A MITEL
PRODUCT
GUIDE

Unify OpenScape Contact Center Enterprise

VoiceXML V11R1

Integration Guide
08/2022

A31003-S22B1-N106-01-7620

Mitel

Notices

The information contained in this document is believed to be accurate in all respects but is not warranted by Mitel Europe Limited. The information is subject to change without notice and should not be construed in any way as a commitment by Mitel or any of its affiliates or subsidiaries. Mitel and its affiliates and subsidiaries assume no responsibility for any errors or omissions in this document. Revisions of this document or new editions of it may be issued to incorporate such changes. No part of this document can be reproduced or transmitted in any form or by any means - electronic or mechanical - for any purpose without written permission from Mitel Networks Corporation.

Trademarks

The trademarks, service marks, logos, and graphics (collectively “Trademarks”) appearing on Mitel’s Internet sites or in its publications are registered and unregistered trademarks of Mitel Networks Corporation (MNC) or its subsidiaries (collectively “Mitel), Unify Software and Solutions GmbH & Co. KG or its affiliates (collectively “Unify”) or others. Use of the Trademarks is prohibited without the express consent from Mitel and/or Unify. Please contact our legal department at iplegal@mitel.com for additional information. For a list of the worldwide Mitel and Unify registered trademarks, please refer to the website: http://www.mitel.com/trademarks.

Contents

1 About this guide

This guide describes how to integrate an Interactive Voice Response (IVR) system with OpenScape Contact Center using the OpenScape Contact Center VoiceXML interface.

1.1 Who should use this guide

This guide is intended for system integrators who want to integrate an interactive voice response (IVR) system with OpenScape Contact Center.

1.2 Formatting conventions

The following formatting conventions are used in this guide:

Bold
This font identifies OpenScape Contact Center components, window and dialog box titles, and item names.

Italic
This font identifies references to related documentation.

`Monospace Font`
This font distinguishes text that you should type, or that the computer displays in a message.

NOTE: Notes emphasize information that is useful but not essential, such as tips or alternative methods for performing a task.

IMPORTANT: Important notes draw special attention to actions that could adversely affect the operation of the application or result in a loss of data.

1.3 Documentation feedback

To report an issue with this document, call the Customer Support Center.

When you call, be sure to include the following information. This will help identify which document you are having issues with.

- **Title:** VoiceXML Integration Guide
- **Order Number:** A31003-S22B1-N106-01-7620

2 About the OpenScape Contact Center VoiceXML Integration

Voice Extensible Markup Language (VoiceXML) is an open interface for integrating IVR systems with OpenScape Contact Center. VoiceXML is a protocol standard used for creating voice-user interfaces that are guided by DTMF tones or spoken voice from the user side and prerecorded messages or synthesized speech resulting from text from the application side.

VoiceXML is designed for creating audio dialogs that feature synthesized speech, digitized audio, recognition of spoken and DTMF key input, recording of spoken input, and telephony. Its major goal is to bring the advantages of Web-based development and content delivery to interactive voice response applications.

The OpenScape Contact Center VoiceXML Integration is supported when connected to the following communication platforms:

- OpenScape Voice
- OpenScape 4000 or HiPath 4000

2.1 Installing the OpenScape Contact Center VoiceXML Integration

The OpenScape Contact Center VoiceXML Integration is distributed with the Web component package that is shipped on the OpenScape Contact Center DVD.

For detailed instructions on how to configure and test the Web component files on the corporate Web server machine to support the OpenScape Contact Center VoiceXML integration, see the *System Management Guide*.

2.2 VoiceXML integration advantages

The OpenScape Contact Center VoiceXML integration provides a VoiceXML interface, as an alternative to the existing proprietary IVR API, to facilitate integration between third-party IVR systems and OpenScape Contact Center. For example, an IVR can be designed to provide callers with self-service functionality and the ability to transfer to a user. Based on the caller's input selection, (for example, a billing question), the IVR can pass the call to OpenScape Contact Center using the VoiceXML interface. OpenScape Contact Center will then route the call to the most appropriate user in the contact center.

This integration provides the following key advantages for OpenScape Contact Center customers:

- Flexibility in leveraging a single corporate Web server or using dedicated servers to support VoiceXML and Web integration features
- Ability to integrate with OpenScape Contact Center in a heterogeneous system environment
- Extends the scope of integration to support both proprietary and standards-based IVR systems

The following diagram illustrates the VoiceXML integration topology.

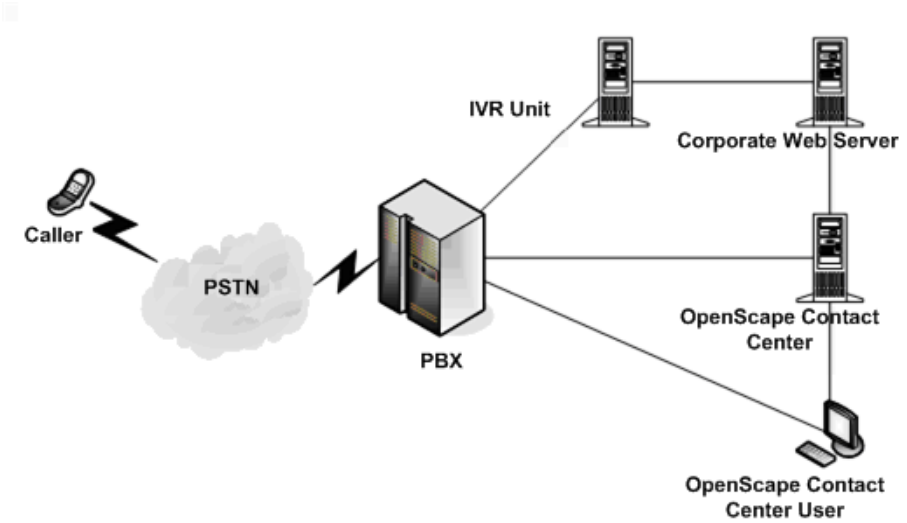

Figure 1 VoiceXML integration topology

NOTE: For a list of the supported corporate Web server platforms refer to the *System Management Guide*.

3 Configuring the OpenScape Contact Center VoiceXML Integration

This chapter describes the various VoiceXML scenarios that you can use to enhance the gathering of your customer's requirements. The scenarios are:

- IVR Hold
- Queue Hold

3.1 IVR Hold configuration

You can use the IVR Hold configuration with an IVR system to hold an enqueued call until it is routed to a user.

NOTE: Since the IVR Hold configuration ties up the IVR extension and temporarily prevents the IVR system from handling incoming calls until the enqueued call has been routed to a user, you must alter the IVR Hold configuration to include more IVR extensions.

The following diagram shows the call flow for the IVR Hold configuration. Refer to the text following the call flow diagram for a description of the numbered steps.

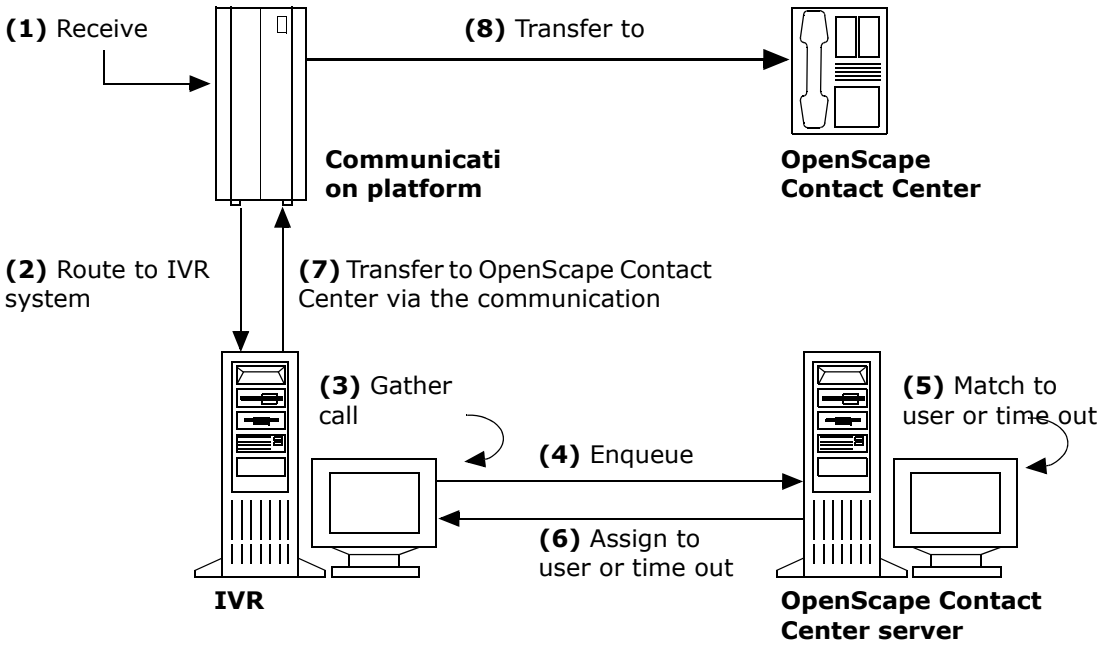

Figure 2 Call flow for the IVR Hold configuration

1. A new call arrives.
2. The call is routed to the IVR system.
3. The IVR system gathers information from the caller as to the purpose of the call, and then uses this information to identify the routing parameters for the call, including the queue.
4. The IVR system sends an enqueue call request to OpenScape Contact Center. The Routing Server then searches for the best available user to handle the call.
5. OpenScape Contact Center then does one of the following:
   - Assigns the call to the best available user
   - Reserves the call for a specific user (optional)

   If the call cannot be assigned to a user by the end of the last call step in the queue, it is routed to a time-out extension.
6. The IVR system queries for the status of the call. If the call times out, the IVR system does one of the following:
   - Transfers the call to a default number specified by the IVR script.

- Prompts the customer for additional information and uses the routing parameters to enqueue the call again. At this time, the process of matching the call to a user is repeated.

7. If the call has been assigned to a user, the IVR system transfers the call to the returned transfer number.
8. OpenScape Contact Center transfers the call to the assigned user.

NOTE: When connected to an OpenScape Voice communication platform, if the transfer fails and the call is reconnected to the IVR system, the IVR application must process the call as a new call.

3.2 Queue Hold configuration

You can use the Queue Hold configuration with an IVR system to transfer calls to an OpenScape Contact Center ACD/MLHG group in the communication platform, where calls wait until eligible users become available to handle the calls.

When a user becomes available, OpenScape Contact Center diverts the call from the OpenScape Contact Center ACD/MLHG group to the user. The Queue Hold configuration frees the IVR extensions for incoming calls by transferring calls to the communication platform while the Routing Server searches for available users.

This configuration reduces IVR extension requirements so that your IVR system can handle a greater number of incoming calls.

The following diagram shows the call flow for the Queue Hold configuration. Refer to the text following the call flow diagram for a description of the numbered steps.

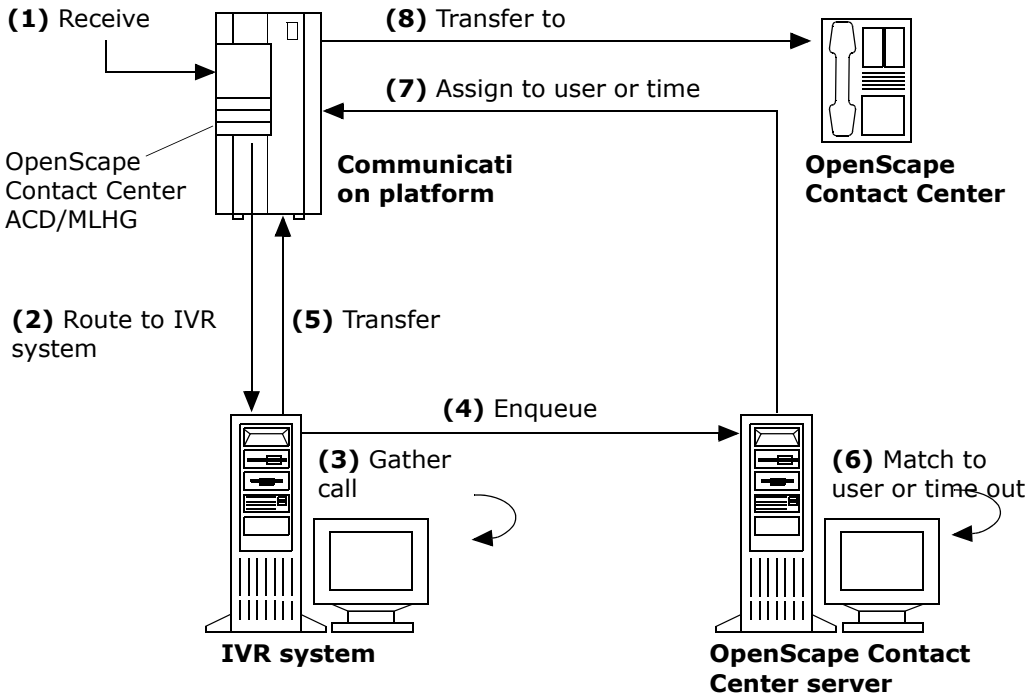

Figure 3 Call flow of the Queue Hold configuration

1. A new call arrives.
2. The call is immediately routed to an IVR system.
3. The IVR system gathers information from the caller as to the purpose of the call, and then uses this information to identify the routing parameters for the call, including the queue.
4. The IVR system sends an enqueue call request to OpenScape Contact Center. The Routing Server then searches for the best available user to handle the call.
5. At this time, the IVR system transfers the call to a transit number, and then to an OpenScape Contact Center ACD/MLHG group.
6. OpenScape Contact Center then does one of the following:
   - Assigns the call to the best available user
   - Reserves the call for a specific user (optional)

   If the call cannot be assigned to a user by the end of the last call step in the queue, it is routed to a time-out extension.

7. The call is assigned to a specific user or time-out target.
8. The call is transferred to the assigned user.

NOTE: When connected to an OpenScape Voice communication platform, if the transfer fails and the call is reconnected to the IVR system, the IVR application must process the call as a new call.

3.3 Writing an IVR script

A customized IVR script gathers information from your customers, provides call management functions, and calls the VoiceXML subdialogs provided with OpenScape Contact Center.

The VoiceXML subdialogs are provided via VoiceXML files and the **hppcwis** servlet (Tomcat) or the **hppcwis.dll** (IIS), which are installed on the corporate Web server.

3.4 VoiceXML subdialogs for IVR systems

OpenScape Contact Center uses OpenScape Contact Center VoiceXML subdialogs to facilitate integration with IVR resources.

The following concepts must be taken into account:

- OpenScape Contact Center is notified, with ANI and DNIS numbers, when the call arrives at the IVR system.
- Coordination between OpenScape Contact Center and the IVR system is necessary to transfer a call from the IVR system to OpenScape Contact Center.
- Calls leaving the IVR system that are either disconnected or transferred out require the Terminate subdialog, which implies call disconnect, to notify OpenScape Contact Center.

The IVR system must provide the ANI and DNIS of the call to OpenScape Contact Center. This information can be obtained by the IVR system through the specific trunking protocol of the communication platform, for example, ISDN (Integrated Services Digital Network). The Initialize subdialog sets the ANI and DNIS locally before this information is sent to OpenScape Contact Center. This subdialog should be called as soon as possible after the IVR system begins processing the arriving call.

An IVR transit number is a pilot number used for integration with an IVR system. If you are using the OpenScape Contact Center VoiceXML integration, a pool of IVR transit numbers is required to transfer and track calls from the IVR system to OpenScape Contact Center. Depending on the type of communication platform being used, OpenScape Contact Center uses an IVR transit number to route calls as follows:

- On the OpenScape 4000 or HiPath 4000 communication platform, the IVR transit number routes calls to an OpenScape Contact Center Route Control Group.
- On the OpenScape Voice communication platform, the IVR transit number routes calls to an OpenScape Contact Center Hunt Group.

The transit number is necessary for tracking the IVR call after it is transferred to OpenScape Contact Center.

OpenScape Contact Center maintains a pool of transit numbers, which are only used for transferring IVR calls to OpenScape Contact Center. The IVR system queries OpenScape Contact Center for a unique transit number, using the GetTransitNumber subdialog. After receiving the transit number, the IVR system must immediately transfer the call. When the call reaches the OpenScape Contact Center RCG/ACD queue/MLHG, OpenScape Contact Center identifies the call using the transit number and can then associate the call information previously received from the IVR system with the incoming call. The transit number then becomes available for use with another IVR call.

NOTE: In the event of an error or time-out, the IVR system may choose to transfer the call to a non-OpenScape Contact Center device (such as voice mail or Xpressions), in which case it is not necessary to call GetTransitNumber.

You can use the GetTransitNumber subdialog differently, depending on whether your communication platform uses Queue Hold or IVR Hold:

- **Queue Hold** — GetTransitNumber must be called immediately before the call is transferred to the OpenScape Contact Center RCG/ACD queue/MLHG, which is typically soon after the call is enqueued. OpenScape Contact Center will then look for a user and divert the call to the assigned user.
- **IVR Hold** — GetTransitNumber must be called immediately before the call is transferred. The call is still transferred to the OpenScape Contact Center RCG/ACD queue/MLHG using a transit number. For IVR Hold this is typically soon after the status of the call returned by the QueryCallStatus subdialog is “Pending” or “Unanswered”.

After transferring the call to the OpenScape Contact Center RCG/ ACD queue/MLHG, OpenScape Contact Center immediately diverts the call to the assigned user or designated time-out number.

If a transit number is requested and the IVR system cannot transfer the call immediately, then the ReleaseTransitNumber subdialog must be called. This makes the transit number available for another call. Before attempting to transfer the call to OpenScape Contact Center again, the GetTransitNumber subdialog must be called again to obtain a new transit number.

If a call at an IVR system is disconnected (by the IVR system or by caller abandon) or transferred out to any number which is not a transit number, the Terminate subdialog is used to notify OpenScape Contact Center.

NOTE: You should add the Terminate subdialog to the hang-up branch of the IVR system and be integrated into other error handling procedures as necessary. If you do not add the subdialog to the hang-up branch, your statistics will not be accurate, because OpenScape Contact Center will not know when the call ended.

3.4.1 Writing an IVR script for the IVR Hold configuration

The following is the suggested flow for an IVR script for the IVR Hold configuration:

1. Prepare the ANI and DNIS information for the call.
2. Initialize the connection between the IVR system and OpenScape Contact Center.
3. Check the system status to ensure that the Routing Server is available.
4. Identify the queue using the IVR script. For example, you could assign a queue to the call based on a combination of the call’s ANI information and the customer’s selection.
5. Enqueue the call using the Enqueue subdialog. Ensure that the parameters specify that the IVR system will hold the call until assigned to a user, using the IVR Hold configuration. This notifies the Routing Server that the IVR system will transfer the call when it is assigned.

6. Check the state of the enqueued call regularly (for example, after every action or IVR subdialog call) and take the appropriate action depending on the results:
   - If the call is assigned, request a transit number from OpenScape Contact Center, and transfer the call to the transit number.
   - Otherwise, if there is a time-out or an error, transfer the call to the default number defined in the IVR script.

NOTE: When a call is successfully enqueued, the IVR script must regularly check the state of the call using the QueryCallStatus subdialog, as well as handle conditions, such as enqueue errors and timed-out calls. In these cases, the IVR script should either transfer the call to a non-OpenScape Contact Center extension or enqueue the call with different parameters.

The diagram below shows a sample IVR script for the IVR Hold configuration.

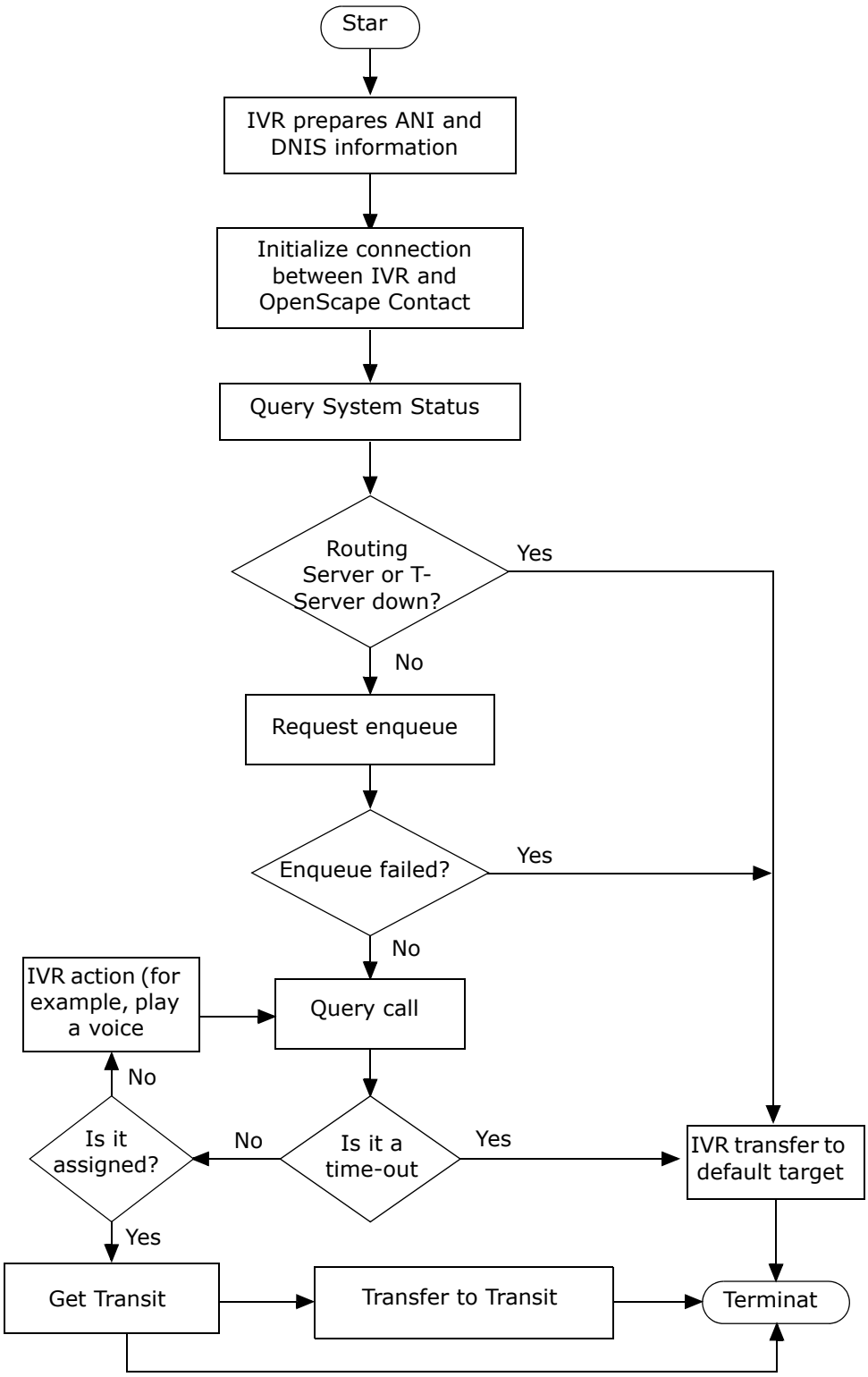

Figure 4 Sample flowchart for an IVR script in the IVR Hold configuration

3.4.2 Writing an IVR script for the Queue Hold configuration

The following is a suggested flow for a customized IVR script for the Queue Hold configuration:

1. Prepare the ANI and DNIS information for the call.
2. Initialize the connection between the IVR system and OpenScape Contact Center.
3. Check the system status to ensure that the Routing Server is available.
4. Identify the queue using the IVR script. For example, you could assign a queue to the call based on a combination of the call's ANI information and the selections the customer makes.
5. Enqueue the call with the Enqueue subdialog. Ensure that the parameters specify the Queue Hold configuration.
6. Request a transit number from OpenScape Contact Center. Transfer the call to the transit number, which will then be placed in an OpenScape Contact Center ACD/MLHG group. OpenScape Contact Center automatically holds the call in the OpenScape Contact Center ACD/MLHG group until the call is assigned to a user.

The diagram below shows a sample IVR script for the Queue Hold configuration.

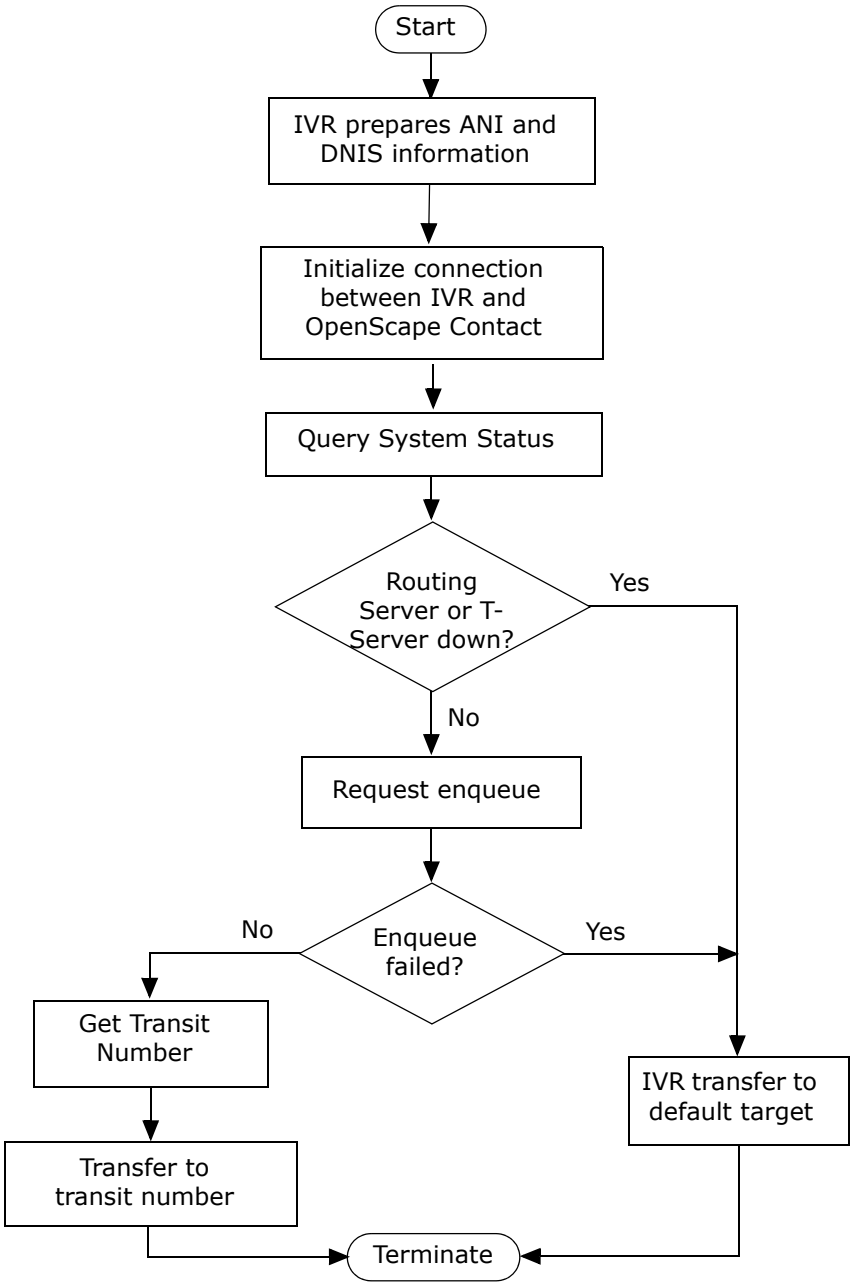

Figure 5 Sample flowchart of a IVR script for the Queue Hold configuration

NOTE: When connected to an OpenScape Voice communication platform, you must ensure that the call is transferred to the Music On Hold Hunt Group.

3.5 Using an IVR system in a multitenant environment

In a multitenant environment, IVR resources are system-level resources that are shared across multiple business units.

If you are deploying a front-end IVR application in a multitenant environment, the IVR system must contain the logic necessary for it to be aware of business units.

For example:

- The IVR system needs to determine which business unit an IVR call will be routed to.
- When an IVR system enqueues a call into the OpenScape Contact Center system, the application must enqueue the call against the correct business unit.
- In the Queue Hold scenario, an IVR system needs to know the correct pilot number for each business unit, and needs to transfer the call to the correct business unit.

Two VoiceXML subdialogs are provided for use in a multitenant environment:

- GetBusinessUnit subdialog – retrieves the business unit for a specified call. For more information, see Section 4.5, "GetBusinessUnit", on page 34.
- SetBusinessUnit subdialog – sets the business unit for a specified call. The SetBusinessUnit subdialog can be called only once for each IVR call. For more information, see Section 4.15, "SetBusinessUnit", on page 61.

NOTE: In a multitenant environment, the SetBusinessUnit subdialog must be called before the following subdialogs are invoked: CreateCallback, Enqueue.

An optional BusinessUnitName parameter can be specified in the QueryQueueStatistics subdialog to indicate the business unit that a queue belongs to. In a non-multitenant environment, the BusinessUnitName parameter is ignored.

4 Using the OpenScape Contact Center VoiceXML subdialogs

This chapter describes how to use the OpenScape Contact Center VoiceXML subdialogs and provides examples of how to create IVR scripts.

NOTE: An error code output parameter is implicit in all subdialogs.

The following code is applicable in the **Syntax** and **Example** subsections for all subdialogs:

```
<script>
<![CDATA[function GetLink(SubDialogName){return(http://
<hostname>/
<VirtualPath>/VXML/+SubDialogName)}]]>
</script>
```

where

- <*hostname*> is the name of your host machine
- <*VirtualPath*> is the path to the virtual directory you created on the corporate Web server.

4.1 CreateCallback

The CreateCallback subdialog attempts to create a callback in the Callback Server. An IVR can indicate that a customer should be called back when a user becomes available. This subdialog supports a maximum of 1000 bytes of contact data.

NOTE: You must call the Initialize subdialog before invoking this subdialog. In a multitenant environment, you must also call the SetBusinessUnit subdialog before invoking this subdialog. For more information, see Section 4.8, “Initialize”, on page 40 and Section 4.15, “SetBusinessUnit”, on page 61.

Syntax

```
<subdialog name="HPPC_CreateCallback"
srcexpr="GetLink('CreateCallback.vxml')">

   <param name="CallID" expr="Call_ID"/>

   <param name="QueueName" expr="m_QueueName"/>

   <param name="Description" expr="'Create callback
   description'"/>

   <param name="TimezoneOffset" expr="-300"/>

   <param name="ContactName" expr="'John Doe'"/>

   <param name="Schedule1" expr="'9053268045;05/25/
   2007;14:00;05/25/2007;14:30'"/>

   <param name="Schedule2" expr="'9053268046;05/25/
   2007;16:00;05/25/2007;16:30'"/>

   <param name="Schedule3" expr="'9053268046;05/26/
   2007;16:00;05/26/2007;16:30'"/>

<filled>

   <assign name="errCode" expr="HPPC_CreateCallback.ErrorCode"/>

   <assign name="ivrCallbackID"
   expr="HPPC_CreateCallback.CallbackID"/>

</filled>

</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to create callback for.
QueueName	Input	32 characters	The callback queue to use for the callback.
Description (optional)	Input	100 characters	A string containing a brief description of the call, which is displayed in the OpenScape Contact Center Agent Portal Web application so that users can identify the call.
TimeZoneOffset	Input	0 to +60*12 or 0 to -60*12	The difference in minutes between the caller's local time and UTC (Coordinated Universal Time).
ContactName	Input	80 characters	The contact name.
CallbackID	Output	17 characters	A unique ID for the callback created.

Table 1 Parameters for the CreateCallback subdialog

Name	Type	Range	Description
Schedule1	Input	47 characters for the telephone number. 81 characters for the entire string.	A string indicating the telephone number where the caller can be reached and the period of time (in the caller's local time) during which the caller wants to receive callbacks. The format is: `telephone number;mm/dd/yyyy;hh:mm;mm/dd/yyyy;hh:mm` where the first date and time (using a 24-hour clock) is the start time, and the second is the end time.
Schedule2 (optional)	Input	47 characters for the telephone number. 81 characters for the entire string.	A string indicating the telephone number where the caller can be reached and the period of time (in the caller's local time) during which the caller wants to receive callbacks. The format is: `telephone number;mm/dd/yyyy;hh:mm;mm/dd/yyyy;hh:mm` where the first date and time (using a 24-hour clock) is the start time, and the second is the end time.
Schedule3 (optional)	Input	47 characters for the telephone number. 81 characters for the entire string.	A string indicating the telephone number where the caller can be reached and the period of time (in the caller's local time) during which the caller wants to receive callbacks. The format is: `telephone number;mm/dd/yyyy;hh:mm;mm/dd/yyyy;hh:mm` where the first date and time (using a 24-hour clock) is the start time, and the second is the end time.

Table 1 Parameters for the CreateCallback subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–314 The specified business unit name was not set or it did not match the name of any business unit in the database. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

–953 The business unit has not been set for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

–1006 You are attempting to create a duplicate callback.

–1028 A schedule time is invalid.

–1029	The callback schedules have no mutual time with the contact center open hours.
–1031	A telephone number in the request is in the excluded numbers list.
–1040	All schedules have already expired.
–1045	The callback queue name is invalid.
–1047	The schedule contains a date that is too far in the future.
Other	Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

The following example requests that OpenScape Contact Center create a callback for the current call (identified by the CallID) using "CBQueue" as the callback queue.

```
<form id="hppcCreateCallback">
   <subdialog name="HPPC_CreateCallback"
   srcexpr="GetLink('CreateCallback.vxml')">
      <param name="CallID" expr="Call_ID"/>
      <param name="QueueName" expr="m_QueueName"/>
      <param name="Description" expr="'Create callback
      description'"/>
      <param name="TimezoneOffset" expr="-300"/>
      <param name="ContactName" expr="'John Doe'"/>
      <param name="Schedule1" expr="'9053268045;05/25/
      2007;14:00;05/25/2010;14:30'"/>
      <param name="Schedule2" expr="'9053268046;05/25/
      2007;16:00;05/25/2010;16:30'"/>
      <param name="Schedule3" expr="'9053268046;05/26/
      2007;16:00;05/26/2010;16:30'"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_CreateCallback.ErrorCode"/>
      <assign name="ivrCallbackID"
      expr="HPPC_CreateCallback.CallbackID"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt cond="PlayWarnings==1">CreateCallback succeed.
      Callback ID is <value expr="ivrCallbackID"/> </prompt>
```

```
      <log>HPPC CallbackID=<value expr="ivrCallbackID"/>,
      Callback ID is <value expr="ivrCallbackID"/>:
      CreateCallback succeed.</log>

   <goto next="#hppcCallbackMenu"/>

   <else/>

      <prompt>An error has occurred in CreateCallback subdialog.
      The error number is <value expr="errCode"/>. </prompt>

   <goto next="#hppcTerminate"/>

   </if>

   </block>

</form>
```

The Callback ID is returned after the callback is created.

Telephone number format

The telephone number parameter must be entered in canonical format as follows:

+[CountryCode] Space [(AreaCode) Space] SubscriberNumber [--Extension]

The telephone number can also be a dialable address obtained from a communication platform or as a result of calling a TAPI function. The following strings are correct telephone numbers:

- +1 (555) 555-0199
- (555) 555-0199
- +1 555-0199
- +1 (555) 555-0199--1212
- 5555550199

The extension part of a telephone number is truncated by the Callback Server before sending it to the T-Server. It is used only when displaying the callback in the Agent Portal Web and Manager applications, so that a user can manually dial the extension, if required.

4.2 DeleteCallback

The DeleteCallback subdialog is used to delete an existing callback that was created using the CreateCallback subdialog. This subdialog takes as its parameter the CallIbackID that was returned from the CreateCallback subdialog.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_DeleteCallback"
srcexpr="GetLink('DeleteCallback.vxml')">
   <param name="CallbackID" expr="ivrCallbackID"/>
<filled>
   <assign name="errCode" expr="HPPC_DeleteCallback.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallbackID	Input	17 characters	The CallbackID returned from a successful CreateCallback request that is to be deleted.

Table 2 Parameters for the DeleteCallback subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–1021 Invalid CallbackID.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcDeleteCallback">
   <subdialog name="HPPC_DeleteCallback"
   srcexpr="GetLink('DeleteCallback.vxml')">
      <param name="CallbackID" expr="ivrCallbackID"/>
   <filled>
```

```
      <assign name="errCode"
      expr="HPPC_DeleteCallback.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt cond="PlayWarnings==1">DeleteCallback succeed.</
      prompt>
   <goto next="#hppcTerminate"/>
   <else/>
      <prompt>An error has occurred in DeleteCallback subdialog.
      The error number is <value expr="errCode"/>. </prompt>
   <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.3 Dequeue

The Dequeue subdialog sends a request to the Routing Server to dequeue a specified call. If you are using the IVR Hold configuration, you can use the Dequeue subdialog to dequeue a call so that the IVR script can transfer the call to an extension or perform some other action with the call.

If the IVR script permits the caller to continue to input choices after the call has been enqueued, you can use further caller input to change the routing of the call. For example, if callers wait too long in queue, they might decide to leave a voice message; the IVR script can then dequeue the call and transfer it to a voice mail extension.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_Dequeue"
srcexpr="GetLink('Dequeue.vxml')">
   <!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
<filled>
   <assign name="errCode" expr="HPPC_Dequeue.ErrorCode"/>
```

```
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to dequeue.

Table 3 Parameters for the Dequeue subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

-308 The CallID specified was not found. You must call Initialize to obtain the CallID.

-914 The operation you are attempting has failed.

-950 VoiceXML is not currently enabled.

-951 The Routing Server is not available.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

The following example sends a request to dequeue a call.

```
<form id="hppcDequeue">
   <subdialog name="HPPC_Dequeue"
   srcexpr="GetLink('Dequeue.vxml')">
      <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="errCode" expr="HPPC_Dequeue.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>Dequeue succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: Dequeue succeed.</log>
      <goto next="#hppcTerminate"/>
   <else/>
```

```
        <prompt>Dequeue failed. Error ID is <value expr="errCode"/
        >. </prompt>
        <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
        <value expr="Call_ID"/>. Dequeue failed. Error ID is
        <value expr="errCode"/>.</log>
        <goto next="#hppcTerminate"/>
    </if>
    </block>
</form>
```

4.4 Enqueue

The Enqueue subdialog requests that the Routing Server enqueue a call and find the best available user to handle the call. This subdialog passes the routing information of a call to the Routing Server, including the queue and the call's initial priority.

NOTE: If you want to associate the contact data with the call, you must set these parameters before enqueuing the call.

Use this subdialog to enqueue calls in either the IVR Hold or the Queue Hold configuration. After a call is enqueued in the IVR Hold configuration, the IVR script must check if the call has been assigned using the QueryCallStatus subdialog, and then transfer the call to the user's extension. After a call is enqueued in the Queue Hold configuration, the call should be immediately transferred to an OpenScape Contact Center ACD/MLHG group.

NOTE: You must call the Initialize subdialog before invoking this subdialog. In a multitenant environment, you must also call the SetBusinessUnit subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40, and Section 4.15, "SetBusinessUnit", on page 61.

Syntax

```
<subdialog name="HPPC_Enqueue"
srcexpr="GetLink(&apos;Enqueue.vxml&apos;)">
    <param name="CallID" expr="Call_ID"/>
    <param name="IVRHold" expr="ivrHold"/>
    <param name="QueueName" expr="&apos;Sales&apos;"/>
    <param name="InitialPriority" expr="m_InitialPriority"/>
    <param name="Description" expr="m_Description"/>
```

```
    <param name="AgentId" expr="m_AgentId"/>
    <param name="AgentWaitTime" expr="m_AgentWaitTime"/>
<filled>
    <assign name=
    "m_EstimatedWait"expr="HPPC_Enqueue.EstimatedWait"/>
    <assign name="m_CallsInQ" expr="HPPC_Enqueue.CallsInQ"/>
    <assign name="errCode" expr="HPPC_Enqueue.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to enqueue.
IVRHold	Input	0 or 1	0 - Indicates the call will be transferred to an ACD/MLHG group for the Queue Hold configuration. 1- Indicates the call will wait at the IVR extension for the IVR Hold configuration.
QueueName	Input	32 characters	The name of the queue for the call.
InitialPriority	Input	1 to 100	Specifies the priority of a call, where 1 is the lowest and 100 is the highest priority.
Description (optional)	Input	100 characters	A string containing a brief description of the call, which is displayed in the OpenScape Contact Center Agent Portal Web application so that users can identify the call.
EstimatedWait	Output	Greater than or equal to 0	The estimated wait time in seconds for this queue.
CallsInQueue	Output	Greater than or equal to 0	The number of calls in queue for the specified queue.
AgentID (optional)	Input	32 characters	The user for whom you want to reserve the call.
AgentWait Time (optional)	Input	32 characters	The maximum time in seconds the call can wait for a reserved user. If this time expires, the call is enqueued to the queue specified by the QueueName.

Table 4 Parameters for the Enqueue subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–305 This code indicates an unknown error (for example, user error or the system is unstable).

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–314 The specified business unit name was not set or it did not match the name of any business unit in the database. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

–400 The system has low system resources.

–904 The specified agent ID did not match the ID of any user in the database.

–905 The specified queue did not match the name of any queue in the database.

-914 The operation you are attempting has failed.

-950 VoiceXML is not currently enabled.

-951 The Routing Server is not available.

–953 The business unit has not been set for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, “Error codes”.

Example

```
<form id="hppcEnqueue">
   <var name="m_EstimatedWait"/>
   <var name="m_CallsInQ"/>
   <subdialog name="HPPC_Enqueue"
   srcexpr="GetLink(&apos;Enqueue.vxml&apos;)">
      <param name="CallID" expr="Call_ID"/>
      <param name="IVRHold" expr="ivrHold"/>
      <param name="QueueName" expr="&apos;Sales&apos;"/>
      <param name="InitialPriority" expr="m_InitialPriority"/>
      <param name="Description" expr="m_Description"/>
      <param name="AgentId" expr="m_AgentId"/>
      <param name="AgentWaitTime" expr="m_AgentWaitTime"/>
   <filled>
      <assign name=
      "m_EstimatedWait"expr="HPPC_Enqueue.EstimatedWait"/>
```

```
        <assign name="m_CallsInQ" expr="HPPC_Enqueue.CallsInQ"/>
        <assign name="errCode" expr="HPPC_Enqueue.ErrorCode"/>
    </filled>
    </subdialog>
    <block>
    <if cond="errCode == 0">
        <prompt>Enqueue succeed.</prompt>
        <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
        <value expr="Call_ID"/>: Enqueue succeed.
        EstimatedWaitTime = <value expr="m_EstimatedWait"/>
        CallsInQ = <value expr="m_CallsInQ"/>
        </log>
        <goto next="#hppcGetTransitNumber"/>
    <else/>
        <prompt>Enqueue failed. Error ID is <value expr="errCode"/
        >. </prompt>
        <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
        <value expr="Call_ID"/>. Enqueue failed. Error ID is
        <value expr="errCode"/>.</log>
        <goto next="#hppcTerminate"/>
    </if>
    </block>
</form>
```

4.5 GetBusinessUnit

NOTE: The GetBusinessUnit subdialog is supported only in a multitenant environment.

The GetBusinessUnit subdialog retrieves the business unit for the specified call.

NOTE: You must call the Initialize and SetBusinessUnit subdialogs before invoking this subdialogs. For more information, see Section 4.8, “Initialize”, on page 40, and Section 4.15, “SetBusinessUnit”, on page 61.

Syntax

```
<subdialog name="HPPC_GetBusinessUnit"
srcexpr="GetLink('GetBusinessUnit.vxml')">
```

```
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_GetBusinessUnit.ErrorCode"/>
      <assign name="BUName"
      expr="HPPC_GetBusinessUnit.BusinessUnitName"/>
   </filled>
   </subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID of the call that you want to retrieve the business unit for.
BusinessUnit Name	Output	32 characters	The name of the business unit for the specified call.

Table 5 Parameters for the GetBusinessUnit subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–400 The system has low system resources.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–953 The business unit has not been set for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

The following example retrieves the business unit name for the specified CallID.

```
<form id="hppcGetBusinessUnit">
   <var name="BUName"/>
   <subdialog name="HPPC_GetBusinessUnit"
   srcexpr="GetLink('GetBusinessUnit.vxml')">
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_GetBusinessUnit.ErrorCode"/>
```

```
      <assign name="BUName"
      expr="HPPC_GetBusinessUnit.BusinessUnitName"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt cond="PlayWarnings==1">Get Business Unit
      succeed.</prompt>
      <prompt>
      Business Unit Name is <value expr="BUName"/>.
      </prompt>
      <log>HPPC Call ID is <value expr="Call_ID"/>: Get Business
      Unit succeed. Business Unit Name is <value expr="BUName"/
      >.</log>
   <else/>
      <prompt>An error has occurred in Get Business Unit
      subdialog. The error number is <value expr="errCode"/>. </
      prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. Get Business Unit failed. Error
      ID is <value expr="errCode"/>.</log>
   </if>
   <goto next="#hppcMainMenu"/>
   </block>
   </form>
```

4.6 GetContactData

The GetContactData subdialog retrieves the contact data for a specified CallID. If you want to associate the contact data with the call, you must set the contact data before enqueuing the call.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, “Initialize”, on page 40.

Syntax

```
<subdialog name="HPPC_GetContactData"
srcexpr="GetLink('GetContactData.vxml')">
   <param name="CallID" expr="Call_ID"/>
   <param name="Key" expr="'queuekey'"/>
<filled>
```

```
   <assign name="m_value" expr="HPPC_GetContactData.Value"/>
   <assign name="errCode" expr="HPPC_GetContactData.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to get the contact data for.
Key	Input	32 characters	The key name for the contact data value.
Value	Output	128 characters	The value returned for the key.

Table 6 Parameters for the GetContactData subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–307 The key name for the contact data value was not set prior to invoking this function.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcGetContactData">
   <subdialog name="HPPC_GetContactData"
   srcexpr="GetLink('GetContactData.vxml')">
      <param name="CallID" expr="Call_ID"/>
      <param name="Key" expr="'queuekey'"/>
   <filled>
      <assign name="m_value" expr="HPPC_GetContactData.Value"/>
      <assign name="errCode"
      expr="HPPC_GetContactData.ErrorCode"/>
   </filled>
   </subdialog>
```

```
   <block>
   <if cond="errCode == 0">
      <prompt>GetContactData succeed. Contact Data Value is
      <value expr="m_value"</prompt>
      <log>HPPC Call ID is <value expr="Call_ID"/>:
      GetContactData succeed.</log>
      <goto next="#hppcSetDisplay"/>
   <else/>
      <prompt>GetContactData failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. GetContactData failed. Error ID
      is <value expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.7 GetTransitNumber

The GetTransitNumber subdialog queries OpenScape Contact Center for a transit number. The IVR system must transfer the call to the transit number received in this request. The transit number expires if the call is not transferred within 30 seconds. The transit number is then available for use with another call.

NOTE: For Queue Hold, this subdialog is called after calling the Enqueue subdialog. For IVR Hold, it is called after receiving Pending or Unanswered status in the QueryCallStatus subdialog.

Syntax

```
<subdialog name="HPPC_GetTransitNumber"
srcexpr="GetLink(&apos;GetTransitNumber.vxml&apos;)">
<!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
<filled>
   <assign name="TransferDest"
   expr="HPPC_GetTransitNumber.TransitNumber"/>
   <assign name="errCode"
   expr="HPPC_GetTransitNumber.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to enqueue.
TransitNumber	Output	80 characters	A pilot number that the IVR will use to transfer the call.

Table 7 Parameters for the GetTransitNumber subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–908 The T-Server is not available.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcGetTransitNumber">
   <subdialog name="HPPC_GetTransitNumber"
   srcexpr="GetLink(&apos;GetTransitNumber.vxml&apos;)">
   <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="TransferDest"
      expr="HPPC_GetTransitNumber.TransitNumber"/>
      <assign name="errCode"
      expr="HPPC_GetTransitNumber.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>GetTransitNumber succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: GetTransitNumber succeed.
      TransitNumber = <value expr="TransferDest"/>
```

```
      </log>
      <goto next="#hppcTransfer"/>
   <else/>
      <prompt>GetTransitNumber failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. GetTransitNumber failed. Error ID
      is <value expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.8 Initialize

The Initialize subdialog initializes the connection to the OpenScape Contact Center servers and must be called before any of the other OpenScape Contact Center VoiceXML subdialogs are invoked. This subdialog clears the ANI and DNIS that were set up during the previous call on this extension. Therefore, invoke this subdialog at the beginning of your IVR script and ensure that OpenScape Contact Center is operational before routing calls.

Syntax

```
<subdialog name="HPPC_Initialize"
srcexpr="GetLink(&apos;Initialize.vxml&apos;)">
<param name="UniqueID" expr="uniqueid"/>
<param name="ANI" expr="ani"/>
<param name="DNIS" expr="dnis"/>
<prompt>Initialize started.</prompt>
<filled>
   <assign name="errCode" expr="HPPC_Initialize.ErrorCode"/>
   <assign name="Call_ID" expr="HPPC_Initialize.CallID"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
UniqueID	Input	16 characters	The device associated with the IVR extension on which the IVR script is running.
ANI (optional)	Input	161 characters	The ANI number of the call. The Statistics Server stores this number in the OpenScape Contact Center database.
DNIS (optional)	Input	161 characters	The DNIS number of the call. The Statistics Server stores this number in the OpenScape Contact Center database.
CallID	Output	18 characters	The CallID for the current call.

Table 8 Parameters for the Initialize subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–908 The T-Server is not available.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcInitialize">
   <subdialog name="HPPC_Initialize"
   srcexpr="GetLink(&apos;Initialize.vxml&apos;)">
      <param name="UniqueID" expr="dnis"/>
      <param name="ANI" expr="ani"/>
      <param name="DNIS" expr="dnis"/>
      <prompt>Initialize started.</prompt>
   <filled>
      <assign name="errCode" expr="HPPC_Initialize.ErrorCode"/>
      <assign name="Call_ID" expr="HPPC_Initialize.CallID"/>
```

```
    </filled>
    </subdialog>
    <block>
    <if cond="errCode == 0">
       <prompt>Initialize succeed.</prompt>
       <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
       <value expr="Call_ID"/>: Initialize succeed.</log>
       <goto next="#hppcQueryRoutingInfo"/>
    <else/>
       <prompt>Initialize failed. Error ID is <value
       expr="errCode"/>.</prompt>
       <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
       <value expr="Call_ID"/>. Initialize failed. Error ID is
       <value expr="errCode"/>.</log>
       <goto next="#hppcTerminate"/>
    </if>
    </block>
</form>
```

4.9 QueryAgentStatus

The QueryAgentStatus subdialog polls for the status of a user associated with a given device. This subdialog will return the status of the user as a string value depending on whether the user is logged on or not. You must specify either the agent ID or the device. If the agent ID is specified and the user is logged on, the device will be filled in and the status returned. If the device is specified and there is a user logged on to the specified device, the agent ID will be filled in and the status returned.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, “Initialize”, on page 40.

Syntax

```
<subdialog name="HPPC_QueryAgentStatus"
srcexpr="GetLink('QueryAgentStatus.vxml')">
   <param name="inAgentID" expr="'1234'"/>
   <param name="inDevice" expr="''"/>
<filled>
   <assign name="m_AgentId"
   expr="HPPC_QueryAgentStatus.outAgentID"/>
```

```
    <assign name="Device" expr="HPPC_QueryAgentStatus.outDevice"/
    >
    <assign name="AgentStatus"
    expr="HPPC_QueryAgentStatus.AgentStatus"/>
    <assign name="errCode"
    expr="HPPC_QueryAgentStatus.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
AgentID	Input and Output	8 characters	A string identifying the user; empty if the Device is provided.
Device	Input and Output	16 characters	The user extension you want to query; empty if AgentID is provided.
AgentStatus	Output	Greater than 0	Indicates the status of a user. 1 - Dialing 2 - Line busy 3 - Ringing 4 - Talking 5 - Line queued 6 - Holding 7 - Consulting 8 - Out of service 9 - Available 10 - Unavailable 11 - Work 12 - Logged off 13 - Unknown 14 - Pending 15 - Processing 16 - Post-processing

Table 9 Parameters for the QueryAgentStatus subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–904 The specified agent ID did not match the ID of any user in the database.

–908 The T-Server is not available.

–909 The specified device did not match any device in the database.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcQueryAgentStatus">
   <var name="AgentStatus"/>
   <var name="Device"/>
   <subdialog name="HPPC_QueryAgentStatus"
   srcexpr="GetLink('QueryAgentStatus.vxml')">
      <param name="inAgentID" expr="'1234'"/>
      <param name="inDevice" expr="'6903'"/>
   <filled>
      <assign name="m_AgentId"
      expr="HPPC_QueryAgentStatus.outAgentID"/>
      <assign name="Device"
      expr="HPPC_QueryAgentStatus.outDevice"/>
      <assign name="AgentStatus"
      expr="HPPC_QueryAgentStatus.AgentStatus"/>
      <assign name="errCode"
      expr="HPPC_QueryAgentStatus.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>Query Agent Status succeed </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: QueryAgentStatus succeed.</log>
      <goto next="#hppcDequeue"/>
   <else/>
      <prompt>Query Agent Status failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. QueryAgentStatus failed. Error ID
      is <value expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.10 QueryCallStatus

The QueryCallStatus subdialog polls for the status of the call associated with a specified CallID. The target to which the call should be transferred is returned by the Extension parameter based on the status of the call (Pending or Unanswered). If the status of the call is Disconnected, then the call should be disconnected by the IVR system (in the IVR Hold scenario).

After a call has been enqueued, regularly invoke the QueryCallStatus subdialog to determine if the call has been assigned so that the IVR system can transfer the call as soon as a user is assigned by the Routing Server. To do this the IVR system should call the GetTransitNumber subdialog. The call can then be transferred to the transit number. For more information, see Section 4.7, "GetTransitNumber", on page 38.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

NOTE: When using the QueryCallStatus subdialog, the position in queue is only returned if the TimeOut parameter is set to 0. If the TimeOut parameter is set to a value that is greater than 0, the QueryCallStatus subdialog waits for the status to change, but does not query the position in queue from the Routing Server. For best results, call the QueryCallStatus subdialog and specify a TimeOut value of 0 to return the position in queue. Then call the QueryCallStatus subdialog again and specify a TimeOut value that is greater than 0 and wait for the status to change.

Syntax

```
<subdialog name="HPPC_QueryCallStatus"
srcexpr="GetLink('QueryCallStatus.vxml')">
   <!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
   <!-- TimeOut in msec. -->
   <param name="TimeOut" expr="1000"/>
<filled>
   <assign name="m_PositionInQ"
   expr="HPPC_QueryCallStatus.PositionInQ"/>
   <assign name="errCode" expr="HPPC_QueryCallStatus.ErrorCode"/
   >
```

```
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID returned from Initialize.
TimeOut	Input	Greater than or equal to 0	The length of time (in milliseconds) spent waiting for a change in call status on the given device. Specifying zero (0) indicates an immediate poll of the current status.
PositionInQ	Output	Greater than or equal to 0	The position of the call in the queue, as returned by the Routing Server, if the TimeOut parameter is set to 0.

Table 10 Parameters for the QueryCallStatus subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 The call is idle.

1 The call is in Queued state. In this case, continue checking the call status.

2 The call is in Pending state. In this case, transfer the call to the OpenScape Contact Center transit number.

3 The call is in Unanswered state. In this case, transfer the call to the OpenScape Contact Center transit number or to another time-out extension.

4 An error has occurred. In this case, transfer the call to a non-OpenScape Contact Center extension.

5 The call must be disconnected.

6 The call must be transferred to the OpenScape Contact Center transit number.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–951 The Routing Server is not available.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, “Error codes”.

Example

```
<form id="hppcQueryCallStatus">
   <var name="m_PositionInQ"/>
   <subdialog name="HPPC_QueryCallStatus"
   srcexpr="GetLink('QueryCallStatus.vxml')">
      <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
      <!-- TimeOut in msec. -->
      <param name="TimeOut" expr="1000"/>
   <filled>
      <assign name="m_PositionInQ"
      expr="HPPC_QueryCallStatus.PositionInQ"/>
      <assign name="errCode"
      expr="HPPC_QueryCallStatus.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
      <prompt cond="PlayWarnings==1">QueryCallStatus finished
      with error code <value expr="errCode"/></prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: QueryCallStatus finished.
      TransferDest = <value expr="TransferDest"/>
      PositionInQ = <value expr="m_PositionInQ"/>
      errCode = <value expr="errCode"/>
   </log>
   <if cond="errCode == 1">
      <prompt>Your position in queue is <value
      expr="m_PositionInQ"/>. Your call is in Queue state.
      Continue to wait</prompt>
      <goto next="#hppcSetDisplay"/>
   <elseif cond="errCode == 2"/>
      <prompt>The call is in Pending state and will be
      transferred.</prompt>
      <goto next="#hppcGetTransitNumber"/>
   <elseif cond="errCode == 3"/>
      <prompt>The call is in Unanswered state and will be
      transferred.</prompt>
      <goto next="#hppcGetTransitNumber"/>
   <elseif cond="errCode == 4"/>
      <prompt>An error has occurred and will be transferred to a
      non-OpenScape Contact Center extension.</prompt>
      <goto next="#hppcErrorTransfer"/>
   <elseif cond="errCode == 5"/>
```

```
            <prompt>The disconnect component was encountered in
            workflow. The call must be terminated.</prompt>

            <goto next="#hppcTerminate"/>

        <elseif cond="errCode == 6"/>

            <prompt>A transfer component was encountered in
            workflow.</prompt>

            <goto next="#hppcGetTransitNumber"/>

        <else/>

            <prompt>An error has occurred in QueryCallStatus
            subdialog. The error number is <value expr="errCode"/>. </
            prompt>

            <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
            <value expr="Call_ID"/>. QueryCallStatus failed. Error ID
            is <value  expr="errCode"/>.</log>

            <goto next="#hppcMainMenu"/>

        </if>

        </block>
    </form>
```

4.11 QueryQueueStatistics

The QueryQueueStatistics subdialog determines the number of calls in queue, the estimated wait time for a call, the average wait time for the queue, the time that the oldest call has been in queue, and the service level for the specified queue.

These calculations assume the call will be enqueued to the Routing Server immediately, regardless of whether the call has been enqueued. You should invoke QueryQueueStatistics before invoking Enqueue to ensure an accurate response from the Routing Server.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, “Initialize”, on page 40.

Syntax

```
<subdialog name="HPPC_QueryQueueStatistics"
srcexpr="GetLink('QueryQueueStatistics.vxml')">

   <!-- call id of current call. Mandatory-->

   <param name="QueueName" expr="m_QueueName"/>

   <param name="BusinessUnitName" expr="m_BusinessUnitName"/>

<filled>

   <assign name="m_CallsInQ"
   expr="GetNumber(HPPC_QueryQueueStatistics.CallsInQ)"/>

   <assign name="m_EstimatedWait"
   expr="GetNumber(HPPC_QueryQueueStatistics.EstimatedWait)"/>

   <assign name="m_AverageWait"
   expr="GetNumber(HPPC_QueryQueueStatistics.AverageWait)"/>

   <assign name="m_OldestCallinQueue"
   expr="GetNumber(HPPC_QueryQueueStatistics.OldestCallinQueue)"
   />

   <assign name="m_ServiceLevel"
   expr="GetNumber(HPPC_QueryQueueStatistics.ServiceLevel)"/>

   <assign name="errCode"
   expr="GetNumber(HPPC_QueryQueueStatistics.ErrorCode)"/>

</filled>

</subdialog>
```

Parameters

Name	Type	Range	Description
QueueName	Input	32 characters	The queue for which the query is performed.
CallsInQueue	Output	Greater than or equal to 0	The number of calls in queue for the specified queue.
EstimatedWait	Output	Greater than or equal to 0	The estimated wait time in seconds for the specified queue.
AverageWait	Output	Greater than or equal to 0	The average wait time in seconds for the specified queue.

Table 11 Parameters for the QueryQueueStatistics subdialog

Name	Type	Range	Description
OldestCall InQueue	Output	Greater than or equal to 0	The time in seconds that the oldest call has been in the queue.
ServiceLevel	Output	0 – 100	The service level for the specified queue.
BusinessUnit Name (optional)	Input	32 characters	The name of the business unit for the specified queue. This parameter is required in a multitenant environment and ignored in a non-multitenant environment.

Table 11 Parameters for the QueryQueueStatistics subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–314 The specified business unit name was not set or it did not match the name of any business unit in the database. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

–905 The specified queue did not match the name of any queue in the database.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–951 The Routing Server is not available.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcQueryQueueStatistics">
   <block>
   <prompt>You query queue statistics for <value
   expr="m_QueueName"/></prompt>
   </block>
      <var name="m_EstimatedWait"/>
      <var name="m_CallsInQ"/>
      <var name="m_AverageWait"/>
      <var name="m_OldestCallinQueue"/>
      <var name="m_ServiceLevel"/>
   <subdialog name="HPPC_QueryQueueStatistics"
   srcexpr="GetLink('QueryQueueStatistics.vxml')">
      <!-- call id of current call. Mandatory-->
```

```
    <param name="QueueName" expr="m_QueueName"/>
<filled>
    <assign name="m_CallsInQ"
    expr="GetNumber(HPPC_QueryQueueStatistics.CallsInQ)"/>
    <assign name="m_EstimatedWait"
    expr="GetNumber(HPPC_QueryQueueStatistics.EstimatedWait)"/
    >
    <assign name="m_AverageWait"
    expr="GetNumber(HPPC_QueryQueueStatistics.AverageWait)"/>
    <assign name="m_OldestCallinQueue"
    expr="GetNumber(HPPC_QueryQueueStatistics.OldestCallinQueu
    e)"/>
    <assign name="m_ServiceLevel"
    expr="GetNumber(HPPC_QueryQueueStatistics.ServiceLevel)"/>
    <assign name="errCode"
    expr="GetNumber(HPPC_QueryQueueStatistics.ErrorCode)"/>
</filled>
</subdialog>
<block>
<if cond="errCode == 0">
    <prompt cond="PlayWarnings==1">QueryQueueStatistics
    succeed.</prompt>
    <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
    <value expr="Call_ID"/>: QueryQueueStatistics succeed.
    CallsInQ = <value expr="m_CallsInQ"/>
    EstimatedWait = <value expr="m_EstimatedWait"/>
    AverageWait = <value expr="m_AverageWait"/>
    OldestCallinQueue = <value expr="m_OldestCallinQueue"/>
    ServiceLevel = <value expr="m_ServiceLevel"/>
    </log>
    <prompt>There are <say-as interpret-as="number"><value
    expr="m_CallsInQ"/></say-as> calls in queue. Estimated
    Wait time <value expr="m_EstimatedWait"/> seconds</prompt>
<if cond="m_CallsInQ > 0">
    <prompt>
    Call center is busy, please create a callback.
    </prompt>
    <goto next="#hppcCreateCallback"/>
    <else/>
    <prompt>
    Trasferring the call to queue <value expr = "m_QueueName"/
    >.
    </prompt>
<goto next="#hppcSetContactData"/>
```

```
    </if>
    <else/>
       <prompt>An error has occurred in QueryQueueStatistics
       subdialog. The error number is <value expr="errCode"/>. </
       prompt>
       <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
       <value expr="Call_ID"/>. QueryQueueStatistics failed.
       Error ID is <value expr="errCode"/>.</log>
    <goto next="#hppcTerminate"/>
    </if>
    </block>
</form>
```

4.12 QueryRoutingInfo

The QueryRoutingInfo subdialog sends a request to determine routing information for a call. The query is based on information collected and submitted by the IVR system as part of an OpenScape Contact Center routing request.

You should enqueue, disconnect, or transfer the call based on the query result.

If a Transfer component is encountered in a workflow, then the code returned is 2. The transfer destination that is returned may be in canonical form, depending on how the Transfer component has been configured. In this case, if you want to transfer the call to this destination using only a hook-flash transfer, or other internal IVR transfer mechanism, then you must translate the number appropriately. If you want to transfer the call using the IVR API, then the parsing will be handled directly by OpenScape Contact Center.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, “Initialize”, on page 40.

NOTE: If you want the contact data values to be considered in the routing decision, you must set these values, using the SetContactData subdialog, before invoking this subdialog. To retain the original contact data values, you must copy and reset the values before invoking the Enqueue subdialog. For more information, see Section 4.16, “SetContactData”, on page 63.

Syntax

```
<subdialog name="HPPC_QueryRoutingInfo"
srcexpr="GetLink(&apos;QueryRoutingInfo.vxml&apos;)">

   <param name="CallID" expr="Call_ID"/>

<filled>

   <assign name="m_Destination"
   expr="HPPC_QueryRoutingInfo.Destination"/>

   <assign name="m_AgentId"
   expr="HPPC_QueryRoutingInfo.AgentId"/>

   <assign name="m_AgentWaitTime"
   expr="HPPC_QueryRoutingInfo.AgentWaitTime"/>

   <assign name="m_Description"
   expr="HPPC_QueryRoutingInfo.Description"/>

   <assign name="m_InitialPriority"
   expr="HPPC_QueryRoutingInfo.InitialPriority"/>

   <assign name="m_EstimatedWait"
   expr="HPPC_QueryRoutingInfo.EstimatedWait"/>

   <assign name="m_CallsInQ"
   expr="HPPC_QueryRoutingInfo.CallsInQ"/>

   <assign name="errCode"
   expr="HPPC_QueryRoutingInfo.ErrorCode"/>

</filled>

</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID of the call for which you want to obtain the appropriate routing information.
Destination	Output	80 characters	Return code 0 – The queue to which the call should be enqueued. Return code 1 – The call should be disconnected. Return code 2 – The target to which the call should be transferred.
AgentID	Output	8 characters	Returns the ID of the user who should be reserved to handle the call. If no AgentID is returned, use the Enqueue subdialog. **Note:** Valid only if the error code is 0.

Table 12 Parameters for the QueryRoutingInfo subdialog

Name	Type	Range	Description
AgentWaitTime	Output	Greater than 0	Returns the maximum time in seconds the caller must wait for a reserved user before the call is released and goes to the specified queue. This is only valid if an AgentID is returned. **Note:** Valid only if the error code is 0.
Description	Output	100 characters	Provides a description of the call, which is displayed in the OpenScape Contact Center Agent Portal Web application so that users can identify the call. **Note:** Valid only if the error code is 0.
InitialPriority	Output	1 to 100	Returns the priority of the call relative to all the other calls queued in the Routing Server, including calls associated with the same queue as the current call. The default is 1. **Note:** Valid only if the error code is 0.
EstimatedWait	Output	Greater than or equal to 0	Returns the estimated wait time in seconds for the specified queue. **Note:** Valid only if the error code is 0.
CallsInQueue	Output	Greater than or equal to 0	Returns the number of calls for the specified queue. **Note:** Valid only if the error code is 0.

Table 12 Parameters for the QueryRoutingInfo subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful. Indicates that the call must be enqueued with the returned information.

1 Successful. Indicates that the call must be disconnected.

2 Successful. Indicates that the call must be transferred to the returned target.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–951 The Routing Server is not available.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

The following example queries the Routing Server for routing information.

```
<form id="hppcQueryRoutingInfo">
   <var name="m_EstimatedWait"/>
   <var name="m_CallsInQ"/>
   <var name="m_Destination"/>
   <subdialog name="HPPC_QueryRoutingInfo"
   srcexpr="GetLink(&apos;QueryRoutingInfo.vxml&apos;)">
   <!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="m_Destination"
      expr="HPPC_QueryRoutingInfo.Destination"/>
      <assign name="m_AgentId"
      expr="HPPC_QueryRoutingInfo.AgentId"/>
      <assign name="m_AgentWaitTime"
      expr="HPPC_QueryRoutingInfo.AgentWaitTime"/>
      <assign name="m_Description"
      expr="HPPC_QueryRoutingInfo.Description"/>
      <assign name="m_InitialPriority"
      expr="HPPC_QueryRoutingInfo.InitialPriority"/>
      <assign name="m_EstimatedWait"
      expr="HPPC_QueryRoutingInfo.EstimatedWait"/>
      <assign name="m_CallsInQ"
      expr="HPPC_QueryRoutingInfo.CallsInQ"/>
      <assign name="errCode"
      expr="HPPC_QueryRoutingInfo.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>QueryRoutingInfo succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: QueryRoutingInfo succeed.
      Destination = <value expr="m_Destination"/>
      <AgentId = <value expr="m_AgentId"/>
      <AgentWaitTime = <value expr="m_AgentWaitTime"/>
      <Description = <value expr="m_Description"/>
      <InitialPriority = <value expr="m_InitialPriority"/>
```

```
      <EstimatedWaitTime = <value expr="m_EstimatedWait"/>
      <CallsInQ = <value expr="m_CallsInQ"/>
      </log>
      <goto next="#hppcSetContactData"/>
   <else/>
      <prompt>QueryRoutingInfo failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. QueryRoutingInfo failed. Error ID
      is <value expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.13 QuerySystemStatus

The QuerySystemStatus subdialog determines the status of the system. This subdialog should be called to verify the system status at the beginning of the IVR script and before enqueuing a call.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_QuerySystemStatus"
srcexpr="GetLink('QuerySystemStatus.vxml')">
<filled>
   <assign name="QSSOverallStatus"
   expr="HPPC_QuerySystemStatus.OverallStatus"/>
   <assign name="QSSServerState"
   expr="HPPC_QuerySystemStatus.ServerState"/>
   <assign name="QSSEntries"
   expr="HPPC_QuerySystemStatus.Entries"/>
   <assign name="errCode"
   expr="HPPC_QuerySystemStatus.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
OverallStatus	Output	0 or 1	0 - Indicates the system is not operational 1 - Indicates the system is operational.
IndividualServerStates	Output	250 characters	A string containing each server name and its state. For example: server-name1=1; server-name2=2
Entries	Output	Greater than or equal to 0	The number of servers in the status string.

Table 13 Parameters for the QuerySystemStatus subdialog

The IVR script should parse the individual server states to find the status of the Routing Server and T-Server. If the Routing Server state is not 1, or the T-Server state is not 8, the call should be transferred to an ACD/MLHG number that routes calls to a backup ACD/MLHG group.

Status values

The following codes indicate the status of the various servers:

0 The server is not operational.

1 The server is operational.

2 The server is inactive. This means that the Administration Server is operational, but it has not received an indication of the status for the specified server.

Only the T-Server uses the following codes:

3 The T-Server is not operational.

4 The T-Server is in the process of shutting down.

5 The T-Server is out of service because it cannot connect to the CSTA provider.

6 The T-Server is in the process of starting.

7 The T-Server is in the process of initializing.

8 The T-Server is operational.

9 The communication platform is overloaded and is having problems carrying out the T-Server's requests.

10 The T-Server is in the process of recovering.

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–400 The system has low system resources.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcQuerySystemStatus">
      <var name="QSSOverallStatus"/>
      <var name="QSSServerState"/>
      <var name="QSSEntries"/>
   <subdialog name="HPPC_QuerySystemStatus"
   srcexpr="GetLink('QuerySystemStatus.vxml')">
   <filled>
      <assign name="QSSOverallStatus"
      expr="HPPC_QuerySystemStatus.OverallStatus"/>
      <assign name="QSSServerState"
      expr="HPPC_QuerySystemStatus.ServerState"/>
      <assign name="QSSEntries"
      expr="HPPC_QuerySystemStatus.Entries"/>
      <assign name="errCode"
      expr="HPPC_QuerySystemStatus.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt cond="PlayWarnings==1">QuerySystemStatus succeed
      </prompt>
      <prompt> The overal status is <value
      expr="QSSOverallStatus"/>. There are <value
      expr="QSSEntries"/> entries in the returned status data.
      see log for details.</prompt>
      <log>HPPC Call ID is <value expr="Call_ID"/>:
      QuerySystemStatus succeed. The overal status is <value
      expr="QSSOverallStatus"/>. There are <value
      expr="QSSEntries"/> entries in the returned status data as
      follows <value expr="QSSServerState"/>.</log>
   <else/>
      <prompt>An error has occurred in QuerySystemStatus
      subdialog. The error number is <value expr="errCode"/>. </
      prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. QuerySystemStatus failed. Error
      ID is <value expr="errCode"/>.</log>
   </if>
```

```
   <goto next="#hppcMainMenu"/>
   </block>
</form>
```

4.14 ReleaseTransitNumber

The ReleaseTransitNumber subdialog allows applications to request that OpenScape Contact Center release a transit number for use by another call. This subdialog must be called before the transit number expires. This means that the transit number will become available for another OpenScape Contact Center call.

The IVR system may use the ReleaseTransitNumber subdialog to inform the T-Server that the transit number is no longer needed. The application must call the GetTransitNumber subdialog again to request a new transit number before attempting to transfer the call to OpenScape Contact Center.

Syntax

```
<subdialog name="HPPC_ReleaseTransitNumber"
srcexpr="GetLink(&apos;ReleaseTransitNumber.vxml&apos;)">
<!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
<filled>
   <assign name="errCode"
   expr="HPPC_ReleaseTransitNumber.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the current call.

Table 14 Parameters for the ReleaseTransitNumber subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–908 The T-Server is not available.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcReleaseTransitNumber">
   <subdialog name="HPPC_ReleaseTransitNumber"
   srcexpr="GetLink(&apos;ReleaseTransitNumber.vxml&apos;)">
      <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_ReleaseTransitNumber.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>ReleaseTransitNumber succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: ReleaseTransitNumber succeed.</
      log>
      <goto next="#hppcTerminate"/>
   <else/>
      <prompt>ReleaseTransitNumber failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. ReleaseTransitNumber failed.
      Error ID is <value expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.15 SetBusinessUnit

NOTE: The SetBusinessUnit subdialog is supported only in a multitenant environment.

The SetBusinessUnit subdialog sets the business unit for the specified call.

NOTE: The SetBusinessUnit subdialog can be called only once for each IVR call.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_SetBusinessUnit"
srcexpr="GetLink('SetBusinessUnit.vxml')">
      <param name="CallID" expr="Call_ID"/>
      <param name="BusinessUnitName" expr="m_BusinessUnitName"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_SetBusinessUnit.ErrorCode"/>
   </filled>
   </subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call that you want to set the business unit for.
BusinessUnit Name	Input	32 characters	The name of the business unit for the specified call.

Table 15 Parameters for the SetBusinessUnit subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–313 The SetBusinessUnit function was already called for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)

–908 The T-Server is not available.

–914 The VXML operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–955 The multitenancy feature is not licensed.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

The following example sets the name of the business unit for the specified call.

```
<form id="hppcSetBusinessUnit">
   <subdialog name="HPPC_SetBusinessUnit"
   srcexpr="GetLink('SetBusinessUnit.vxml')">
      <param name="CallID" expr="Call_ID"/>
      <param name="BusinessUnitName" expr="m_BusinessUnitName"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_SetBusinessUnit.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt cond="PlayWarnings==1">Set Business Unit
      succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: Set Business Unit <value expr =
      "m_BusinessUnitName"/> succeed.</log>
      <prompt>Your call is assigned to business unit <value expr
      = "m_BusinessUnitName"/>.</prompt>
   <else/>
      <prompt>An error has occurred in Set Business Unit
      subdialog. The error number is <value expr="errCode"/>. </
      prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. Set Business Unit failed. Error
      ID is <value expr="errCode"/>.</log>
```

```
</if>
<goto next="#hppcMainMenu"/>
</block>
</form>
```

4.16 SetContactData

The SetContactData subdialog sets the contact data for a specified CallID and adds the key-value pair if it does not exist. If you want the contact data to be available for the call, you must set the contact data before enqueuing the call.

NOTE: You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_SetContactData"
srcexpr="GetLink(&apos;SetContactData.vxml&apos;)">
   <!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
   <param name="Key" expr="&apos;Name&apos;"/>
   <param name="Value" expr="&apos;John Doe&apos;"/>
<filled>
   <assign name="errCode" expr="HPPC_SetContactData.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to set the contact data for.
Key	Input	32 characters	The key name for the contact data value.
Value	Input	128 characters	The value to be set.

Table 16 Parameters for the SetContactData subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–908 The T-Server is not available. The Routing Server attempted to contact the T-Server but failed for an unknown reason.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

–951 The Routing Server is not available.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcSetContactData">
   <subdialog name="HPPC_SetContactData"
   srcexpr="GetLink(&apos;SetContactData.vxml&apos;)">
      <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
      <param name="Key" expr="&apos;Name&apos;"/>
      <param name="Value" expr="&apos;John Doe&apos;"/>
   <filled>
      <assign name="errCode"
      expr="HPPC_SetContactData.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>SetContactData succeed.</prompt>
```

```
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: SetContactData succeed.</log>

      <goto next="#hppcEnqueue"/>

   <else/>

      <prompt>SetContactData failed. Error ID is <value
      expr="errCode"/>. </prompt>

      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. SetContactData failed. Error ID
      is <value expr="errCode"/>.</log>

      <goto next="#hppcTerminate"/>

   </if>

   </block>

</form>
```

4.17 SetDisplay

The SetDisplay subdialog sets the telephone display for the first answering user of the specified call.

NOTE: This subdialog is only available on the OpenScape 4000 or HiPath 4000 communication platform. You must call the Initialize subdialog before invoking this subdialog. For more information, see Section 4.8, "Initialize", on page 40.

Syntax

```
<subdialog name="HPPC_SetDisplay"
srcexpr="GetLink('SetDisplay.vxml')">

   <param name="CallID" expr="Call_ID"/>

   <param name="DisplayInfo" expr="'Hello this is John'"/>

<filled>

   <assign name="errCode" expr="HPPC_SetDisplay.ErrorCode"/>

</filled>

</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID of the call for which you want to set the display.
DisplayInfo	Input	240 characters	The string that is displayed on the answering user device.

Table 17 Parameters for the SetDisplay subdialog

Error Codes

The most common codes returned by this subdialog are as follows:

0	Successful.
–306	One or more of the parameters is either incorrect or is of the wrong type.
–308	The CallID specified was not found. You must call Initialize to obtain the CallID.
–908	The T-Server is not available.
–914	The operation you are attempting has failed.
–950	VoiceXML is not currently enabled.
Other	Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcSetDisplay">
   <subdialog name="HPPC_SetDisplay"
   srcexpr="GetLink('SetDisplay.vxml')">
      <param name="CallID" expr="Call_ID"/>
      <param name="DisplayInfo" expr="'Hello this is John'"/>
   <filled>
      <assign name="errCode" expr="HPPC_SetDisplay.ErrorCode"/>
   </filled>
</subdialog>
   <block>
   <if cond="errCode == 0">
      <prompt>SetDisplay succeed </prompt>
      log>HPPC Call ID is <value expr="Call_ID"/>: SetDisplay
      succeed.</log>
      goto next="#hppcGetTransitNumber"/>
   <else/>
      <prompt>SetDisplay failed. Error ID is <value
      expr="errCode"/>. </prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. SetDisplay failed. Error ID is
      <value  expr="errCode"/>.</log>
      <goto next="#hppcTerminate"/>
   </if>
   </block>
</form>
```

4.18 Terminate

The Terminate subdialog performs termination and clean up. It must be called when the IVR system has finished using the OpenScape Contact Center VoiceXML functionality.

NOTE: If the GetTransitNumber subdialog was called, you must call the ReleaseTransitNumber subdialog before invoking this subdialog.

Syntax

```
<subdialog name="HPPC_Terminate"
srcexpr="GetLink(&apos;Terminate.vxml&apos;)">
<!-- call id of current call. Mandatory-->
   <param name="CallID" expr="Call_ID"/>
<filled>
   <assign name="errCode" expr="HPPC_Terminate.ErrorCode"/>
</filled>
</subdialog>
```

Parameters

Name	Type	Range	Description
CallID	Input	18 characters	The CallID for the call you want to terminate.
TermType	Input	0, 1, or 2	0 – Indicates the call was terminated as a result of an internal issue. 1 – Indicates the call was transferred out of scope. 2 – Indicates the call was abandoned.

Table 18 Parameters for the Terminate subdialog

Error codes

The most common codes returned by this subdialog are as follows:

0 Successful.

–306 One or more of the parameters is either incorrect or is of the wrong type.

–308 The CallID specified was not found. You must call Initialize to obtain the CallID.

–400 The system has low system resources.

–914 The operation you are attempting has failed.

–950 VoiceXML is not currently enabled.

Other Any other code indicates failure. For more information about a specific code, see Chapter 5, "Error codes".

Example

```
<form id="hppcTerminate">
   <subdialog name="HPPC_Terminate"
   srcexpr="GetLink(&apos;Terminate.vxml&apos;)">
      <!-- call id of current call. Mandatory-->
      <param name="CallID" expr="Call_ID"/>
   <filled>
      <assign name="errCode" expr="HPPC_Terminate.ErrorCode"/>
   </filled>
   </subdialog>
   <block>
      <if cond="errCode == 0">
      <prompt>Terminate succeed.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: Terminate succeed.</log>
   <else/>
      <prompt>Terminate failed. Error ID is <value
      expr="errCode"/>.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>. Terminate failed. Error ID is
      <value expr="errCode"/>.</log>
   </if>
   </block>
   <block>
      <prompt>Good bye.</prompt>
      <log>HPPC IVRID=<value expr="ivrCallID"/>, HPPC Call ID is
      <value expr="Call_ID"/>: IVR Call is finished</log>
   </block>
```

- `</form>`

5 Error codes

This chapter describes all of the codes returned by the VoiceXML subdialogs. For more information on how these codes relate to the specific subdialogs, see Chapter 4, "Using the OpenScape Contact Center VoiceXML subdialogs".

Code	Description
0	Successful. QueryRoutingInfo — Indicates that the call must be enqueued with the returned information.
1	QueryCallStatus — Indicates that the call is in Queued state. In this case, continue checking the call status. QueryRoutingInfo — Indicates that the call must be disconnected.
2	QueryCallStatus — Indicates that the call is in Pending state. In this case, transfer the call to the transit number. QueryRoutingInfo — Indicates that the call must be transferred to the returned target.
3	QueryCallStatus — Indicates that the call is in Unanswered state. In this case, transfer the call to the transit number or to another time-out extension.
4	QueryCallStatus — Indicates that an error has occurred. In this case, transfer the call to a non-OpenScape Contact Center extension.
5	QueryCallStatus — Indicates that the call must be disconnected.
6	QueryCallStatus — The call must be transferred to the OpenScape Contact Center transit number.
0009	The request could not be completed due to an internal error occurring within a Web component servlet.
1003	The request could not be completed due to a socket communication error between the IIS server and the Web Interaction Server.
–304	Initialize was not called before invoking this function. You should call Initialize at the start of your script.
–305	This code indicates an unknown error (for example, user error or the system is unstable).
–306	One or more of the parameters is either incorrect or is of the wrong type.
–307	The key name for the contact data value was not set prior to invoking this function.
–308	The CallID specified was not found. You must call Initialize to obtain the CallID.
–311	The function does not support unmonitored IVR calls.

Table 19 Error codes

Code	Description
–312	Wrong sequence of operations (for example, SetCallInfo called after QueryCallInfo).
–313	The SetBusinessUnit function was already called for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)
–314	The specified business unit name was not set or it did not match the name of any business unit in the database. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)
–400	The system has low system resources.
–701	The request timed out before the function could be completed. It may be an indication of network problems or a busy server.
–801	Unable to connect to one of the OpenScape Contact Center servers. Either the server name is invalid or the server is not operational.
–901	There was no status information available. The Administration Server is not operational but the Routing Server is still available. In this case, the status of the other servers is unknown. This error is returned only by the QuerySystemStatus function.
–903	There was no call associated with the specified CallID. Ensure that all IVR extensions are represented in the database.
–904	The specified agent ID did not match the ID of any user in the database.
–905	The specified queue did not match the name of any queue in the database.
–907	The T-Server is not available. This error means the Routing Server could not enqueue the call because the T-Server was not available.
–**908**	The T-Server is not available.
–**909**	The specified device did not match any device in the database.
–**911**	The function contained data that became corrupted for an unknown reason. The function was therefore unable to connect with the Routing Server. The function was unsuccessful.
–913	The function passed a “To Device” that was not valid.
–914	The operation you are attempting has failed.
–915	Unable to determine the queue and other routing information.
–916	The Routing Server encountered an error in the flow execution.
–917	The telephone number cannot be translated correctly by TAPI.
–927	All registered transit numbers are busy.
–928	No transit number has been allocated for this CallID.
–929	No transit numbers have been configured.
–950	VoiceXML is not currently enabled.
–951	The Routing Server is not available.

Table 19 Error codes

Code	Description
–953	The business unit has not been set for this call. (This error is applicable only when OpenScape Contact Center is configured as a multitenant system.)
–955	The multitenancy feature is not licensed.
–1006	You are attempting to create a duplicate callback.
–1021	Invalid CallbackID.
–1028	A schedule time is invalid.
–1029	The callback schedules have no mutual time with the contact center open hours.
–1031	A telephone number in the request is in the excluded numbers list.
–1040	All schedules have already expired.
–1045	The callback queue name is invalid.
–1047	The schedule contains a date that is too far in the future.
53006	The Web Interaction Server cannot connect with the T-Server.
53007	The Web Interaction Server cannot connect with the Routing Server.
53012	There is not enough internal memory to complete the requested action. Contact your support representative.
53015	The Web Interaction Server has lost connectivity with the corporate Web server. Contact your support representative.
53018	The Web Interaction Server has lost connectivity with the corporate Web server. Contact your support representative.
53028	The Web Interaction Server could not complete the requested action. Contact your support representative.
53503	The Web Interaction Server could not complete the requested action. Contact your support representative.
53504	The Web Interaction Server is not compatible with the current version of the application. Contact your support representative.
53508	The Web Interaction Server does not support the requested action.
53510	VoiceXML is not currently enabled.

Table 19 Error codes

Code	Description
Other	Any other negative code indicates failure. If another code is returned: ● Ensure that all site, queue, and user names (or ID numbers) in the IVR script match the names (or ID numbers) in the production copy of the OpenScape Contact Center database. ● Ensure that all communication platform resources are properly entered into the production copy of the OpenScape Contact Center database. ● Contact your next level of support.

Table 19 Error codes

Index