



Avaya Aura® Call Center Elite Multichannel Call Routing Server User Guide

Release 6.4
Issue 2
April 2015

Notice

While reasonable efforts have been made to ensure that the information in this document is complete and accurate at the time of printing, Avaya assumes no liability for any errors. Avaya reserves the right to make changes and corrections to the information in this document without the obligation to notify any person or organization of such changes.

Documentation disclaimer

"Documentation" means information published by Avaya in varying mediums which may include product information, operating instructions and performance specifications that Avaya may generally make available to users of its products and Hosted Services. Documentation does not include marketing materials. Avaya shall not be responsible for any modifications, additions, or deletions to the original published version of documentation unless such modifications, additions, or deletions were performed by Avaya. End User agrees to indemnify and hold harmless Avaya, Avaya's agents, servants and employees against all claims, lawsuits, demands and judgments arising out of, or in connection with, subsequent modifications, additions or deletions to this documentation, to the extent made by End User.

Link disclaimer

Avaya is not responsible for the contents or reliability of any linked websites referenced within this site or documentation provided by Avaya. Avaya is not responsible for the accuracy of any information, statement or content provided on these sites and does not necessarily endorse the products, services, or information described or offered within them. Avaya does not guarantee that these links will work all the time and has no control over the availability of the linked pages.

Warranty

Avaya provides a limited warranty on Avaya hardware and software. Refer to your sales agreement to establish the terms of the limited warranty. In addition, Avaya's standard warranty language, as well as information regarding support for this product while under warranty is available to Avaya customers and other parties through the Avaya Support website: <http://support.avaya.com> or such successor site as designated by Avaya. Please note that if You acquired the product(s) from an authorized Avaya Channel Partner outside of the United States and Canada, the warranty is provided to You by said Avaya Channel Partner and not by Avaya.

Licenses

THE SOFTWARE LICENSE TERMS AVAILABLE ON THE AVAYA WEBSITE, [HTTP://SUPPORT.AVAYA.COM/LICENSEINFO](http://SUPPORT.AVAYA.COM/LICENSEINFO) OR SUCH SUCCESSOR SITE AS DESIGNATED BY AVAYA, ARE APPLICABLE TO ANYONE WHO DOWNLOADS, USES AND/OR INSTALLS AVAYA SOFTWARE, PURCHASED FROM AVAYA INC., ANY AVAYA AFFILIATE, OR AN AVAYA CHANNEL PARTNER (AS APPLICABLE) UNDER A COMMERCIAL AGREEMENT WITH AVAYA OR AN AVAYA CHANNEL PARTNER. UNLESS OTHERWISE AGREED TO BY AVAYA IN WRITING, AVAYA DOES NOT EXTEND THIS LICENSE IF THE SOFTWARE WAS OBTAINED FROM ANYONE OTHER THAN AVAYA, AN AVAYA AFFILIATE OR AN AVAYA CHANNEL PARTNER; AVAYA RESERVES THE RIGHT TO TAKE LEGAL ACTION AGAINST YOU AND ANYONE ELSE USING OR SELLING THE SOFTWARE WITHOUT A LICENSE. BY INSTALLING, DOWNLOADING OR USING THE SOFTWARE, OR AUTHORIZING OTHERS TO DO SO, YOU, ON BEHALF OF YOURSELF AND THE ENTITY FOR WHOM YOU ARE INSTALLING, DOWNLOADING OR USING THE SOFTWARE (HEREINAFTER REFERRED TO INTERCHANGEABLY AS "YOU" AND "END USER"), AGREE TO THESE TERMS AND CONDITIONS AND CREATE A BINDING CONTRACT BETWEEN YOU AND AVAYA INC. OR THE APPLICABLE AVAYA AFFILIATE ("AVAYA").

Avaya grants You a license within the scope of the license types described below, with the exception of Heritage Nortel Software, for which the scope of the license is detailed below. Where the order documentation does not expressly identify a license type, the applicable license will be a Designated System License. The

applicable number of licenses and units of capacity for which the license is granted will be one (1), unless a different number of licenses or units of capacity is specified in the documentation or other materials available to You. "Software" means computer programs in object code, provided by Avaya or an Avaya Channel Partner, whether as stand-alone products, pre-installed on hardware products, and any upgrades, updates, patches, bug fixes, or modified versions thereto. "Designated Processor" means a single stand-alone computing device. "Server" means a Designated Processor that hosts a software application to be accessed by multiple users. "Instance" means a single copy of the Software executing at a particular time: (i) on one physical machine; or (ii) on one deployed software virtual machine ("VM") or similar deployment.

License type(s)

Designated System(s) License (DS). End User may install and use each copy or an Instance of the Software only on a number of Designated Processors up to the number indicated in the order. Avaya may require the Designated Processor(s) to be identified in the order by type, serial number, feature key, Instance, location or other specific designation, or to be provided by End User to Avaya through electronic means established by Avaya specifically for this purpose.

Concurrent User License (CU). End User may install and use the Software on multiple Designated Processors or one or more Servers, so long as only the licensed number of Units are accessing and using the Software at any given time. A "Unit" means the unit on which Avaya, at its sole discretion, bases the pricing of its licenses and can be, without limitation, an agent, port or user, an e-mail or voice mail account in the name of a person or corporate function (e.g., webmaster or helpdesk), or a directory entry in the administrative database utilized by the Software that permits one user to interface with the Software. Units may be linked to a specific, identified Server or an Instance of the Software.

Copyright

Except where expressly stated otherwise, no use should be made of materials on this site, the Documentation, Software, Hosted Service, or hardware provided by Avaya. All content on this site, the documentation, Hosted Service, and the product provided by Avaya including the selection, arrangement and design of the content is owned either by Avaya or its licensors and is protected by copyright and other intellectual property laws including the sui generis rights relating to the protection of databases. You may not modify, copy, reproduce, republish, upload, post, transmit or distribute in any way any content, in whole or in part, including any code and software unless expressly authorized by Avaya. Unauthorized reproduction, transmission, dissemination, storage, and or use without the express written consent of Avaya can be a criminal, as well as a civil offense under the applicable law.

Third Party Components

"Third Party Components" mean certain software programs or portions thereof included in the Software or Hosted Service may contain software (including open source software) distributed under third party agreements ("Third Party Components"), which contain terms regarding the rights to use certain portions of the Software ("Third Party Terms"). As required, information regarding distributed Linux OS source code (for those products that have distributed Linux OS source code) and identifying the copyright holders of the Third Party Components and the Third Party Terms that apply is available in the products, Documentation or on Avaya's website at: <http://support.avaya.com/Copyright> or such successor site as designated by Avaya. You agree to the Third Party Terms for any such Third Party Components.

Note to Service Provider

The product or Hosted Service may use Third Party Components subject to Third Party Terms that do not allow hosting and require a Service Provider to be independently licensed for such purpose. It is your responsibility to obtain such licensing.

Preventing Toll Fraud

"Toll Fraud" is the unauthorized use of your telecommunications system by an unauthorized party (for example, a person who is not a corporate employee, agent, subcontractor, or is not working on your company's behalf). Be aware that there can be a risk of Toll Fraud associated with your system and that, if Toll Fraud occurs, it can

result in substantial additional charges for your telecommunications services.

Avaya Toll Fraud intervention

If You suspect that You are being victimized by Toll Fraud and You need technical assistance or support, call Technical Service Center Toll Fraud Intervention Hotline at +1-800-643-2353 for the United States and Canada. For additional support telephone numbers, see the Avaya Support website: <http://support.avaya.com> or such successor site as designated by Avaya. Suspected security vulnerabilities with Avaya products should be reported to Avaya by sending mail to: securityalerts@avaya.com.

Trademarks

The trademarks, logos and service marks ("Marks") displayed in this site, the Documentation, Hosted Service(s), and product(s) provided by Avaya are the registered or unregistered Marks of Avaya, its affiliates, or other third parties. Users are not permitted to use such Marks without prior written consent from Avaya or such third party which may own the Mark. Nothing contained in this site, the Documentation, Hosted Service(s) and product(s) should be construed as granting, by implication, estoppel, or otherwise, any license or right in and to the Marks without the express written permission of Avaya or the applicable third party.

Avaya is a registered trademark of Avaya Inc.

All non-Avaya trademarks are the property of their respective owners. Linux[®] is the registered trademark of Linus Torvalds in the U.S. and other countries.

Downloading Documentation

For the most current versions of Documentation, see the Avaya Support website: <http://support.avaya.com>, or such successor site as designated by Avaya.

Contact Avaya Support

See the Avaya Support website: <http://support.avaya.com> for product or Hosted Service notices and articles, or to report a problem with your Avaya product or Hosted Service. For a list of support telephone numbers and contact addresses, go to the Avaya Support website: <http://support.avaya.com> (or such successor site as designated by Avaya), scroll to the bottom of the page, and select Contact Avaya Support.

Contents

Chapter 1: Introduction	6
Purpose	6
Intended audience	6
Related resources	6
Documentation	6
Training	8
Viewing Avaya Mentor videos	8
Support	9
Chapter 2: Overview	10
VDN registration	10
Calls received to a routing VDN	11
Calls received to a monitored VDN	11
Call flow	11
Licensing	13
Error logging	13
Diagnostic testing error logging	14
Chapter 3: Installation	15
Chapter 4: Administration	16
Configuring Call Routing Server	16
[Error Logging]	16
[Telephony Services]	17
[Call Routing Server]	18
[AS Client Extensions]	18
[ASG Simple SQL]	18
[ASG Script Host]	20
[ASG SOAP]	20
[ASG Rules Engine]	21
Chapter 5: Plug-ins	23
Call routing through SQL Plug-in	23
Call routing through Script Plug-in	25
Call routing through SOAP Plug-in	26
Call routing through Rules Plug-in	28
Plug-in installation	29
Plug-in configuration	29
Sections	29
Result parameters	33
Transfer parameters	33
Fixed Value Parameters	34
Error log file	35

Events sent from Call Routing Server to Plug-in.....	35
CallDelivered.....	36
CallEstablished.....	37
ConnectionCleared.....	38
CallCleared.....	38
RouteRequest.....	38
RouteEnd.....	39
RouteUsed.....	39
TServerLinkStatus.....	40
Events sent from Plug-in to Call Routing Server.....	40
RouteSelect.....	40
RouteEnd.....	40
ClearCall.....	41
ClearConnection.....	41
Script COM Interface.....	41
Script COM Object-Parameter Set.....	41
Script COM - Properties.....	42
SendReturnEvent.....	42
CurrentEventName.....	42
Script COM - Methods.....	43
ASSendReturnEvent.....	43
ASLogErrorInformation.....	43

Chapter 1: Introduction

Purpose

This guide provides information about Call Routing Server that enables intelligent call routing for inbound calls in Call Center Elite Multichannel.

Intended audience

This guide is intended for sales engineers, solution architects, implementation engineers, and system administrators who use Call Center Elite Multichannel. You can use this guide as a source of information for Call Routing Server that is provided in Avaya Aura® Call Center Elite Multichannel.

Related resources

Documentation

The following table lists the related documents for the Avaya Aura® Call Center Elite Multichannel product. You can download the documents from the Avaya Support website at <http://support.avaya.com/>.

Title	Description	Audience
<i>Installing Avaya Aura® Call Center Elite Multichannel</i>	Provides product overview, supported products, installation, configuration, and licensing requirements for Avaya Aura® Call Center Elite Multichannel.	• Implementation engineers
<i>Avaya Aura® Call Center Elite Multichannel Configuration Client Developer Guide</i>	Provides information about the Configuration Client Developer application, which	• Programmers

Table continues...

Title	Description	Audience
	is a control that facilitates an application to transparently access the configuration information from any location.	
<i>Administering Avaya Aura® Call Center Elite Multichannel</i>	Provides information about how to manage databases, configure Call Center Elite Multichannel services, and administer Avaya Aura® Communication Manager.	• Sales engineers • Solution architects • Implementation engineers • System administrators
<i>Avaya Aura® Call Center Elite Multichannel Overview Guide</i>	Provides an overview of the Call Center Elite Multichannel features.	• Sales engineers • Implementation engineers • System administrators
<i>Avaya Aura® Call Center Elite Multichannel Desktop User Guide</i>	Provides information about Call Center Elite Multichannel Desktop and describes how to use Call Center Elite Multichannel Desktop to receive, view, and respond to voice and multimedia work items.	• Sales engineers • Solution architects • Implementation engineers • System administrators • End users
<i>Avaya Aura® Call Center Elite Multichannel Application Management Service User Guide</i>	Provides information about how to install and administer Application Management Service in Call Center Elite Multichannel.	• Sales engineers • Solution architects • Implementation engineers
<i>Avaya Aura® Call Center Elite Multichannel Upgrade and Migration Guide</i>	Provides information about how to upgrade or migrate Avaya Aura® Call Center Elite Multichannel from Release 6.2.x or 6.3.x to Release 6.4. The Upgrade Sequence section in the respective upgrading chapters provides a high-level overview of the process.	• Implementation engineers • Solution architects
<i>Avaya Aura® Call Center Elite Multichannel Reporting User Guide</i>	Provides information about the reports for Agents, Customers, Interaction, Program and Schedule, Skills, and VDNs. This guide also provides information about historical reports and real-time reports.	• Sales engineers • Solution architects • Implementation engineers

Table continues...

Title	Description	Audience
<i>Avaya Aura® Call Center Elite Multichannel TTrace User Guide</i>	Provides information about TTrace and its components, helps you to understand the TTraceConsole and TTraceConfig user interfaces, and explains the operations that you can perform using TTraceConsole and TTraceConfig.	• Sales engineers • Solution architects • Implementation engineers

Training

The following courses are available on the Avaya Learning website at <http://www.avaya-learning.com>. After logging in to the website, enter the course code or the course title in the **Search** field and press **Enter** or click > to search for the course.

Course code	Course title
5C00092W	Avaya Aura® Call Center Elite Multichannel Overview
10C00010E	Knowledge Access: Avaya Aura® Call Center Elite Multichannel Implementation
10C00094V	Avaya Aura® Call Center Elite Multichannel Implementation and Maintenance
4302	Avaya Aura® Call Center Elite Multichannel Implementation Test
0C00060E	Knowledge Collection Access: Avaya Aura® Call Center Elite Portfolio
E: Self-paced in virtual campus	
W: Web (online) course	
V: Virtual	

Viewing Avaya Mentor videos

Avaya Mentor videos provide technical content on how to install, configure, and troubleshoot Avaya products.

About this task

Videos are available on the Avaya Support website, listed under the video document type, and on the Avaya-run channel on YouTube.

Procedure

- To find videos on the Avaya Support website, go to support.avaya.com and perform one of the following actions:
 - In **Search**, type `Avaya Mentor Videos` to see a list of the available videos.

- In **Search**, type the product name. On the Search Results page, select **Video** in the **Content Type** column on the left.
- To find the Avaya Mentor videos on YouTube, go to www.youtube.com/AvayaMentor and perform one of the following actions:
 - Enter a key word or key words in the Search Channel to search for a specific product or topic.
 - Scroll down Playlists, and click the name of a topic to see the available list of videos posted on the website.

 Note:

Videos are not available for all products.

Support

Go to the Avaya Support website at <http://support.avaya.com> for the most up-to-date documentation, product notices, and knowledge articles. You can also search for release notes, downloads, and resolutions to issues. Use the online service request system to create a service request. Chat with live agents to get answers to questions, or request an agent to connect you to a support team if an issue requires additional expertise.

Chapter 2: Overview

Call Routing Server facilitates intelligent call routing for inbound calls. The routing depends on the received call data that matches customer information, contact center statistics, or agent availability.

Call Routing Server:

- Manages the TSAPI Link to Application Enablement Services (AES) Server
- Monitors VDNs
- Manages VDN licensing
- Registers for routing services
- Receives call events
- Issues routing instructions
- Loads and manages generic plug-ins, such as SQL plug-in that gives server access to the SQL server databases

Call Routing Server supports a single connection to Avaya Communication Manager.

VDN registration

On startup, Call Routing Server requires configuration information to run through Configuration Client. Call Routing Server extracts the configuration information from the `ActiveCallRoutingServer.ini` file. Call Routing Server uses the primary telephony link from the configuration information and establishes a connection with AES Server.

If the primary telephony link is unavailable, Call Routing Server uses the secondary telephony link from the configuration information and establishes a connection.

After a connection is set up between Call Routing Server and AES Server, Call Routing Server sends monitoring and routing registration requests for Vector Directory Numbers (VDNs) and loads the generic plug-ins. The VDNs and generic plug-ins are listed in the configuration.

Calls received to a routing VDN

When a routing request is received from a controlled VDN, Call Routing Server passes the routing request to correct generic plug-in. The passed event contains the information required for the return routing decision.

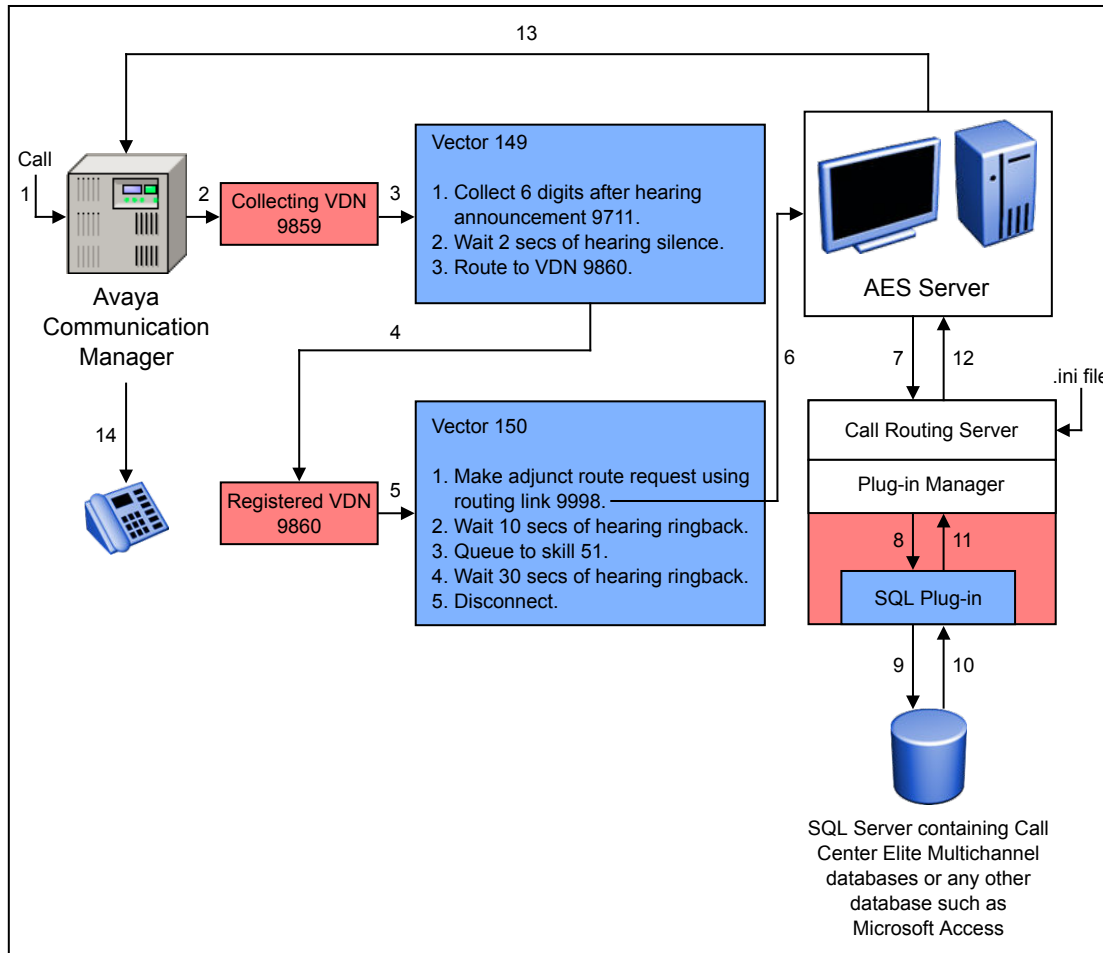
Call Routing Server contains an internal list of outstanding routing requests. A routing request is removed from the internal list of Call Routing Server when AES Server receives response for the routing request from a plug-in or the RouteEnd event.

Calls received to a monitored VDN

When a monitored VDN processes a call, a sequence of call events is generated. Call Routing Server passes the call events to appropriate generic plug-ins connected to Call Routing Server. Call Routing Server does not track these calls.

Call flow

The following diagram depicts how Call Routing Server handles a routing request.



A vector sends an adjunct routing request to AES Server using a routing extension link. AES Server routes the request to Call Routing Server that registered the associated VDN. Call Routing Server receives the request and collects digits from the request.

After collecting the digits, Call Routing Server accesses an SQL Server database using SQL Plug-in to get information related to the collected digits. The SQL Server database can be a Call Center Elite Multichannel database or any other database such as Microsoft Access database.

The SQL Server database returns an extension number where the call must be sent. The extension number can be a skill, split, agent DN, DDI, or international number.

When Call Routing Server processes the extension number, a route selection request passes to AES Server and Avaya Communication Manager. If Avaya Communication Manager does not receive the call destination within the wait-time specified in the vector, Call Routing Server processes the next routing request in the list.

*** Note:**

- Only one Call Routing Server can register an adjunct routing VDN at a time.

- The routing request fails if the number used for the SQL query or the number returned by the SQL Server database contains any blank spaces. For example, "7764500 ".

Licensing

To run Call Routing Server, you do not need to buy any additional Call Center Elite Multichannel licenses, because Call Routing Server uses the TSAPI Advanced licenses. These licenses can be provided for free if Named Licensing is configured. Call Routing Server requires a secure TSAPI Link to AES Server.

Avaya Communication Manager requires an AES license to let the adjunct link work. Therefore, you must ensure that your system has an AES ADVANCED SMALL SWITCH or AES ADVANCED LARGE SWITCH license.

Error logging

An agent logs the error information related to its operations in a series of log files. The agent creates a new log file for each day of the week. The name of the error log file specifies the application and the day of the week. For example: `MonACRS.log`.

When a log file reaches the size limit specified in the configuration, the log file is archived and the system creates a new log file.

Note:

The archive stores only one log file. If a log file reaches the specified maximum size limit, it overrides the previously archived log file.

The logging level retrieved from the application configuration data determines the type of error in error log.

The levels of error logging are:

- 0. No error logging takes place.
- 1. Logs fatal, major, and minor errors and trace information.
- 2. Logs fatal, major, and minor errors.
- 4. Logs fatal and major errors.
- 8. Logs fatal errors only.

Each log file records the selected logging level, date, time, location, and description of every error that occurs.

Error log files are automatically saved into the current working folder of the application. The current working folder is the folder where the executable file of the application is stored.

Diagnostic testing error logging

Diagnostic testing error logging is another error log level. Using this error logging level, you can create log files that do not override each other when the file size reaches the specified maximum size limit. This logging level is for diagnostic purpose only and is achieved by adding 128 to one of the logging level values specified in the levels of error logging.

For example, if you specify Error Log Level = 129, the system creates new error log files continuously. The new error log files contain fatal, major, and minor errors and trace information.

Each file has a unique name based on the date (year, month, and day) and time (hours, minutes, and seconds) when the file is created.

For example, `YYYYMMDDHHMMSSASMediaDirector.log`.

YYYY = Year	MM = Month	DD = Day
HH = Hour	MM = Minute	SS = Second

Warning:

To prevent a multitude of log files affecting your available disk space, you must use this error logging level only for short duration while carrying out diagnostic testing.

Chapter 3: Installation

For installation instructions, see *Installing Avaya Aura® Call Center Elite Multichannel*.

In addition to the standard installation steps, this document explains:

- How to use command line parameters during installation to dictate where configuration information is retrieved.
- How to change the source of configuration from the local .ini file to Configuration Server if an application is already installed.
- How to automatically install an application using a set of predefined selection options.

You can download *Installing Avaya Aura® Call Center Elite Multichannel* from the Avaya Support website: <http://support.avaya.com>.

Chapter 4: Administration

Configuring Call Routing Server

Procedure

1. Click **Start > Run**.
2. In the Run dialog box, type `services.msc`.
3. Click **OK**.
4. Stop the **AS Call Routing Server** service.
5. Go to `C:\Program Files (x86)\Avaya\Avaya Aura CC Elite Multichannel\Server\Call Routing Server`.
6. Open the `ActiveCallRoutingServer.ini` file in a text editor, such as Notepad.
7. Edit the information in the following sections:
 - [\[Error Logging\]](#) on page 16
 - [\[Telephony Services\]](#) on page 17
 - [\[Call Routing Server\]](#) on page 18
 - [\[AS Client Extensions\]](#) on page 18
 - [\[ASG Simple SQL\]](#) on page 18
 - [\[ASG Script Host\]](#) on page 20
 - [\[ASG SOAP\]](#) on page 20
 - [\[ASG Rules Engine\]](#) on page 21

 Note:

Do not change section names or parameter names.

[Error Logging]

The following table lists and describes the parameters in the Error Logging section.

Name	Description
Error Log File Path	The location where you want to store the error log files. By default, this parameter is blank. If you do not set a value for this parameter, the error log files are stored to the current working folder of the application. The current working folder is the folder where the executable file of the application is stored.
Error Log Level	A value that determines which level of error detail must be saved in the log file. For a list of error logging levels, see Error logging on page 13.
Maximum Error Log File Size KB	The maximum amount of information that an error log file must store before the system archives the log file and creates a new file. The default value for this parameter is 1000 KB. The minimum size that you can set for an error log file is 100 KB. If a log file reaches the specified maximum size limit, it overrides the previously archived log file. If you select the diagnostic testing error log level in Error Log Level, the system creates a new file with a new name every time the log file reaches the specified maximum size limit.
Error Log File Extension	The extension of error log files for the application. Extension refers to part of the file name and the file type extension, such as log. The application automatically precedes the default extension with the day of the week when it creates the error logs.

[Telephony Services]

The following table lists and describes the parameters in the Telephony Services section.

Name	Description
Primary Link Name	The primary link name that Call Routing Server uses to establish a connection with AES Server and Avaya Communication Manager.
Primary User Name	The user name for the primary link.
Primary User Password	The password for the primary link. By default, Call Center Elite Multichannel encrypts this password. For more information about configuration commands, see <i>Installing Avaya Aura® Call Center Elite Multichannel</i> .
Secondary Link Name	The secondary link name that Call Routing Server uses to establish a connection with AES Server and Avaya Communication Manager when the primary link is unavailable.
Secondary User Name	The user name for the secondary link.
Secondary User Password	The password for the secondary link. By default, Call Center Elite Multichannel encrypts this password. For more information about configuration commands, see <i>Installing Avaya Aura® Call Center Elite Multichannel</i> .

[Call Routing Server]

The following table lists and describes the parameters in the Call Routing Server section.

Name	Description
Routing VDN List	The VDNs for which Call Routing Server issues the RouteRequest events. You must separate the VDNs with commas and use a hyphen to specify a range. For example, Routing VDN List=9885,9888,9910-9916.
Monitored VDN List	The VDNs for which Call Routing Server issues the monitor request commands. The VDNs listed with this configuration setting generate the CallDelivered, CallEstablished, and CallCleared events. You must separate VDNs with commas and use a hyphen to specify a range. For example, Monitored VDN List=4401-4424,4637.

[AS Client Extensions]

This section lists the loadable generic plug-ins. Each parameter in this section has the format `Friendly name=Plug-in section name`. For example, `SQL Plug-in=ASG Simple SQL`, `Rules Plug-in=ASG Rules Engine`, `Script Plug-in=ASG Script Host`, and `SOAP Plug-in=ASG SOAP`.

`Plug-in section name` points to the section that contains configuration data for the plug-in.

 Note:

The familiar names can have spaces.

[ASG Simple SQL]

The following table lists the SQL Plug-in parameters and their sample values.

Name	Value
Enable Client	True
Client Library Name	ASGSimpleSQL.dll
Display Name	Link to SQL Database
Display Icon	someicon.ico
Event Type	Native
Configuration Section Name	Check Caller Digits
Connection String	A connection string can be configured using the Database Server Name, Database Name, and Database User Name parameters. You can use the

Table continues...

Name	Value
	%%ENCRYPT command to encrypt the connection string if it contains a password. For more information about the %%ENCRYPT command, see <i>Installing Avaya Aura® Call Center Elite Multichannel</i>.
Database Server Name	The name of the database server.
Database Name	The name of the database.
Database User Name	The user name to access the database.
Database User Password	The password to access the database.
SQL Query	SELECT * FROM tblChecking WHERE CallerDigits=cstr(%UserEnteredCode%)
SQL Stored Procedure Name	The name of the SQL stored procedure.
SQL Stored Procedure Parameter Sequence	The parameter sequence in the SQL stored procedure.
Event Name	RouteRequest
Event Filter Name	VDN
Event Filter Value	9888
Return Event Name	RouteSelect
Return Event Parameter Sequence	%RouteRegisterReqID%, %RoutingCrossRefID%, RouteSelected = \$Agent\$
Return Event Allow Multiple Events	False
Return Event Total Record Count Name	RecordCount
Return Event Current Record Count Name	CurrentRecord
No Record Event Name	RouteSelect
No Record Event Parameter Sequence	%RouteRegisterReqID%, %RoutingCrossRefID%, RouteSelected = 8574
Error Event Name	RouteSelect
Error Event Parameter Sequence	%RouteRegisterReqID%, %RoutingCrossRefID%, RouteSelected = 8553
Thread Pool Size	2
Maximum Queued Requests	10

*** Note:**

- Type the event parameters accurately.
- Use uppercase letters as mentioned in parameter names.
- Do not add spaces between words.

[ASG Script Host]

The following table lists the Script Plug-in parameters and their sample values.

Name	Value
Enable Client	True
Client Library Name	ASGScriptHost.dll
Display Name	Link to ASG Script Host
Display Icon	someicon.ico
Event Type	Native
Configuration Section Name	Check Caller CLI
Script Language	VBScript
Script File Name	CheckCaller.vbs
Script Start Function	main
Event Name	RouteRequest
Event Filter Name	VDN
Event Filter Value	9888
Return Event Name	RouteSelect
Return Event Parameter Sequence	%RouteRegisterReqID%,%RoutingCrossRefID%
Error Event Name	RouteEnd
Error Event Parameter Sequence	%RouteRegisterReqID%,%RoutingCrossRefID%,%ErrorValue%,%VDN%
Thread Pool Size	2
Maximum Queued Requests	10

*** Note:**

- Type the event parameters accurately.
- Use uppercase letters as mentioned in parameter names.
- Do not add spaces between words.

[ASG SOAP]

The following table lists the SOAP Plug-in parameters and their sample values.

Name	Value
Enable Client	True

Table continues...

Name	Value
Client Library Name	ASGSOAP.dll
Display Name	Link to SOAP Services
Display Icon	someicon.ico
Event Type	Native
Configuration Section Name	Check Customer Details
SOAP URL	http://fitzgers/SOAPSample/Service1.asmx?WSDL
SOAP Service	Service1
SOAP Port	Service1Soap
SOAP Method	ActiveCallRoute
SOAP Method Parameter Sequence	CollectedDigits = %UserEnteredCode%, %UUI%, %VDN%, %UCID%
Event Name	RouteRequest
Event Filter Name	VDN
Event Filter Value	9888
Return Event Name	RouteSelect
Return Event Parameter Sequence	%RouteRegisterReqID%, %RoutingCrossRefID%, \$*\$
Thread Pool Size	2
Maximum Queued Requests	10

*** Note:**

- Type the event parameters accurately.
- Use uppercase letters as mentioned in parameter names.
- Do not add spaces between words.

[ASG Rules Engine]

The following table lists the SQL Plug-in parameters and their sample values.

Name	Value
Enable Client	True
Client Library Name	ASGRules.dll
Display Name	Rules Engine
Display Icon	someicon.ico
Event Type	Native
Event Name	RouteRequest

Table continues...

Name	Value
Event Filter Name	VDN
Event Filter Value	9885
Rule1	When RouteRequest And UserEnteredCode > Nothing Do ReturnEvent RouteSelect, %RouteRegisterReqID%, %RoutingCrossRefID%, RouteSelected =%UserEnteredCode%, UI=%UserEnteredCode% Then Stop
Rule2	When RouteRequest Always Do ReturnEvent RouteSelect, %RouteRegisterReqID%, %RoutingCrossRefID%, RouteSelected = 8888
Thread Pool Size	2
Maximum Queued Requests	10

*** Note:**

- Type the event parameters accurately.
- Use uppercase letters as mentioned in parameter names.
- Do not add spaces between words.

Chapter 5: Plug-ins

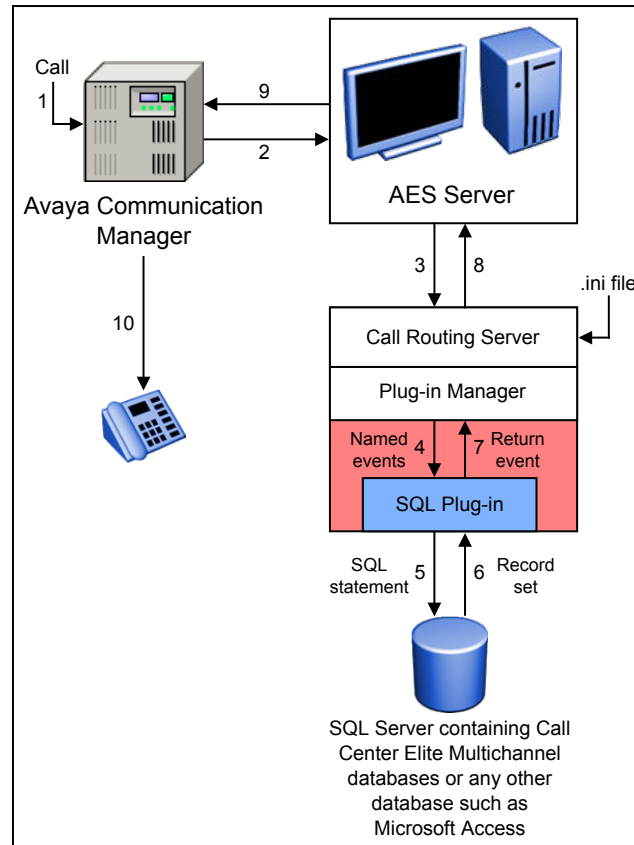
For information about the plug-ins, see the following documents:

- *Avaya Aura® Call Center Elite Multichannel Overview*
- *Avaya Aura® Call Center Elite Multichannel Desktop User Guide*

Call routing through SQL Plug-in

Call Routing Server routes the incoming calls based on the digits entered by the caller. To route the incoming calls to an appropriate agent, Call Routing Server interacts with SQL Plug-in.

The following diagram illustrates how Call Routing Server interacts with SQL Plug-in to route the incoming calls.



Call Routing Server receives an event from AES Server and starts matching the received information with the following values in each section of the `ActiveCallRoutingServer.ini` file:

- Event Name
- Event Filter Name
- Event Filter Value

Call Routing Server stops matching if it receives a `RouteRequest` event from VDN 9888*.

If the received information matches the values, Call Routing Server passes the event to SQL Plug-in. After receiving the event, SQL Plug-in executes an SQL query on the database mentioned in the Connection String parameter in the `ActiveCallRoutingServer.ini` file.

- If the SQL query matches the caller-entered digits with the digits in the `CallerDigits` column of the database table, SQL Plug-in returns a `RouteSelect` event containing the string in the `Agent` column of the database table. SQL Plug-in also returns the `RouteRegisterReqID` and `RoutingCrossRefID` parameters for identification of the route request.
- If the SQL query does not match the caller-entered digits with the digits in the `CallerDigits` column of the database table, SQL Plug-in instructs Call Routing Server to route the call to the extension 8574.

If execution of the SQL query fails because of an error, the `RouteSelect` event requests for routing the call to extension 8553.

After receiving the RouteSelect event, Call Routing Server passes the RouteSelect event to AES Server. AES Server passes the event to Avaya Communication Manager for routing the call to the agent.

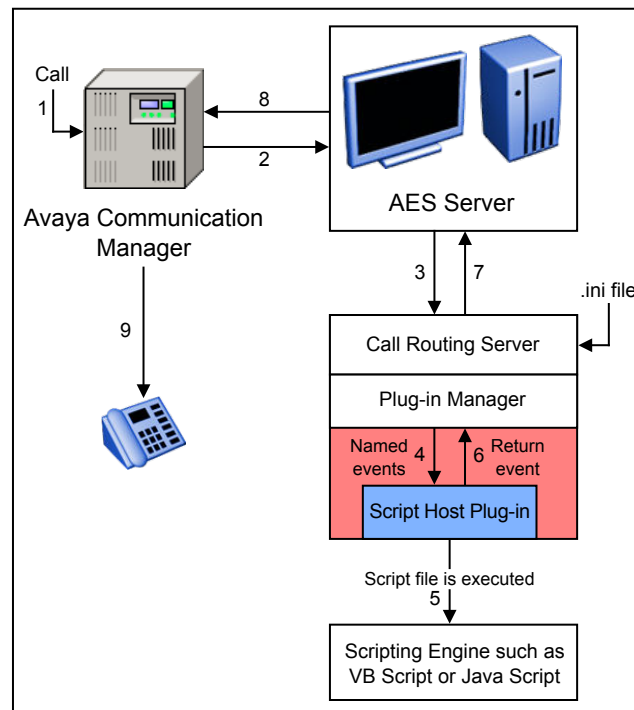
! Important:

The VDN 9888* must be registered as a routing VDN before Call Routing Server can receive RouteRequest events from it

Call routing through Script Plug-in

Call Routing Server routes the incoming calls based on the digits entered by the caller. To route the incoming calls to an appropriate agent, Call Routing Server interacts with Script Plug-in.

The following diagram illustrates how Call Routing Server interacts with Script Plug-in to route the incoming calls.



Call Routing Server receives an event from AES Server and starts matching the received information with the following values in each section of the `ActiveCallRoutingServer.ini` file:

- Event Name
- Event Filter Name
- Event Filter Value

Call Routing Server stops matching if it receives a RouteRequest event from VDN 9888*.

If the received information matches the values, Call Routing Server passes the event to Script Plug-in. After receiving the event, Script Plug-in runs the script mentioned in the Script File Name parameter in the `ActiveCallRoutingServer.ini` file. To search the script, Script Plug-in accesses the folder that contains the executable file for Call Routing Server.

- If the script matches the caller-entered digits with the digits 1234, Script Plug-in returns a RouteSelect event to Call Routing Server. The RouteSelect event instructs Call Routing Server to route the call to extension 8574.
- If the script does not match the caller-entered digits with the digits 1234, Script Plug-in instructs Call Routing Server to route the call to extension 8553.

Script Plug-in returns the RouteSelect event and the value of the extension to Call Routing Server as the RouteSelected parameter. Script Plug-in also returns the RouteRegisterReqID and RoutingCrossRefID parameters for identification of the route request.

If execution of the script fails because of an error, the RouteSelect event requests Call Routing Server to route the call to extension 8887.

After receiving the RouteSelect event, Call Routing Server passes the RouteSelect event to AES Server. AES Server passes the event to Avaya Communication Manager for routing the call to the agent.

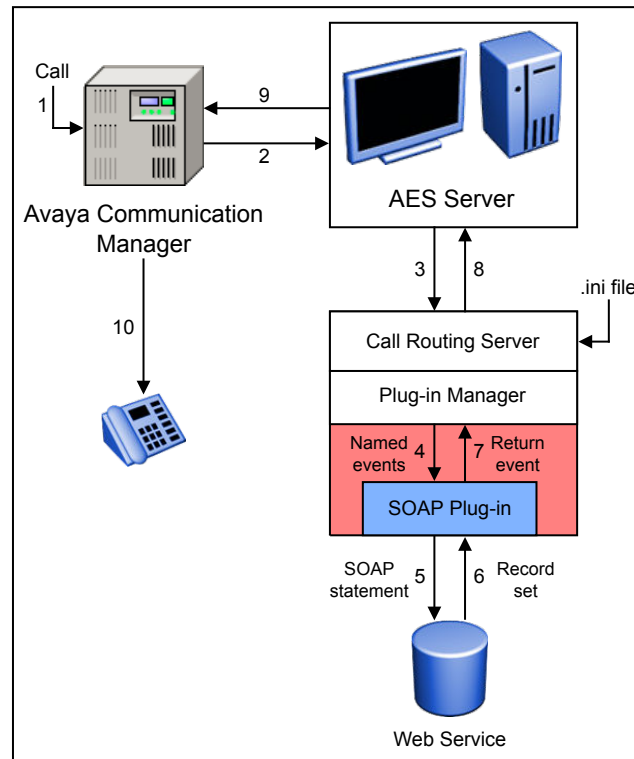
 Important:

The VDN 9888* must be registered as a routing VDN before Call Routing Server can receive RouteRequest events from it

Call routing through SOAP Plug-in

Call Routing Server routes the incoming calls based on the digits entered by the caller. To route the incoming calls to an appropriate agent, Call Routing Server interacts with SOAP Plug-in.

The following diagram illustrates how Call Routing Server interacts with SOAP Plug-in to route the incoming calls.



Call Routing Server receives an event from AES Server and starts matching the received information with the following values in each section of the `ActiveCallRoutingServer.ini` file:

- Event Name
- Event Filter Name
- Event Filter Value

Call Routing Server stops matching if it receives a `RouteRequest` event from VDN 9888*.

If the received information matches the values, Call Routing Server passes the event to SOAP Plug-in. After receiving the event, SOAP Plug-in connects to the web service mentioned in the SOAP URL parameter in the `ActiveCallRoutingServer.ini` file.

The SOAP connection passes the Collected Digits and UII variables with the `RouteRequest` event. For a selected case of collected digits, the service returns a routing destination and account information within the SOAP return.

SOAP Plug-in returns the following parameters for identification of the route request:

- `RouteRegisterReqID`
- `RoutingCrossRefID`

SOAP Plug-in sends the following parameters from the SOAP return:

- `RouteSelected`
- `UII`

After receiving the RouteSelect event, Call Routing Server passes the RouteSelect event to AES Server. AES Server passes the event to Avaya Communication Manager for routing the call to the agent.

 Important:

The VDN 9888* must be registered as a routing VDN before Call Routing Server can receive RouteRequest events from it

Call routing through Rules Plug-in

Call Routing Server routes the incoming calls based on the digits entered by the caller. To route the incoming calls to a particular agent, Call Routing Server interacts with Rules Plug-in.

Call Routing Server receives an event from AES Server and starts matching the received information with the following values in each section of the `ActiveCallRoutingServer.ini` file:

- Event Name
- Event Filter Name
- Event Filter Value

Call Routing Server stops matching if it receives a RouteRequest event from VDN 9888*.

If the received information matches the values, Call Routing Server passes the event to Rules Plug-in. After receiving the event, Rules Plug-in executes the rules configured in the `ActiveCallRoutingServer.ini` file.

Rules Plug-in executes the first rule only if the received event is the RouteRequest event carrying the digits collected with the call.

- If the first rule requirements are met, Rules Plug-in returns a RouteSelect event to Call Routing Server. The RouteSelect event instructs Call Routing Server to call the extension entered by the user and display the extension number to the agent as UUI. Rules Plug-in also returns the RouteRegisterReqID and RoutingCrossRefID parameters for identification of the route request.
- If the first rule requirements are not met, Rules Plug-in executes a second rule and returns a RouteSelect event to Call Routing Server. The RouteSelect event instructs Call Routing Server to route the call to extension 8888.

After receiving the RouteSelect event, Call Routing Server passes the RouteSelect event to AES Server. AES Server passes the event to Avaya Communication Manager for routing the call to the agent.

 Important:

The VDN 9888* must be registered as a routing VDN before Call Routing Server can receive RouteRequest events from it

Plug-in installation

Call Center Elite Multichannel plug-ins are automatically installed when you install applications that support the plug-in integration.

Plug-in configuration

For information about how to configure Rules Plug-in, see *Avaya Aura® Call Center Elite Multichannel Desktop User Guide*.

The following table list and describes the parameters that you must set to configure SOAP Plug-in, Scripts Plug-in, and SQL Plug-in. These parameters are present in the `ActiveCallRoutingServer.ini` file.

Name	Description
Enable Client	A value that determines whether the controlling application enables the plug-in on startup. To enable the plug-in on startup, you must set the value of this parameter to True.
Client Library Name	The name of the generic plug-in. For example, ASGSOAP.dll.
Display Name	The text that displays on the user interface of the client.
Display Icon	An icon that displays on the user interface of the client. This icon represents the running processes of plug-ins.
Event Type	A value that specifies whether the plug-in loaded by the server wants to do maintenance tasks or real work. If you set this value to Native, it specifies that the plug-in wants to interface for native server functionality.

Sections

All parameters related to a plug-in must appear as one section and repeat the data to create more sections. You can use a section if the received event and filter values match the configured event and filter parameters.

Applicable for Plug-in	Parameters
SOAP Plug-in	SOAP URL
	SOAP Service
	SOAP Port
	SOAP Method
SOAP Plug-in, Script Plug-in, SQL Plug-in	Configuration Section Name
	Event Name
	Event Filter Name

Table continues...

Applicable for Plug-in	Parameters
	Event Filter Value
	Return Event Name
	Return Event Parameter Sequence
Script Plug-in	Script Language
	Script File Name
	Script Start Function
SQL Plug-in	Connection String
	Database Server Name
	Database Name
	Database User Name
	Database User Password
	SQL Query
	SQL Stored Procedure Name
	SQL Stored Procedure Parameter Sequence

The following table lists and describes the parameters related to plug-ins.

Name	Description
Configuration Section Name	A unique name to identify the configuration set. If the value for this parameter is empty, the plug-in populates the value <code>UnNamed</code> plus a number indicating the position of the section in the configuration. For example, if the name for the fourth configuration set not specified, the plug-in populates it with a default name of <code>UnNamed4</code>.
SOAP URL	The URL of the SOAP server. Using this URL, the SOAP toolkit can load the Web Service Description Language (WSDL) file that describes the offered SOAP services. For example, www.AvayaActiveEnterprise.com/TestService/TestSoap?WSDL.
SOAP Service	The service in the WSDL file that contains the operation specified in the Simple Object Access Protocol (SOAP) request message.
SOAP Port	The name of the port in the WSDL file that contains the operation specified in the SOAP request message.
SOAP Method	The name of the method which is executed on the SOAP service.
SOAP Method Parameter Sequence	A comma-delimited sequence of parameters. The parameters are passed to the SOAP toolkit in the same order in which they are listed. This sequence contains the following types of information: • Static Information: Static information has the form <code>NAME=VALUE</code>. • Received Information: The information received from the controlling application. This information contains a leading and trailing <code>%</code> character.

Table continues...

Name	Description
	When a parameter is specified as absent in a received parameter sequence, an empty string is used. For example, SOAP Method Parameter Sequence=CallingNumber = %CallingNumber%, ID=27.
Event Name	The name of the event sent by the controlling application that runs the configuration set. This name represents a unique combination when it is combined with the Event Filter Name and Event Filter Value parameters. The Event Name parameter must have a unique value when the Event Filter Name parameter is empty. If the Event Name parameter does not have a unique value, the first entry in the configuration sequence is used.
Event Filter Name	The name expected in the parameter list which is sent with the event from the controlling application. This filter facilitates multiple configuration sets to have the same Event Name, but to be identified by a received parameter value. If this parameter is empty, filtering does not occur.
Event Filter Value	The value that the plug-in attempts to match with the received event and event filter. If the received event and event filter match, the configuration set is in the correct sequence. For example, if a configuration set has Event Filter Name =VDN and Event Filter Value =9830, then any event received with a matching Event Name must also contain a set VDN/9830.
Return Event Name	The name of the event that send the results obtained by the configuration set to the controlling application. If this parameter is empty, the plug-in does not send a return event.
Return Event Parameter Sequence	A list of parameters that the plug-in returns to the controlling application with the return event. If the Return Event Parameter Sequence parameter is empty, no parameters are returned. The parameters in a list that are separated by commas come from the following sources: • Script Language: The language that this script runs. The script language must match with the name registered by the script engine during installation. • Script File Name: The name of the script file that runs when the event is received. If you specify a file path, then the plug-in runs the script from that path. Otherwise, the plug-in searches for the script in the working folder, which is the same folder that contains the executable of the controlling application. • Script Start Function: With Script Start Function, you can specify an entry point in the script. If this parameter is empty, it is considered that the script is self-describing and has a known entry point.
Connection String	An ADO connection string that is used for database connectivity. If this parameter is empty, the Database Server Name, Database Name, and Database User Name parameters form the connection string. You can use the %%ENCRYPT command to encrypt this string if it contains a password.

Table continues...

Name	Description
	For more information about the %% ENCRYPT command, see <i>Installing Avaya Aura® Call Center Elite Multichannel</i> .
Database Server Name	The name of the server where the database is located. If the Connection String parameter contains a value, this parameter is ignored.
Database Name	The name of the database. If the Connection String parameter contains a value, this parameter is ignored.
Database User Name	The user name required to get access to the database. If this parameter is left blank, integrated security is assumed. If the Connection String parameter contains data, this entry is ignored.
Database User Password	The password for the database user name. By default the password is <code>CCEUser0</code> before encryption. For encryption information, see <i>Installing Avaya Aura® Call Center Elite Multichannel</i> . If the Connection String parameter contains data, this entry is ignored.
SQL Query	The SQL statement that runs when the event is received. This statement contains placeholder characters so that the system can insert the values received with the event into the string before forwarding to the database. A placeholder is the name of the variable received from the controlling application with a leading and trailing % character. For example, SQL Query=Select * from tblCustomers where CustomerNumber=%CallingNumber% or CustomerNumber=%UUI%. If an event is received with a parameter CallingNumber equal to 92996015 and a UUI equal to 111, the resultant SQL statement sent forward is Select * from tblCustomers where CustomerNumber=92996015 or CustomerNumber=111. If an event is received that does not contain the correct parameters, empty strings replace the values. The SQL Query and SQL Stored Procedure Name parameters are mutually exclusive. If both parameters are present, SQL Query takes precedence.
SQL Stored Procedure Name	The name of the stored procedure that is executed when the plug-in receives the event. The SQL Query and SQL Stored Procedure Name parameters are mutually exclusive. If both parameters are present, SQL Query takes precedence.
SQL Stored Procedure Parameter Sequence	A comma-delimited sequence of parameters. This sequence is passed to the stored procedure. This sequence consists of parameters that are extracted from the received event parameters and the fixed value parameters. • Extracted Parameters: If the sequence parameter contains a leading and trailing % character, the associated value is extracted from the parameters received with the event. • Fixed Value Parameters: Fixed value parameters have the form NAME=VALUE. For example, SQL Stored Procedure Parameter Sequence=%UUI%, %AParty%,CollectedDigits=44. This example sends three parameters to the stored procedure called UUI, AParty, and CollectedDigits. The

Table continues...

Name	Description
	parameters UUI and AParty contain the values extracted from the received event parameters. The CollectedDigits parameter has the fixed value 44.

Result parameters

Using the result parameters, you can send the results generated from the SOAP service to the controlling application. The result parameters contain a leading and trailing \$ character. For example, if the configuration set contains an entry `Return Event Parameter Sequence = $Destination$`, the returned parameter is from the **Destination** field in the returned record set and the parameter set key is `Destination`.

You can use the returned parameter set to change the key/value by using the format `KEYNAME=$SQL VALUE$`. For example, if the configuration set contains the entry `Return Event Parameter Sequence=Destination=$VDN$`, the returned parameter is from the **VDN** field in the returned record set and the parameter set key is `Destination`.

To return all fields in the resultant record set, you can use the parameter `$*$`. When you use `$*$`, the key/value pairs in the event parameter list contain the keys and values extracted from the record set.

For SQL Plug-in, you can specify the SQL result field by the numeric position in the returned record set. You can also specify the SQL result field using the format `$_NUMERIC LOCATION_`. For example, if the configuration set contains the entry `Return Event Parameter Sequence = Destination = $_1_`, the returned parameter is from the first field in the returned record set and the parameter set key is `Destination`.

Transfer parameters

Using the transfer parameters, you can send an important sequence and the tracking information received with the original event to the controlling application. Using this information, the controlling application can match the original event with the return event. The transfer parameters contain a leading and trailing \$ character.

For example, if the configuration set contains the entry `Return Event Parameter Sequence = %RouteRegisterReqID%, %RoutingCrossRefID%`, the return event parameter list contains the entries `RouteRegisterReqID` and `RoutingCrossRefID`. The values are copied from the parameter list received from the controlling application.

To transfer every parameter from the received event to the returned event, you can use the parameter `%*%`.

Fixed Value Parameters

Using the fixed value parameters, you can include fixed key/value data in the parameter list returned to the controlling application. The fixed value parameters have the form **NAME = VALUE**.

For example, if the configuration set contains the entry `Return Event Parameter Sequence = Destination=8579`, the return event parameter list contains a key entry `Destination` with a fixed value `8579`.

The parameter types can appear in any order in the parameter sequence. The order in which the parameter types appear dictates the order in which they are placed into the parameter list.

For example, if the configuration set contains the entry `Return Event Parameter Sequence = Destination = 8579, %RouteRegisterReqID%`, the return event parameter list has a key entry `Destination` with a fixed value `8579`. The return event parameter list also has a key of `RouteRegisterReqID` with a value copied from the original received event.

Parameters	Applicable for Plug-in
Return Event Allow Multiple Events	SOAP Plug-in, SQL Plug-in
Thread Pool Size	SOAP Plug-in, Script Plug-in, SQL Plug-in
Maximum Queued Requests	SOAP Plug-in, Script Plug-in, SQL Plug-in
Error Event Name	Script Plug-in, SQL Plug-in
Error Event Parameter Sequence	Script Plug-in, SQL Plug-in
Return Event Total Record Count Name	SQL Plug-in
Return Event Current Record Count Name	SQL Plug-in
No Record Event Name	SQL Plug-in
No Record Event Parameter Sequence	SQL Plug-in

The following table lists and describes the fixed value parameters.

Name	Description
Return Event Allow Multiple Events	If you set this parameter to True, the controlling application can receive multiple return events with this parameter. If you set this parameter to False, only the first record in the record set is returned.
Thread Pool Size	The number of threads that this configuration section can use to dispatch event requests for processing. Multiple threads are required because each configuration section receive multiple simultaneous events from the controlling application. All requests require to be processed parallelly. If requests are

Table continues...

Name	Description
	received when no threads are immediately available, the requests are queued for the next available thread. The thread pool size can range between 1 and 10. The default value is 2.
Maximum Queued Requests	The maximum number of requests that can be queued to the thread pool.
Error Event Name	The name of the event passed to the controlling application when any error occurs that prevents the execution of the specified stored procedure or SQL statement. This includes all ADO errors, including failure to open the connection to the specified database. If this parameter is empty, the event specified in the Return Event Name parameter is used. If the Return Event Name is used, the parameter sequence from the Error Event Parameter Sequence is used. If the Error Event Parameter Sequence is unavailable, the parameter sequence from the Return Event Parameter Sequence is used.
Error Event Parameter Sequence	The sequence of parameters which is passed with the error event. This sequence has the same format as Return Event Parameter Sequence.
Return Event Total Record Count Name	With this parameter, you can alter the name of the event parameter that contains the number of records received with the SQL query. If this parameter is blank, no information about the number of records received returns. For example, Return Event Total Record Count Name= NumberOfRecords.
Return Event Current Record Count Name	The system alters the name of the event parameter that contains the number of the current record in a multi record. If this parameter is blank, no information about the current record number returns. For example, Return Event Current Record Count Name= CurrentRecord.

Error log file

The error information related to operations of the plug-ins is stored in the error log file of the controlling application. For example, you can find the error log files for Call Routing Server at the same location where the executable file for Call Routing Server is stored.

Events sent from Call Routing Server to Plug-in

This section describes the events that are sent from Call Routing Server to the plug-in. For information about the parameter described in the events, see *Avaya Application Enablement Services TSAPI for Avaya Communication Manager Programmer's Reference Guide*.

CallDelivered

This event starts when the call is delivered to a monitored VDN. This event starts when Call Routing Server receives the cstaCallDelivered event from AES Server.

The following parameters pass with the CallDelivered event:

- VDN
- CallID
- CallingDevice
- CalledDevice
- AlertingDevice
- LastRedirectionDevice
- Cause
- DeliveredType
- Split
- UserEnteredCode
- UUI
- TrunkGroup
- TrunkGroupMember
- UCID
- LAIType
- LAIPriority
- LAIHours
- LAIMinutes
- LAISecods
- LAISourceVDN
- OCICallingDevice
- OCICalledDevice
- OCITrunkMember
- OCITrunkGroup
- OCUserEnteredCode
- OCUIUI
- OCUCID
- OCILAIType

- OCILAIPriority
- OCILAIHours
- OCILAIMinutes
- OCILAISeconds
- OCILAISourceVDN

CallEstablished

This event starts when the call is answered and triggered by receiving the cstaCallEstablished event from AES Server.

The following parameters pass with the CallEstablished event:

- VDN
- CallID
- AnsweringDevice
- CallingDevice
- CalledDevice
- Cause
- LastRedirectionDevice
- Split
- UserEnteredCode
- UUI
- TrunkGroup
- TrunkGroupMember
- UCID
- LAIType
- LAIPriority
- LAIHours
- LAIMinutes
- LAISeconds
- LAISourceVDN
- OCICallingDevice
- OCICalledDevice
- OCITrunkMember
- OCITrunkGroup

- OCUserEnteredCode
- OCUIUI
- OCUCID
- OCILAIType
- OCILAIPriority
- OCILAIHours
- OCILAIMinutes
- OCILAISecods
- OCILAISourceVDN

ConnectionCleared

This event is mapped directly from the cstaConnectionCleared event received from AES Server.

The following parameters pass with the ConnectionCleared event:

- VDN
- CallID
- ReleasingDevice
- Cause
- UUI

CallCleared

This event is mapped directly from the cstaCallCleared event received from AES Server.

The following parameters pass with the CallCleared event:

- VDN
- CallID
- Cause
- Reason

RouteRequest

This event is mapped directly from the cstaRouteRequest event received from AES Server.

The following parameters pass with the RouteRequest event:

- RouteRegisterReqID

- RouteCrossRefID
- CurrentRoute
- CallID
- CallingDevice
- TrunkGroup
- TrunkGroupMember
- UserEnteredCode
- LAIType
- LAIPriority
- LAIHours
- LAIMinutes
- LAISecods
- LAISourceVDN
- UUI
- UCID
- VDN

RouteEnd

This event is mapped directly from the cstaRouteEnded event received from AES Server.

The following parameters pass with the RouteEnd event:

- RouteRegisterReqID
- RoutingCrossRefID
- ErrorValue
- VDN

RouteUsed

This event is mapped directly from the RouteUsedEvent event received from AES Server.

The following parameters pass with the RouteUsedEvent:

- EventName
- Comma-Separated Event Arguments

TServerLinkStatus

This event is broadcasted to all extensions when the status of the AES Server link changes.

The following parameters pass with the TServerLinkStatus event:

- LinkStatus
- IsPrimaryLink

Events sent from Plug-in to Call Routing Server

This section describes the events that are sent from the plug-in to Call Routing Server. For information about the parameter described in the events, see *Avaya Application Enablement Services TSAPI for Avaya Communication Manager Programmer's Reference Guide*.

The plug-in instructs Call Routing Server to perform tasks using a fixed command. The command contains mandatory (M) and optional (O) parameters. If a mandatory parameter is missing, the command request is ignored.

RouteSelect

This event instructs Call Routing Server to direct a call to a specified destination.

This event expects the following parameters:

- RouteRegisterReqID (M)
- RoutingCrossRefID (M)
- RouteSelected (M)
- RouteUsedReq (O)
- CallingDevice (O)
- DirectAgentCallSplit (O)
- UserProvidedCode (O)
- UII (O)

RouteEnd

This event instructs Call Routing Server to end the routing association on a specific call.

This event expects the following parameters:

- RouteRegisterReqID (M)

- RoutingCrossRefID (M)
- ErrorValue (M)

ClearCall

This event instructs Call Routing Server to clear the call.

This event expects the following parameter:

- CallID (M)

ClearConnection

This event instructs Call Routing Server to clear a calling party from the call.

The event expects the following parameters:

- CallID (M)
- Device (M)
- UUI (O)

Script COM Interface

The scripting interface IActiveScript allows custom COM objects to be exposed to the script. This interface provides the script access to the host parameters and methods. The Script Plug-in exposes several COM interfaces to allow the script to operate effectively. Most script interfaces expect dual COM interfaces and various passed parameter. The Script Plug-in exposes the interfaces that conform to this requirement.

Script COM Object-Parameter Set

Parameters received from the controlling application are made available to the script. This is achieved by creating a COM object that can be accessed from the script. This COM object is created and populated with the parameters received with ASHandleClientEvent. It must also be added to the script site using the AddNamedItem function. The COM object is named in the script as ASParameterList.

COM Object Definition-ParameterItem

The COM object exposed to the script has the following properties and methods:

- Count: This property returns a count for the number of items in the parameter set.

- **Item:** This method takes a single parameter and returns the matching `ParameterItem` object from the list. The supplied parameter is either the index in the parameter list or the key of the parameter item.
- **Add:** This method takes two parameters and adds a new value to the parameter list. The passed parameters include the parameter name and parameter value.

COM Object Definition – Parameter Item

The event data passed with the event is a listing of `ParameterItem` objects. Each object represents a key/value pair.

These properties are implemented in the COM object as:

- **Key:** Returns the name of the parameter item.
- **Value:** Returns the value of the parameter item.

Script COM - Properties

SendReturnEvent

Syntax:

```
SendReturnEvent As Boolean
```

Description:

The script sets the `SendReturnEvent` property to suppress sending of a return event automatically by the plug-in. When this property is set to `False`, the plug-in automatically sends the return event specified in the configuration as the script ends. The default value for this property is `False` and this property is Read-only.

CurrentEventName

Syntax:

```
CurrentEventName As String
```

Description:

The script uses this property to determine the name of the event received from the controlling application that causes the script instance to run. This property is Read-only.

Script COM - Methods

ASSendReturnEvent

Syntax:

```
ASSendReturnEvent (EventName, ParameterSet)
```

Description:

The script uses this method to call a named event from the script code.

Parameters:

- EventName
- ParameterSet

ASLogErrorInformation

Syntax:

```
ASLogErrorInformation(ErrorLevel, ErrorText)
```

Description:

The script uses this method to log execution details in the error log of the controlling application.

Parameters:

- ErrorLevel
- ErrorText

Index

A

add	41
administration	16
ASLogErrorInformation	43
ASSendReturnEvent	43
audience	6

C

CallCleared	38
CallDelivered	36
CallEstablished	37
call events	11
call flow	11
call routing server	10
Call Routing Server	35
calls received	11
ClearCall	41
ClearConnection	41
configuration	33
configure	16
ConnectionCleared	38
count	41
CurrentEventName	42

D

destination	33
diagnostic testing error logging	14

E

error log file	35
error logging	13
events	35 , 37

F

fixed value parameters	34
------------------------------	--------------------

I

information	33
ini file	16
install	15
item	41

K

key	42
-----------	--------------------

L

licensing	13
-----------------	--------------------

M

monitored VDN	11
monitoring registration request	10

P

parameter	33
parameters	33
plug-in	35
purpose	6

R

related documentation	6
RouteEnd	39 , 40
RouteRequest	38
RouteSelect	40
RouteUsed	39
routing	11
routing extension link	11
routing registration request	10
routing request	11

S

SendReturnEvent	42
support	9

T

tracking	33
training	8
transfer	33
TServerLinkStatus	40

V

value	42
VDN registration	10
videos	8