxml/fips_enable = false
```

http-client-side-port-range

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: A number between `1` and `65535` followed by a "`-`" and then another number between `1` and `65535`; with the second number larger than the first number.
Value changes: During startup. Changes to this option are not applied dynamically.

Specifies the port range of the socket used for an HTTP client side connection. An empty value, or lack of option, indicates that the default should be used.

For example:

```
scxml/http-client-side-port-range = 1000-2000
```

http-enable-continue-header

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `false`
Valid values: `true, false`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies whether to enable or disable the `100-continue` header in the HTTP 1.1 post request.

For example:

```
scxml/http-enable-continue-header = true
```

http-max-age-local-file

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `60000`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum age of the local file in milliseconds.

For example:

```
scxml/http-max-age-local-file = 65000
```

http-max-cache-entry-count

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `1000`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum number of entries that can be stored in the cache.

For example:

```
scxml/http-max-cache-entry-count = 1250
```

http-max-cache-entry-size

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `100000 (100K)`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum size of each cache entry in bytes.

For example:

```
scxml/http-max-cache-entry-size = 125000
```

http-max-cache-size

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `10000000 (10 MB)`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum size of the HTTP cache in bytes.

For example:

```
scxml/http-max-cache-size = 12500000
```

http-max-redirections

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `5`
Valid values: `0 - 100`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum number of times to follow the `Location:header` in the HTTP response. Set to `0` to disable HTTP redirection.

For example:

```
scxml/http-max-redirections = 10
```

http-no-cache-urls

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Comma-delimited set of strings
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets a comma-delimited list of substrings that would prevent a response from being cached, if the URL contains any of the substrings.

For example:

```
scxml/http-no-cache-urls=myserver.com, yourserver.com
```

http-proxy

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: <empty string>
Valid values: Valid IP Address or URL: Port
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the HTTP proxy server (if applicable). If specified, all HTTP fetches done by the Orchestration Server will be done via this proxy server.

For example:

```
scxml/http-proxy = 127.0.0.1:3128
scxml/http-proxy = myproxy:3128
```

http-ssl-ca-info

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: <empty string>
Valid values: Valid file name
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the file name holding one or more CA certificates with which to verify the peer. This is only useful when `ssl-verify-peer=true`.

For example:

```
scxml/http-ssl-ca-info = myca.cer
```

http-ssl-ca-path

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: ""
Valid values: Valid path
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the path holding one or more CA certificates with which to verify the peer. The certificate directory must be prepared using the `openssl c_rehash` utility.

For example:

```
scxml/http-ssl-ca-path = c:\ssl\ca
```

http-ssl-cert

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Valid file name
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the file name of the SSL certificate.

For example:

```
scxml/http-ssl-cert = mycert.crt
```

Note: Spaces are not valid in the directory name that is specified in the file path for this option. For example `C:\Program Files\Certificates\my_cert.pem` is not a valid path.

http-ssl-cert-type

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `PEM`
Valid values: `PEM, DER`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the SSL Certificate Type:

- `PEM` – PEM encoded certificate
- `DER` – DER encoded certificate

For example:

```
scxml/http-ssl-cert-type = DER
```

http-ssl-cipher-list

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: String value
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the cipher list as defined by OpenSSL. Paste the following url into your web browser to see valid cipher list formats:

```
http://www.openssl.org/docs/apps/ciphers.html#CIPHER_LIST_FORMAT
```

For example:

```
scxml/http-ssl-cipher-list=RC4-SHA
```

http-ssl-key

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Valid path or file name
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the path or file name of the SSL private key.

For example:

```
scxml/http-ssl-key = c:\ssl\mykey.key
```

Note: Spaces are not valid in the directory name that is specified in the file path for this option. For example `C:\Program Files\Certificates\my_cert.pem` is not a valid path.

http-ssl-key-type

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `PEM`
Valid values: `PEM, DER`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the SSL Key Type:

- `PEM` – PEM encoded key
- `DER` – DER encoded key

For example:

```
scxml/http-ssl-key-type = DER
```

http-ssl-random-file

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Valid file name
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the file which is read from in order to see the random engine for SSL. The more random the specified file is, the more secure the SSL connection will become.

For example:

```
scxml/http-ssl-random-file=c:\ssl\random-seed
```

http-ssl-verify-host

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `disable`
Valid values: `disable, common, match`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies how the common name from the peer certificate should be verified during the SSL handshake.

- `disable` – the connection succeeds regardless of the names in the certificate
- `common` – the certificate must contain a "Common Name" field, but the field's contents are not validated
- `match` – the certificate must indicate correct server name, or the connection will fail

For example:

```
scxml/http-ssl-verify-host = match
```

http-ssl-verify-peer

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `false`
Valid values: `true, false`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies whether the system should verify the peer's certificate. When this is `true`, either `http-ssl-ca-path` or `http-ssl-ca-info` would be set.

For example:

```
scxml/http-ssl-verify-peer = true
```

http-ssl-version

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `default`
Valid values: `String (default, TLSv1, SSLv2 or SSLv3)`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the SSL version to be used. By default, the system will determine the correct version to use. However, this option may be useful when some servers make it difficult to determine the correct SSL version.

For example:

```
scxml/http-ssl-version = SSLv2
```

https-proxy

Option section: `scxml`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Valid IP Address or URL: Port

Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the HTTPS proxy server (if applicable). If specified, all HTTPS fetches done by the Orchestration Server will be done via this proxy server.

For example:

```
scxml/https-proxy = 127.0.0.1:3128
```

max-compiler-cache-size

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `10000000`
Valid values: non-negative integer
Value changes: Takes effect after restart. Changes to this option are not applied dynamically.

The maximum amount of memory (in bytes) allowed for the `<xi:include>` compiler cache.

max-compiler-cached-docs

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `1000`
Valid values: non-negative integer
Value changes: Takes effect after restart. Changes to this option are not applied dynamically.

The maximum number of items that the `<xi:include>` compiler cache can have.

max-compiler-cached-doc-size

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `10000000`
Valid values: non-negative integer
Value changes: Takes effect after restart. Changes to this option are not applied dynamically.

The maximum size, in bytes, allowed to cache the results of the compiled `<xi:include>` document.

max-includes

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `500`
Valid values: `0 to 10000`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum number of documents that may be included using `<xi:include>`.

For example:

```
scxml/max-includes = 200
```

max-preprocessor-cache-size

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `10000000`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the amount of memory, in bytes, that the `<xi:include>` pre-processor cache can use.

For example:

```
scxml/max-preprocessor-cache-size = 20000000
```

max-preprocessor-cached-docs

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `1000`
Valid values: Any integer greater than or equal to `0`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the maximum number of items that the `<xi:include>` pre-processor cache can have.

For example:

```
scxml/max-preprocessor-cache-docs = 2000
```

max-preprocessor-cached-doc-size

Option section: `scxml`
Configuration object: ORS `Application` object

Default value: `10000000`
Valid values: non-negative integer
Value changes: Takes effect after restart. Changes to this option are not applied dynamically.

The maximum size, in bytes, of the cached results of the preprocessed `<xi:include>` documents.

password

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: <empty string>
Valid values: String value
Value changes: During startup. Changes to this option are not applied dynamically.

This option specifies the SSL key password.

For example:

```
scxml/password = agent007
```

persistence-default

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `false`
Valid values: `true, false`
Value changes: Takes effect after restart. Changes to this option are not applied dynamically.

When set to `false`, all persistence capabilities with the SCXML engine are suppressed.

persistence-max-active

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `10000`
Valid values: `100 - 1000000`
Value changes: During startup. Changes to this option are not applied dynamically.

This option allows setup of a maximum number of active sessions that the SCXML engine will keep in memory.

For example:

```
scxml/persistence-max-active = 5000
```

session-processing-threads

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `8`
Valid values: Any integer greater than or equal to `1`
Value changes: During startup. Changes to this option are not applied dynamically.

This option sets the number of threads in the thread pool. The recommended value is two times the number of cores.

For example:

```
scxml/session-processing-threads = 16
```

system-id

Option section: `scxml`
Configuration object: ORS `Application` object
Default value: `-1`
Valid values: Any integer greater than or equal to `-1`
Value changes: During startup. Changes to this option are not applied dynamically.

This option allows setup of the Orchestration Server (ORS) system ID in order to have unique session IDs across ORS instances. If `-1` is specified, ORS creates a session ID based on the IP address of the host running ORS.

For example:

```
scxml/system-id = 11
```

x-server-trace-level

Option section: `log`
Configuration object: ORS `Application` object
Default value: `0`
Valid values: `0 - 3`
Value changes: as soon as committed to Configuration Server

This option specifies the level of tracing to be enabled for the Orchestration Server.

For example:

```
log/x-server-trace-level = 2
```

x-server-gcti-trace-level

Option section: `log`
Configuration object: ORS `Application` object

Default value: `0`
Valid values: `0 - 3`
Value changes: as soon as committed to Configuration Server

This option specifies the level of GCTI tracing to be enabled for the Orchestration Server. It controls how much detail should be in the logs for GCTI-related events, such as those from T-Server.

For example:

```
log/x-server-gcti-trace-level = 2
```

x-server-config-trace-level

Option section: `log`
Configuration object: ORS `Application` object
Default value: `0`
Valid values: `0 - 3`
Value changes: as soon as committed to Configuration Server

This option specifies the level of configuration tracing to be enabled for the Orchestration Server. It controls how much detail should be in the logs for Configuration-related events, such as reading from Configuration Server and reacting to dynamic changes.

For example:

```
log/x-server-config-trace-level = 2
```

x-print-attached-data

Option section: `log`
Configuration object: ORS `Application` object
Default value: `0`
Valid values: `0, 1`
Value changes: as soon as committed to Configuration Server

This option specifies whether or not attached data should be formatted and printed in the logs.

- `0`—Suppress printing attached data
- `1`—Format and print attached data

For example:

```
log/x-print-attached-data = 1
```

om-memory-optimization

Option section: `mcr`
Configuration object: ORS `Application` object
Default value: `true`
Valid values: `true, false`

Value changes: take effect immediately.

This option specifies whether memory optimization will be in effect for the current ORS. When the value is set to `true`, Orchestration Server will remove passive multimedia interactions from memory cache.

For example:

```
mcr/om-memory-optimization = true
```

om-max-in-memory

Option section: `mcr`
Configuration object: ORS `Application` object
Default value: `100`
Valid values: `1 to 2000`
Value changes: take effect immediately.

This option specifies a maximum number of interactions to store in memory cache before interactions will begin to be removed from the cache (oldest first), when `om-memory-optimization` is set to `true`. The value is in *thousands*, so that the default value of `100` represents 100,000 interactions that are to be stored in memory cache. The maximum value for this option is `2000`, which represent 2,000,000 interactions..

For example:

```
mcr/om-max-in-memory = 100
```

om-delete-from-memory

Option section: `mcr`
Configuration object: ORS `Application` object
Default value: `1`
Valid values: `1 to 2000000`
Value changes: take effect immediately.

This option specifies how many calls should be deleted when the value of `om-max-in-memory` has been attained.

For example:

```
mcr/om-delete-from-memory = 1
```

name

Option section: `cluster`
Configuration object: ORS `Application` object
Default value:<empty string?
Valid values: Any valid string.
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option provides the string name of the Orchestration Server cluster that this Orchestration Server application (node) belongs to.

All Orchestration Server applications within the same cluster will have the same value for this option. For example, if there are three Orchestration Server applications installed, to put them all into a single cluster, specify for the all of them the option:

```
cluster/name = routing_cluster
```

super_node

Option section: `cluster`
Configuration object: ORS `Application` object
Default value: `false`
Valid values: `true, false`
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option, if set to `true`, specifies that this node should act as a Super Node in the cluster. Refer to "Cluster Deployment" on page 52 in Chapter 3: "Persistence, High Availability, and Load Balancing".

Note: When configuring and deploying clustering, carefully select how many and which nodes will be assigned as Super Nodes. The number of Super Nodes should not be too small (one, for example) or too large (hundreds, for example).

```
cluster/super_node = true
```

hostname

Option section: `web_services`
Configuration object: ORS `Application` object
Default value:<empty string>
Valid values: Any valid string.
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option provides the string for a hostname or IP address to identify the host for a particular web service that the Web Services functional module will use.

```
web_services/hostname = http://www.myurl.com/webservice_host
```

port

Option section: `web_services`
Configuration object: ORS `Application` object
Default value: <empty string>

Valid values: Any positive integer less than 65535.
Value changes: take effect during startup. Changes to this option are not applied dynamically.

This option provides the value for the port associated with the hostname or IP address identified in the `hostname` option. Use any unassigned port value integer less than `65535`.

```
web_services/hostname = http://www.myurl.com/webservice_host
```

application

Option section: `Orchestration`
Configuration object: `DN` (`Extension` or `RoutePoint`), or `Interaction Queue`
Default value: `none` (this is a required option)
Valid values: `any valid URL`
Value changes: For the next interaction that hits this resource

This option specifies the URL of the SCXML document to load. The URL can be any one of the following protocols:

- file:<path>
- http://<url>
- script:<name of script object >

The use of `script:` allows an indirect reference to a script object of type `CfgEnhancedRouting`, which can contain the application URL, parameters, and other configuration values. The URL can also contain *parameters* which will be passed to the Application server. The values shown in Table 6 on page 125 are substituted at run-time based on the information in the interaction.

Table 6: URL Parameter Elements for application option

Formatting Element	Description
[DNIS]	The `DNIS` attribute of the interaction
[ANI]	The `ANI` attribute of the interaction
[DN]	The `ThisDN` attribute of the interaction
[CED]	The `CollectedDigits` attribute of the interaction
[EMAILFROM]	E-mail from address
[EMAILTO]	E-mail to address
[EMAILSUBJECT]	E-mail subject

Table 6: URL Parameter Elements for application option (Continued)

Formatting Element	Description
[UDATA]	Expanded to the entire user data of the interaction in the format: `name1=value1&name2=value2&·`
[UDATA.*]	Expanded to the entire user data of the interaction in the format: `name1:value1,name2:value2,·`

[UDATA.name]	Expanded to the value of a specific user data key of the interaction value
	For example:
	`&servicetype=[UDATA.ServiceType]`
	could resolve to:
	`&servicetype=CreditCards`

For example:

```
application = http://xserver.genesyslab.com:80/NewCallReq.asp

application=
http://xserver.genesyslab.com:80/NewCallReq.asp?ani=[ANI]&dnis=[DNIS]&s
ervicetype=[UDATA.ServiceType]

application = script:orsscript
```

Users can provide an alternate URL utilizing the following rules.

The value of the `application` section can be done as follows:

`application = <URL1><single space><URL2>`

For example:

```
application = http://host1/RouteToDN1.scxml http://host1/RouteToDN2.scxml
```

The same is applicable for file:<path>. A single space is used between the two file paths.

alternate-url

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `none` (This option is not required)
Valid values: any valid URL
Value changes: For the next interaction that hits this resource

This option specifies the URL of the SCXML document to load. This is attempted if the value of the `url` option cannot be fetched or compiled (for example: application server is temporarily down, network error, script is malformed, and so on).

The URL can be any one of the following protocols:

- file:<path>
- http://<url>
- https://<url>

The URL can also contain *parameters* which will be passed to the Application server. The values shown in Table 8 are substituted at run-time based on the information in the interaction.

Table 7: URL Parameter Elements for alternate-url option

Formatting Element	Description
[DNIS]	The `DNIS` attribute of the interaction
[ANI]	The `ANI` attribute of the interaction
[DN]	The `ThisDN` attribute of the interaction
[CED]	The `CollectedDigits` attribute of the interaction
[EMAILFROM]	E-mail from address
[EMAILTO]	E-mail to address
[EMAILSUBJECT]	E-mail subject
[UDATA]	Expanded to the entire user data of the interaction in the format: `name1=value1&name2=value2&·`
[UDATA.*]	Expanded to the entire user data of the interaction in the format: `name1:value1,name2:value2,·`

[UDATA.name]	Expanded to the value of a specific user data key of the interaction value
	For example:
	`&servicetype=[UDATA.ServiceType]`
	could resolve to:
	`&servicetype=CreditCards`

Examples Simple case:

```
alternate-url = http://xserver.genesyslab.com:80/NewCallReq.asp
```

With parameters:

```
alternate-url =
https://xserver.genesyslab.com:80/NewCallReq.asp?ani=[ANI]&dnis=[DNIS]&
servicetype=[UDATA.ServiceType]
```

fetch-timeout

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `5000` [5000 ms is 5 seconds]
Valid values: Any integer greater than or equal to `0`
Value changes: For the next document fetch

This option specifies the time (in milliseconds) before a document fetch is abandoned. If the SCXML document cannot be retrieved within this timeout value, the fetch will be abandoned and a fetch for the alternate-url will be attempted.

The actual fetch waiting time can be up to 10 ms more than specified. If `fetch-timeout` is `0` or is not specified, the system will wait indefinitely for the document to be fetched.

For example:

`Application/ fetch-timeout = 0` (no timeout)

`Application/ fetch-timeout = 500` (wait up to 500 ms to fetch the document)

`Application/ fetch-timeout = 60000` (wait up to 1 minute)

http-useragent

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `GOES/8.1`
Valid values: Any string
Value changes: For the next document fetch

This option specifies the string to use in the HTTP header field `User Agent`, which identifies the application to the web server.

For example:

```
http-useragent = MYAGENT
```

http-version

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `1.1`
Valid values: `1.0, 1.1`
Value changes: For the next document fetch

This option specifies the HTTP version to use when fetching documents from the application server.

For example:

```
http-version = 1.1
```

max-age

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `0`
Valid values: Any integer greater than or equal to `0`
Value changes: For the next document fetch

This option is an HTTP Cache-Control value which tells the Orchestration Server how long an application script can be cached. If another interaction causes the same URL to be fetched, and it is within the `max-age` value specified, the cached version will be used instead of fetching a new version from the Application server.

Note: The application can only be cached if the URL is the same for a particular cached version. If application parameters are specified which are unique for each invocation (for example, `ani=[ANI]` parameter), the application will be fetched each time regardless of this value.

The value of `max-age` is the time to cache *in seconds*. If `max-age` is `0` or is not specified, the system will not cache this application script.

For example:

`Application/ max-age = 0` (no caching)

`Application/ max-age = 1000` (application contents can be cached for 1 second)

`Application/ max-age = 60000` (application contents can be cached for 1 minute)

max-duration

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `2000`
Valid values: `1 to 10000`
Value changes: For the next document fetch

This option sets the maximum time, in milliseconds, that a single ECMAScript script can take. Applies to `<script>`, `expr` or `cond`. For `max-loop-count` and `max-duration`, if the limit is exceeded, the script is aborted and an error event is fired.

The Orchestration Server will attempt to terminate hung sessions gracefully. If they do not respond within the `session-hung-timeout` time, they will be terminated.

For example:

```
max-duration = 5000
```

max-loop-count

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `5000`
Valid values: `1 to 100000000`
Value changes: For the next document fetch

This option sets the maximum number of loops that can be performed in a single ECMAScript script. This applies to `<script>`, `expr` or `cond`. For `max-loop-count` and `max-duration`, if the limit is exceeded, the script is aborted and an error event is fired.

The Orchestration Server will attempt to terminate hung sessions gracefully. If they do not respond within the `session-hung-timeout` time, they will be terminated. See `session-hung-timeout` for more information.

For example:

```
max-loop-count = 10000
```

max-stale

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `0`
Valid values: Any integer greater than or equal to `0`
Value changes: For the next document fetch

This option determines the number of seconds to extend the life of a cached file. If the cached file would have expired 120 seconds ago, but `max-stale` is set to `300`, the local cached file will be sent back to the platform without first verifying the status of the file from the Application server.

If `max-stale` is `0` or is not specified, the system will not cache this application.

For example:

`Application/ max-stale = 0` (no caching)

`Application/ max-stale = 1000` (application contents can be 1 second old before verifying the status from the Application server)

url

Option section: `Application`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `none` (this is a required option)
Valid values: `any valid URL`
Value changes: For the next interaction that hits this resource

This option specifies the URL of the SCXML document to load. The URL can be any one of the following protocols:

- file:<path>
- http://<url>

The URL can also contain *parameters* which will be passed to the Application server. The values shown in Table 8 are substituted at run-time based on the information in the interaction.

Table 8: URL Parameter Elements for url option

Formatting Element	Description
[DNIS]	The `DNIS` attribute of the interaction
[ANI]	The `ANI` attribute of the interaction
[DN]	The `ThisDN` attribute of the interaction
[CED]	The `CollectedDigits` attribute of the interaction
[EMAILFROM]	E-mail from address
[EMAILTO]	E-mail to address
[EMAILSUBJECT]	E-mail subject
[UDATA]	Expanded to the entire user data of the interaction in the format: `name1=value1&name2=value2&·`

Table 8: URL Parameter Elements for url option (Continued)

Formatting Element	Description
[UDATA.*]	Expanded to the entire user data of the interaction in the format: `name1:value1,name2:value2,` ·

[UDATA.name]	Expanded to the value of a specific user data key of the interaction value
	For example:
	`&servicetype=[UDATA.ServiceType]`
	could resolve to:
	`&servicetype=CreditCards`

For example:

```
url = http://xserver.genesyslab.com:80/NewCallReq.asp

url =
http://xserver.genesyslab.com:80/NewCallReq.asp?ani=[ANI]&dnis=[DNIS]&s
ervicetype=[UDATA.ServiceType]
```

{Parameter Name}

Option section: `ApplicationParms`
Configuration object: Script object (`CfgEnhancedRouting`)
Default value: `none`
Valid values: Any string
Value changes: For the next interaction that hits this resource

This option specifies a string that represents a parameter value to be passed to the application.

The `ApplicationParms` section contains the values for data elements that may be referred to within the SCXML application. The parameters can be statically defined for each application (for example, `Service = Sales`) or contain substitution values that will be substituted at run-time based on the information in the interaction, as specified in Table 9 on page 132, for example:

```
Segment = [UDATA.CustomerSegment]
```

Table 9: Parameter Elements for ApplicationParms

Formatting Element	Description
[DNIS]	The `DNIS` attribute of the interaction
[ANI]	The `ANI` attribute of the interaction

Table 9: Parameter Elements for ApplicationParms (Continued)

Formatting Element	Description
[DN]	The `ThisDN` attribute of the interaction
[CED]	The `CollectedDigits` attribute of the interaction
[EMAILFROM]	E-mail from address
[EMAILTO]	E-mail to address
[EMAILSUBJECT]	E-mail subject

[UDATA.name]	Expanded to the value of a specific user data key of the interaction value
	For example:
	`[UDATA.ServiceType]`
	could resolve to:
	`"CreditCards"`

For example:

```
ApplicationParms =
                Service = Sales
                EmailSubject = [EMAILSUBJECT]
                Segment = [UDATA.CustomerSegment]
                AppServer = http://myappserver:8080[1]
```

Operational Reporting

Operational Reporting is a new feature of Orchestration Server that is available beginning with release 8.1.0. It provides Orchestration with the capability to inform administrators about the SCXML session workload that is being serviced throughout the Orchestration deployment. You can see what types of sessions are being run, how they are distributed, how many active sessions there are at a given point in time, and so on. Additionally, the solution allows you to search for sessions that match specific criteria. This feature is primarily used for monitoring *active* sessions. An active session is defined as any Orchestration session that has not reached its final state.

1.Universal Resource Locators (URL) that are passed in as parameters are not resolved by the Orchestration Server. They are passed to the application as a string. If necessary, the application can retrieve the value that is associated with the URL by using the `<fetch>` operation.

The Operational Reporting feature of ORS is implemented as a new mode of operation within the Orchestration Server.

Configuring Operational Reporting

Operational Reporting runs within the Genesys Administrator 8.1 graphical user interface (GUI) as an option within the `Monitoring` tab. Details about accessing and running the Operational Reporting interface are provided within the *Framework 8.1 Genesys Administrator Help* file.

In order for Operational Reporting to function, you must configure all Super node within your Orchestration deployment with Reporting ports.

Note: Adding or updating the Reporting ports (as described in the procedures that follow) leads to ORS stopping and restarting.

You must have completed the steps and procedures that were described earlier in this chapter before attempting to configure a Super node:

Note: The term *Master Super node* is used in this section, however, it is not easily identified. To ensure a Master Super node is configured in a cluster, Genesys recommends that you configure all Super nodes with Reporting ports. Any one of the configured Super nodes in a cluster might become the Master Super node (although it will not be apparent).

Follow one of the detailed procedures that follows:

Configuring a Super Node for Operation Reporting

You can use the `Provisioning` tab within Genesys Administrator or you can use Configuration Manager to configure an Orchestration Server application to function as a Super node (Master Super node in a cluster).

Use one of the following two procedures to perform this configuration.

Procedure: Configuring a Super node for Operational Reporting Using Genesys Administrator

Start of procedure

1. Log in to Genesys Administrator with your usual credentials.
2. Select the `Provisioning` tab if it is not already selected.

3. Select the Environment bar in the Navigation pane, and click Applications to see a list of available applications and application folders.
4. Navigate to the Orchestration Server instance/node that you will configure for Operational Reporting.
5. Double-click the name of the Orchestration Server application that you are configuring for Operational Reporting.
6. On the Configuration tab, open the Server Info section by clicking the down arrow beside Server Info.
7. Scroll to Listening Ports and click Add.
8. On the Port Info dialog box's General tab, enter rts_ipc in the Port ID field.

   ORS will use the rts_ipc port to identify the port for Member node connectivity.
9. Enter any unique port value in the Port field and click OK.
10. Click Add again within Listening Ports.
11. On the Port Info dialog box's General tab, enter rts_client in the Port ID field.

    ORS will use the rts_client port to identify the port for Operational Reporting interface (client) connectivity (Genesys Administrator).
12. Enter any unique port value in the Port field and click OK.
13. Click Save or Save & Close to save the port definitions for this ORS application or node.
14. Click OK to save.

End of procedure

This ORS application or node will now function as a Super node for Operational Reporting. Refer to the *Framework 8.1 Genesys Administrator Help* for details on how to access and use Operational Reporting within Genesys Administrator.

Procedure: Configuring a Super node for Operational Reporting Using Configuration Manager

Start of procedure

1. Log in to Configuration Manager with your usual credentials.
2. Open the Environment list in the Navigation pane, and click Applications to see a list of available applications and application folders.

3. Navigate to the Orchestration Server instance/node that you will configure configure for Operational Reporting.
4. Double-click the name of the Orchestration Server application you are configuring for Operational Reporting.
5. Click the `Server Info` tab.
6. Click the `Add Port` button.
7. On the `New Port Info` dialog box's `Port Info` tab, enter `rts_ipc` for the Port ID.

   ORS will use the `rts_ipc` port to identify the port for Member node connectivity.
8. Enter any unique port value in the `Communication Port` field and click `OK`.
9. Click `Add Port` again.
10. On the `New Port Info` dialog box's `Port Info` tab, enter `rts_client` for the Port ID.

    ORS will use the `rts_client` port to identify the port for Operational Reporting interface (client) connectivity (Genesys Administrator).
11. Enter any unique port value in the `Communication Port` field and click `OK`.
12. Click `OK` to save the port definitions for this ORS application or node.

End of procedure

This ORS application or node will now function as a Super node for Operational Reporting. Refer to the *Framework 8.1 Genesys Administrator Help* for details on how to access and use Operational Reporting within Genesys Administrator.

Accessing and Using the Operational Reporting Interface

The user interface for Orchestration's Operational Reporting runs inside of Genesys Administrator release 8.1 or later. Operational Reporting is available as a *Monitoring* function in Genesys Administrator, and is accessed from the `Monitoring` tab.

The *Framework 8.1 Genesys Administrator Help* system contains information on how to access and use the Operational Reporting function. In the Help file's Table of Contents, open `Monitoring Your Environment`, and then open `Orchestration` and review the Help topics before attempting to run Operational Reporting.

Chapter

6 Business Logic for Advice of Charge

This chapter describes how Orchestration Server supports the Business Logic for Advice of Charge (AoC) and includes the following topic:

- Implementation of Business Logic for Advice of Charge, page 137

Implementation of Business Logic for Advice of Charge

Orchestration Server supports the SIP Server Advice of Charge (AoC) feature by implementing the business logic to insert charge messages. This feature enables SIP Server to act as a Charging Determination Point (CDP) to specify the charges for using a service. SIP Server sends the CDP data to a Charging Generation Point (CGP) in Secure SIP (SIPS) `INFO` messages action. The charge information is initiated by using the `PrivateServiceRequest` action. Functionality for this feature is based on the 3GPP Specification, TS 32.280. The Private Service function is described in detail in the

<privateservice> Request

Used in conjunction with SIP Server 8.1.0, the `<privateservice>` request enables users to implement the Advice of Charge (AoC) feature in their solution. This action enables an application to pass data and request services that are supported only by certain T-Servers and are not covered by general feature requests. Request services can include Set Feature, AoC, change T-Server behavior, and so on. The `<privateservice>` request is equivalent to the T-Server `TPrivateService` function. See the applicable T-Server documentation for information about the request `TPrivateService` function.

Attribute Details

Table 10 contains the Private Service request attributes and their descriptions. There are no default values for any of these attributes.

Table 10: Private Service Request Attributes

Attribute name	Type	Valid Values	Description
`requestid` (not required)	Location expression	Any valid location expression	Represents the location of the request ID that is returned in the request. Any data model expression evaluating to a data model location. The location's value is configured as an internally generated unique string identifier that is associated with the action that is sent. If this attribute is not specified, the event identifier is dropped. This identifier can be tested by the completion event handler to distinguish between multiple outstanding requests. If this attribute is not specified, the identifier can be acquired from the fetch completion event. Every request must receive a unique identifier.
`serviceid` (required)	Value expression	Any value expression that returns a valid integer	A value expression that returns an integer to indicate the type of information that is passed or the service that is requested. It is specific to the T-Server that is handling the call. Before you configure this value, check the T-Server documentation for your switch.
`interactionid` (required)	Value expression	Any value expression that returns a valid integer	A value expression that returns the `_genesys.FMname.interactions[x].g_uid` interaction ID that is associated with this request. Non voice interactions can result in the generation of an `error.voice.privateservice` error.
`resource` (not required)	Value expression	Any valid string or resource object	A value expression that returns the DN of the controlling agent or route point for which the information is provided. This attribute corresponds to the `thisDN` parameter within the TLib `TPrivateService` method. Before you configure this value, check the T-Server documentation for your switch. The `<ixn:privateservice>` resource attribute is mandatory in this release.
`udata` (not required)	ECMAScript object	Any valid ECMAScript object	An ECMAScript object that contains the list of key/value pairs are attached to the call in question.

Table 10: Private Service Request Attributes (Continued)

Attribute name	Type	Valid Values	Description
`reasons` (not required)	ECMAScript object	Any valid ECMAScript object	An ECMAScript object that contains the list of key/value pairs that provide additional information associated with this Private Service request, and is intended to specify reasons for and results of actions taken by the user.
`extentions` (not required)	ECMAScript object	Any valid ECMAScript object	An ECMAScript object that contains the list of key/value pairs that provide an additional data structure that is intended to take into account the switch-specific features that cannot be described by other parameters, or the list of key/value pairs in the original structure of the user data that is associated with this Private Service request.
`hints` (not required)	Value expression	Any valid ECMAScript object	A value expression that returns the ECMAScript object that contains information that might be used by the implementing functional module while performing this action. This information might consist of protocol-specific parameters, protocol selection guidelines, or other related data. The meaning of these hints is specific to the implementing functional module.
Note: For more information about the expressions and data values in this table, see the *State Chart XML (SCXML): State Machine Notation for Control Abstraction, W3C Working Draft 26 April 2011* W3C Specification.			

Determining the Target T-Server

The target T-Server to which the Private Service request is submitted is determined by the following factors:

- If the resource attribute contains both a switch and DN, the switch is used to locate the T-Server to which the request is submitted.
- If the resource attribute contains only the switch, the switch is used to locate the T-Server to which the request is submitted and the thisDN parameter value of the underlying TLib TPrivateService method is not populated.
- If the resource attribute contains only a DN, the switch and associated T-Server are determined by the interaction ID. The switch is determined, based on the first party that references the DN resource.
- If the resource is not provided, the T-Server is determined by the last party within the associated party entries for the associated interaction. In this case, the target T-Server does not provide a resource (thisDN parameter).

If the target T-Server cannot be determined by using the provided information, an `error.voice.privateservice` error is generated. See the following example:

```
<state id="do_private_service">
<datamodel>
<data id="reqid"/>
</datamodel>
<onentry>

          <script>
               var myuserdata = {details : {name: "Smith, John", age
: 45} };
                var myreasons = {code: "New Update"};
                var myextensions = { keyname : "Its value"};
                var myhints = {handle_responses : "false"};
          </script>

          <ixn:privateservice requestid="_data.reqid"
          serviceid="1234"
interactionid="_genesys.ixn.interactions[0].g_uid"
          resource="'9000'"
          udate = "myuserdata"
          reasons = "myreasons"
          extensions = "myextensions"/>

</onentry>
<transition event="voice.privateservice.done" target="statex"/>
<transition event="error.voice.privateservice" target="statey"/>
</state>
```

Children

There are no child objects.

Events

The following events can be generated as part of this action, but are specific to the service and T-Server implementation. Therefore, Genesys recommends that you refer to the appropriate T-Server manual for information about how and when to generate them.

- `voice.privateservice.done`—This event is sent when the request is accepted and sent by Orchestration Server. It is not an indication that the T-Server handled or accepted the event.
- `error.voice.privateservice`—This event is sent if, for any reason, the request itself fails.

Chapter

7 SCXML Application Support

This chapter contains the following topics:

- Creating SCXML-Based Applications, page 141
- Deploying, page 143
- Samples, page 144

Creating SCXML-Based Applications

You can create SCXML-based applications by using the following methods:

- Any simple text editor such as Notepad or an XML-based editor, with which you are already comfortable.
- The Genesys Composer GUI, which has both an SCXML text editor view and a graphical editor view. Here you place, connect, and configure blocks similar to IRD's strategy-building objects. For more information on using this GUI, consult the *Genesys Composer 8.1*.x *Help*. Also see the *Genesys Composer 8.1 Deployment Guide*.

Genesys Composer

Genesys Composer has a drag and drop visual designer for creating workflows when building SCXML-based applications (see Figure 24).

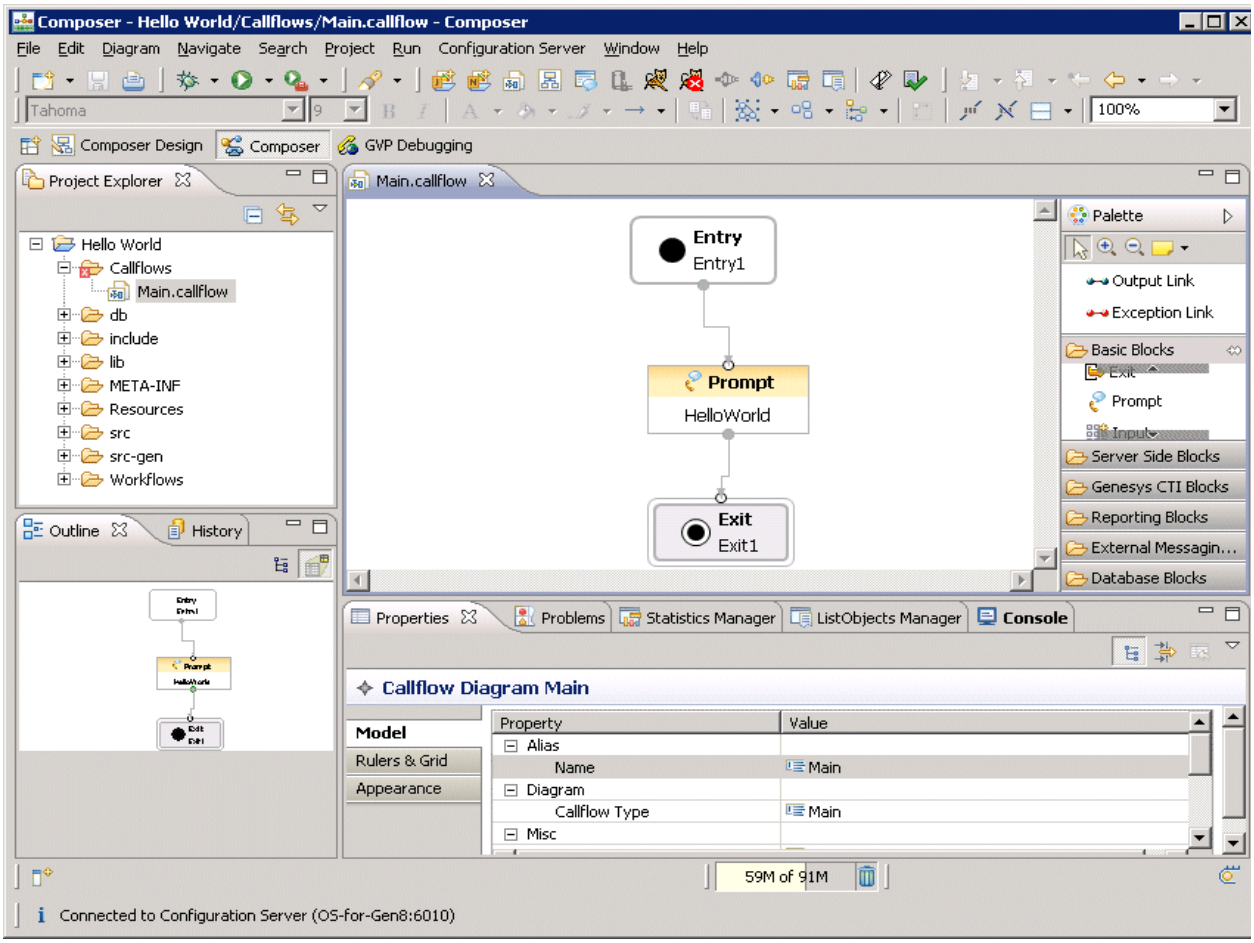

Figure 24: Composer

Composer supports Inbound Voice and some eServices functions.

Routing strategy developers use an approach where they first define a visual workflow model and then generate SCXML code from it. Composer features include:

- A “palette” of blocks (similar to IRD’s strategy-building objects) for generating SCXML.
- Sample workflows and call flows (templates) are provided for quick start and rapid development.
- Extensive validation checking to prevent users from making mistakes while setting the block properties.
- Error and warning markers in the workflow designer area that help in speedy resolution of the problem(s).

Composer Code Generator

The workflow designer area provides a code generator for generating SCXML code. A single static SCXML page is generated per workflow. The generated code can act as a learning tool for those developers who prefer to write their own SCXML applications. The code can be re-used to quickly generate new SCXML pages and then hand-edited using the SCXML editor. A template library is provided and developers can add their own custom snippets to the template library.

Composer Testing

Developers can use Composer to test their SCXML applications. Support for `Run` mode is provided. The SCXML testing function opens a direct TCP connection with URS for simulating the call and interacting with the URS. For routing decisions like target selection, Composer provides choices to the developer for simulating calls.

Both IIS and Tomcat are bundled as part of Composer. No configuration in Configuration Server is needed for simulating calls. For making live calls, the Route Point must be provisioned manually in Configuration Manager or Genesys Administrator.

Deploying

Composer does not yet support automatic deployment of routing applications to an Application Server. However, for purposes of testing, you can use one of the following methods:

- Manually copy files to a folder on the Application Server (for IIS).
- Use Composer's `Run as Call Flow` option.

Procedure: Manual deployment to an Application Server (IIS)

Start of procedure

To deploy manually deploy an SCXML-based application (created inside or outside of Composer) if Microsoft Internet Information Services is installed and running on the computer:

1. Navigate to `C:\Inetpub\wwwroot` folder.
2. Copy files with extensions `.xml` or `.scxml` to that folder.

End of procedure

Samples

Orchestration Server supports only SCXML documents. The *Genesys 8.1 SCXML Samples* document contains numerous examples of SCXML routing applications. The samples are not designed for use in a Production environment. Instead, use them to get started configuring your own applications, subroutines, and list objects. Consider them as guides when developing your own objects adjusted to your company's specific business needs.

Note: The *Genesys 8.1 SCXML Samples* provides examples of various types of voice and eServices SCXML routing applications. The information includes application flows and the properties of the various application-building objects. If you need an example of how to use application-building objects, start with this guide.

Chapter

8 Installing Orchestration Server

Before installing Orchestration Server, you must:

- Install and configure the required Framework components: T-Server, Stat Server, and the Configuration Layer and Management Layer components.
- Configure the Orchestration Server as described in Chapter 5 on page 63.

Once the above tasks are done, you are ready to install the configured Orchestration Server as described in this chapter.

This chapter includes the following topics:

- Installation Package Location, page 146
- Installing on Windows Operating Systems, page 146
- Installing on UNIX-Based Platforms, page 150

Note: For a list of supported operating systems and databases, see the *Genesys Supported Operating Environment Reference Manual* which is available on the Genesys Technical Support web site at `http://genesyslab.com/support`.

Warning! Genesys does not recommend installation of its components via a Microsoft Remote Desktop connection. The installation should be performed locally.

Task Summary: Installing Orchestration Server

Objective	Related Procedures and Activities
Locate the installation package	See "Installation Package Location" on page 146
Install components on Windows	Procedure: Installing Orchestration Server on Windows using the Installation Wizard, on page 147
Install components on Linux	Procedure: Installing Orchestration Server on a UNIX platform, on page 151

Installation Package Location

The installation package, whether on CD or from an FTP site, contains a setup folder for Orchestration Server.

When FTP delivery is used, there are separate setup folders for Windows and Linux.

Note: If you install several instances of Orchestration Server component on the same computer, a separate shortcut is created for each one, based on the `Application` name stored in Configuration Layer.

Installing on Windows Operating Systems

The installation process does not present the option of installing a server component as a service. By default, starting with 7.5, all server components (excluding Genesys Desktop and Multimedia/eServices components) are installed as services in automatic startup mode.

If you wish to use Management Layer and SCI, you must also install LCA on the Orchestration Server host computer as documented in the *Framework 8.1 Deployment Guide*.

Procedure:
Installing Orchestration Server on Windows using the Installation Wizard

Start of procedure

Note: The Orchestration Server `Application` object must already be configured before you begin the installation.

1. Double-click `setup.exe`.
   - If Orchestration Server was downloaded from an FTP site, the file is located in the download directory.

   The Install Shield opens the `Welcome` screen.
2. Click `Next`. The `Connection Parameters to the Configuration Server` screen appears (see Figure 25).

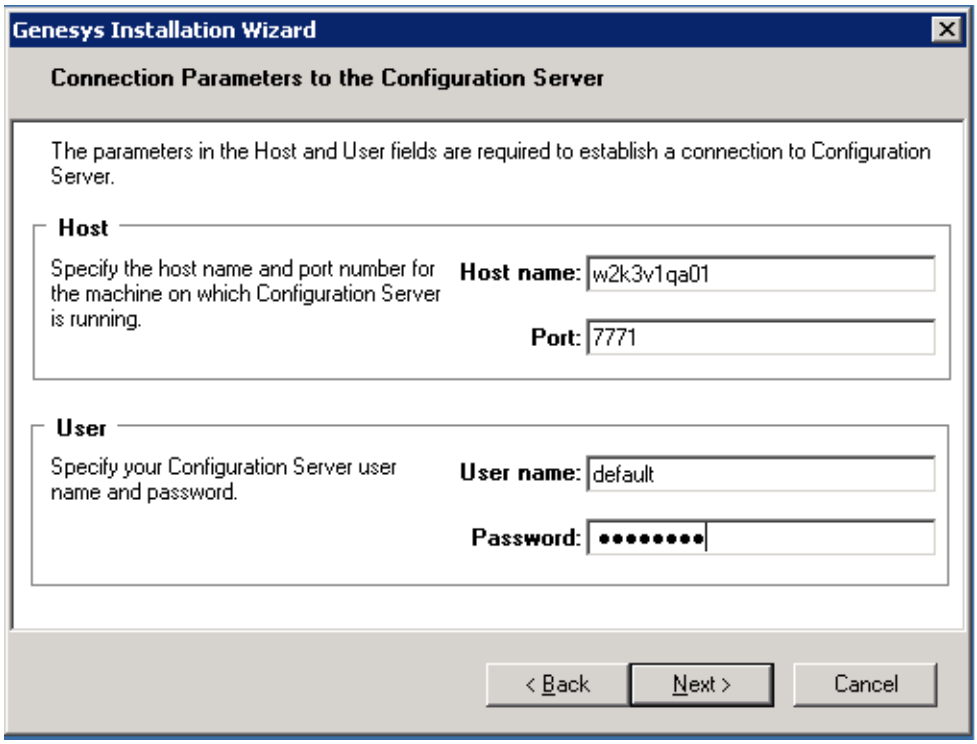

Figure 25: Connection Parameters to the Genesys Configuration Server

3. Under `Host`, specify the host name and port number for the computer on which Configuration Server is running. This is the main "listening" port entered in the `Server Info` tab for Configuration Server, which is used for authentication in the Configuration Manager login dialog box.
4. Under `User`, enter the user name and password used for logging on to Configuration Server.

5. Click `Next` to open the `Client Side Port Configuration` screen. (see Figure 26).

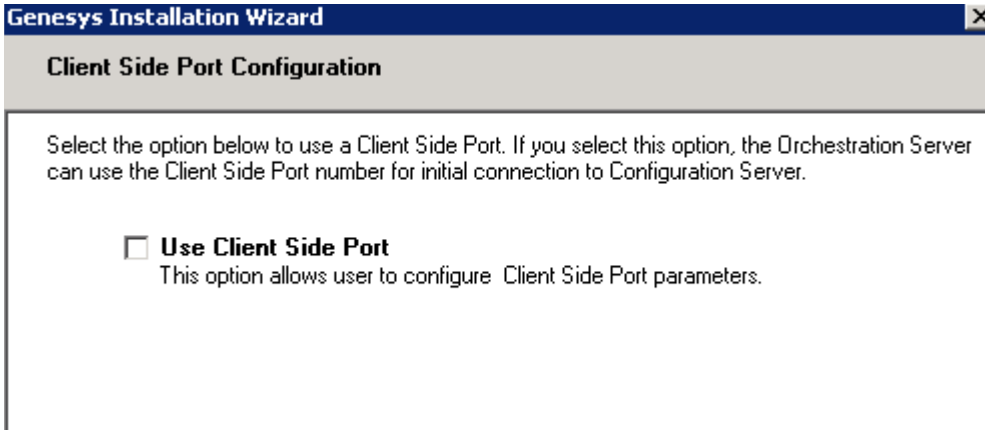

Figure 26: Client-Side Port Configuration

6. If you are setting up client-side port configuration for the initial connection to Configuration Server as described in the *Genesys 8.1 Security Deployment Guide*, select the `Use Client Side Port` check box to revel additional fields (see Figure 27) You also set up the client-side port when using the RESTful interface (refer to "Load Balancing for RESTful Interface" on page 54).

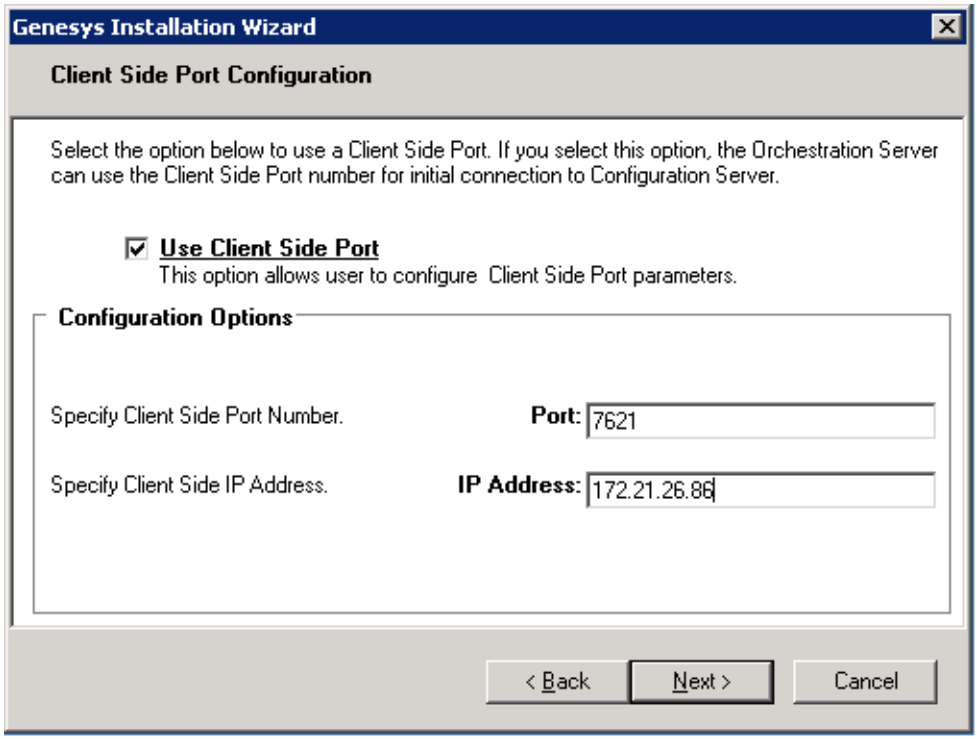

Figure 27: Client-Side Port Configuration Screen, Configuration Options

7. Specify the following parameters:

- `Port`—Enter any free port number (this is *not* the `Listening` port in the `Server Info` tab of the Orchestration Server `Application` object).
- `IP Address`—Enter the IP Address of the computer on which you are installing and running the Orchestration Server `Application`.

Note: After entering this information, the installation process will add the necessary command line arguments (`-transport-address` and `-transport-port`) for connecting to Configuration Server during Application startup.

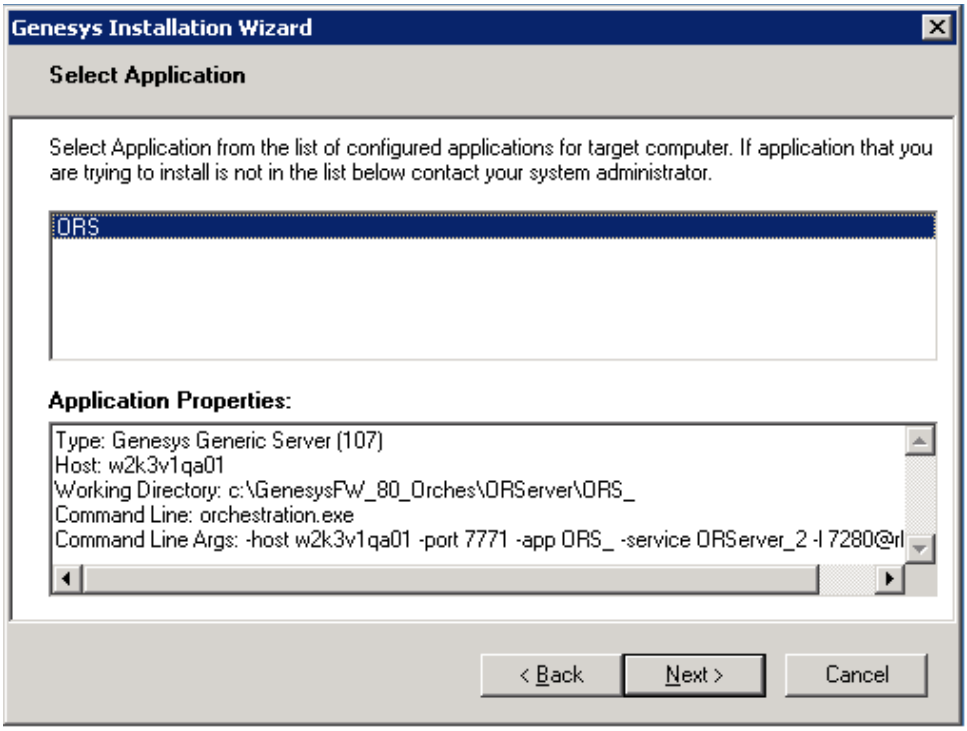

Figure 28: Select Application

8. Click `Next`. The `Select Application` screen appears. Figure 28 shows example entries.
9. On the `Select Application` screen, select the Orchestration Server `Application` that you are installing. The `Application Properties` area shows the `Type`, `Host`, `Working Directory`, `Command Line` executable, and `Command Line Arguments` information previously entered in the `Server Info` and `Start Info` tabs of the selected Orchestration Server `Application` object.
10. Click `Next`. The `Choose Destination Location` screen appears (see Figure 29).

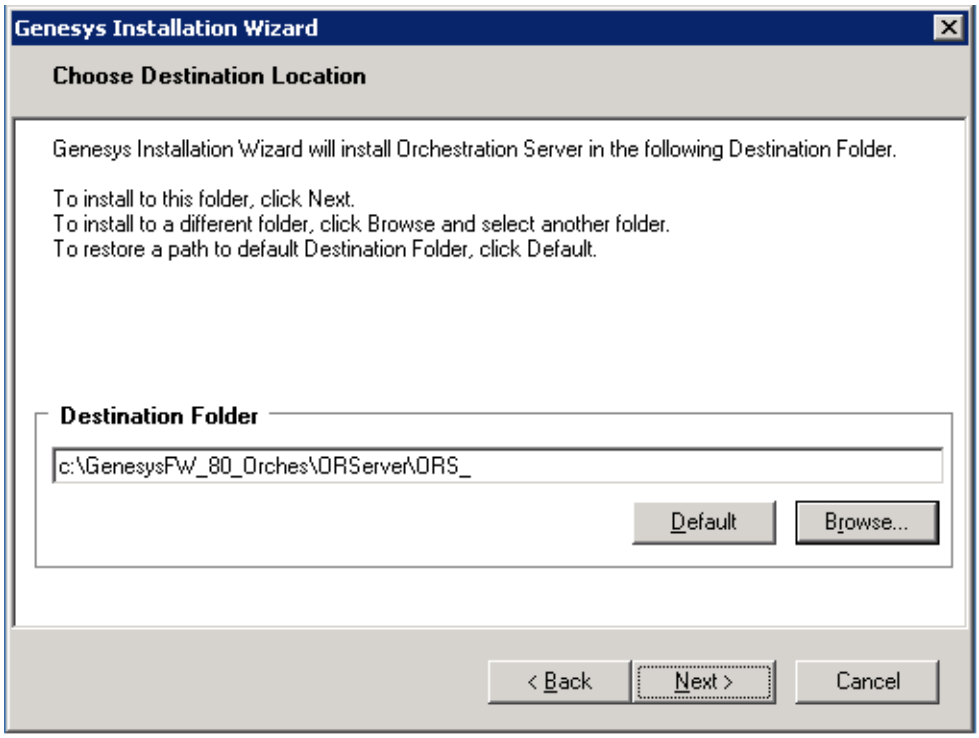

Figure 29: Choose Destination Location

11. Under `Destination Folder`, keep the default or browse for the installation location for Orchestration Server.
12. Click `Next`. The `Ready to Install` screen appears.
13. Click `Next` on the `Ready to Install` screen. The Genesys Installation Wizard indicates it is performing the requested operation for Orchestration Server. When through, the `Installation Complete` screen appears.
14. Click `Finish` on the `Installation Complete` screen.

End of procedure

Installing on UNIX-Based Platforms

Warning! Before you start the installation, make sure all instances of Orchestration Server are already installed on your computer and shut down. If you do not do this, you will not be able to back up your files if you want to use the same installation directory for another version of those components.

Installing Orchestration Server on a UNIX-Based Platform

Complete the procedures in this section to install Orchestration Server on a UNIX platform. This release of Orchestration Server supports:

- Red Hat Enterprise Linux AS 4
- Red Hat Enterprise Linux AS 5

Procedure: Installing Orchestration Server on a UNIX platform

Start of procedure

1. Go to the directory where the installation is created.
2. Copy all files to a temporary directory.

   Note: Files included in the installation package require permission to execute.

3. Run the installation script by typing `./install.sh` (see "Installation Package Location" on page 146).
4. When prompted, enter the host name of the computer where Orchestration Server will be installed or press the `Enter` key for the supplied entry.
5. When prompted, enter the following information about your Configuration Server:
   - `Configuration Server Host name`
   - `Network port`
   - `User name`
   - `Password`
6. Prompts appear regarding securing connections between Orchestration Server and Configuration Server.

   ```
   Client Side Port Configuration
   Select the option below to use a Client Side Port. If you select
   this option, the application can use Client Side Port number for
   initial connection to Configuration Server.
   Do you want to use Client Side Port option (y/n)
   ```

7. When prompted, type either `Y` for yes or `N` for no. The instructions below assume you typed `Y`.
8. Enter an IP address or press `Enter` for the supplied entry after the following prompt:

   ```
   Client Side IP Address (optional), the following values can be used:
   ```

9. Choose the Orchestration Server `Application` to install after this prompt (which may list several Orchestration Servers):

```
Please choose which application to install:
1 : <ORS_application>
```

10. Enter the destination directory for the installation after this prompt:

    ```
    Please enter full path of the destination directory for
    installation:
    ```

    After you enter the destination directory, the installation continues. A message appears that starts with `xtracting tarfile: d`

11. When this instruction appears:

    ```
    There are two versions of this product available: 32-bit and 64-bit.
    Please enter 32 or 64 to select which version to use,
    ```

    enter `32` or `64` according to the Linux type that you use.

End of procedure

As soon as the installation process is finished, a message appears announcing that installation was successful. The process created a directory, with the name specified during the installation, containing Orchestration Server.

Note: If you wish to use Management Layer and SCI, you must also install LCA on the Orchestration Server host computer as documented in the *Framework 8.1 Deployment Guide*.

Chapter

9 Starting and Stopping Procedures

This chapter provides instructions for starting and stopping Orchestration Server (ORS) either with the Solution Control Interface (SCI), or manually.

This chapter includes the following topics:

- Prestart Information, page 153
- Starting Orchestration Server, page 154
- Stopping, page 156
- Non-Stop Operation, page 157
- Version Identification, page 158

Prestart Information

Before Orchestration Server can be started, it must be configured and installed. For more information, see Chapter 5, "Configuring Orchestration Server," on page 63 and Chapter 8, "Installing Orchestration Server," on page 145.

Before starting Solution Control Interface (SCI), start the Configuration Server, and Solution Control Server. Make sure that Local Control Agent (LCA) is running.

Note: SCI, Solution Control Server, and LCA are a part of the Management Layer. See the *Framework 8.1 Management Layer User's Guide* for information about these components.

Starting Orchestration Server

Task Summary: Starting Orchestration Server

Objective	Related Procedures and Actions
Start Orchestration Server using Solution Control Interface	Procedure: Starting Orchestration Server with Solution Control Interface, on page 155
Start Orchestration Server on UNIX-Based Platforms	Procedure: Starting Orchestration Server on UNIX-Based Platforms, on page 155

Start Orchestration Server from the Solution Control Interface (SCI). Figure 30 shows an example that could include Orchestration Server as well as other server applications.

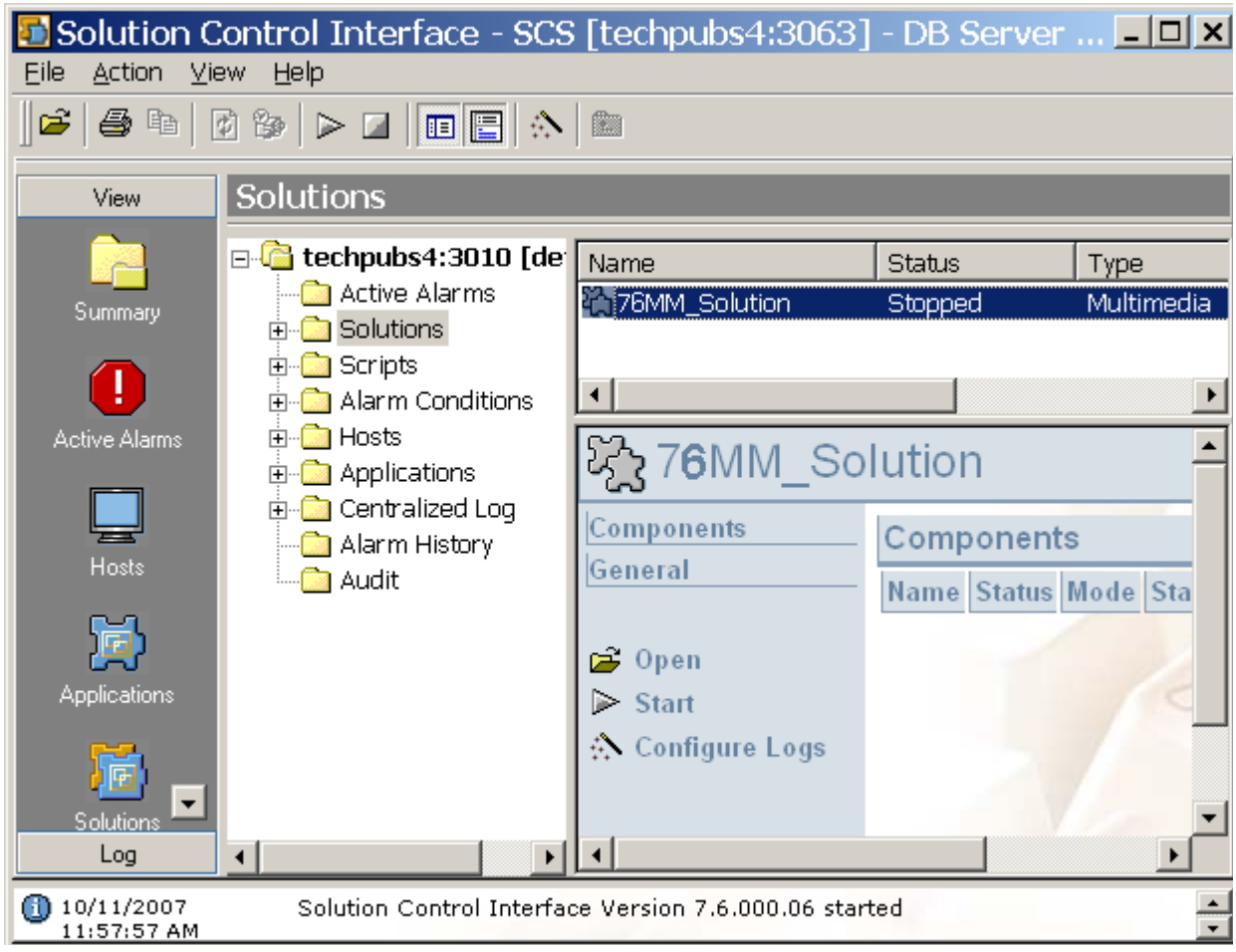

Figure 30: Solution Control Interface

Procedure:
Starting Orchestration Server with Solution Control Interface

Start of procedure

1. Start the Solution Control Interface.
2. Go to the `Solutions` view.
3. Right-click on the desired solution and select `Start` from the shortcut menu.
   OR
   Select the desired solution and choose `Action > Start` on the menu bar.

End of procedure

Starting Orchestration Server Manually

This section describes how to manually start Orchestration Server (ORS). For information on manually starting Framework components necessary to use Orchestration Server, see the *Framework 8.1 Deployment Guide*.

To start Orchestration Server manually, select `Start > All Programs > Genesys Solutions > Routing > Orchestration Server [ORS]> Start Orchestration Server`.

Note: The above path is the default location. If you installed the software at a different location, navigate to the appropriate location to start ORS.

You can also start Orchestration Server as follows:

`Programs > Administrative Tools > Services > Genesys Orchestration Server`, and choose `Run` or `Stop`.

You can also start from the command line or the `Start Info` tab for ORS in Solution Control Interface.

ORS runs automatically after rebooting Windows. To change this: choose `ORS` in `Services > Properties`, startup type `"manual"`.

Procedure:
Starting Orchestration Server on UNIX-Based Platforms

Installation of ORS creates a `run.sh` file.

You can start Orchestration Server on UNIX platforms by just running this file which contains:

```
./orchestration -host <name of Configuration Server host> -port
<name of Configuration Server port> -app <name of ORS Application>
-l <the full path and name of license file>
```

Start of procedure

1. Open a terminal window.
2. Log in.
3. Choose the appropriate directory.
4. Run the `run.sh` file.

End of procedure

Note: The text in angle brackets (`<text>`) above indicates the variables you enter that are unique to your environment and are required. Your information should replace the text and the brackets. See the example below for clarification.

The following is an example of a `run.sh` file:

```
./orchestration -host Daemon -port 5010 -app "OR_Server" -l "/FLEXlm
/license.dat"
```

Quotation marks are required before and after the name of the ORS application. The license file path must also be enclosed in quotation marks.

When ORS is started, a window opens and messages are sent regarding its status. ORS also establishes connections to all servers listed in the `Connections` tab of the Orchestration Server Application object.

Stopping

This section describes how to stop Orchestration Server by using the Solution Control Interface

Task Summary: Stopping Orchestration Server

Objective	Related Procedures and Activities
Stop Orchestration Server using Solution Control Interface	Procedure: Stopping Orchestration Server using Solution Control Interface, on page 157

Procedure:
Stopping Orchestration Server using Solution Control Interface

Orchestration Server should be stopped using the Solution Control Interface (SCI).

Start of procedure

1. Start the Solution Control Interface.
2. Go to the `Applications` view.
3. Right-click on the desired application and select `Stop` from the shortcut menu.
   OR
   Select the desired application and choose `Action > Stop` on the menu bar.

End of procedure

The command to stop the application is sent to Solution Control Server, which uses Local Control Agent to terminate the application.

SCI reports a successful stop of the application. When stopped, Orchestration Server's status changes from `Started` to `Stopped`.

Non-Stop Operation

The *non-stop operation* (NSO) feature enables ORS to continue to run even if it encounters problems. NSO prevents a shutdown in the event of failures. This works by allowing ORS to operate on two levels that are designated by the command-line parameters described below.

Built-in NSO provides the option of running ORS in non-stop operation mode (NSO).

Note: When ORS is started, non-stop operation is disabled by default.

The command-line parameter `-nco` is used to control non-stop operation. ORS built with NSO support runs in NSO only if one of the following arguments is specified in the command line:

`-nco xcount/xthreshold`	Where `xcount` (exception counts) is the number of faults allowed during a specified interval before the application exits and `xthreshold` (exception threshold) is the time interval in seconds. The values must be separated by a slash.
`-nco`	Starts NCO with default parameters (six faults in 10 seconds)

Examples:

```
orchestration -host ra -port 2000 -app orchestration -nco
orchestration -host ra -port 2000 -app orchestration -nco 100/1
```

See the Framework documentation on T-Servers for more information about faults.

Version Identification

To print the ORS version number to the log, use `-v`, `-version`, or `-V` in the command line. This option does not actually start ORS. It just prints the version number to the log and then exits.

Chapter

10 Uninstalling Orchestration Server

This chapter describes how to uninstall Orchestration Server. It includes the following topics:

- Removing Orchestration Server with Genesys Administrator, page 160
- Removing Orchestration Server Manually, page 160

Task Summary: Uninstalling Orchestration Server

Objective	Related Procedures and Activities
Uninstall Orchestration Server in Genesys Administrator	Procedure: Removing the Orchestration Server component with Genesys Administrator, on page 160
Uninstall Orchestration Server manually	Procedure: Manually removing Orchestration Server on Windows, on page 160 Procedure: Manually removing Orchestration Server on UNIX-Based Platforms, on page 161

Removing Orchestration Server with Genesys Administrator

This section describes how to remove the Orchestration Server component with Genesys Administrator.

Procedure: Removing the Orchestration Server component with Genesys Administrator

Purpose:

If you are using Genesys Administrator in your environment, you can uninstall the Orchestration Server component directly from the Genesys Administrator interface.

Start of procedure

1. Log in to Genesys Administrator.
2. Locate the Orchestration Server component that you wish to remove.
3. Click `Uninstall`.

End of procedure

Removing Orchestration Server Manually

This section describes how to remove the Orchestration Server manually.

Procedure: Manually removing Orchestration Server on Windows

Start of procedure

Do the following on each machine that hosts Orchestration Server:

1. From the Windows `Start` menu, open the Control Panel (`Start > Control Panel`) and click `Add or Remove Programs`.
2. At the `Add or Remove Programs` dialog box, select `Genesys Orchestration Server <version number> [ORS]`.
3. Click `Remove`, and follow the instructions to remove Orchestration Server.

4. Using Windows Explorer, browse to the `GCTI` main directory and delete the complete Orchestration Server subdirectory, including all subfolders, if any folders or files remain.

End of procedure

Procedure: Manually removing Orchestration Server on UNIX-Based Platforms

Start of procedure

- For the directory in which each Orchestration Server component is installed, run the following command:

  ```
  rm -R
  ```

Note: If an instance of the component is running, the command will fail.

End of procedure

Appendix

Installing and Configuring Apache Cassandra Server

This Appendix describes how to install and configure Apache Cassandra Server *in a single-node cluster* or a *multi-node cluster* to prepare it for use with Orchestration Server.

Note: The preceding link is not live because third-party links often change. To ensure you are accessing the correct link for information, check the Apache web site.

This Appendix includes the following sections:

- Installing, Configuring, Starting, and Testing a Cassandra Node, page 164
- Sample Cassandra Output, page 173

Genesys recommends that all nodes in a Cassandra cluster be deployed on the same platform type (all Windows-based or all Linux-based). The operating system version is not a factor in this consideration, except for the minimum operating system version that Cassandra requires.

ORS 8.1.1 and later releases support Cassandra 0.7x and later versions. Cassandra requires an installed Java 6.

For Linux installations, the UNIX style paths (and other UNIX conventions) should be used.

Note: Genesys recommends that Apache Cassandra Server be deployed from the CD/DVD; however, later updates of the server may be available from the Apache web site. If you find a more recent version of the software, contact Genesys Technical Support to determine support.

Installing, Configuring, Starting, and Testing a Cassandra Node

The following is an overview of the steps that are involved in the installation and configuration of a single Cassandra node. Use this procedure only if you are downloading the install image from the Cassandra web site:

1. Download the install image, extract it, and create the required directories.
2. Configure Cassandra Server.
3. Start the server.
4. Test the server.

Task Summary: Installing/Configuring a Cassandra Node

Objective	Related Procedures and Actions
Download the install image, extract it, and create the required directories.	Procedure: Downloading and extracting the install image, on page 164
Configure Cassandra Server.	Procedure: Configuring Cassandra Server, on page 165
Start the server.	Procedure: Starting Cassandra Server, on page 169
Test the server.	Procedure: Testing Cassandra Server

Downloading and Extracting the Install Image

This section describes the following procedure:

- "Downloading and extracting the install image"

Note: You can skip this procedure if you are installing Cassandra from the CD/DVD.

Procedure: Downloading and extracting the install image

Purpose: To retrieve the Cassandra Server install image from the Apache web site, download it to a local drive, and extract the contents of the compressed file.

Start of procedure

1. Download the install image from `http://cassandra.apache.org/`.

   Download the most current stable version.
2. Copy the install image file to a computer that will serve as the Cassandra node—for example: `C:\CassandraServerNodeOne`.
3. Extract the install image into the chosen directory.
4. In the `installation` directory, create the following sub-directories:
   - `commitlog`
   - `logs`
   - `data`
   - `saved_caches`
5. In the `data` directory, create the following sub-directories:
   - `system`
   - `Orchestration`

End of procedure

Next Steps

- Continue with Procedure: Configuring Cassandra Server.

Configuring Cassandra Server

This section describes the following procedures:

- "Configuring Cassandra Server"
- "Configuring a multi-node Cassandra cluster"

Configuring Cassandra Server consists of *copying* the `storage-conf.sample` file from the ORS installation directory to the Cassandra installation `conf` directory and *editing* the contents of this file and several other files provided in the Cassandra installation package. This procedure describes these steps.

Procedure: Configuring Cassandra Server

Purpose: To configure Cassandra Server in a cluster to use as persistent data storage for ORS.

Start of procedure

1. Copy the `orchestration-schema-sample.txt` file from the ORS installation directory to the Cassandra installation directory, inside the `conf` directory. For example, to: `C:\CassandraServerNodeOne\conf`

   Note: The `orchestration-schema-sample.txt` file does not require modification if you are configuring a single-node cluster. Entries in this file can be modified to optimize performance by customizing the deployment. For further details about how to modify this file, see the Cassandra documentation at `http://www.datastax.com`.

2. If you are configuring on Windows:
   a. Set the `CASSANDRA_HOME` environment variable to the installation directory. For example, `C:\CassandraServerNodeOne`.
   b. Set the `JAVA_HOME` environment variable to the installation directory of the java jre (or jdk). For example, `JAVA_HOME=C:\Program Files\Java\jre6`

   On UNIX systems `JAVA_HOME` is in the `Cassandra.in.sh` file in the `install-path/bin` directory, as shown in the example that follows.

   ```
   JAVA_HOME=/jre16/jre1.6.0_20/
   ```

   Note: In this case, the `cassandra.in.sh` file includes the following items:

   ```
   #MAX_HEAP_SIZE="4G" #HEAP_NEWSIZE="800M"
   JMX_PORT="8080"
   ```

   Review these items and refer to the comments in the `cassandra.in.sh` file to determine if any modifications are required and, if so, apply the appropriate settings.

   Modify the `JMX_PORT` port only if there is a conflict with an existing port assignment. If you choose a different port, use the this same port on all of the Cassandra nodes.

3. Open the `cassandra.yaml` file and refer to the comments to determine if any modifications are required.
4. Modify the `Cluster_Name:'Test Cluster'` entry to assign an actual name for the cluster. This name must be *unique*, as nodes in the cluster use this name to identify themselves as members of the cluster. For example,

   ```
   cluster_name: 'ClusterOne'
   ```

5. Set the `initial_token` entry. For example, for a single node Cassandra cluster, set the entry as follows: `initial_token: '0'`
6. Set the paths to the storage locations on the server platform for this node.

Note: The example provided in this procedure is for Windows-style paths. Adjust file paths accordingly if using the UNIX platform.

See the following examples for each directory:

- `commitlog_directory: C:\CassandraServerNodeOne\commitlog`
- `data_file_directories: C:\CassandraServerNodeOne\data`
- `saved_caches_directory: C:\CassandraServerNodeOne\saved_caches`

Note: Genesys recommends that you place the `commitlog` and `data` directories on *different* disk drives for optimal performance. Also make sure the directories exist or, if they do not exist, create them.

7. The `listen_address` and `rpc_address` comments in the `cassandra.yaml` file describe what needs to be done for these entries: most often for a single node they may be left empty, defaulting to the `hostname` of the platform. The `rpc_address` comment defines the client listen address and defaults to `9160`. If this port conflicts with an existing configured port, provide a non-conflicting entry for the `rpc_address` comment. If `9160` is not currently used, you can keep the default value.

   See the following examples:

   - `listen_address: localhost`
   - `rpc_address: localhost`
   - `rpc_port: 9160`

   Note: For a *multi-node cluster*, use the IP address of the platform for the addresses.

8. For a *single-node cluster*, if the IP address `127.0.0.1` resolves correctly, it is not necessary to modify the `<seeds>` entry. Otherwise, enter the IP address of the node (for example, `seeds: 127.0.0.1`).

9. A potential port conflict with URS might occur for `storage_port 7000`. This can be modified for Cassandra to remove the conflict by setting the `storage_port` entry to something other than `7000`. Choose a port that does not conflict with any other existing port usage. This port must be the *same* for all nodes in the Cassandra cluster.

   For example:

   ```
   storage_port: 7001
   ```

10. Modify the `log4j-server.properties` entry, as follows:

```
# Edit the next line to point to your logs directory
log4j.appender.R.File=C:/CassandraServerNodeOne/logs/system.log
```

> **Note:** The `log4j.appender.R.File` entry must have the forward slash as a separator.

End of procedure

Next Steps

- If you need to configure a multi-node Cassandra cluster, continue with Procedure: Configuring a multi-node Cassandra cluster.
- Otherwise, continue with Procedure: Starting Cassandra Server.

Procedure: Configuring a multi-node Cassandra cluster

Purpose: To configure a multi-node Cassandra cluster. There is important and relevant third-party information available regarding multi-node Cassandra clusters. At the time of this writing, this content is located at the following location:

```
http://wiki.apache/cassandra/MultinodeCluster
```

Start of procedure

1. The procedure for installation and configuration of a multi-node cluster is essentially the same as what was presented above. First, review Procedure: Configuring Cassandra Server for multi-node issues regarding `seeds`, `listen_address`, and `rpc_address` comments.
2. Provide a *replication factor,* which is an integer less than or equal to the number of nodes in the cluster. The value indicates the number of nodes to "copy" the data to on writes, and the number to read from on reads, depending on the consistency level specified.

   The replication factor determines the success of data access in Cassandra, where consistency is achieved if:

   ```
   [READ replica count] + [WRITE replica count] > [Replication Factor]
   ```

   or

   ```
   R + W > N
   ```

For Orchestration Server, the consistency level is `ONE`, which means that reads and writes will be successful if at least one node of the replica set is available. For example, if there are three nodes in the cluster, and the replication factor is 2, the read or write will be successful if either one of the nodes fail.

Refer to the content at `http://wiki.apache.org/cassandra/ArchitectureOverview` for more detail.

3. Assign an *initial token*.

   The initial token assignment is critical for use of the Random Partitioner, which is what is used with Orchestration Server. Cassandra uses the MD5 hash of the keys provided by a client to determine which Cassandra node in a cluster will "own"" the data and processing associated with the write or read. The Cassandra cluster can be viewed as a ring, and the key space should be divided equally over all the nodes. This means the initial token should be `2**127/N`, then multiplied by the node number, `0`, `1`,`2`,.`N`, as below for a three node cluster:

   ```
   (2**127)/3 = 56713727820156410577229101238628035242
   ```

   Then the tokens would be:

   ```
   ((2**127)/3)*0 = 0
   ((2**127)/3)*1 = 56713727820156410577229101238628035242
   ((2**127)/3)*2 = 113427455640312821154458202477256070484
   ```

   More detail can be found at:
   `http://wiki.apache.org/cassandra/Operations`.

End of procedure

Next Steps

- Continue with Procedure: Starting Cassandra Server.

Starting Cassandra Server and Loading the Schema

This section describes the following procedures:

- "Starting Cassandra Server"
- "Loading the Cassandra Schema"

Procedure: Starting Cassandra Server

Purpose: To start up the Cassandra Server after it has been installed and configured as described in earlier procedures.

Start of procedure

1. Under Windows, start Cassandra Server from the installation directory (for example, `C:/CassandraServerNodeOne`).

   `.\bin\cassandra.bat`

2. Under Linux, start Cassandra Server by issuing the following:

   `./bin/cassandra` (from the Cassandra installation directory)

End of procedure

Next Steps

- Load the schema. See Procedure: Loading the Cassandra Schema.

Procedure: Loading the Cassandra Schema

Purpose: To load the schema after the Cassandra Server is started

Summary

After all nodes in the cluster are started (which might be only one), execute one of the commands in this procedure (for Windows or Linux) from the Cassandra installation directory. Execute these commands on a single node only; Cassandra distributes the schema from the executing node to all nodes in the cluster.

Start of procedure

1. Execute the following command on Windows:

   ```
   .\bin\cassandra-cli.bat -host ip-address-of-cassandra-host
   --file conf/orchestration-schema-sample.txt
   ```

   where `ip-address-of-cassandra-host` is the host IP. For example, `135.225.58.81`.

2. Execute the following command on Linux:

   ```
   ./bin/cassandra-cli
   ```

   The following a sample output of the Cassandra schema load for a single

node on Windows:

```
Starting Cassandra Client
Connected to: "ClusterOne" on 172.21.82.194/9160
8d4c8263-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
Authenticated to keyspace: Orchestration
8dae5174-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8dbc8245-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8dc86926-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8dd47717-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8dec44d8-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8e068399-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8e17256a-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
8e230c4b-a9ae-11e0-9d1c-82171c8401e9
Waiting for schema agreement...
... schemas agree across the cluster
```

End of procedure

Next Steps

- Continue with Procedure: Testing Cassandra Server.

Testing Cassandra Server

This section describes the following procedure:

- "Testing Cassandra Server"

Procedure: Testing Cassandra Server

Purpose: To test whether the Cassandra Server is able to connect to an Orchestration Cluster after it has been installed, configured, and started as described in earlier procedures.

Start of procedure

1. Under Windows (from the install directory, such as `C:\CassandraServerNodeOne`:

   `.\bin\cassandra-cli.bat`

   The client also requires `JAVA_HOME` and `CASSANDRA_HOME` environment variables to be set.

   As an alternative, the `cassandra-cli.bat` file in the `bin` directory may be modified to set `JAVA_HOME` and `CASSANDRA_HOME` as follows (if these are not to be set in the system environment variables):

   `JAVA_HOME=C:\Program Files\Java\jre6` (the installation root of `jrel6`)

   `CASSANDRA_HOME=C:\CassandraServerNodeOne`

2. Enter Cassandra Help, by typing `help;` at the `[default@unknown]` prompt once the client starts. For example:

   ```
   Starting Cassandra Client
   Welcome to cassandra CLI.

   Type 'help;' or '?' for help. Type 'quit;' or 'exit;' to quit.
   [default@unknown] help;
   ```

   Cassandra Client displays a command list showing you the types of operations you may perform.

3. Attempt to make a connection. Type the following, substituting the name of your Cassandra Server and port for the information in the example below:

   `[default@unknown] connect DWS/9160;`

   If a message similar to the one below displays, the server is ready to run with Orchestration.:

   `Connected to: "ClusterOne" on DWS/9160`

4. Execute the following command to indicate the keyspace, in this case, `Orchestration:`

   `[default@unknown] use Orchectration;`

5. Other commands in the Cassandra Client Help list may be used to set/get/delete data. See the following examples:

```
[default@Orchestration] set
Session['D7FE00BD045104C20001']['D7FE00BD045104C20001'] = 'This is
the session content in JSON form';
Value inserted.

[default@Orchestration] get Session['D7FE00BD045104C20001'];
=> (column=D7FE00BD045104C20001,
value=5468697320697320746865207365737373696f6e20636f6e74656e7420696e2
04a534f4e20666f726d, timestamp=131016872
9128000)
```

End of procedure

Sample Cassandra Output

The following is an example of output from Cassandra for reference and informational purposes.

```
Starting Cassandra Server
 INFO 16:34:59,346 Logging initialized
 INFO 16:34:59,362 Heap size: 1067057152/1067057152
 INFO 16:34:59,362 JNA not found. Native methods will be disabled.
 INFO 16:34:59,378 Loading settings from
file:/C:/Cassandra/apache-cassandra-0.7.6-2/conf/cassandra.yaml
DEBUG 16:34:59,471 Syncing log with a period of 10000
 INFO 16:34:59,471 DiskAccessMode is standard, indexAccessMode is mmap
DEBUG 16:34:59,534 setting auto_bootstrap to false
DEBUG 16:34:59,581 Initializing system.IndexInfo
DEBUG 16:34:59,596 Starting CFS IndexInfo
DEBUG 16:34:59,596 key cache capacity for IndexInfo is 1
DEBUG 16:34:59,596 row cache capacity for IndexInfo is 0
DEBUG 16:34:59,612 Initializing system.Schema
DEBUG 16:34:59,612 Starting CFS Schema
 INFO 16:34:59,612 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-9
DEBUG 16:34:59,612 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-9
DEBUG 16:34:59,628 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-9: 16 ms.
DEBUG 16:34:59,628 key cache contains 0/0 keys
```

```
 INFO 16:34:59,628 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-10
DEBUG 16:34:59,628 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-10
DEBUG 16:34:59,628 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-10: 0 ms.
DEBUG 16:34:59,628 key cache contains 0/0 keys
 INFO 16:34:59,643 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-11
DEBUG 16:34:59,643 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-11
DEBUG 16:34:59,643 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-11: 0 ms.
DEBUG 16:34:59,643 key cache contains 0/0 keys
DEBUG 16:34:59,643 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-9 to list of
files tracked for system.Schema
DEBUG 16:34:59,643 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-10 to list
of files tracked for system.Schema
DEBUG 16:34:59,643 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Schema-f-11 to list
of files tracked for system.Schema
DEBUG 16:34:59,643 key cache capacity for Schema is 3
DEBUG 16:34:59,643 row cache capacity for Schema is 0
DEBUG 16:34:59,643 Initializing system.Migrations
DEBUG 16:34:59,643 Starting CFS Migrations
 INFO 16:34:59,659 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-10
DEBUG 16:34:59,659 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-10
DEBUG 16:34:59,659 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-10: 0
ms.
DEBUG 16:34:59,659 key cache contains 0/0 keys
 INFO 16:34:59,659 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-11
DEBUG 16:34:59,659 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-11
DEBUG 16:34:59,659 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-11: 0
ms.
DEBUG 16:34:59,659 key cache contains 0/0 keys
 INFO 16:34:59,659 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-9
DEBUG 16:34:59,659 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-9
```

```
DEBUG 16:34:59,659 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-9: 0 ms.
DEBUG 16:34:59,659 key cache contains 0/0 keys
DEBUG 16:34:59,659 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-10 to
list of files tracked for system.Migrations
DEBUG 16:34:59,659 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-11 to
list of files tracked for system.Migrations
DEBUG 16:34:59,659 adding
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\Migrations-f-9 to
list of files tracked for system.Migrations
DEBUG 16:34:59,659 key cache capacity for Migrations is 3
DEBUG 16:34:59,659 row cache capacity for Migrations is 0
DEBUG 16:34:59,659 Initializing system.LocationInfo
DEBUG 16:34:59,674 Starting CFS LocationInfo
 INFO 16:34:59,674 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-6
DEBUG 16:34:59,674 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-6
DEBUG 16:34:59,674 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-6: 0
ms.
DEBUG 16:34:59,674 key cache contains 0/0 keys
 INFO 16:34:59,674 Opening
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-5
DEBUG 16:34:59,674 Load statistics for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-5
DEBUG 16:34:59,674 INDEX LOAD TIME for
C:\Cassandra\apache-cassandra-0.7.6-2\data\system\LocationInfo-f-5: 0
ms.
DEBUG 16:34:59,674 key cache contains 0/0 k
```

Related Documentation Resources

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

Universal Routing

- *Universal Routing 8.1 Deployment Guide,* which describes installation, configuration, and deployment of the Universal Routing Server and Custom Server components that work with the Orchestration Server.
- *Universal Routing 8.1 Reference Manual,* which describes and defines routing strategies, IRD objects, Universal Routing Server and other server functions and options, number translation, pegs, and statistics used for routing.
- *Universal Routing 8.1 Business Process User's Guide*. This guide contains step-by-step instructions for creating interaction workflows (business processes), which direct incoming customer interactions through various processing objects. The goal is to generate an appropriate response for the customer.
- *Universal Routing 8.1 Strategy Samples*, which simplifies strategy configuration for first-time users of the strategy development tool, Interaction Routing Designer. To achieve this goal, this document supplies examples of simple voice and e-mail routing strategies that can be used as general guides during the design stage.
- *Universal Routing 8.0 Routing Application Configuration Guide* (previously *Universal Routing 7.0 Routing Solutions Guide*), which contains information on the various types of routing that can be implemented, including skills-based routing, business-priority routing, and share agent by service level agreement routing. It also summarizes cost-based routing, proactive routing, and a SIP/instant message solution.

- *Universal Routing 8.1 Interaction Routing Designer Help,* which describes how to use Interaction Routing Designer to create routing strategies. It also describes Interaction Design view where you create business processes that route incoming interactions through various processing objects with the goal of generating an appropriate response for the customer.

eServices and Other

- *eServices 8.0 Deployment Guide,* which includes a high-level overview of features and functions of Genesys eServices together with architecture information and deployment-planning materials. It also introduces you to some of the basic concepts and terminology used in this product.
- *eServices 8.0 User's Guide,* which provides overall information and recommendations on the use and operation of Genesys eServices.
- *eServices 8.0 Open Media Interaction Models Reference Manual*, which presents a set of basic interaction models, showing the components involved and the messaging (requests, events) among them.
- "Universal Routing and Multimedia/eServices Log Events" in *Framework 8.1 Combined Log Events Help,* which is a comprehensive list and description of all events that may be recorded in Management Layer logs.

Genesys

- *Genesys 8.0 Proactive Routing Solution Guide*, which documents a solution that enables you to proactively route outbound_preview interactions to Genesys Agent Desktop, as well as to completely process Calling List and Do Not Call List records solely from the logic of a routing strategy without agent intervention.
- *Genesys Events and Models Reference Manual*, which provides information on most of the published Genesys events and their attributes, and an extensive collection of models describing core interaction processing in Genesys environments.
- *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 Technical Support for more information.
- Release Notes and Product Advisories for this product, which are available on the Genesys Technical Support website at http://genesyslab.com/support.

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

- *Genesys Supported Operating Environment Reference Manual*
- *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 Genesys 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.

For additional system-wide planning tools and information, see the release-specific listings of System Level Documents on the Genesys Technical Support website, accessible from the `system level documents by release` tab in the Knowledge Base `Browse Documents` Section.

Genesys product documentation is available on the:

- Genesys Technical Support website at `http://genesyslab.com/support`.
- Genesys Documentation Library DVD, which you can order by e-mail from Genesys Order Management at `orderman@genesyslab.com`.

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:

8‘fr_ref_04-2011_v8.1.001.00

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, may 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 11 describes and illustrates the type conventions that are used in this document.

Table 11: Type Styles

Type Style	Used For	Examples
Italic	• Document titles • Emphasis • Definitions of (or first references to) unfamiliar terms • Mathematical variables 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 181).	Please consult the *Genesys Migration Guide* for more information. Do *not* use this value for this option. A *customary and usual* practice is one that is widely accepted and used within a particular industry or profession. The formula, $x + 1 = 7$ where *x* stands for. . .

Table 11: Type Styles (Continued)

Type Style	Used For	Examples
Monospace font (Looks like `teletype` or `typewriter text`)	All programming identifiers and GUI elements. This convention includes: • The *names* 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. Also used for any text that users must manually enter during a configuration or installation procedure, or on a command line.	Select the `Show variables on screen` check box. In the `Operand` text box, enter your formula. Click `OK` to exit the `Properties` dialog box. T-Server distributes the error messages in `EventError` events. If you select `true` for the `inbound-bsns-calls` option, all established inbound calls on a local agent are considered business calls. Enter `exit` on the command line.
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 (< >)	A placeholder for a value that the user must specify. This might be a DN or a port number specific to your enterprise. **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.	`smcp_server -host <confighost>`

Index

Symbols

[] (square brackets) . . . 181
< > (angle brackets) . . . 181

A

agent
  capacity rules . . . 59
agent capacity rules . . . 60
angle brackets . . . 181
Application
  Configuration Manager login dialog box . . . 69
application
  SCXML-based . . . 20
application changes and ORS . . . 70, 76
Application Servers
  supported in ORS 8.1 . . . 24
audience, for document . . . 14
auto-restart
  ORS . . . 74

B

brackets
  angle . . . 181
  square . . . 181
business attributes . . . 101
business-priority routing . . . 177

C

Cassandra
  configuring . . . 165
  downloading/extracting image . . . 164
  installing . . . 164
  sample output . . . 173
  starting . . . 169
  storage-conf.xml . . . 165
  testing . . . 171
Cassandra Server . . . 163
changes to applications . . . 70, 76
CIM (Customer Interaction Management) Platform . . . 12, 13
client side port definition . . . 29, 151
  ORS . . . 76, 148
clustering
  configuration . . . 85
clusters . . . 52
  deployment . . . 52
  failure recovery . . . 54
  load balancing method . . . 54
Command Line
  ORS . . . 74
commenting on this document . . . 15
components
  Configuration Manager . . . 60
  Configuration Server . . . 60
  DB Server . . . 60
  Message Server . . . 60
  Stat Server . . . 60, 145
  T-Server . . . 60, 145
  Universal Routing Server . . . 60
Composer . . . 142
Configuration Layer . . . 145
Configuration Manager . . . 60
configuration object changes and ORS . . . 70, 76
configuration options
  {Parameter Name}, ApplicationParms section . . . 132
  alternate-url . . . 126
  application . . . 125
  cassandra-listenport . . . 110
  cassandra-nodes . . . 110
  debug_port . . . 119
  fetch-timeout . . . 128
  fips_enable . . . 110
  hostname . . . 124
  http-client-side-port-range . . . 111
  http-enable-continue-header . . . 111
  http-max-age-local-file . . . 111

http-max-cache-entry-count 112
http-max-cache-entry-size 112
http-max-cache-size 112
http-max-redirections 113
http-no-cache-urls 113
http-proxy 113
https-proxy 117
http-ssl-ca-info 114
http-ssl-ca-path 114
http-ssl-cert 114
http-ssl-cert-type 115
http-ssl-cipher-list 115
http-ssl-key 115
http-ssl-key-type 116
http-ssl-random-file 116
http-ssl-verify-host 116
http-ssl-verify-peer 117
http-ssl-version 117
http-useragent 128
http-version 128
max-age 129
max-compiler-cached-docs 118
max-compiler-cached-doc-size 118
max-compiler-cache-size 118
max-duration 129
max-includes 119
max-loop-count 130
max-preprocessor-cached-docs 119
max-preprocessor-cached-doc-size 119
max-preprocessor-cache-size 119
max-stale 130
mcr-pull-interval 109
mcr-pull-limit 109
name 123
om-delete-from-memory 123
om-max-in-memory 123
om-memory-optimization 122
password 120
persistence-default 120
persistence-max-active 120
port 124
primary-webdav-url 109, 110
session-hung-timeout 109
session-processing-threads 121
super_node 124
system-id 121
url 131
x-print-attached-data 122
x-server-config-trace-level 122
x-server-gcti-trace-level 121
x-server-trace-level 121
Configuration Server 60
port for ORS 69
Configuration Server Proxy 74
configuring
premise T-Server 102
Stat Server 102
Connections tab
ORS 74
conventions
in document 180
type styles 180
coordinating customer interactions and dialogues 25
cost-based routing 177

D

DB Server 60
document
audience 14
change history 16
conventions 180
errors, commenting on 15
version number 180
documentation
eServices User's Guide 178
Events and Models Reference Manual . . 178
IRD Help 178
Log Events Help 178
Proactive Routing Solution Guide 178
Universal Routing Business Process User's Guide 177
Universal Routing Deployment Guide . . 177
Universal Routing Reference Manual . . . 177
Universal Routing Routing Application Configuration Guide 177
Universal Routing Strategy Samples . . . 177

E

ECMAScript 21
E-mail Accounts 101
eServices Open Media Interaction Models . 178
External Service object 102

F

font styles
italic 180
monospace 181
Framework 57
Functional Modules 22

G

General tab
ORS 73

H

help file
  Interaction Routing Designer Help 178
high availability 52
Host name
  Configuration Manager login dialog box . . . 69

I

installation order 59
installation package directories 146
installing ORS
  Linux 151
  Windows 147
intended audience 14
interaction attributes 101
italics 180

L

Language Business Attribute 101
Linux
  Orchestration Server installation 151
load balancing
  eServices interactions 54
  RESTful web services 54
  voice interactions 54
logs
  Management Layer 178

M

Management Layer 145
Management Layer logging 178
media channels 13
Media Type 102
memory optimization
  configuration 83
Message Server 60
modeling/managing customer service process 25
monospace font 181

N

Network Routing reporting 59
Non-Stop Operation 157

O

Open Media concept 102
Operational Reporting 133
options
  options list 103
orchestrating resources and product services . 26
Orchestration 24
  definition 19
  eServices interaction routing 36
  platform capabilities 24
Orchestration Server
  About 11
  architecture 31
  auto-restart 74
  clustering
    configuring 85
  Configuration Server Proxy 74
  deployment 57
  eServices interaction routing 36
  eServices interactions 34
  high availability 52
  Linux installation 151
  memory optimization
    configuring 83
  persistent storage
    configuring 49
    deploying 48
    operation 46
  persistent storage design 45
  shutdown time 74
  startup time 74
  voice interactions 32
  Windows installation 147
Orchestration Server options, table 104
ORS cluster configuration 85
ORS delays 70, 76
ORS memory optimization configuration . . . 83

P

password
  Configuration Manager login dialog 69
persistence 45
Persistent Storage
  configuration 49
  deployment 48
  design 45
  operation of 46
Port
  Configuration Manager login dialog box . . 69
port info for ORS 73
premise
  T-Server 102
pre-start information 153

R

Real-Time Metrics Engine 59
reporting 59

e-mail and chat interactions 59
Resource Capacity Wizard 60
routing
strategy samples 177

S

Samples 177
SCXML application support 20
SCXML session startup 55
Secure port configuration 73
security 28
Server Info tab
ORS 73
service history binding 26
Share Agents by Service Level Agreement . 177
Solution Control Interface 155
square brackets 181
Start Info tab
ORS 73
starting
Orchestration Server 155
Orchestration Server in Linux 155
Orchestration Server using SCI 155
Solution Control Interface 155
Stat Server 60, 102, 145
Real-Time Metrics Engine 59
stopping
Enterprise Routing solution 157
storage-conf.xml 165
strategy
samples 177
supported Application Servers 24

T

Tenants tab
ORS 73
Third Party Server 102
T-Server 60, 145
premise 102
type styles
conventions 180
italic 180
monospace 181
typographical styles 180

U

uninstalling ORS
Linux 161
Windows 160
with Genesys Administrator 160
Universal Routing Server 60
User name
Configuration Manager login dialog box . . 69
User password
Configuration Manager login dialog box . . 69

V

version identification 158
version numbering, document 180
virtual queues 59

W

Windows
Orchestration Server installation 147
Working Directory
ORS 73

X

XML 20