Orchestration Server 8.0

Deployment Guide

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 website or on the Documentation Library DVD, which is available from Genesys upon request. For more information, contact your sales representative.

Notice

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

Your Responsibility for Your System’s Security

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

Trademarks

Genesys, 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: 80ors_dep_10-2010_v8.0.101.00

Table of Contents

Appendix

Supplements

Index

List of Procedures

Preface

Welcome to the *Orchestration Server 8.0 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 all 8.x release(s) of this product.

This preface contains the following sections:

For information about related resources and about the conventions that are used in this document, see the supplementary material starting on page 149.

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.

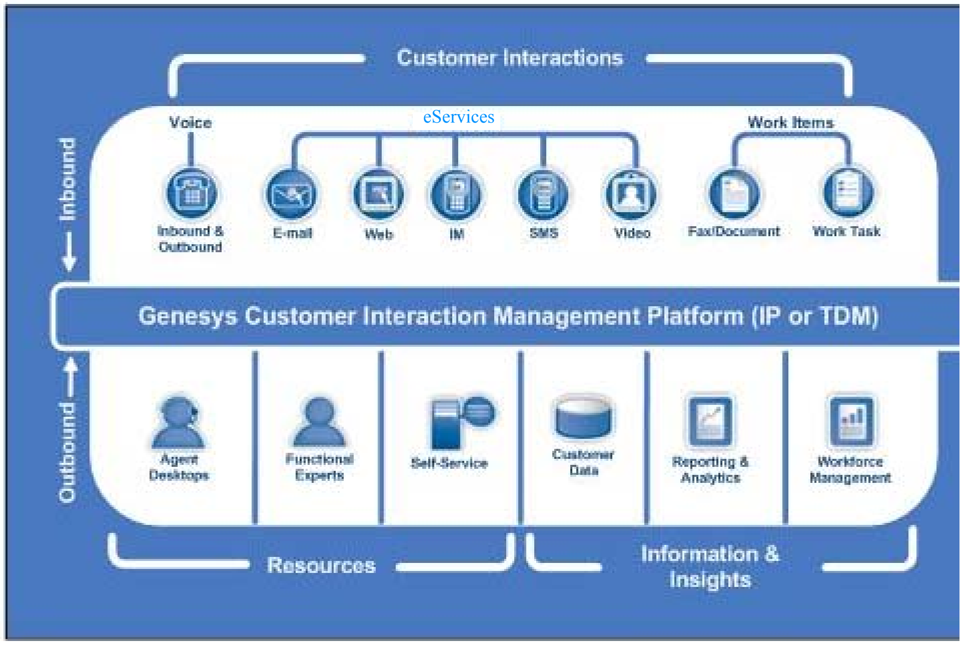


Figure 1: Customer Interaction Management Platform

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.

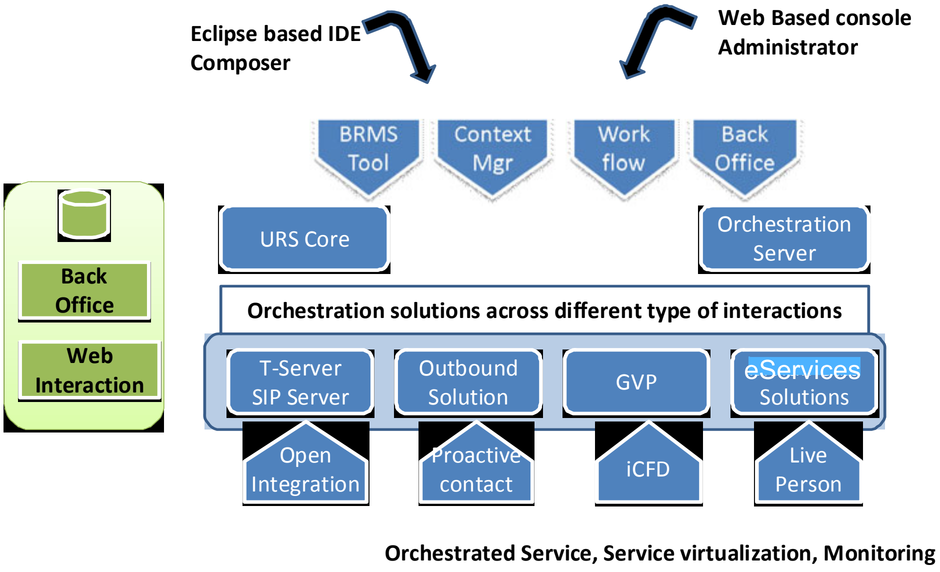


Figure 2: An Example of an Orchestration Solution

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

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
- Leveraging 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 8.0. 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)-1276-45-7002	support@genesyslab.co.uk
Before contacting technical support, refer to the *Genesys Technical Support Guide* for complete contact information and procedures.		

Region	Telephone	E-Mail
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) +91-(022)-3918-0537	support@genesyslab.com.au
Japan	+81-3-6361-8950	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 is the first release of the *Orchestration Server 8.0 Deployment Guide*. In the future, this section will list topics that are new or that have changed significantly since the first release of this document.

Chapter

1 Orchestration Server Overview

The Orchestration Server (ORS) is responsible for executing orchestration logic (SCXML) provided by an Application Server. The responsibility of the Universal Routing Server (URS) is to provide a necessary service to Orchestration Server; by executing routing strategies (providing traditional Universal Routing Server functionality).

Please refer to the *Genesys Universal Routing 8.0 Deployment Guide* for a thorough overview of routing, routing strategies, Universal Routing capabilities, features, licensing, security, and migration issues.

This chapter includes these sections:

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
- Leveraging 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, combinations, and more.

Starting with version 8.0, 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 with the Genesys Interaction Routing Language (IRL)[1], the Orchestration Server executes routing strategies that are written in SCXML (State Chart EXtensible Markup Language). SCXML-based strategies are hosted on an Application Server and provided to the Routing solution based upon parameters that are contained in HTTP requests.

What is SCXML?

SCXML is an XML-based markup language 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 can also be used for sequential workflow solutions.

SCXML provides the backbone logic for orchestration applications. The following must also be present in order to have a complete application:

- Functional modules (which URS provides) to achieve domain-specific functions such as queue handling, eServices, and interaction handling. ORS uses these functional modules from URS for queue, target and statistics information.
- Business logic which provides the business-specific processing that needs to be orchestrated. Any type of interaction can be “orchestrated” through this platform, based on business logic and customer profile data. Business logic can be defined through an external application like Business Rules Management System (BRMS) tools, and customer data/profile can be provisioned through Genesys Context Manager.

Why SCXML?

Customer service occurs in a highly event-driven, asynchronous environment. From a routing strategy 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.

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

- 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:
    - Built using the Genesys Composer Eclipse-based environment (refer to the *Genesys Composer 8.0 Deployment Guide* and *Genesys Composer 8.0 Help*)
    - Built by hand 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 Strategies

Orchestration Server 8.0 with Universal Routing 8.0 supports SCXML plus ECMAScript as a routing language through ECMA Script profiles. It 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 8.0 extensions to the SCXML executable content are grouped by Functional Module as identified in the *Genesys 8.0 SCXML Technical Reference*. (see Table 2 on page 20).

Reference Documents

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

Table 1: External Reference Documents

Document Title	File Name
State Chart XML (SCXML): State Machine Notation for Control Abstraction	http://www.w3.org/TR/scxml/
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 strategy language:

Table 2: Genesys Reference Documents

Document Title
Genesys 8.0 SCXML Technical Reference
Genesys 8.0 SCXML Strategy Samples

Simple Sample of SCXML-Based Strategies

The following XML code sample for a simple voice routing strategy provides a general introduction to SCXML-based strategies (route to target eServices strategy requires use of the `<ixn:redirect>` method):

```
<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 8.0 SCXML Technical Reference* document referenced in Table 2. The preceding example uses the Queue Functional Module.

Universal Routing 8.0 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 thenamespace 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 strategies 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 note on Tomcat below.

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:

[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. Th e terms “customer service process” and “customer service application” are used to describe this.

- Deliver a consistent and tailored customer experience across interaction channels, no matter where the customer chooses to transact with the enterprise, and regardless of whether the transaction cycle spans over multiple interactions and 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 across 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.
  - The service process spans across contact center and back-office touch points (IVR, e-mail, SMS, web, fax, agents, knowledge experts in branch office) as a customer transacts with the enterprise.
  - 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 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, 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

Licensing

There is no all-inclusive Orchestration Server licensing. You must set up licensing for Orchestration components such as Universal Routing Server, Orchestration Server, Context Management Services (CMS), GVP, Outbound, Composer, and BRMS, individually. To fully utilize the range of functionality that Orchestration provides, customers are encouraged to acquire these components.

Note: Development or sales of third party tools and applications which require or utilize Genesys Universal Routing or Genesys Orchestration Server functionality, including access to Genesys Universal Router web interface to provision Target information, Statistics, invoking session, etc., requires a Router Connector License. Please contact your account executive to get more information on its applicability and pricing.

Security

As described in the *Genesys 8.0 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 8.0 security is presented below. For detailed information on how to implement security within Genesys, see the *Genesys 8.0 Security Deployment Guide*. For information on deploying 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.

Orchestration Server provides the flexibility and security of defining the client-side port of the client/server connection. 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 8.0 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 this main section:

- Orchestration Server Architecture, page 27

Orchestration Server Architecture

The following figures (Figure 3 and Figure 4) illustrate the high-level architecture of Orchestration Server 8.0.

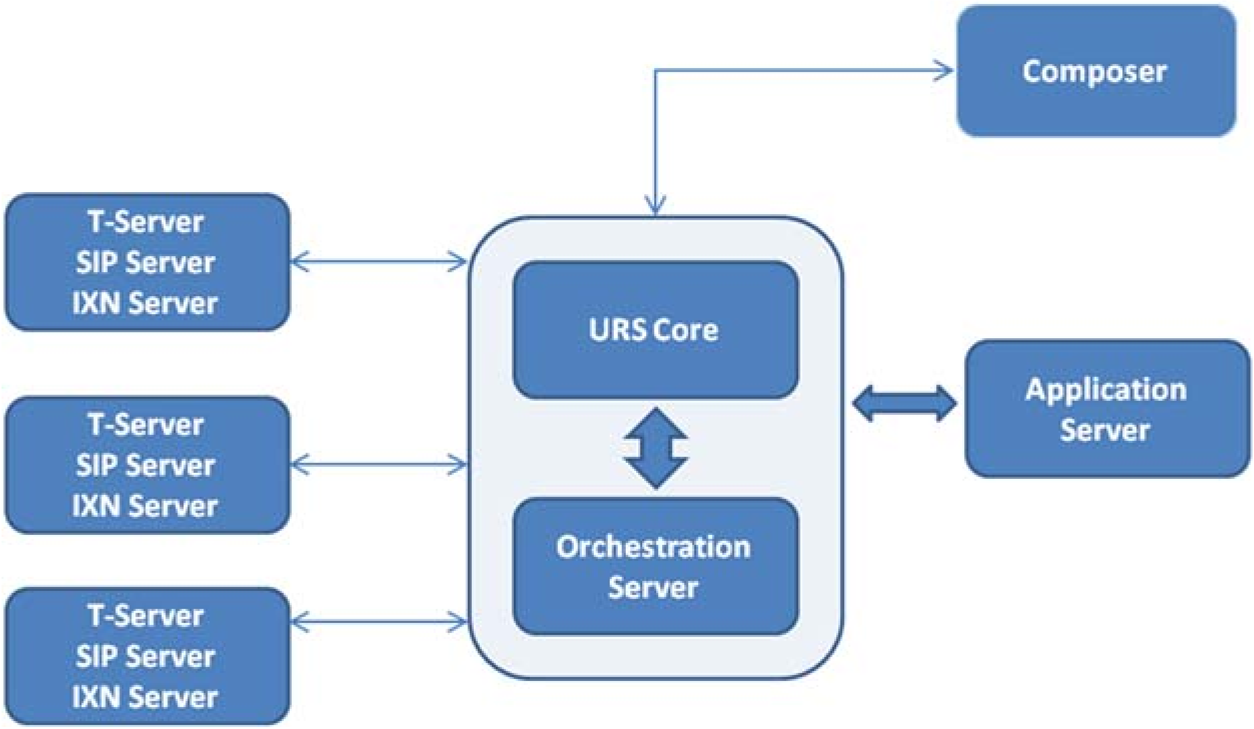


Figure 3: Orchestration Server Architecture

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 strategies and an Application Server provisions SCXML applications to ORS.

Voice Interactions

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

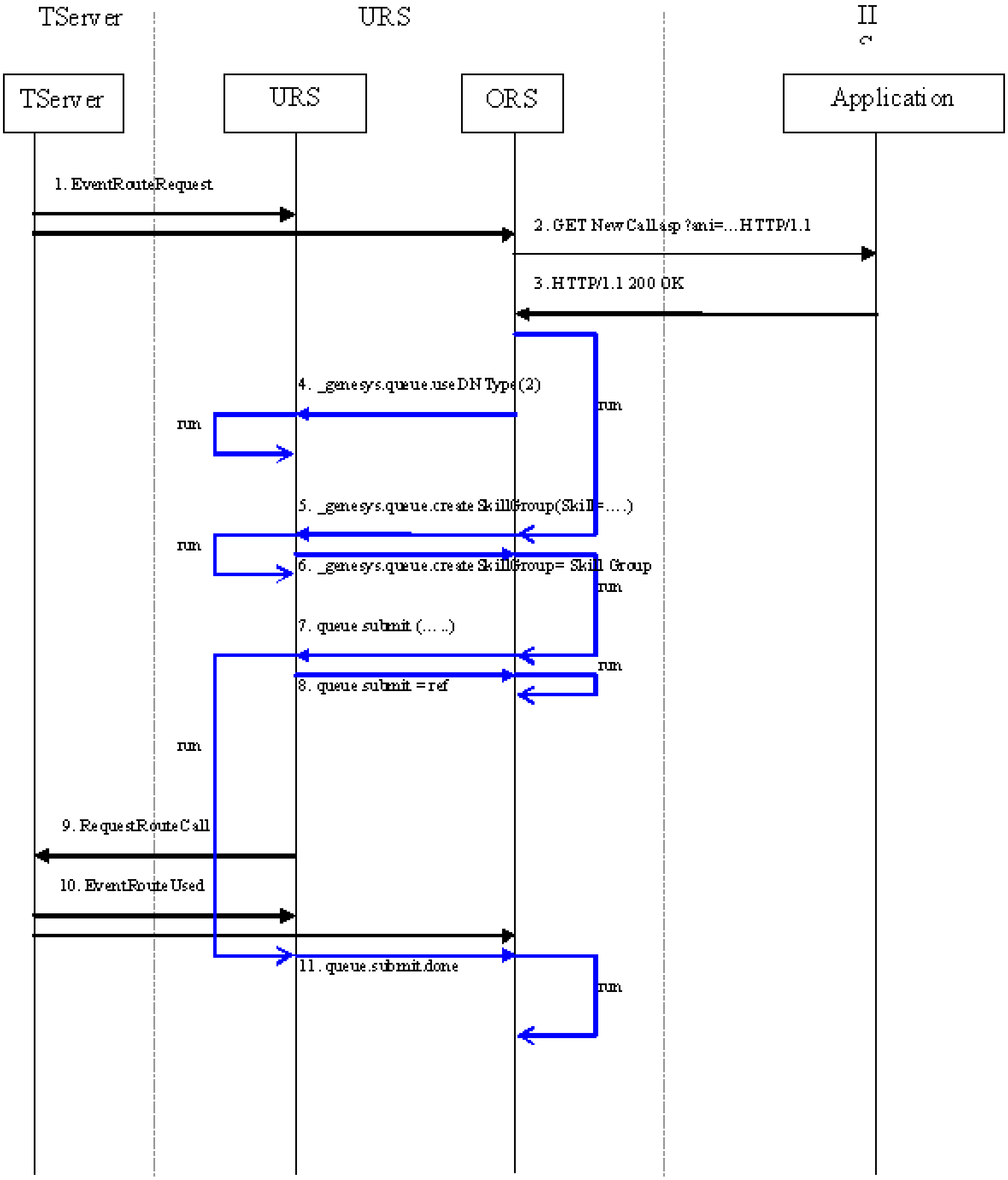

TServer
URS
II
S
TServer
URS
ORS
Application
1. EventRouteRequest
2. GET NewCall.asp ?ani=...HTTP/1.1
3. HTTP/1.1 200 OK
4. _genesys.queue.useDNType(2)
run
run
5. _genesys.queue.createSkillGroup(Skill=....)
run
6. _genesys.queue.createSkillGroup= Skill Group
run
7. queue.submit (.... ..)
run
8. queue.submit = ref
run
9. RequestRouteCall
10. EventRouteUsed
11. queue.submit.done
run

Figure 4: Sample Interaction Flow

The numbered sequences in the above figure are identified below:

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, then the 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 strategy.

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

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`).

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.

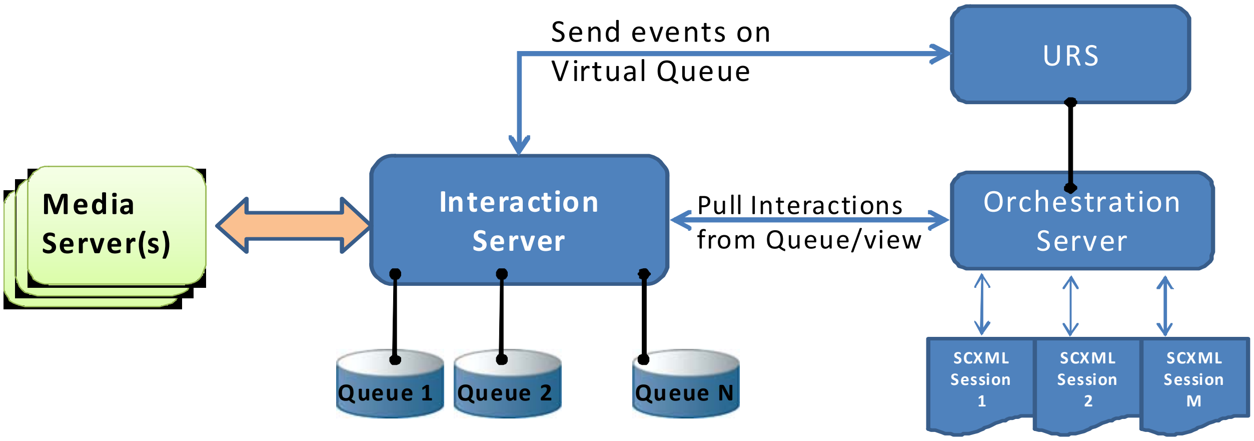


Figure 5: eServices-Specific ORS Architecture

The following are key design principals 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.

Basic eServices Interaction Routing

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

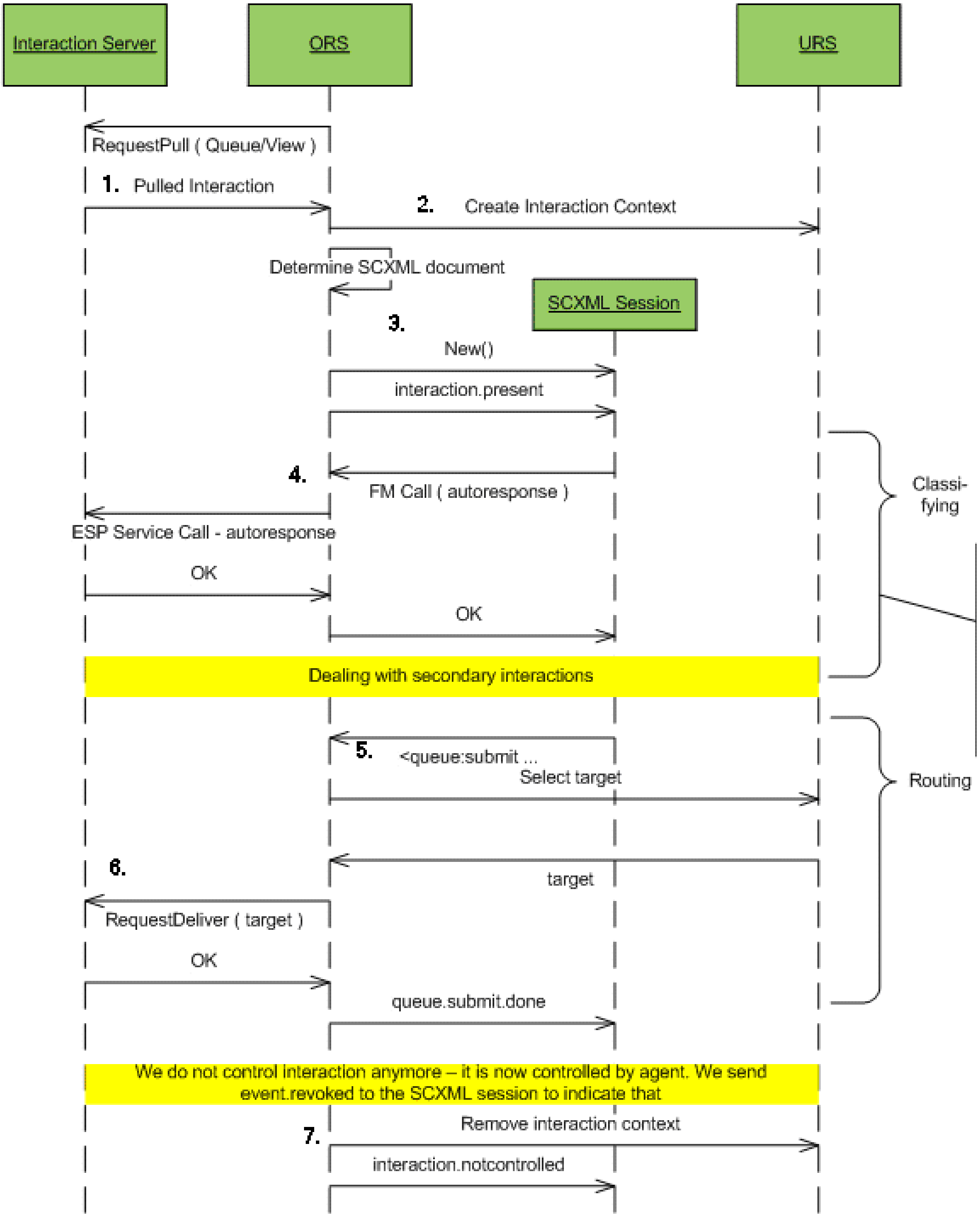

Interaction Server
ORS
URS
RequestPull (Queue/View)
1. Pulled Interaction
2. Create Interaction Context
Determine SCXML document
SCXML Session
3.
New()
interaction.present
4.
FM Call (autoresponse)
ESP Service Call - autoresponse
OK
OK
Classi-
fying
Dealing with secondary interactions
5. <queue:submit ...
Select target
Routing
target
6.
RequestDeliver (target)
OK
queue.submit.done
We do not control interaction anymore – it is now controlled by agent. We send event.revoked to the SCXML session to indicate that
7.
Remove interaction context
interaction.notcontrolled

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 strategy 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.

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

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 regarding persistence in the Orchestration 8.0.1 implementation.

- The persistence layer allows storage and retrieval of data related to active SCXML sessions. The amount of data stored is sufficient to fully restore the SCXML session 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 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 8.0.1, 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. This release supports Cassandra 6.x.
- 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 as required to restore the session context. 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 simply respond to an extended idle state for that session.

Persistent Storage Operation

The datastore type is determined by configuration.

The 8.0.1 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.

This release supports Cassandra 6.x.

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 39, and configuration options are described in Chapter 5, "Configuring Orchestration Server," beginning on page 53.

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. Once the Cassandra operation is complete there is no further action required.

Session Persistence

SCXML sessions are the current SCXML representation of an active session. These are constructed with the documents mentioned above. 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 39, and configuration options are described in Chapter 5, "Configuring Orchestration Server," beginning on page 53.

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. Once the Cassandra operation is complete there is no further action 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, and
- 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 7, 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 42 for a description of multiple ORS instances configured as clusters. All Orchestration Server instances are active, and all work with persistent storage.

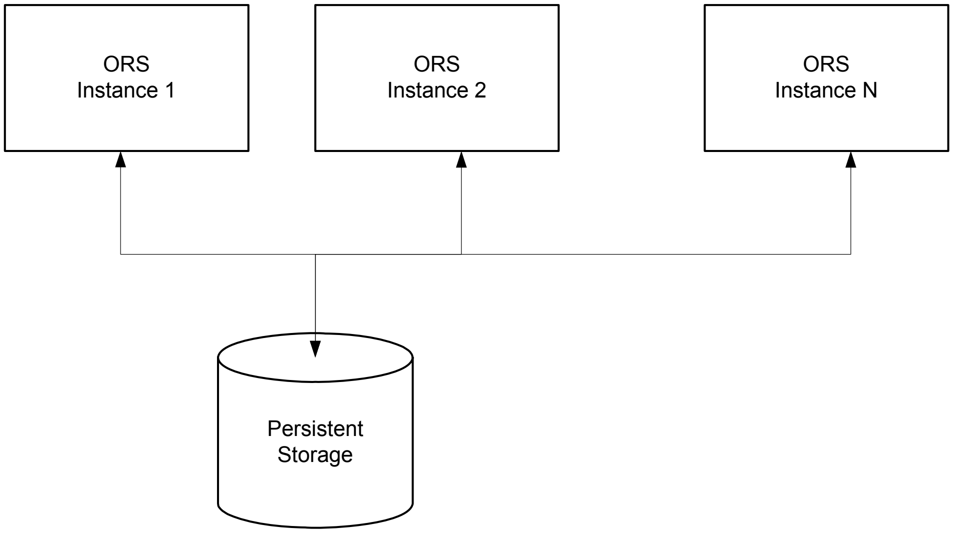


Figure 7: Multiple Orchestration Server Instances Running

As shown in Figure 8 on page 39, 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.

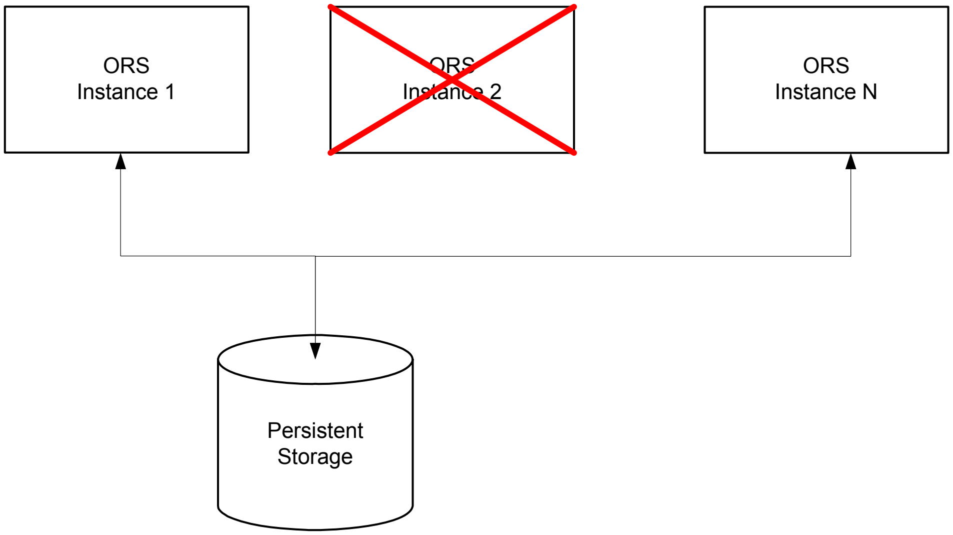


Figure 8: Remaining Orchestration Server Instances Work With Persistent Storage

Configuring Persistent Storage

Using persistent storage functionality in SCXML applications requires the following:

- Enable persistence in the ORS instance that will run the SCXML application and select the persistence type (Cassandra).
- Set values for the configuration options appropriate to the persistence type.
- Enable persistence in the SCXML application itself.

Enabling Persistence in ORS and Selecting Persistence Type

By default, persistence in ORS is disabled. Enabling persistence in ORS requires proper configuration of the ORS application involving these general steps:

1. Open the `persistence` section and set the ORS configuration option `enabled` to `true`.
2. Add the ORS configuration option `type` and enter a value of `cassandra`.

Refer to the detailed procedure below.

Procedure: Enabling persistent storage in ORS

Purpose: To prepare Orchestration Server to use persistent storage.

Start of procedure

1. Open Configuration Manager as described in Procedure: Logging into Configuration Manager, on page 57.
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 `enabled` option.
7. In the resulting `Edit Option` dialog box, change the `Option Value to true.`
8. Click `OK` to save.
9. Right-click inside the persistence options window and select `New` from the shortcut menu.
10. In the resulting `Edit Option` dialog box, in the `Option Name` field, type the following: `type.`

11. In the Option Value field, type cassandra to use the NoSQL Cassandra persistence method. This is the only persistence method supported in release 8.0.1.

    Refer to the section "Orchestration Server Option Descriptions" on page 87 for a more detailed description of this configuration option.
12. Click OK to save.

End of procedure

Next Steps

- Go to Procedure: Configuring the Cassandra persistence type in ORS.

Configuring the Cassandra Persistence Type in ORS

There are additional configuration options that you must add and set to configure the Cassandra persistence type. These are the general steps:

1. Add the ORS configuration option cassandra-listenport and enter a value for the Cassandra client port.
2. Add the ORS configuration option cassandra-nodes and enter values for the hostnames (or IP addresses) that will constitute the individual nodes in the Cassandra cluster.

Refer to the detailed procedure below.

Procedure: Configuring the Cassandra persistence type in ORS

Purpose: To prepare Orchestration Server to use the Cassandra persistence method.

Start of procedure

1. Right-click inside the persistence options window and select New from the shortcut menu.
2. In the resulting Edit Option dialog box, in the Option Name field, type the following: cassandra-listenport.
3. In the Option Value field, type the port number that represents the Cassandra client connection port.

   For example:

   ```
   9160
   ```

   Refer to the section "Orchestration Server Option Descriptions" on page 87 for a more detailed description of this configuration option.

4. Click `OK` to save.
5. Right-click inside the persistence options window again and select `New` from the shortcut menu.
6. In the resulting `Edit Option` dialog box, in the `Option Name` field, type the following: `cassandra-nodes`.
7. 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 87 for a more detailed description of this configuration option.
8. 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`.

The default value for this attribute is `true`, so no special steps are required to activate persistence for an SCXML application. You can *disable* persistence for an SCXML application by setting this attribute value to `false`.

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 9 on page 44 shows an example of a cluster deployment as described above.

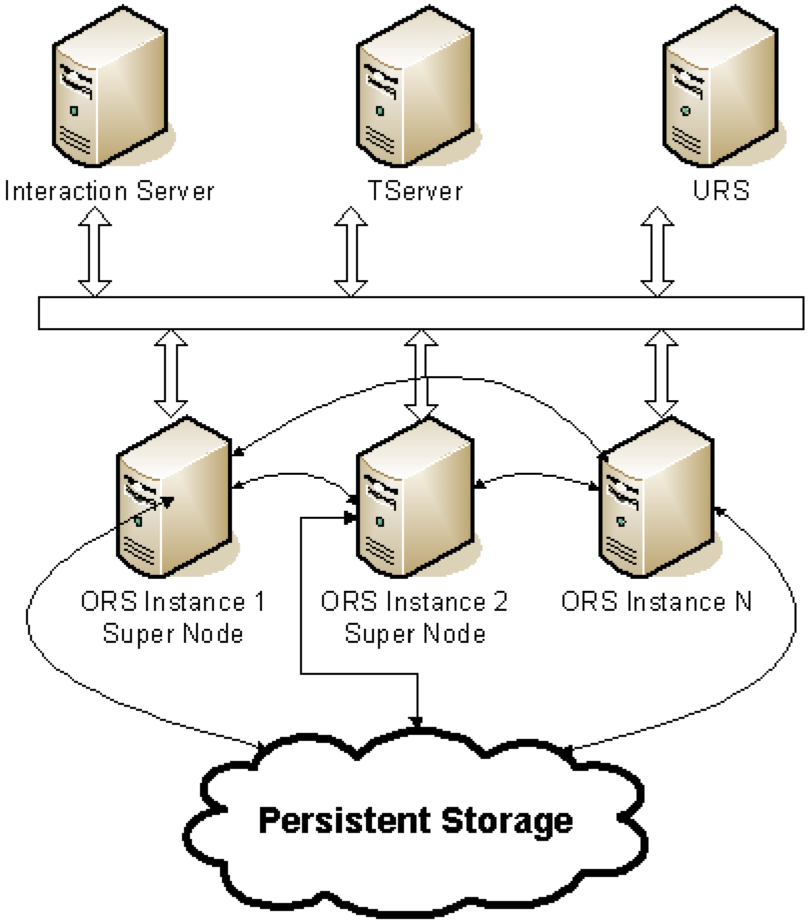


Figure 9: Typical ORS Cluster Deployment

The figure illustrates a typical cluster deployment, where:

- 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 (in this case, two).

To configure clustering in ORS, refer to the section “Configuring ORS Clustering” on page 72 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 115.

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 8.0 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

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 53.

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 49).

For information on installation and configuration of Framework components, see the *Framework 8.0 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 27 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 and Resource Capacity Wizard)

2. Routing Components (URS)
3. eServices components

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

4. Orchestration Server component (ORS)
5. Genesys Agent Desktop (optional)

Order for Individual Components

The individual components for deploying 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. License Manager
5. 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.
6. 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.

7. Message Server(s) (if not already installed with other Management Layer components)
8. Universal Routing Server
9. After installation of the above components, install eServices as described in the *eServices 8.0 Deployment Guide*.
10. 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 8.0 Resource Capacity Planning Guide*. For summary information on agent-capacity rules and how they can affect routing, see “Configuring the Template Options in the orchestration Section” on page 65.
11. Orchestration Server

What Each Component Does

Below 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*.

Chapter

5 Configuring Orchestration Server

Before you can install the Orchestration Server component as described in Chapter 7 on page 113, you must configure it.

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

See the *Universal Routing 8.0 Deployment Guide* for information on configuring Universal Routing Server (URS), Custom Server, and Interaction Routing Designer (IRD).

How to Configure

This document describes how to manually install Orchestration Server.

This 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 8.0 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 57
Import the appropriate Application Template for Orchestration Server.	Procedure: Importing the application template for Orchestration Server, on page 60
Create the Orchestration Server Application object.	Procedure: Creating/Configuring the Orchestration Server Application object, on page 61
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 63 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 65
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 66
Configure the template options in the persistence section: • enabled	Procedure: Viewing/changing template options in the persistence section, on page 67
Add and set non-template options in the persistence section: • cassandra-listenport • cassandra-nodes • type	Procedure: Adding and setting non-template options in the orchestration section, on page 66

Task Summary: Configuring With Configuration Manager (Continued)

Objective	Related Procedures and Actions
Configure the template options in the `scxml` section: • `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-preprocessor-cache-size` • `max-preprocessor-cached-docs` • `session-processing-threads`	Procedure: Viewing/changing template options in the scxml section, on page 69
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-key-password` • `http-ssl-random-file` • `https-proxy` • `persistence-max-active` • `system-id`	Procedure: Adding and setting non-template options in the scxml section, on page 70

Task Summary: Configuring With Configuration Manager (Continued)

Objective	Related Procedures and Actions
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 71
(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 72
(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 74
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 76
Configure options specified in Script objects of type CfgEnhancedRouting (Enhanced Routing Script): • 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 78 Procedure: Adding and setting the {Parameter Name} option in the ApplicationParms section of CfgEnhancedRouting Script objects, on page 79

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"

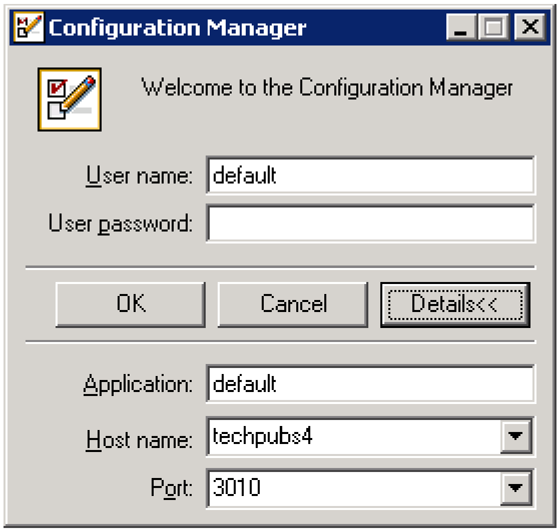
Procedure: Logging into Configuration Manager

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 8.0 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 10 shows an example.



Welcome to the Configuration Manager

User name: default

User password:

Application: default

Host name: techpubs4

Port: 3010

Figure 10: 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 machine where Configuration Server is running.
Port:	Port number used by Configuration Server.

3. Click `OK` in the login dialog box to open Configuration Manager.

Figure 11 on page 59 shows a sample Configuration Manager for a Multi-`Tenant` environment.

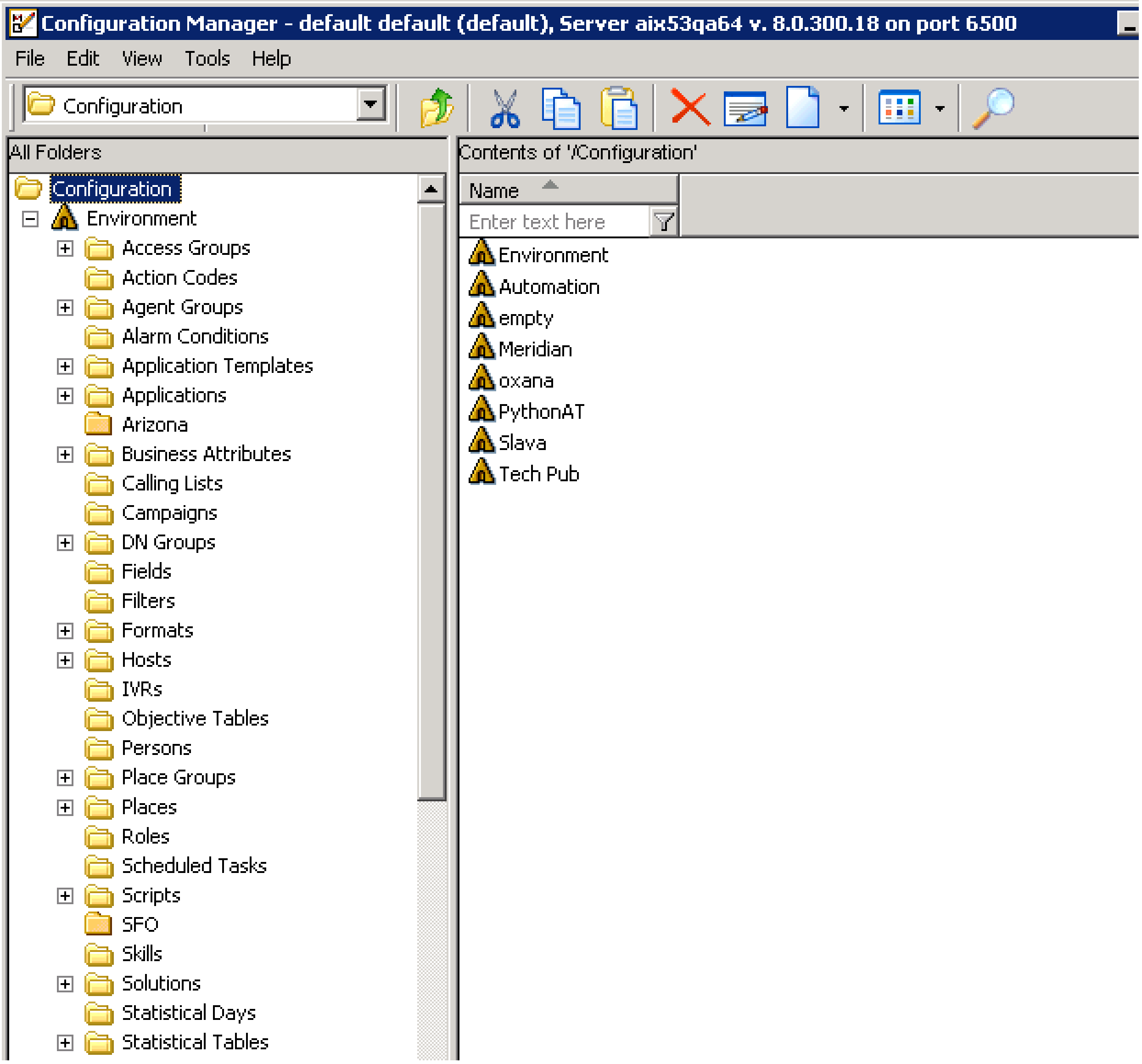


Figure 11: Configuration Manager

Note: If you wish to use the Management Layer and its Solution Control Interface (SCI as shown in Figure 25 on page 126) to stop and start applications, you must install Local Control Agent (LCA) as documented in the *Framework 8.0 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 60

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 57.
2. In Configuration Manager, select `Environment` > `Application Templates`.
3. Right-click `Application Templates`. From the shortcut menu that opens, select `Import Application Template`.
4. 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 may vary.)

   The file name is `OR_Server_801.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.
5. Select this file and click `Open`.
6. In the `Properties` dialog box, click `OK`.

End of procedure

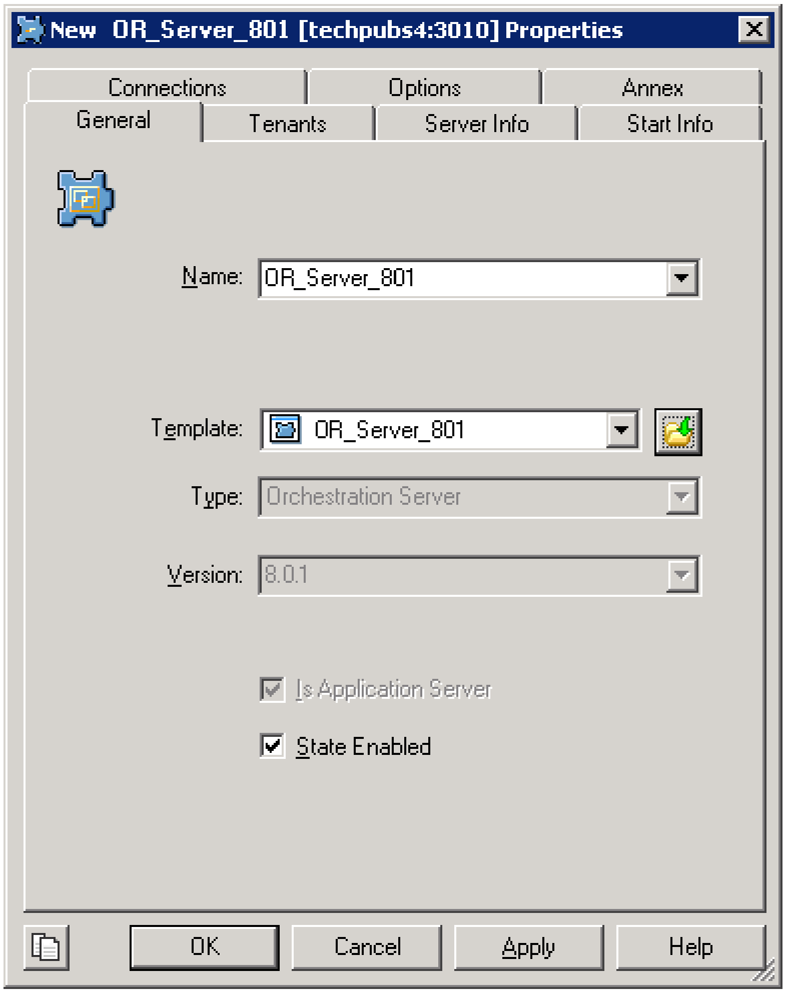
Next Steps

- 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 57.
2. Import the template as described on page 60.
3. In Configuration Manager, select `Environment` > `Applications`.
4. Right-click either the Applications folder or the subfolder where 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 you just imported, and double-click it to open the Orchestration Server `Application` object. Figure 12 shows an example.


New OR_Server_801 [techpubs4:3010] Properties

Name: OR_Server_801

Template: OR_Server_801

Type: Orchestration Server

Version: 8.0.1

☑ Is Application Server

☑ State Enabled

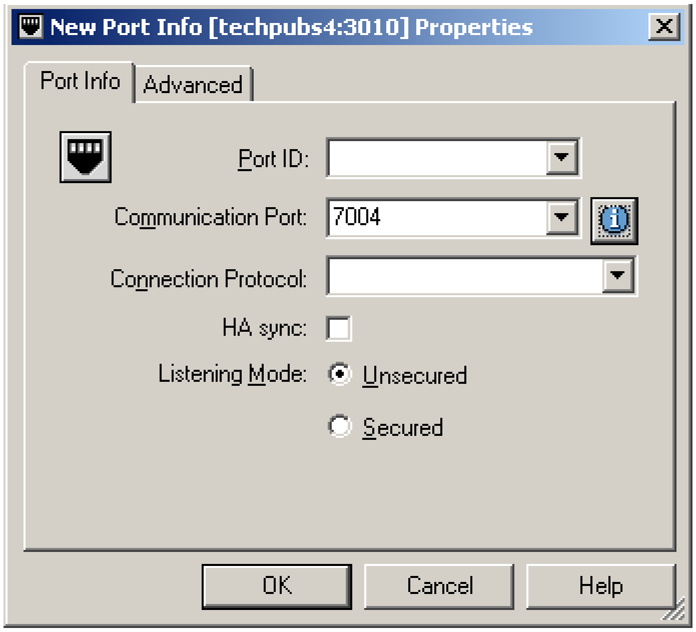
Figure 12: Orchestration Server Application Object

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 13).



Port ID:

Communication Port: 7004

Connection Protocol:

HA sync: □

Listening Mode: ● Unsecured ○ Secured

Figure 13: New Port Info Dialog Box

Note: For information on this dialog box, see the Port Info Tab topic in the *Framework 8.0 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, then 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> -l <the full path and name of license file>)`

  The license file path must also be enclosed in quotation marks.

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 associated with client-side port connections, see the *Genesys 8.0 Security Deployment Guide*.

End of procedure

Next Steps

- Continue with Procedure: Viewing/changing template options in the orchestration section, on page 65

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 *Genesys 8.0 Security Deployment Guide*. Table 3 on page 26 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 Options 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 57.
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 to another valid value.

   Refer to the section “Orchestration Server Option Descriptions” on page 87 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 57.
   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 87, 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

- enabled

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 57.
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 `enabled` option.
7. In the resulting `Edit Option` dialog box, keep the default `Option Value` `false,` or change to `true`.

   Refer to the section "Orchestration Server Option Descriptions" on page 87 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-listenport`
- `cassandra-nodes`
- `type`

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 57.

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 one of the option names listed in "Persistence section non-template options" on page 67 above.
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 87, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 87 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 above to add another option in this section.

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

- `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-preprocessor-cache-size`
- `max-preprocessor-cached-docs`
- `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 57.
   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 68 above.
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 87 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 above 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-key-password`

- `http-ssl-random-file`
- `https-proxy`
- `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 57.
   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 69 above.
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 87, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 87 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 above 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 57.
   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" above.
5. In the `Option Value` field, type the default value as indicated in the section "Orchestration Server Option Descriptions" on page 87, or type a different valid value.

   Also refer to the section "Orchestration Server Option Descriptions" on page 87 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 above to add another option in this section.

End of procedure

Configuring ORS Clustering

Refer to Chapter 3, "Persistence, High Availability, and Load Balancing," on page 35 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 39 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 hostname or IP address you'll 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 below.

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 57.
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 the following: `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 87 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 the following: `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 87 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

To identify a web service (hostname and port), do the following:

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 below.

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 57.
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 the following: 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 87 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 the following: 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 87 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 below.

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 57.
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`, then 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 8.0 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 57.
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 101 for a full description of this configuration option its valid values. Table 6, “URL Parameter Elements for application option,” on page 101 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

- `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 57.
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 77 above.
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 87, or type a different valid value.

   Also refer to the section “Orchestration Server Option Descriptions” on page 87 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 105 for a full description of this configuration option its valid values. Table 7, “URL Parameter Elements for url option,” on page 105 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

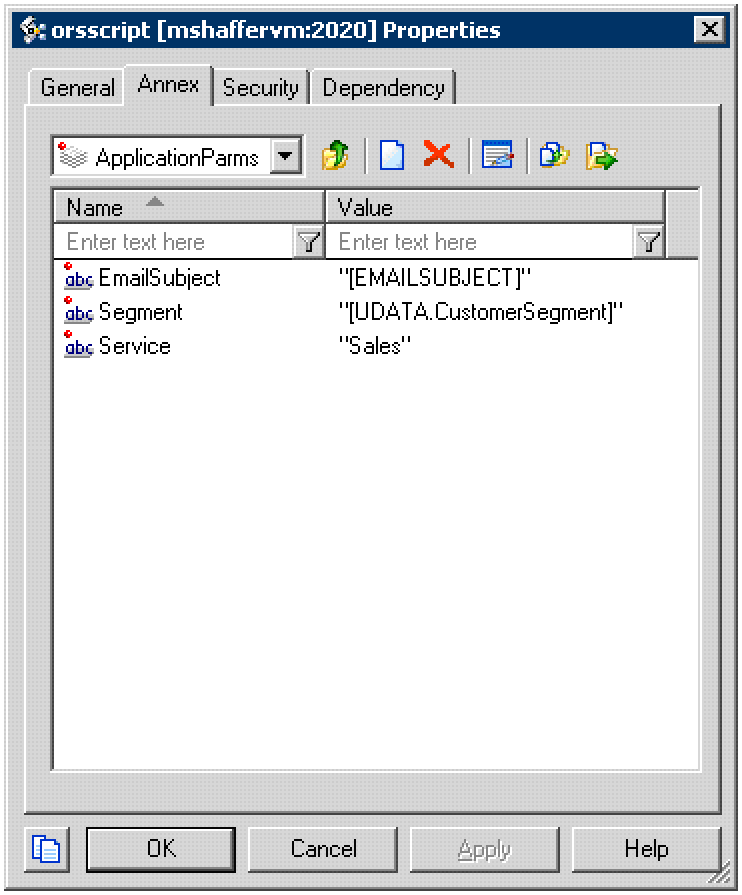
In addition, an option can be used to specify a string that represents a parameter value to be passed to the application. The ApplicationParms section contains the values for data elements that may be referred to within the SCXML application.

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 57.
   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 106 for a full description of this configuration option its valid values. Table 8, "Parameter Elements for ApplicationParms," on page 107 provides useful information about parameters that can be added. Figure 14 below shows an example of the use of the `ApplicationParms` section.



ApplicationParms

Name	Value
EmailSubject	"[EMAILSUBJECT]"
Segment	"[UDATA.CustomerSegment]"
Service	"Sales"

Figure 14: Script Object Annex Tab for ApplicationParms

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

End of procedure

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` may 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 8.0 Stat Server User's Guide*.

Premise T-Server for Network Routing

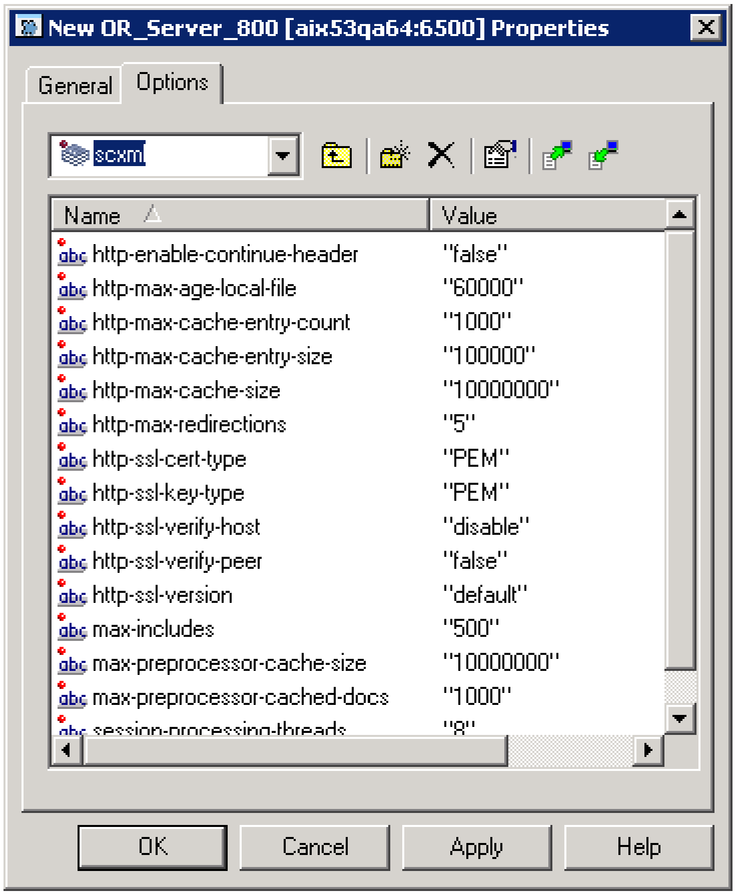
Refer to the appropriate T-Server document for configuring a premise T-Server application for Network Routing.

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 15 shows an example of the properties window of the Orchestration Server `Application` object in Configuration Manager with the `scxml` section visible.



Name	Value
http-enable-continue-header	"false"
http-max-age-local-file	"60000"
http-max-cache-entry-count	"1000"
http-max-cache-entry-size	"100000"
http-max-cache-size	"10000000"
http-max-redirections	"5"
http-ssl-cert-type	"PEM"
http-ssl-key-type	"PEM"
http-ssl-verify-host	"disable"
http-ssl-verify-peer	"false"
http-ssl-version	"default"
max-includes	"500"
max-preprocessor-cache-size	"10000000"
max-preprocessor-cached-docs	"1000"
session-processing-threads	"8"

Figure 15: 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`, and `log` sections of the `Options` tab of the Orchestration Server `Properties` dialog box. If you are configuring clustering, you also 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 where 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*, then in alphabetical order.

Table 5: Orchestration Server Options

Name	Brief Description	Detailed Description Page
orchestration section (ORS `Application` object)		
mcr-pull-interval	The number of milliseconds between attempts to pull interactions from the queues and/or views managed by the Orchestration Server.	87
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.	87
persistence section (ORS `Application` object)		
cassandra-listenport	Specifies the Cassandra client connection port when using Cassandra-based persistence method (`type = cassandra`)	88
cassandra-nodes	Semi-colon-separated list of hostnames or IP addresses when using Cassandra-based persistence method (`type = cassandra`)	88
enabled	Specifies whether persistence should be enabled or disabled	89
type	Specifies the type (method) of persistence operation to use: `cassandra`	89

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
scxml section (ORS `Application` object)		
http-enable-continue-header	Specifies whether to enable or disable the `100-continue` header in the HTTP 1.1 post request.	89
http-max-age-local-file	The maximum age of a local file, in milliseconds	90
http-max-cache-entry-count	The maximum number of entries that can be stored in the cache	90
http-max-cache-entry-size	The maximum size of each cache entry, in bytes	90
http-max-cache-size	The maximum size of the HTTP cache, in bytes.	91
http-max-redirections	The maximum number of times to follow the `Location:header` in the HTTP response	91
http-no-cache-urls	A comma-delimited list of substrings to prevent a response from being cached.	91
http-proxy	The HTTP proxy server (if applicable).	91
http-ssl-ca-info	The file name holding one or more CA certificates with which to verify the peer.	92
http-ssl-ca-path	The path holding one or more CA certificates with which to verify the peer.	92
http-ssl-cert	The file name of the SSL certificate.	92
http-ssl-cert-type	The SSL Certificate type (PEM or DER)	93
http-ssl-cipher-list	The cipher list as defined by OpenSSL.	93
http-ssl-key	The path or file name of the SSL private key.	93
http-ssl-key-password	The SSL key password.	94
http-ssl-key-type	The SSL Key type (PEM or DER)	94
http-ssl-random-file	The file which is read from in order to see the random engine for SSL.	94
http-ssl-verify-host	Specifies how the common name from the peer certificate should be verified during the SSL handshake	95

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
http-ssl-verify-peer	Specifies whether or not the system should verify the peer's certificate	95
http-ssl-version	The SSL version to be used	95
https-proxy	The HTTPS proxy server (if applicable).	96
max-includes	The maximum number of documents that may be included using `<xi:include>`	96
max-preprocessor-cache-size	The amount of memory, in bytes, that the `<xi:include>` pre-processor cache can use	96
max-preprocessor-cached-docs	The maximum number of items that the `<xi:include>` pre-processor cache can have	97
persistence-max-active	Allows setup of a maximum number of active sessions that the SCXML engine will keep in memory.	97
session-processing-threads	The number of threads in the thread pool	97
system-id	Allows setup of the Orchestration Server (ORS) system ID in order to have unique session IDs across ORS instances.	98
log section (ORS `Application` object)		
all buffering expire segment verbose	These are standard Genesys log options.	n/a
x-server-trace-level	The level of tracing to be enabled for the Orchestration Server	98
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.	98

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
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	98
x-print-attached-data	Specifies whether or not attached data should be formatted and printed in the logs	99
cluster section (ORS `Application` object)		
name	The string name of the Orchestration Server cluster this ORS application node belongs to	99
super_node	Specifies whether the node should act as a Super Node in the cluster	100
web_services section (ORS `Application` object)		
hostname	The IP address or hostname to expose externally for the Web Services functional module.	100
port	The port number associated with the IP address or hostname identified for the web services in the `hostname` option.	100
Orchestration section (DN Extension or RoutePoint, or Interaction Queue)		
application	Specifies the URL of the SCXML document to load.	101
Application section (Script object `CfgEnhancedRouting`)		
fetch-timeout	Specifies the time (in milliseconds) before a document fetch is abandoned.	102
http-useragent	The string to use in the HTTP header field `User Agent`, which identifies the application to the web server.	103
http-version	The HTTP version to use when fetching documents from the application server	103
max-age	Tells the Orchestration Server how long an application script can be cached.	103
max-duration	The maximum duration, in milliseconds, of a single ECMAScript script	104

Table 5: Orchestration Server Options (Continued)

Name	Brief Description	Detailed Description Page
max-loop-count	The maximum number of loops that can be performed in a single ECMAScript script	104
max-stale	The number of seconds to extend the life of a cached file.	105
url	Specifies the URL of the SCXML document to load.	105
ApplicationParms section (Script object `CfgEnhancedRouting`)		
{Parameter Name}	Specifies a string that represents a parameter value to be passed to the application.	106

The "Orchestration Server Option Descriptions" section below 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: `1 to 2147483647`
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
```

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 iterationsof 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: `unknown`
Valid values: `[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 option must be specified if `cassandra` is chosen for the `type` option (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
```

enabled

Option section: `persistence`
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, if set to `true`, specifies that persistence should occur.

For example:

```
persistence/enabled = true
```

type

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

This option must be specified if the `enabled` option is set to `true`, which enables persistence operation. This option sets the persistence method. The only valid option for the 8.0.1 release is `cassandra` (Cassandra-based persistence method). Refer to Chapter 3, “Persistence, High Availability, and Load Balancing,” on page 35 for details on this persistence method.

For `cassandra`, you must also specify values for both the `cassandra-listenport` and `cassandra-nodes`

For example:

```
persistence/type = cassandra
```

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: ""
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: "" (blank)

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: ""
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: ""
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
```

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: ""
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: ""
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
```

http-ssl-key-password

Option section: scxml
Configuration object: ORS Application object
Default value: ""
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/http-ssl-key-password = agent007
```

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: ""
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: "" `(blank)`
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-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
```

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
```

name

Option section: `cluster`
Configuration object: ORS `Application` object
Default value: `none`
Valid values: `[any string name]`
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 43 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).

```
cluster/super_node = true
```

hostname

Option section: `web_services`
Configuration object: ORS `Application` object
Default value: `{empty}`
Valid values: `[any string name]`
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}`
Valid values: `[any positive integer less than 64000]`
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 `64000`.

```
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 101 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
[UDATA]	Expanded to the entire user data of the interaction in the format: name1=value1&name2=value2&·

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.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
```

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/8.0`
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. Ifapplication 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, thescript is aborted and an error eventis 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 7 are substituted at run-time based on the information in the interaction.

Table 7: 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

Table 7: URL Parameter Elements for url option (Continued)

Formatting Element	Description
[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`

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 8 on page 107, for example:

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

Table 8: Parameter Elements for ApplicationParms

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.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]

[1] URLs passed in as parameters are not resolved by the Orchestration Server. They are passed to the strategy as a string. If necessary, the strategy can retrieve the value associated with the URL by using the `<fetch>` operation.

Chapter

6 SCXML Strategy Support

This chapter contains the following topics:

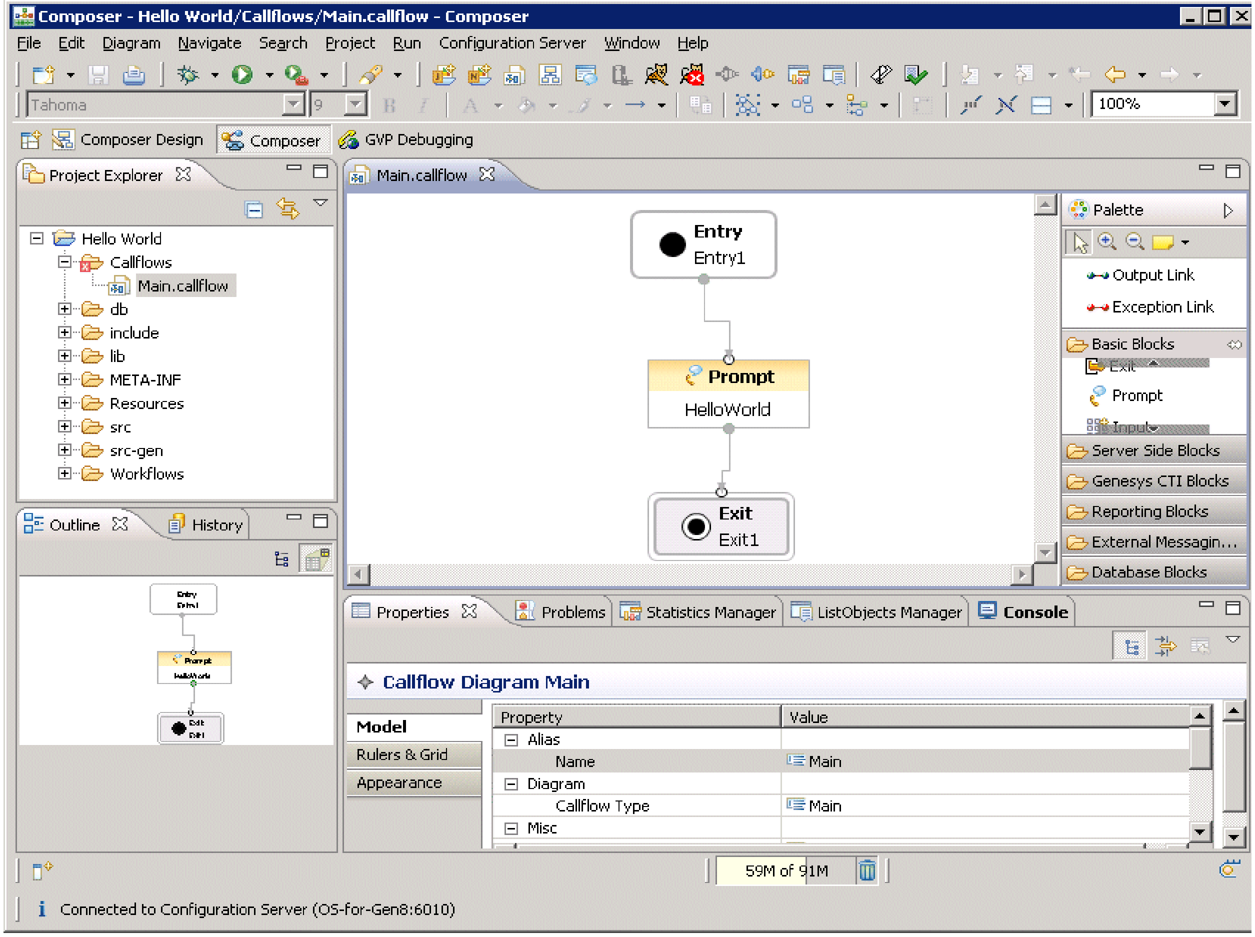
Creating SCXML-Based Strategies

You can create SCXML-based strategies using the following methods:

- Any simple text editor such as Notepad or an XML-based editor that you are already comfortable with.
- 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 8.0.x Help*. Also see the *Genesys Composer 8.0 Deployment Guide*.

Genesys Composer

Genesys Composer has a drag and drop visual designer for creating workflows when building SCXML-based strategies (see Figure 16).



Callflow Diagram Main

Property	Value
⊟ Alias	
Name	Main
⊟ Diagram	
Callflow Type	Main
⊟ Misc	

Connected to Configuration Server (OS-for-Gen8:6010)

Figure 16: Composer

Composer supports Inbound Voice and some eServices functionalities.

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 callflows (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 strategy (created inside or outside of Composer) if Microsoft Internet Information Services is installed and running on the machine:

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

End of procedure

Strategy Samples

Orchestration Server supports only SCXML routing strategies. The *Genesys 8.0 SCXML Samples* document contains numerous SCXML routing strategy examples. The samples are not designed for use in a Production environment. Instead, use them to get started configuring your own strategies, subroutines, and list objects. Consider them as guides when developing your own objects adjusted to your company's specific business needs.

Note: *Genesys 8.0 SCXML Samples* provides examples of various types of voice and eServices SCXML routing strategies. The information includes strategy flows and the properties of the various strategy-building objects. If you need an example of how to use strategy-building objects, start with this guide.

Chapter

7 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 53.

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:

Note: For a list of supported operating systems and databases, see *Genesys Supported Operating Environment Reference Manual* available on the Genesys Technical Support website 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 114
Install components on Windows	Procedure: Installing Orchestration Server on Windows using the Installation Wizard, on page 115
Install components on Linux	Procedure: Installing Orchestration Server on a UNIX platform, on page 122

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 machine as documented in the *Framework 8.0 Deployment Guide*.

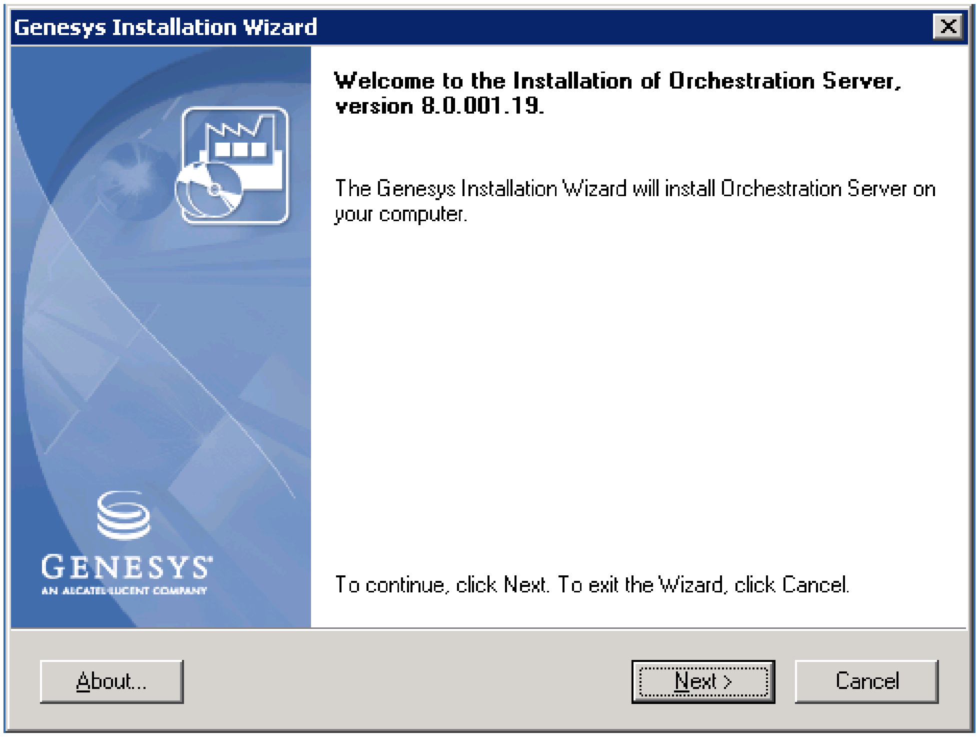
Procedure: Installing Orchestration Server on Windows using the Installation Wizard

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.

   Install Shield opens the `Welcome` screen (see Figure 17).



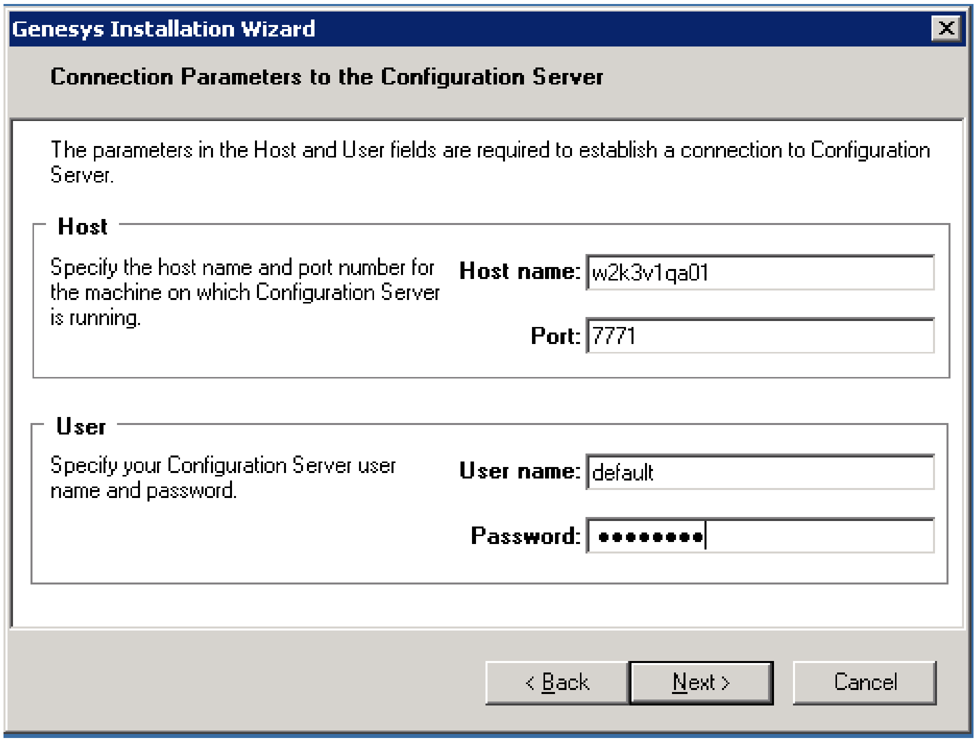
Welcome to the Installation of Orchestration Server, version 8.0.001.19.

The Genesys Installation Wizard will install Orchestration Server on your computer.

To continue, click Next. To exit the Wizard, click Cancel.

Figure 17: Welcome Screen, ORS Installation Wizard

2. Click `Next`. The `Connection Parameters to the Configuration Server` screen appears (see Figure 18).



Connection Parameters to the Configuration Server

The parameters in the Host and User fields are required to establish a connection to Configuration Server.

Host

Specify the host name and port number for the machine on which Configuration Server is running.

Host name: w2k3v1qa01

Port: 7771

User

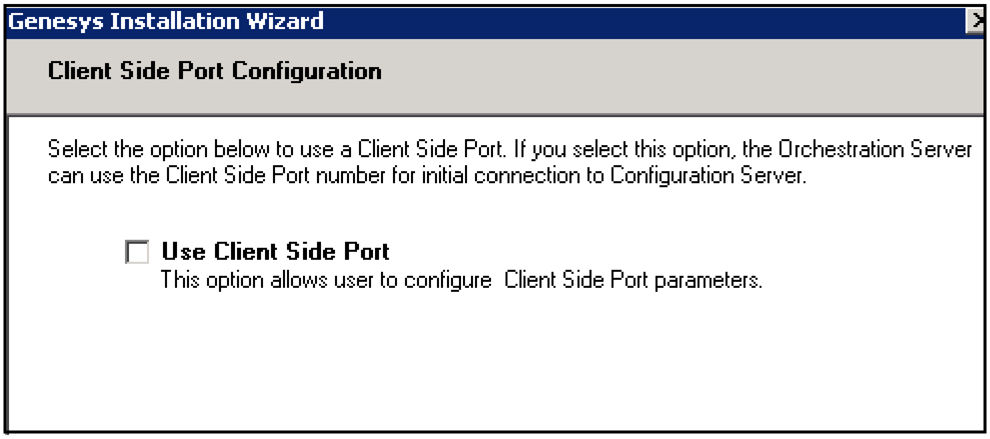
Specify your Configuration Server user name and password.

User name: default

Password: ●●●●●●●●

Figure 18: Connection Parameters to the Genesys Configuration Server

3. Under Host, specify the host name and port number for the machine 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 19).



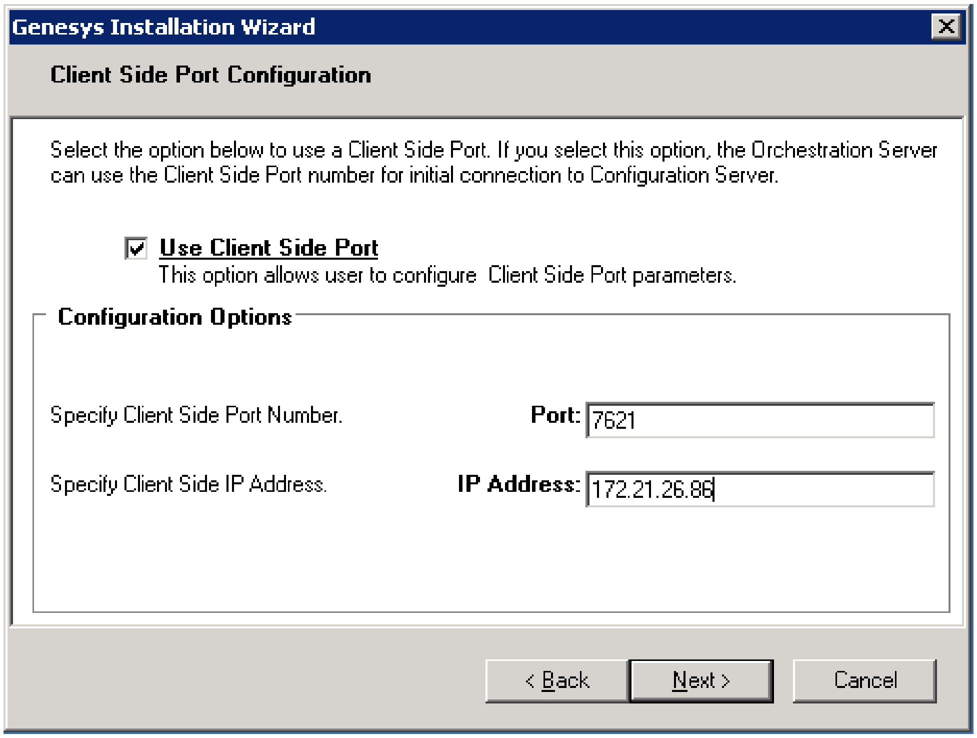
Client Side Port Configuration

Select the option below to use a Client Side Port. If you select this option, the Orchestration Server can use the Client Side Port number for initial connection to Configuration Server.

□ Use Client Side Port
This option allows user to configure Client Side Port parameters.

Figure 19: 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 8.0 Security Deployment Guide*, select the `Use Client Side Port` check box to drop down additional fields (see Figure 20) You also set up the client-side port when using the RESTful interface (refer to “Load Balancing for RESTful Interface” on page 45).



Genesys Installation Wizard

Client Side Port Configuration

Select the option below to use a Client Side Port. If you select this option, the Orchestration Server can use the Client Side Port number for initial connection to Configuration Server.

☑ Use Client Side Port
This option allows user to configure Client Side Port parameters.

Configuration Options

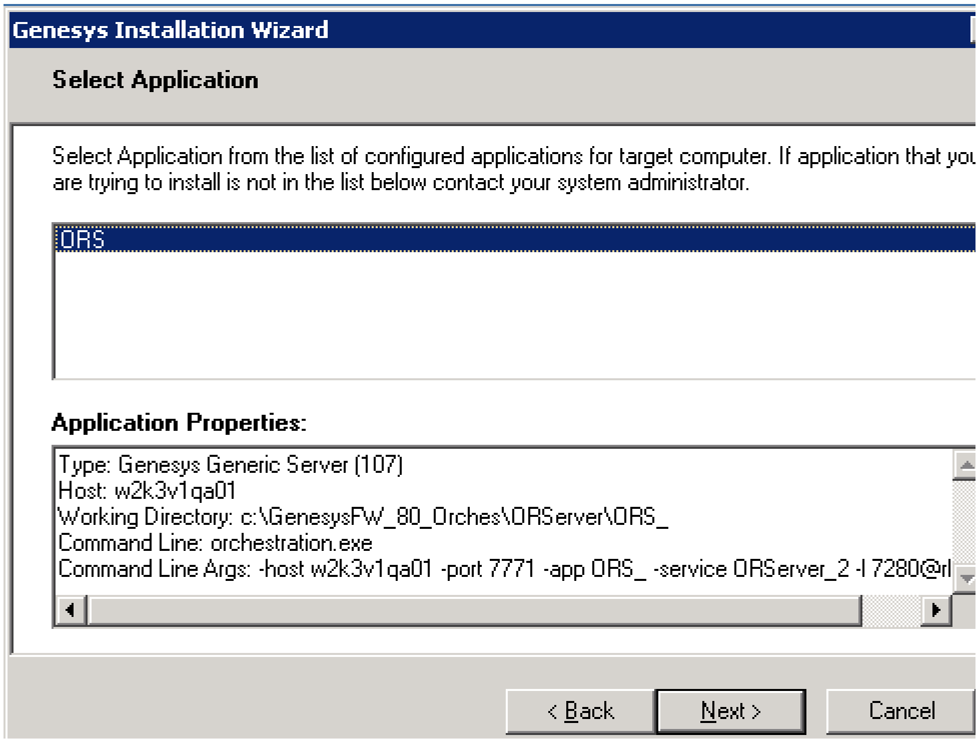
Specify Client Side Port Number. Port: 7621

Specify Client Side IP Address. IP Address: 172.21.26.86

Figure 20: 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 machine where 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.



Select Application from the list of configured applications for target computer. If application that you are trying to install is not in the list below contact your system administrator.

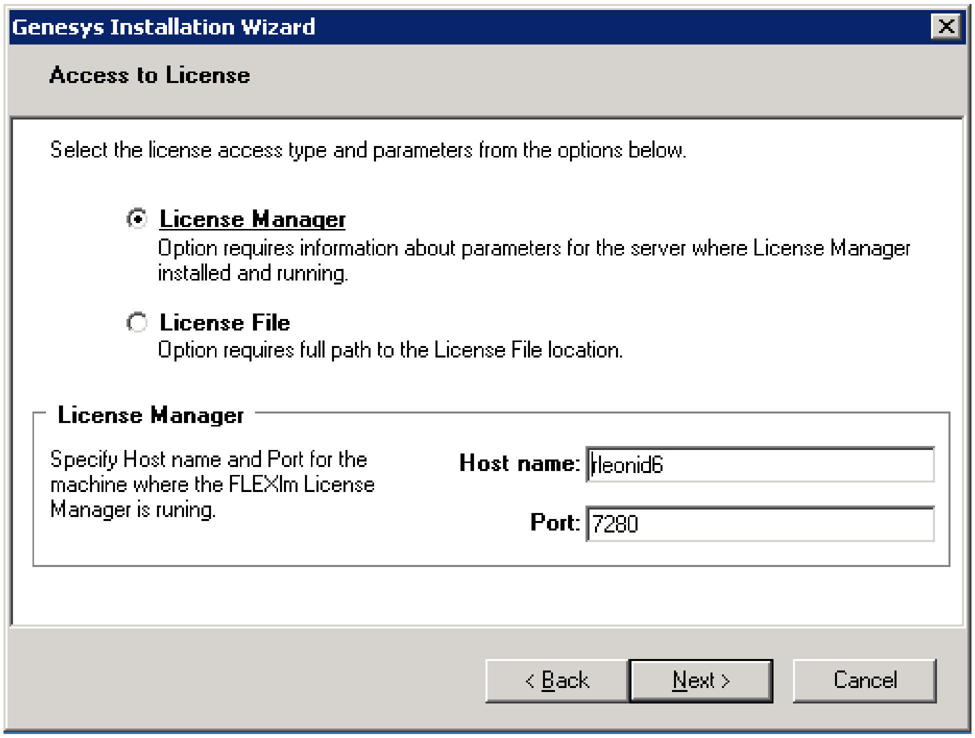
ORS

Application Properties:

Type: Genesys Generic Server (107)
Host: w2k3v1qa01
Working Directory: c:\GenesysFW_80_Orches\ORServer\ORS_
Command Line: orchestration.exe
Command Line Args: -host w2k3v1qa01 -port 7771 -app ORS_ -service ORServer_2 -l 7280@rl

Figure 21: Select Application

8. Click Next. The Select Application screen appears. Figure 21 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 Access to License screen appears. Figure 22 shows the screen with License Manager selected.



Genesys Installation Wizard

Access to License

Select the license access type and parameters from the options below.

● License Manager
Option requires information about parameters for the server where License Manager installed and running.

○ License File
Option requires full path to the License File location.

License Manager

Specify Host name and Port for the machine where the FLEXlm License Manager is runing.

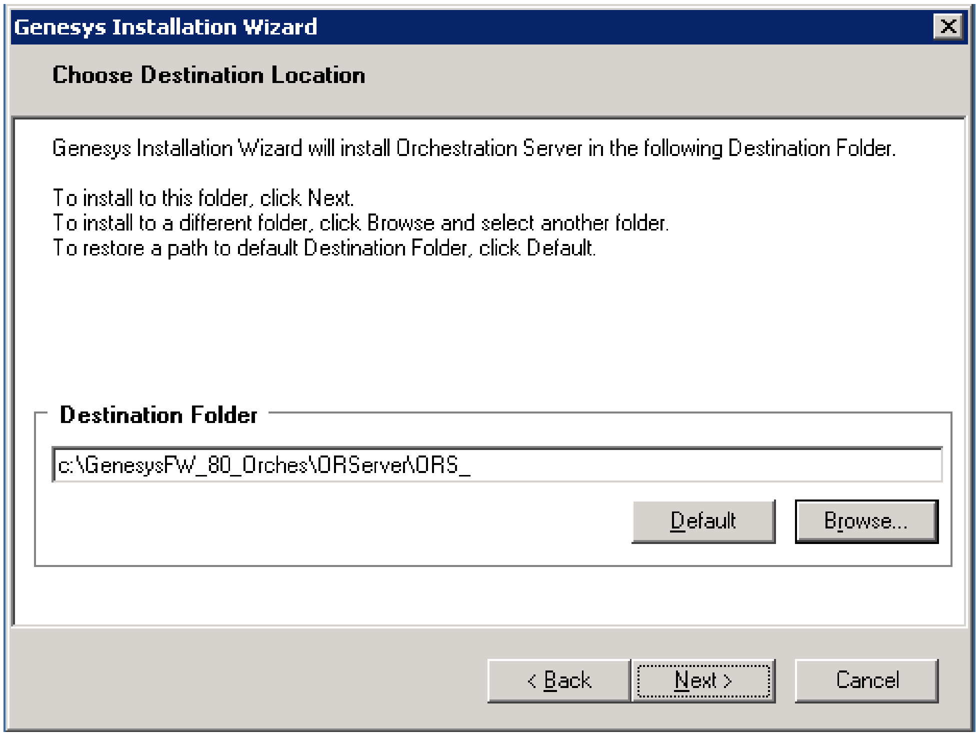
Host name: leonid6

Port: 7280

Figure 22: Access to License

11. Select one of the following options:
    - `License Manager`. Requires host name and port information for the server where License Manager is installed and running as shown in Figure 22.
    - `License file`. Requires the full path to the license file location.

    These instructions assume you select the `License Manager` radio button.
12. Click `Next`. The `Choose Destination Location` screen appears (see Figure 23).



Genesys Installation Wizard

Choose Destination Location

Genesys Installation Wizard will install Orchestration Server in the following Destination Folder.

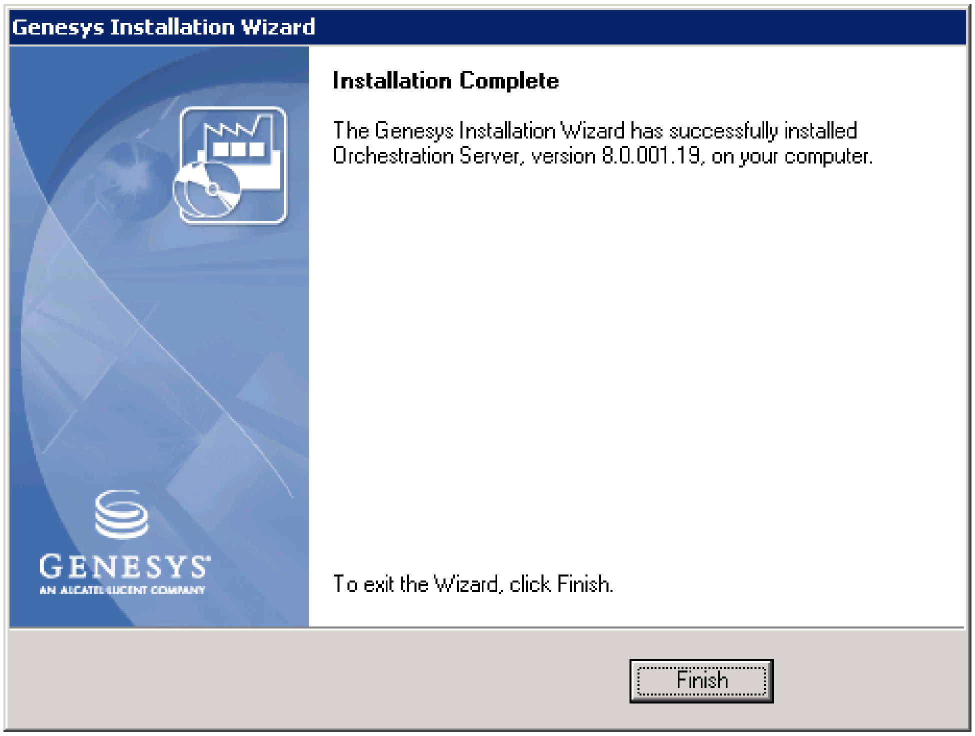
To install to this folder, click Next.
To install to a different folder, click Browse and select another folder.
To restore a path to default Destination Folder, click Default.

Destination Folder

c:\GenesysFW_80_Orches\ORServer\ORS_

Figure 23: Choose Destination Location

13. Under `Destination Folder`, keep the default or browse for the installation location for Orchestration Server.
14. Click `Next`. The `Ready to Install` screen appears.
15. 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 (see Figure 24).



Genesys Installation Wizard

Installation Complete

The Genesys Installation Wizard has successfully installed Orchestration Server, version 8.0.001.19, on your computer.

To exit the Wizard, click Finish.

Figure 24: Installation Complete

16. Click `Finish` on the `Installation Complete` screen.

End of procedure

Installing on UNIX-Based Platforms

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

Installing Orchestration Server on a UNIX-Based Platform

Complete the following directions 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
- Solaris SPARC 32/64-bit

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 114).
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 Hostname`
   - `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 (Solaris) type that you use.

12. Respond to this prompt:

```
Please select the format for your license location. Enter the number
1 or 2.
1) Full path to the license file
2) License Manager port and host
```

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 machine as documented in the *Framework 8.0 Deployment Guide*.

Chapter

8 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

Before Orchestration Server can be started, it must be configured and installed. For more information, see Chapter 5, "Configuring Orchestration Server," on page 53 and Chapter 7, "Installing Orchestration Server," on page 113.

Before starting Solution Control Interface (SCI), start the Configuration Server, and Solution Control Server. Make sure that Local Control Agent (LCA) is running.

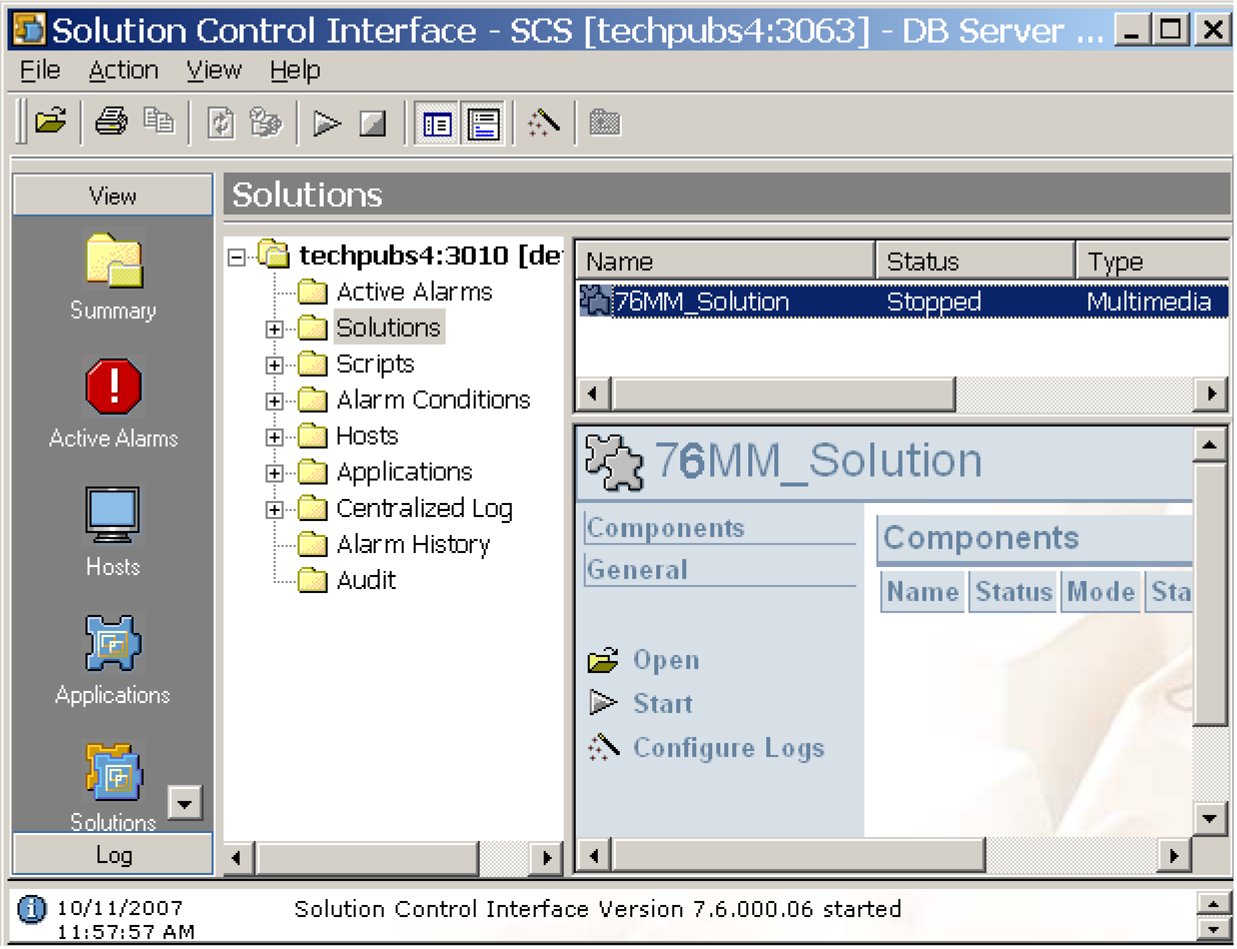
Note: SCI, Solution Control Server, and LCA are a part of the Management Layer. See the *Framework 8.0 Management Layer User's Guide* for information about these components.

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 127
Start Orchestration Server on UNIX-Based Platforms	Procedure: Starting Orchestration Server on UNIX-Based Platforms, on page 127

Start Orchestration Server from the Solution Control Interface (SCI). Figure 25 shows an example that could include Orchestration Server as well as other server applications.



Name	Status	Type
76MM_Solution	Stopped	Multimedia

```
10/11/2007 11:57:57 AM   Solution Control Interface Version 7.6.000.06 started
```

Figure 25: 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 8.0 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 mustalso 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 listedin 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 129

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 simply prints the version number to the log and then exits.

Chapter

9 Uninstalling Orchestration Server

This chapter describes how to uninstall Orchestration Server. It includes the following topics:

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 132
Uninstall Orchestration Server manually	Procedure: Manually removing Orchestration Server on Windows, on page 132 Procedure: Manually removing Orchestration Server on UNIX-Based Platforms, on page 133

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 be able to use it as data storage in the Orchestration Persistent Storage component. Additional information may be found at `http://www.riptano.com/`.

Note: Because third-party links may change, the link shown above (accurate at the time of this writing) is not live.

This Appendix includes the following sections:

We recommend 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.

Cassandra requires an installed Java 6.

For Linux installations, the UNIX style paths (and other UNIX conventions) should be used.

Installing, Configuring, Starting, and Testing a Cassandra Node

The following is an overview of the steps involved in the installation and configuration of a single Cassandra node:

1. Download the install image and extract it.
2. Configure Cassandra Server (copy and edit the `storage-conf.xml` file).
3. Start the server.

4. Test the server.

Task Summary: Installing/Configuring a Cassandra Node

Objective	Related Procedures and Actions
Download the install image and extract it.	Procedure: Downloading and extracting the install image, on page 137
Configure Cassandra Server by copying and editing the `storage-conf.xml` file.	Procedure: Configuring Cassandra Server (copying/editing the storage-conf.xml file), on page 138
Start the server.	Procedure: Starting Cassandra Server, on page 142
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"

Procedure: Downloading and extracting the install image

Purpose: To retrieve the Cassandra Server install image from the Apache website, 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. At the time of this writing, the most current stable version is `apache-cassandra-0.6.5-bin.tar.gz`.
2. Copy the install image file to a machine that will serve as the Cassandra node, for example: `C:\CassandraServerNodeOne` .
3. Extract the install image into the chosen directory.

End of procedure

Next Steps

- Continue with Procedure: Configuring Cassandra Server (copying/editing the storage-conf.xml file)

Configuring Cassandra Server

This section describes the following procedures:

- "Configuring Cassandra Server (copying/editing the storage-conf.xml file)"
- "Configuring a multi-node Cassandra cluster"

Configuring Cassandra Server consists of *copying* the `storage-conf.xml` file and *editing* its contents. This procedure describes these steps.

Procedure: Configuring Cassandra Server (copying/editing the storage-conf.xml file)

Purpose: To configure Cassandra Server by copying and modifying the contents of the `storage-conf.xml` file.

Start of procedure

1. Copy the `storage-conf.xml` file from the ORS installation directory to the Cassandra installation directory, inside a `conf` sub-directory. For example, to: `C:\CassandraServerNodeOne\conf` .
2. Under Windows, set the environment variable `CASSANDRA_HOME` to the installation directory (for example: `C:\CassandraServerNodeOne`).

   As an alternative, the `Cassandra.bat` file in the `bin` directory may be modified to set `JAVA_HOME` and `CASSANDRA_HOME` as follows:

   `JAVA_HOME=C:\Program Files\Java\jre6` (the installation root of `jre16`)

   `CASSANDRA_HOME=C:\CassandraServerNodeOne`

   Under 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/`
3. Open the `storage-conf.xml` file that you just copied into the `C:\CassandraServerNodeOne\conf` sub-directory (for example).
4. Inside the `<ClusterName>` tag, assign a 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:

   `<ClusterName>Cluster1</ClusterName>`
5. Set the paths to the storage locations on the server platform for this node.

   > Note: The example provided in this release is for UNIX-style paths. Adjust pathnames accordingly if using the Windows platform.

   For example:

   ```
   <CommitLogDirectory>C:/CassandraServerNodeOne/data/commitlog
   </CommitLogDirectory>


   <DataFileDirectories>
       <DataFileDirectory>C:/CassandraServerNodeOne/data/data
       </DataFileDirectory>
   </DataFileDirectories>
   ```

Note: The `commitlog` and `data` directories should be placed on *different* disk drives for optimal performance. Also make sure the directories exist, and create them if they do not exist.

6. The <`ListenAddress`> and <`ThriftAddress`> comments in the `.xml` 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 <`ThriftPort`> defines the client listen address and defaults to `9160`. If this port conflicts with an existing configured port, provide a non-conflicting entry in within the `<ThriftPort>` tag. If `9160` is not currently used, you can keep the default value.

For example:

```
<ListenAddress></ListenAddress>


<ThriftAddress></ThriftAddress>


<ThriftPort>9160</ThriftPort>
```

Note: For a *multi-node cluster*, use the IP address of the platform for the addresses.

7. Change the `<Seed>` entry to the host on which Cassandra Server is installed for a single-node cluster.

For multi-node cluster deployment, the `<Seed>` should point to the host(s) that is(are) deemed to be the most reliable. As an example, for a four node cluster consisting of `rh01`, `rh02`, `rh03`, and `rh04`, you can specify `rh01` and `rh03` as seeds. It is not advisable to set more than 50% of the nodes as seeds.

For example:

`<Seed>rh01</Seed>` (change `rh01` to the hostname of the platform)

8. A potential port conflict with URS may occur for port `7000`. This can be modified for Cassandra to remove the conflict by setting the port in the `<StoragePort>` tag 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:

```
<StoragePort>7001</StoragePort>
```

9. Finally, the `log4j.properties` must be modified:

```
# output messages into a rolling log file as well as (DEBUG
OPTIONAL) stdout
log4j.rootLogger=DEBUG,stdout,R

# rolling log file
log4j.appender.R=org.apache.log4j.RollingFileAppender
log4j.appender.R.maxFileSize=20MB
log4j.appender.R.maxBackupIndex=50
log4j.appender.R.layout=org.apache.log4j.PatternLayout
log4j.appender.R.layout.ConversionPattern=%5p [%t] %d{ISO8601} %F
(line %L) %m%n

# Edit the next line to point to your logs directory

log4j.appender.R.File=/cassandra/apache-cassandra-0.6.5/logs/system
.log

# Application logging options (OPTIONAL)
log4j.logger.com.facebook=DEBUG
log4j.logger.com.facebook.infrastructure.gms=DEBUG
log4j.logger.com.facebook.infrastructure.db=DEBUG
```

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:

```
http://wiki.apache.org/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 (copying/editing the storage-conf.xml file) for multi-node issues regarding `<Seed>`, `<ThriftAddress>`, and `<ListenAddress>`.

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 `QUORUM`, defined as `N/2+1`, which means that reads and writes will be successful for the following conditions:

   ```
   W=1, R=N
   W=N, R=1
   ```

   `W=Q, R=Q` where `Q = N / 2 + 1`

   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, `1,2,.N`, as below for a three node cluster:

   ```
   (2**127)/3 = 56713727820156410577229101238628035242
   ((2**127)/3)*2 = 113427455640312821154458202477256070484
   ((2^127)/3)*3 = 170141183460469231731687303715884105726
   ```

   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

This section describes the following procedure:

- "Starting Cassandra Server"

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` (or double click on `Cassandra.bat` in the `bin` directory)
2. Under Linux, start Cassandra Server by issuing the following:

   `./bin/cassandra` (from the Cassandra installation directory)

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
   ```

   Also, if you want to start the client by double-clicking `cassandra-cli.bat` and *do not add the sets as above*, then this line:

   ```
   if NOT DEFINED CASSANDRA_HOME set CASSANDRA_HOME=%CD%
   ```

   should be modified to the following:

   ```
   if NOT DEFINED CASSANDRA_HOME set CASSANDRA_HOME=%~dp0..
   ```

2. Enter Cassandra Help, by typing `help` at the `cassandra>` 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.
   cassandra> 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:

   ```
   cassandra> connect DWS/9160
   ```

   If a message similar to the one below displays, the server is ready to run with Orchestration.:

   ```
   Connected to: "Orchestration Cluster CLUSTER1" on DWS/9160
   ```

4. Other commands in the Cassandra Client Help list may be used to set/get/delete data. The following is example:

```
cassandra> set
Orchestration.OrsSession['D7FE00BD-045104C2-0001']['SessionContent'] =
'This will be the session content'
Value inserted.

cassandra> get Orchestration.OrsSession['D7FE00BD-045104C2-0001']
=> (column=SessionContent, value=This will be the session content,
timestamp=1275945227869000)
Returned 1 results.
```

End of procedure

Sample Cassandra Output

The following is an example of output from Cassandra for reference and informational purposes.

```
Starting Cassandra Server
Listening for transport dt_socket at address: 8888
DEBUG [main] 2010-08-16 19:34:16,310 DatabaseDescriptor.java (line 166)
Loading settings from
C:\CassandraNodeOne-0.6.5\apache-cassandra-0.6.5\conf\storage-conf.xml
DEBUG [main] 2010-08-16 19:34:16,388 DatabaseDescriptor.java (line 216)
Syncing log with a period of 10000
DEBUG [main] 2010-08-16 19:34:16,419 DatabaseDescriptor.java (line 414)
setting autoBootstrap to true
DEBUG [main] 2010-08-16 19:34:16,435 DatabaseDescriptor.java (line 493)
setting hintedHandoffEnabled to true
DEBUG [main] 2010-08-16 19:34:16,482 CassandraDaemon.java (line 90)
opening keyspace system
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
LocationInfo as 0
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
HintsColumnFamily as 1
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
Document as 2
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
Schedule as 3
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
Session as 4
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
SessionIDServerInfoRIndex as 5
```

```
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
ScheduleRIndex as 7
DEBUG [main] 2010-08-16 19:34:16,497 Table.java (line 117) adding
ORS8010018 as 8
DEBUG [main] 2010-08-16 19:34:16,513 Table.java (line 117) adding
SessionIDServerInfo as 9
DEBUG [main] 2010-08-16 19:34:16,591 ColumnFamilyStore.java (line 133)
Starting CFS LocationInfo
DEBUG [main] 2010-08-16 19:34:16,591 SSTableTracker.java (line 117) key
cache capacity for LocationInfo is 1
DEBUG [main] 2010-08-16 19:34:16,607 ColumnFamilyStore.java (line 133)
Starting CFS HintsColumnFamily
DEBUG [main] 2010-08-16 19:34:16,607 SSTableTracker.java (line 117) key
cache capacity for HintsColumnFamily is 1
DEBUG [main] 2010-08-16 19:34:16,607 CassandraDaemon.java (line 90)
opening keyspace Orchestration
DEBUG [main] 2010-08-16 19:34:16,607 ColumnFamilyStore.java (line 133)
Starting CFS Document
DEBUG [main] 2010-08-16 19:34:16,607 SSTableTracker.java (line 117) key
cache capacity for Document is 200000
DEBUG [main] 2010-08-16 19:34:16,607 ColumnFamilyStore.java (line 133)
Starting CFS Schedule
DEBUG [main] 2010-08-16 19:34:16,607 SSTableTracker.java (line 117) key
cache capacity for Schedule is 200000
DEBUG [main] 2010-08-16 19:34:16,607 ColumnFamilyStore.java (line 133)
Starting CFS Session
DEBUG [main] 2010-08-16 19:34:16,607 SSTableTracker.java (line 117) key
cache capacity for Session is 200000
DEBUG [main] 2010-08-16 19:34:16,607 ColumnFamilyStore.java (line 133)
Starting CFS SessionIDServerInfoRIndex
DEBUG [main] 2010-08-16 19:34:16,622 SSTableTracker.java (line 117) key
cache capacity for SessionIDServerInfoRIndex is 200000
DEBUG [main] 2010-08-16 19:34:16,622 ColumnFamilyStore.java (line 133)
Starting CFS ScheduleRIndex
DEBUG [main] 2010-08-16 19:34:16,622 SSTableTracker.java (line 117) key
cache capacity for ScheduleRIndex is 200000
DEBUG [main] 2010-08-16 19:34:16,622 ColumnFamilyStore.java (line 133)
Starting CFS ORS8010018
DEBUG [main] 2010-08-16 19:34:16,622 SSTableTracker.java (line 117) key
cache capacity for ORS8010018 is 200000
DEBUG [main] 2010-08-16 19:34:16,622 ColumnFamilyStore.java (line 133)
Starting CFS SessionIDServerInfo
DEBUG [main] 2010-08-16 19:34:16,622 SSTableTracker.java (line 117) key
cache capacity for SessionIDServerInfo is 200000
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for
LocationInfo
```

```
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for
HintsColumnFamily
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for Document
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for Schedule
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for Session
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,654
CompactionManager.java (line 553) Estimating compactions for
SessionIDServerInfoRIndex
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 553) Estimating compactions for
ScheduleRIndex
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 553) Estimating compactions for ORS8010018
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 553) Estimating compactions for
SessionIDServerInfo
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
LocationInfo would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
HintsColumnFamily would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
Document would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
Schedule would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
Session would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
SessionIDServerInfoRIndex would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
ScheduleRIndex would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
ORS8010018 would be useful
DEBUG [COMPACTION-POOL:1] 2010-08-16 19:34:16,669
CompactionManager.java (line 91) Checking to see if compaction of
SessionIDServerInfo would be useful
 INFO [main] 2010-08-16 19:34:16,685 SystemTable.java (line 139) Saved
Token not found. Using 13805985883188715835535869616050412168
```

```
 INFO [main] 2010-08-16 19:34:16,685 SystemTable.java (line 145) Saved
ClusterName not found. Using Orchestration Cluster DWS
 INFO [main] 2010-08-16 19:34:16,685 CommitLogSegment.java (line 50)
Creating new commitlog segment
C:/CassandraNodeOne-0.6.5/apache-cassandra-0.6.5/data/commitlog\CommitL
og-1282012456685.log
 INFO [main] 2010-08-16 19:34:16,716 StorageService.java (line 326)
Starting up server gossip
 INFO [main] 2010-08-16 19:34:16,732 StorageService.java (line 341)
This node will not auto bootstrap because it is configured to be a seed
node.
 INFO [main] 2010-08-16 19:34:16,778 CassandraDaemon.java (line 108)
Binding thrift service to DWS/172.21.82.194:9160
 INFO [main] 2010-08-16 19:34:16,778 CassandraDaemon.java (line 148)
Cassandra starting up...
DEBUG [GC inspection] 2010-08-16 19:34:17,716 GCInspector.java (line
112) GC for ParNew: 6 ms, 15464776 reclaimed leaving 9352984 used; max
is 1173946368
DEBUG [GC inspection] 2010-08-16 19:34:17,716 GCInspector.java (line
112) GC for ConcurrentMarkSweep: 0 ms, 0 reclaimed leaving 0 used; max
is 0
DEBUG [Timer-0] 2010-08-16 19:34:18,747 LoadDisseminator.java (line 36)
Disseminating load info ...
```

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 8.0 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 8.0 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 8.0 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 8.0 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 8.0 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 8.0 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 7 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:

80fr_ref_06-2008_v8.0.001.00

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

Screen Captures Used in This Document

Screen captures from the product graphical user interface (GUI), as used in this document, 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 9 describes and illustrates the type conventions that are used in this document.

Table 9: 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 153).	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 9: 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