IVR Driver for VoiceGenie 7.2

User Reference Guide

About Genesys

Genesys Telecommunications Laboratories, Inc., a subsidiary of Alcatel-Lucent, is 100% focused on software for call centers. Genesys recognizes that better interactions drive better business and build company reputations. Customer service solutions from Genesys deliver on this promise for Global 2000 enterprises, government organizations, and telecommunications service providers across 80 countries, directing more than 100 million customer interactions every day. Sophisticated routing and reporting across voice, e-mail, and Web channels ensure that customers are quickly connected to the best available resource—the first time. Genesys offers solutions for customer service, help desks, order desks, collections, outbound telesales and service, and workforce management. Visit www.genesyslab.com for more information.

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

Notice

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

Your Responsibility for Your System's Security

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

Trademarks

Genesys, the Genesys logo, and T-Server are registered trademarks of Genesys Telecommunications Laboratories, Inc. All other trademarks and trade names referred to in this document are the property of other companies. The Crystal monospace font is used by permission of Software Renovation Corporation, www.SoftwareRenovation.com.

Technical Support from VARs

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

Technical Support from Genesys

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

Region	Telephone	E-Mail
North and Latin America	+888-369-5555 or +506-674-6767	support@genesyslab.com
Europe, Middle East, and Africa	+44-(0)-1276-45-7002	support@genesyslab.co.uk
Asia Pacific	+61-7-3368-6868	support@genesyslab.com.au
Japan	+81-3-6361-8950	support@genesyslab.co.jp

Prior to contacting technical support, please refer to the Genesys Technical Support Guide for complete contact information and procedures.

Ordering and Licensing Information

Complete information on ordering and licensing Genesys products can be found in the Genesys 7 Licensing Guide.

Released by

Genesys Telecommunications Laboratories, Inc. www.genesyslab.com

Document Version: 72ivd_vg_urg_v7.2.200.01 04-2008

Table of Contents

Chapter

1 Preface

Welcome to the *IVR Driver for VoiceGenie 7.2 User Reference Guide*. This book describes the IVR Driver for VoiceGenie, which is the component designed to integration the VoiceGenie platform with Genesys Customer Interaction Management (CIM).

This document is valid only for the 7.2 release of this product.

This chapter includes the following sections:

Interactive voice response (IVR) technology has emerged as an integral part of contact centers, financial institutions, and the travel industry. IVR components are used to provide the initial interface when a client calls a business. Using IVRs, businesses can realize significant savings and customers can conduct their business more efficiently.

The IVR Driver for VoiceGenie architecture simplifies the integration of VoiceGenie VoiceXML applications with the Genesys CIM environment. The IVR Driver for VoiceGenie is built on the Genesys IVR Server XML specification for interaction with Genesys suite of T-Servers, Universal Routing and Agent Desktops. For more information on the interoperability of the IVR Driver for VoiceGenie with other Genesys components, see Architecture.

New in Release 7.2

The following changes have been implemented in release 7.2 of the IVR Driver for VoiceGenie:

- The URL reference to the servlet has been modified to /gcti/ivrdriver_vg. Backwards compatibility with /GenesysMessages/SubdialogHandler has been retained, but is not recommended.
- Improved support for IVR Server messages including treatments, attached data and call control.

Note: To take advantage of the full range of IVR Driver for VoiceGenie 7.2 functions, you must use IVR Server 7.2, Genesys Framework 7.2 and Genesys 7.2 licensing.

Intended Audience

This guide, primarily intended for IVR developers, operations personnel, contact center administrators, and contact center managers, assumes that you have a basic understanding of:

- Computer-telephony integration (CTI) concepts, processes, terminology, and applications.
- VoiceGenie Media Server platform.
- VoiceXML scripting and features.
- Network design and operation.
- Your own network configurations.

You should also be familiar with Genesys Framework architecture and functions.

Chapter Summaries

In addition to this opening chapter, this guide contains these chapters and an appendix:

- Chapter 2, "IVR Driver for VoiceGenie Overview," generally discusses deployment of IVR Driver for VoiceGenie and provides a description and illustrations of the IVR Driver for VoiceGenie architecture.
- Chapter 3, "Pre-Installation Setup for the IVR Driver for VoiceGenie," describes the pre-installation tasks for the IVR Driver for VoiceGenie.
- Chapter 4, "Installing the IVR Driver for VoiceGenie," describes how to install the IVR Driver for VoiceGenie.

- Chapter 5, "Starting and Stopping the IVR Driver for VoiceGenie," describes how to start and stop the IVR Driver for VoiceGenie.
- Chapter 6, "Developers Reference," describes the VoiceXML subdialog interface and message types.
- An Appendix, "VoiceXML Example" provides an example of an VoiceXML application.

Document Conventions

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

Document Version Number

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

72ivd_vg_urg_v7.2.000.00

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

Type Styles

Italic

In this document, italic is used for emphasis, for documents' titles, for definitions of (or first references to) unfamiliar terms, and for mathematical variables.

Examples:

- Please consult the *Genesys 7 Migration Guide* for more information.
- *A customary and usual practice* is one that is widely accepted and used within a particular industry or profession.
- Do *not* use this value for this option.
- The formula, $x + 1 = 7$ where x stands for . . .

Monospace Font

A monospace font, which looks like teletype or typewriter text, is used for all programming identifiers and GUI elements.

This convention includes the *names* of directories, files, directorys, configuration objects, paths, scripts, dialog boxes, options, text and list boxes, operational modes, all buttons (including radio buttons), check boxes, commands, tabs, CTI events, and error messages; the values of options; logical arguments and command syntax; and code samples.

Examples:

- Select the Show variables on screen check box.
- Click the Summation button.
- In the Properties dialog box, enter the value for the host server in your environment.
- In the Operand text box, enter your formula.
- Click OK to exit the Properties dialog box.
- The following table presents the complete set of error messages T-Server® distributes in EventError events.
- If you select true for the inbound-bsns-calls option, all established inbound calls on a local agent are considered business calls.

Monospace is also used for any text that users must manually enter during a configuration or installation procedure, or on a command line:

Example:

- Enter exit on the command line.

Screen Captures Used in This Document

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

Square Brackets

Square brackets indicate that a particular parameter or value is optional within a logical argument, a command, or programming syntax. That is, the parameter's or value's presence is not required to resolve the argument, command, or block of code. The user decides whether to include this optional information. Here is a sample:

```
smcp_server -host [/flags]
```

Angle Brackets

Angle brackets indicate a placeholder for a value that the user must specify. This might be a DN or port number specific to your enterprise. Here is a sample:

smcp_server -host <confighost>

Related Resources

Consult these additional resources as necessary:

- The *Genesys Technical Publications Glossary*, which ships on the Genesys Documentation Library CD, and which provides a comprehensive list of the Genesys and CTI terminology and acronyms used in this document.
- The Release Notes and Product Advisories for this product, which are available through your Account or Professional Services representative.

Information on supported hardware and third-party software is available on the Genesys Technical Support website in the following documents:

- *Genesys 7 Supported Operating Systems and Databases*
- *Genesys 7 Supported Media Interfaces*

Genesys product documentation is available on the:

- Genesys Technical Support website at http://genesyslab.com/support.
- Genesys Documentation Library CD, which you can order by e-mail from Genesys Order Management at mailto:orderman@genesyslab.com.

Making Comments on This Document

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

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

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

Chapter

2 IVR Driver for VoiceGenie Overview

This chapter describes the architecture of the IVR Driver for VoiceGenie and how the IVR Driver for VoiceGenie is used in this solution. The chapter also discusses deployment tasks and tips. It includes the following sections:

Deployment

This *User Reference Guide* describes how to install the IVR Driver for VoiceGenie and configure it.

For the IVR Driver for VoiceGenie to run successfully, the IVR Server must be installed and running. For information about installing and configuring the IVR Server, see the *IVR Interface Option 7.2 IVR Server System Administrator's Guide*.

The IVR Driver for VoiceGenie operates within an Apache Tomcat® application server as a *servlet*. Genesys recommends that you install Apache Tomcat with the IVR Driver for VoiceGenie servlet on each VoiceGenie Media Server platform.

For information about integrating a VoiceGenie application with the IVR Server, see "Developers Reference." For an example of voice application programming, see the "VoiceXML Example."

Architecture

This section describes the architecture of the IVR Driver for VoiceGenie. This software application integrates VoiceGenie software and hardware with the Genesys Framework. The IVR Driver for VoiceGenie architecture is described in this section.

IVR Server

IVR Server, a key component of Genesys Framework, provides the following functionality:

- Tracks call flow
- Interfaces multiple drivers with multiple T-Servers
- Works with other Genesys servers (such as T-Server, Stat Server, and Universal Routing Server)
- Can be used in load-sharing or warm standby mode

Genesys provides these configuration modes for the IVR Server:

- IVR-Behind-Switch is a basic configuration in which a T-Server connected to the premise switch (using CTI links) can monitor the call activity on IVR channels. See IVR-Behind-Switch Configuration for more information.
- In the IVR-In-Front configuration, a Computer Telephony Integration (CTI) link is not involved with the call processing. See IVR-In-Front Configuration for more information.
- In the IVR Network T-Server configuration, the IVR Server is a link to a user-provided Network IVR application. The Service Control Point (SCP) and a Network T-Server are used to redirect calls to the Network IVR for processing. In this mode, IVR Server functions as a Network T-Server.

Library Usage

IVR Server is built with 7.*x* T-Library and can thereby connect to T-Servers. IVR Server supports connection to a regular T-Server, the TServer_IVR function of an IVR Server operating in IVR-In-Front mode, and a Network T-Server.

IVR-Behind-Switch Configuration

In the IVR-Behind-Switch configuration, an incoming call arrives at the premise switch before going to the vendor-provided IVR (see Figure 1). The premise switch and T-Server are at the same site.

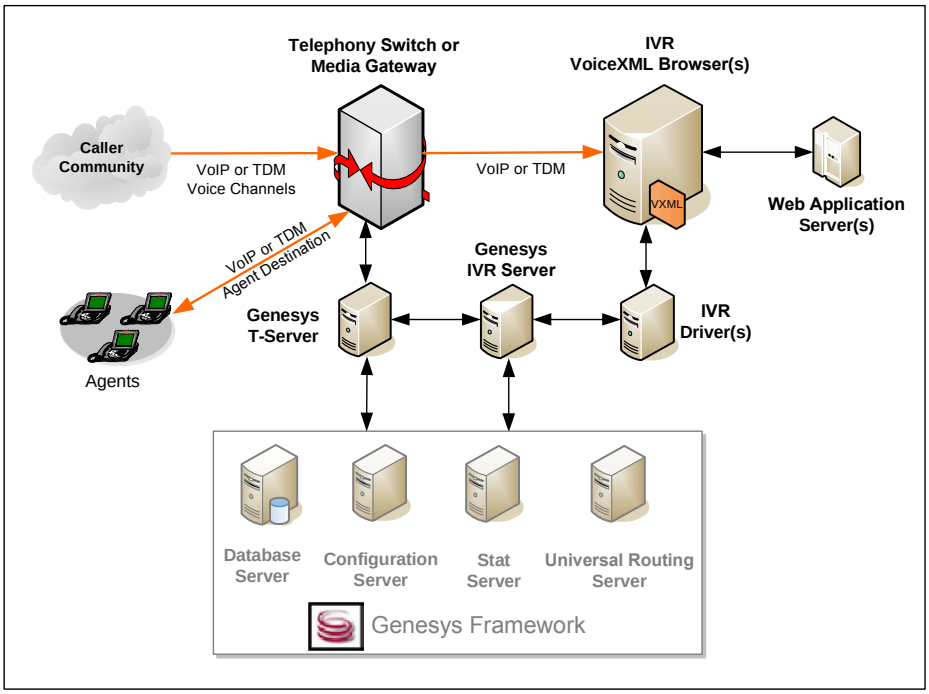


Figure 1: IVR-Behind-Switch Configuration

In this configuration, a physical T-Server is connected to a premise switch, and the IVR is connected directly to both the switch (through phone lines) and the IVR Server (through data lines). The IVR Server communicates with T-Server and Stat Server.

IVR-In-Front Configuration

If a vendor-provided IVR is connected directly to the PSTN (Public Switched Telephone Network), without a premise switch, the configuration is called IVR-In-Front. In the Site A configuration shown in Figure 2, there is no T-Server to connect to, because there is no premise switch. The TServer_IVR function resides within the IVR Server.

An IVR Server operating in IVR-In-Front mode supports IVRs that are connected directly to a PSTN, by performing functions similar to those of a regular T-Server. If an IVR is considered a termination point for incoming calls, no premise switch is involved, and no local T-Server receives notification of the incoming call. Instead, the IVR Server provides this functionality.

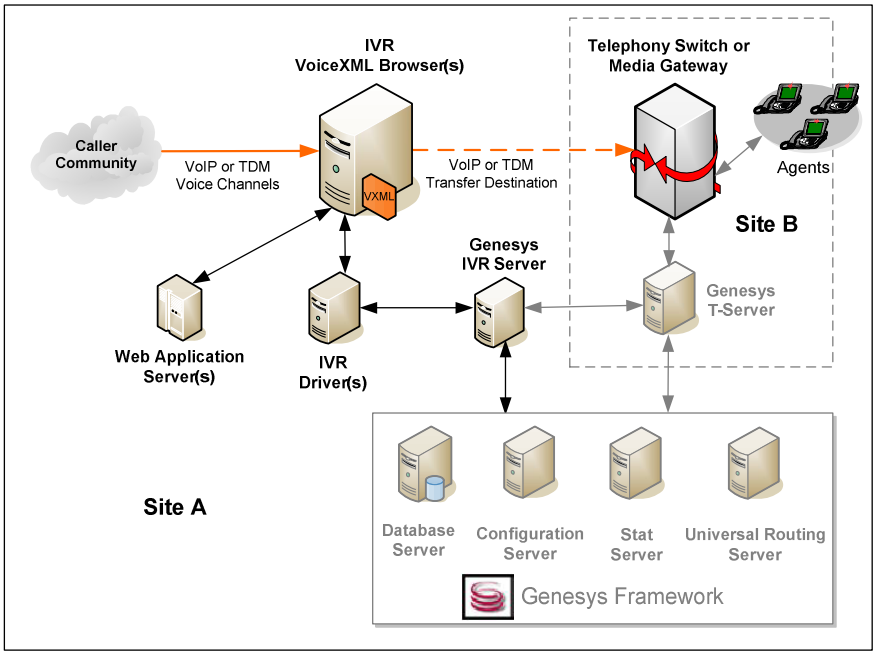


Figure 2: IVR-In-Front Configuration

In the IVR-In-Front configuration shown in Figure 2, Site A is configured for IVR-In-Front mode. The IVR Server communicates with the IVR, the Universal Routing Server, and Stat Server. The IVR Server also simulates a T-Server, which enables it to communicate with other T-Servers, such as the T-Server at Site B. The IVR at Site A is physically connected to the public telephony network for phone lines, and to the IVR Server for data lines.

Site B includes a physical switch connected to a T-Server, which, in turn, provides data to agents in an agent pool.

This distributed configuration across Sites A and B enables coordinated Call Data Transfers.

IVR Driver for VoiceGenie

The IVR Driver for VoiceGenie component integrates the VoiceGenie Media Server platform with the Genesys environment. This adds a set of operations to the VoiceGenie VoiceXML application. All interactions between the VoiceXML application and Genesys Framework are based on the request-response architecture of the IVR Library, and use a TCP/IP connection.

The major functions provided by the IVR Driver for VoiceGenie include:

- Telephony function support (such as transfer, conference, answer, and release).
- Call data manipulation (such as attach, update, and delete).
- Treatment handling for agent hold and parking, and similar operations.

Each VoiceGenie Media Server platform requires the IVR Driver for VoiceGenie to operate in the Genesys environment.

IVR Server Connection Instances

The IVR Driver for VoiceGenie can support up to six IVR Server connection instances. The relationship with the IVR Server is based on the port number or *channel*. Each of these connection instances is designed to handle the following actions:

- Opens a socket connection to the IVR Server at a predefined port (see Defining IVRs and IVR Ports). The socket connection remains open to support multiple requests from the VoiceXML application.

  Note: You can open more than one port connection by sending additional loginrequest messages.

- Sends a reply to Keep-Alive messages to and from the IVR Server. Sends Keep-Alive messages every 5 seconds and waits for a reply. If a response is not received within a specified amount of time, the connection will attempt 5 retries before closing the socket, and then reopening it. Sockets have a configurable timeout value, so that the IVR port is not perpetually blocked.
- Checks active calls periodically, and if an active call is not found, it sends an endcall message to the IVR Server.

Managing Connection Failures

The IVR Driver for VoiceGenie performs the following connection management when failure conditions occur.

Note: Unless otherwise noted, the configuration properties stated below are located in the genesysmessage.properties file.

- If a connection or login error occurs, the IVR Driver for VoiceGenie closes and attempts to reopen the socket.
- If the value of the configuration property maxcontinuousIOErrors is exceeded, the IVR Driver for VoiceGenie closes and attempts to reopen the socket after the amount of time specified in the configuration property iorestartperiod.
- When a call comes into a channel, it uses the socket defined in the channelmapping.properties file (see Modifying Configuration Properties). This file can have more than one socket ID delimited by commas. Each socket ID is selected by order if failover is set to loadbalance or according to the defined primary and backup if failover is set to primary-backup.

- If the value of the configuration property failover is set to loadbalance, and a socket connection cannot be established, the channel skips it and the next socket ID is selected. If a previously-failed socket connection is re-established, it is included in the selection order of socket IDs.
- If failover is set to primary-backup, when the primary server is down, the channel uses the backup server. Each new connection will attempt to use the primary server before failing over to the backup server.

Note: The IVR Driver for VoiceGenie performs periodic call clean-up that includes sending an endcall message for any inactive call that did not complete successfully. If a new call arrives on the same port that did not complete with an endcall, the IVR Driver for VoiceGenie will send an endcall prior to the newcall.

Chapter

3 Pre-Installation Setup for the IVR Driver for VoiceGenie

Before you can successfully install the IVR Driver for VoiceGenie, you must perform the steps described in this chapter. This chapter includes the following sections:

Configuring Genesys Framework

Before you configure the IVR Driver for VoiceGenie, you must complete the tasks described in this section.

Component Compatibility

Important: Before you can configure Configuration Manager objects and install IVR Interface Option 7.2, you must install a supported release of Genesys Framework. For IVR-Behind-Switch configurations, you must also install a compatible release of Genesys T-Server. For more information, see the *Framework 7.2 Deployment Guide*.

Installing the IVR Server

The IVR Driver for VoiceGenie is intended to be used with the IVR Server. You must install the IVR Server and configure it in Genesys Configuration Manager before you install the IVR Driver for VoiceGenie.

You can install the IVR Server on any computer belonging to the site where the IVR Driver for VoiceGenie product is used. However, for performance and reliability, Genesys recommends that you install the IVR Server on a different machine than the one running Apache Tomcat and the IVR Driver for VoiceGenie. Also, make sure that the operating system of the host where the IVR Server is installed matches the operating system on which the IVR Server was built.

For help on installing and configuring the Genesys IVR Server, see the *IVR Interface Option 7.2 IVR Server System Administrator's Guide*.

Defining IVRs and IVR Ports

You can use the IVR Interface Option Wizard to define an IVR and to define an entire range of IVR ports at once. For information on how to use the Wizard, see the wizard configuration chapter in the *IVR Interface Option 7.2 IVR Server System Administrator's Guide*.

You can also define an IVR or a single IVR port manually in Configuration Manager. For information on how to configure IVRs and IVR ports manually, see the manual configuration chapter in the *IVR Interface Option 7.2 IVR Server System Administrator's Guide*.

Once you have defined an IVR object in Configuration Manager (using the wizard or manually), you must configure the IVR for use with IVR Driver for VoiceGenie, using the steps described in Configuring the IVR Driver for VoiceGenie.

Configuring the IVR Driver for VoiceGenie

This section describes how to configure the IVR Driver for VoiceGenie.

Deploying an Apache Tomcat Application Server

Before you can install the IVR Driver for VoiceGenie, you must first acquire a Apache Tomcat Application Server and deploy it in your environment. For more information, see "Installing the IVR Driver for VoiceGenie."

The IVR Driver for VoiceGenie should be deployed to Apache Tomcat as a web archive (WAR) container. You can decide whether to autodeploy or manually deploy the WAR container. Please change the **maxProcessors** parameter to 100 in {TOMCAT_HOME}\conf\server.xml file for 8080 connector. For more information on Apache Tomcat, see the Apache Tomcat documentation and http://tomcat.apache.org/.

Modifying Configuration Properties

The IVR Driver for VoiceGenie provides several configuration property files located in the [tomcat_home]/webapps/gcti/WEB-INF/classes/config directory. You must modify the content of these property files before using the IVR Driver for VoiceGenie. Properties are stored in the following three property files, which you can edit using a text editor:

- channelcallednum.properties—Maps the channel (callidref) to the calledNum value, except when using SIP (see note below):
  XB01T01=1025
  XB01T02=1026
  XB01T03=1027
  ..
  XB01T24=1048
- channelmapping.properties—Maps the channel (callidref) to the socket ID, except when using SIP (see note below). The socket ID ranges from 0 to 5 inclusive (depending on the number of predefined destination IP addresses and port numbers) in the following format:
  XB01T01=0,1,2
  XB01T02=0,1,2
  XB01T03=0,1,2
  ..
  XB01T23=0,1,2

  Note: For Session Initiation Protocol (SIP), you must use the DNIS identifier instead of the callidref variable for the channelcallednum.properties and channelmapping.properties files—for example, 9090=1025 and 9090=0,1,2.

- genesysmessage.properties—Contains the majority of configuration files. Table 1 lists, describes, and provides examples for each of these files.

Table 1: genesysmessage.properties Configuration Files

Property	Description	Example
numsockets	The number of socket handler IDs (6 maximum).	numsockets=2
socket[x].port, socket[x].host, socket[x].clientname, socket[x].shouldLogin	The port of the IVR Server, the host where the IVR Server is running, and the IVR client name as defined in Configuration Manager (for login purposes) for each socket handler. [*x*] should be replaced	socket0.port=5100 socket0.host=TSERVER1 socket0.clientname= VG_ISERVER1 socket0.shouldLogin=true socket1.port=5101 socket1.host=TSERVER1

Property	Description	Example
	by a number from 0 to 5 inclusive, depending on the number of connections that you want to define. **Note:** shouldLogin must always be set to true.	socket1.clientname= VG_ISERVER2
maxcontinuousIOErrors	If more than this number of continuous connection errors occur, the servlet waits the amount of time set in iorestartperiod before it attempts to reconnect to the IVR Server.	maxcontinuousIOErrors=2
iorestartperiod	If the maxcontinuousIOErrors limit is reached, the servlet waits for this period of time (in milliseconds) before it attempts to reconnect to the IVR Server.	iorestartperiod=30000
defaultmessagetimeout	The period of time to wait for a response message before returning status = F.	defaultmessagetimeout=5000
defaultlogintimeout	The period of time to wait for a login reply before the socket is marked as unusable.	defaultlogintimeout=60000
reportstatus	Indicates that the login response message (loginresponse) for this request should include its status property. This property must always be set to true.	reportstatus=true
dtdurl	The URL of the DTD to be used in the message. This should be Iserver.dtd.	dtdurl=IServer.dtd
fulldtdurl	The URL of the DTD that validates against the retrieved response messages.	fulldtdurl=http://localhost: 8080/gcti/dtd/ IServer.dtd

Property	Description	Example
keepaliveperiod	The amount of time (in milliseconds) that the connected socket sends Keep-Alive requests to the IVR Server.	keepaliveperiod=5000
serverdead	The number of attempts for Keep-Alive sent by the socket without a response before reopening a new connection.	serverdead=5
loglevel	The logging detail 1 – 6 to write to the log directory.	loglevel=3
logmethod	Must be set to either 1 (to write each line) or 2 (to append the log at the end of the call).	logmethod=1
systembuffer	The number of lines in the system logs to be buffered in memory. When the number of lines in the buffer reaches this value, the logs are written to the log file.	systembuffer=1
logdirectory	The directory where the log files are stored. This is relative to the IVR Driver for VoiceGenie directory.	logdirectory=logs
sessionmonitorcommand	The session monitor command in Command Line Console (CLC) to get the list of active sessions (sscm for CLC 6.2, snapshot callmgr for CLC 6.4).	sessionmonitorcommand= snapshot callmgr
sessionmonitorport	The CLC port (by default, 8999).	sessionmonitorport=8999
sessionmonitorhost	The host name of the VoiceGenie Media Server platform. This must be set to localhost.	sessionmonitorhost=localhost

Property	Description	Example
sessionmonitorperiod	The period of time (in milliseconds) until the next monitoring session occurs.	sessionmonitorperiod=300000
sessionmonitor	Enables session monitoring.	sessionmonitor=false
maxcalllength	The amount of time before the session monitor considers the call to be in an undefined state and cleans up this session.	maxcalllength=180000
failover	The failover method to be used. It can be either primary-backup or loadbalance. The default setting is loadbalance.	failover=loadbalance
sysout	If set to true, logging messages are sent to the Apache Tomcat log mechanism; otherwise, they are sent to the system console. This is typically set to false.	sysout=false
callednumbermapping	Overwrites the channelcallednumber. properties file. This parameter is used along with the callednumberlist parameter. **Note:** This is only used when set to dynamic. Otherwise, it is static.	callednumbermapping= dynamic
callednumberlist	This is the list of ports that will be assigned in round-robin mode to the new calls. This parameter is used along with the callednumbermapping parameter.	callednumberlist=101,102, 103,104,105,106

Managing Channel Identifiers

Within this document the terminology of channel ID's and CalledNum are synonymous, they are used to represent the same concept of identifying the channel that the call arrived for port synchronization with the Genesys CTI framework.

When operating as IVR In-Front, the call is received by the IVR before Genesys CTI is aware of the call existence. It is the responsibility of the IVR to select a channel and have it named consistently with the definition of IVR Ports as defined in the Genesys Framework Configuration Management Environment (CME). To ensure name consistency, it may be necessary to use ECMA code in the VoiceXML to format a channel of call as it arrived in the IVR. Examples are ports defined as VXMLI IP Address + (1 through 100 formatted as three digit numbers). In this case it is necessary to extract the IP Address and format the arriving channel into a three digit number before using in the NewCall message interaction.

When operating as IVR Behind, the call is already known by the T-Server/I-Server before it is directed to the IVR where a port is pre-selected. In this case, the IVR Driver for VoiceGenie provides that port DN as the CalledNum for reporting and tracking purposes.

Static and Dynamic Channel Identification

Generally the IVR Driver for VoiceGenie is configured with static channel mapping using the properties files and the genesysmessages.properties property callednumbermapping is set to `static` (default). This setting respects the message parameter channeled to contain the channel identified that the call appeared on.

When implementing in a hosted or leveraged IVR configuration, the number of ports managed for CTI interaction may be less than all the IVR ports that the voice browsers are physically supporting. In this architecture the physical port may need to be decoupled from the Genesys CTI CME port as many more ports would need to be defined in CME than would be utilized.

To decouple the channel and port definition, a dynamic channel allocation method is provided with the IVR Driver for VoiceGenie that enables the channel to be selected from a range of possible entries.

To use the dynamic channel allocation capability, set the callednumbermapping property to `dynamic`. The channels that can then be selected is a comma separated list in the property callednumberlist. Ensure there are sufficient channels listed to meet the port specification and that channels defined do not clash with other instances of the IVR Driver for VoiceGenie.

Chapter

4 Installing the IVR Driver for VoiceGenie

This chapter describes how to install the IVR Driver for VoiceGenie and includes the following information:

Before you install the IVR Driver for VoiceGenie, you should complete the tasks described in "Pre-Installation Setup for the IVR Driver for VoiceGenie."

Identifying the IVR Driver for VoiceGenie Version

To identify the version of the IVR Driver for VoiceGenie:

- Do one of the following:
  - If the IVR Driver for VoiceGenie is already installed, open a web browser to http://[Media Server host name]:8080/gcti/version.html
  - If the IVR Driver for VoiceGenie is not installed, navigate to the IVR Driver for VoiceGenie .war file, and open the version.html file, which lists the IVR Driver for VoiceGenie version.

Supported IVR Versions

The IVR Driver for VoiceGenie and this *User Reference Guide* support only the following VoiceGenie Media Server platform versions and operating systems:

- VoiceGenie Media Server platform on Microsoft Windows Server 2000 with Java 1.4.1
- VoiceGenie Media Server platform on Microsoft Windows Server 2003 with Java 1.4.1
- VoiceGenie Media Server platform on Red Hat Linux 3.0 with Java 1.4.1

Installing the IVR Driver for VoiceGenie

The IVR Driver for VoiceGenie runs as a servlet in Apache Tomcat. To install the IVR Driver for VoiceGenie:

1. Install Apache Tomcat as described in the Apache Tomcat documentation and http://tomcat.apache.org/, and configure it to run as either a Windows service or a Linux process.
2. If possible, configure Apache Tomcat to listen on port 8080. If 8080 cannot be selected, record the specified port and replace it where 8080 is referenced in this document.
3. Install gcti.war on the Apache Tomcat platform. This file installs the IVR Driver for VoiceGenie on the same computer where Apache Tomcat is installed.

   Notes: When you copy the war file into the [tomcat_home]/webapps/ directory, Apache Tomcat installs the war file automatically, provided the following statement is included in the [tomcat_home]/conf/server.xml file:

   <Host name="localhost" debug="0" appBase="webapps" unpackWARS="true" autoDeploy="true">

   The default Linux [tomcat_home] value is /usr/local/tomcat.

4. Modify the configuration files (see Modifying Configuration Properties).

Configuring the Apache Tomcat JVM Heap

The IVR Driver for VoiceGenie requires more JVM *heap* (an internal memory pool that tasks use to dynamically allocate memory as needed) than is specified in the Tomcat default memory setting. The procedure to adjust the heap depends on your operating system.

Apache Tomcat on Windows Implementation

To adjust the heap on Windows operating systems running Apache Tomcat 5 and later:

1. Open the Configure Tomcat application from the Start menu.

   Note: You must stop Apache Tomcat before starting this application.

2. Click the Java tab in the Configuration dialog box. A list of predefined options appear in the Java Options text box.
3. Append your Java options to the list—for example, to set the server mode and allow 256 MB of heap memory, append the following line:
   -server -Xmx256m

To adjust the heap on Apache Tomcat 4.1 or earlier:

1. Open Control Panel and double-click System. The System Properties dialog box appears (see Figure 3).

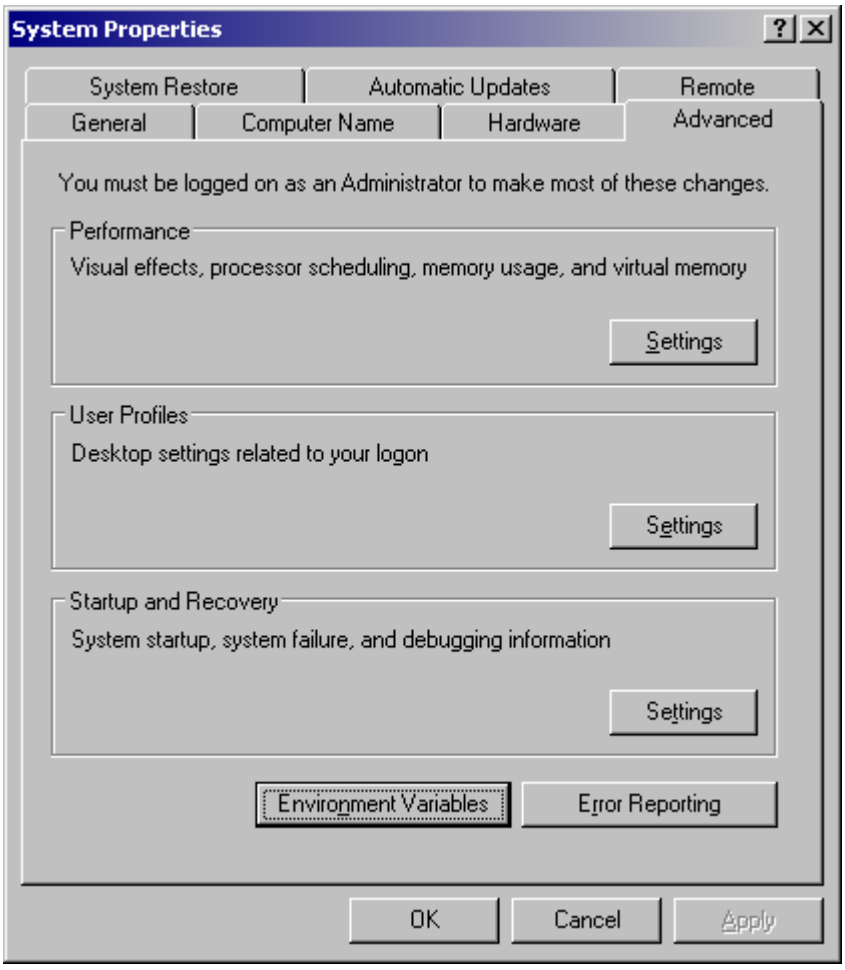


Figure 3: System Properties Dialog Box

2. On the Advanced tab, click Environment Variables. The Environment Variables dialog box appears (see Figure 4).

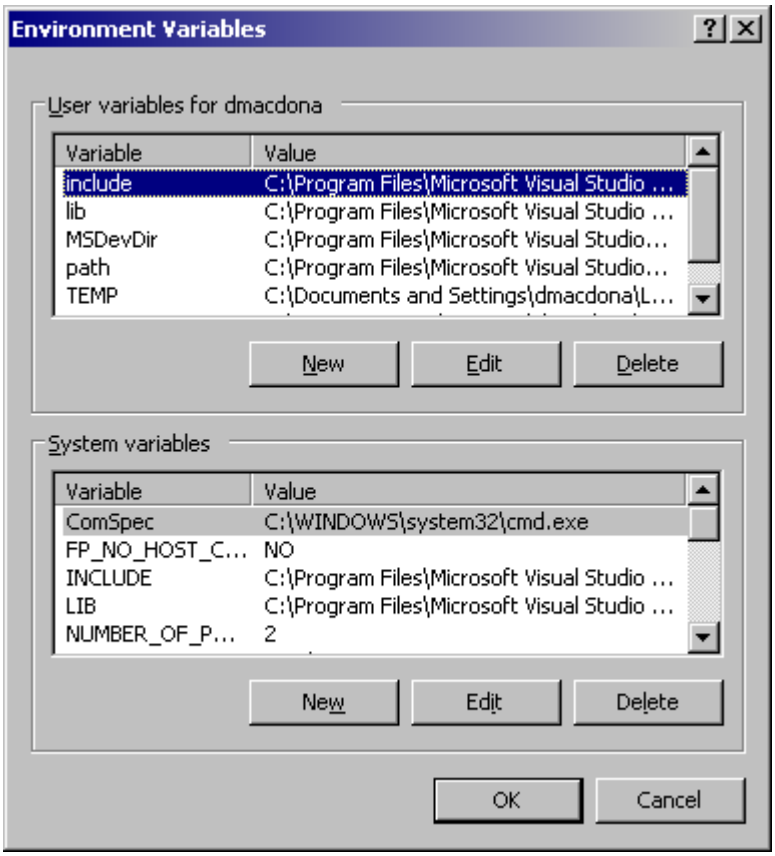


Figure 4: Environment Variables Dialog Box

3. In the System variables section, click New. The New System Variable dialog box appears (see Figure 5).

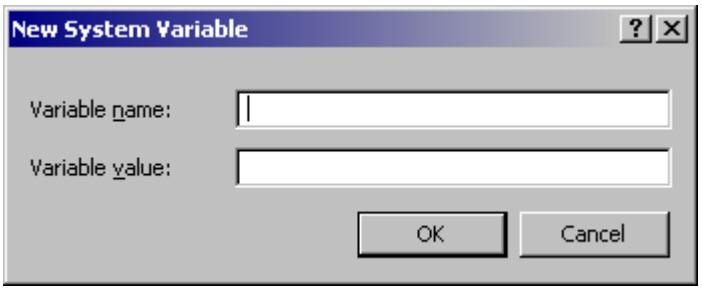


Figure 5: New System Variable Dialog Box

4. In the Variable name box, enter CATALINA_OPTS.
5. In the Variable value box, enter -server -Xmx256m.
6. Click OK to create the CATALINA_OPTS environment variable. You are returned to the Environment Variables dialog box (see Figure 4).
7. Click OK to return to the System Properties dialog box (see Figure 3).
8. Click OK.

Apache Tomcat Linux Implementation

To adjust the heap on Linux operating systems:

1. Open the catalina.sh file, which is located in the usr/local/Tomcat/bin directory.
2. Add the following line:
   export JAVA_OPTS="-Xms256m -Xmx256m"
3. Append the Java options to the CATALINA_OPTS environment variable. For example: -server -Xmx256m

Chapter

5 Starting and Stopping the IVR Driver for VoiceGenie

This chapter describes how to start and stop the IVR Driver for VoiceGenie, which you can do only after you have properly installed and configured:

- IVR Server
- IVR Driver for VoiceGenie

For additional information about the installation and configuration of the IVR Server, see the *IVR Interface Option 7.2 IVR Server System Administrator's Guide*.

After installing and configuring the IVR Server and the IVR Driver for VoiceGenie, you must start Apache Tomcat. To stop the IVR Driver for VoiceGenie, stop Apache Tomcat.

Chapter

6 Developers Reference

The IVR Driver for VoiceGenie provides a VoiceXML subdialog interface that can be invoked within a VoiceXML script. After you install, configure, and start the IVR Driver for VoiceGenie, you can use and test your Genesys integration from within the VoiceGenie Media Server platform VoiceXML applications.

This chapter describes the IVR Driver for VoiceGenie operation and message types that this driver supports. It contains the following sections:

IVR Driver for VoiceGenie Operation

To invoke the operation of the IVR Driver for VoiceGenie, it is necessary to use the subdialog element, which enables VoiceXML to call remote procedures using an object name. Each object name contains a URL reference, list of input parameters, that return properties as members, and a method attribute that must be set to "post". For more information on the subdialog element, refer to the VoiceXML standard text available at http://www.w3.org/TR/voicexml20/.

When invoking the IVR Driver for VoiceGenie, the following VoiceXML script example can be used and customized for each operation requested:

```
<var name="callId" expr="session.com.voicegenie.telephone.primarychan"/>
<var name="channelId" expr="session.connection.callidref"/>
<var name="dnis" expr="session.connection.dnis"/>
<var name="ani" expr="session.connection.ani"/>
<var name="callControlMode" expr="'Network'"/>
<var name="messagetype" expr="'NewCall'"/>
<var name="endCause" expr="'Normal'"/>
```

```
<subdialog name="newcall" src="http://your-host:8080/gcti/iserver"
namelist="callId channelId messagetype callcontrolmode" method="post">
<filled>
    <!-- perform results processing -->
    <if cond="newcall.status=='S'" >
        <!-- success handler -->
        ---
    <else />
        <!-- failure handler -->
    </if>
</filled>
</subdialog>
```

Parameter Constraints

Some functions described in this chapter use key-value pairs. The following constraints apply:

- Characters with a value less than 0x20 are not valid in key names or data values. The only exceptions are the characters 0x09, 0x0A, and 0x0D, which correspond to the ASCII control characters TAB, LINE FEED, and CARRIAGE RETURN. No other ASCII control characters are allowed.
- Keys in key-value pairs cannot include the (.) and (:) characters. An error message is generated if a key includes either of these characters.
- You cannot issue a getdata message from the IVR Driver for VoiceGenie by using a key that contains a colon (although other Genesys software may be able to access this type of key-value pair).
- The combined key-value pairs (including delimiters) on a given call instance can total no more than 16,000 bytes. The length of any individual key or value is limited only by this total.

Input Parameters

Each IVR Driver for VoiceGenie operation contains a namelist of standard parameters, followed by a unique list that is specific to the operation invoked. These names are expected by the internals of the IVR Driver for VoiceGenie and must not be changed. Table 2 lists the standard parameters, and provides a description and example values for each parameter.

Table 2: Standard Input Variables

Name	Description	Example
callid	Uniquely identifies the call between the VoiceXML browser instance and the IVR Driver for VoiceGenie.	session.com.voicegenie.telephone.primarychan

Name	Description	Example
channelid	The channel identifier or port of the call as determined from the IVR.	session.connection.callidref
messagetype	The message name of the operation to invoke.	'newcall'

All other operation-specific parameters conform to their names as stated in Message Types, except where discussed below for uDataEx and extnsEx parameter lists.

uDataEx and extnsEx Input Parameter Lists

There are two input and output parameters that are lists of strings:

- uDataEx—an optional parameter that you can include in almost every message from the IVR Driver for VoiceGenie to the IVR Server. When the IVR Server receives data in the uDataEx parameter list, it attaches the data to the call as user data, thus providing a convenient way to combine attached data with another request. For example, sending routerequest with the uDataEx parameter list attached is equivalent to calling setdata followed by a routerequest.
- extnsEx—*Message Extensions* are optional variables included in some messages. They can be used by routing strategies, IVR scripts, and the client application. The IVR Server does not make any interpretation of these values. It simply forwards them between the Universal Routing Server (URS) and the client application.

Parameter lists uDataEx or extnsEx must be submitted in a specific format. Each named element of the array must appear using a specific naming convention. For example, the follow two data elements could contain the following information:

- the first element (Customer_ID) contains the following information:
  - name[0]=CreditID
  - type[0]=Str
  - val[0]=1234
  - name[1]=SecurityID
  - type[1]=Str
  - val[1]=joe
- the second element (Customer_Name) contains the following information:
  - name=Customer_Name
  - type=Str
  - val=Joseph

The parameter list to be submitted should be formatted as follows:

uDataEx_totalelements=2
uDataEx_0name=Customer_ID
uDataEx_0totalelements=2
uDataEx_00name=CreditID
uDataEx_00type=str
uDataEx_00val=1234
uDataEx_01name=SecurityID
uDataEx_01type=str
uDataEx_01val=joe
uDataEx_1name=Customer_Name
uDataEx_1type=str
uDataEx_1val=Joseph

The numeric increment represents the position in the parameter list, equivalent to an array of strings where the top level increments (0, 1, 2, 3, and so on), and inner levels (00, 01, 02, and so on) where the initial digit ties to the top level and the second digit is the increment. A second top level of 1 would contain an inner list of 11, 12, 13, and so on. An example name list would be:

... namelist="callId channelId messagetype requestId action uDataEx_totalelements uDataEx_0name uDataEx_0totalelements uDataEx_00name uDataEx_00type uDataEx_00val uDataEx_01name uDataEx_01type uDataEx_01val uDataEx_1name uDataEx_1type uDataEx_1val" ...

Output Properties

Upon completion of the subdialog operation, the object name of the subdialog is populated with standard and specific properties associated with the operation invoked. Table 3 lists the standard output properties.

Table 3: Standard Input Variables

Name	Description
status	Set to S if the operation invokes successfully, F if not. See vg_error for a description of the error.
vg_error	Description of the error if status is F.

To access the property values, you must use European Computer Manufacturer's Association (ECMA) scripting to validate the content of the status property of subdialog objects—as in the following example:

```
<if cond="NewCall.status='F'">
    <log>
        <!-- Log the error message and returned event -->
```

```
            <value expr="NewCall.vg_error" />
            <value expr="NewCall.event" />
        </log>
</if>
```

uDataEx Output

The uDataEx output consists of an array of objects, and each object has the following properties:

- name—The name of the object (always available).
- type—If this is unavailable, val is also unavailable, but elements must be available.
- val—If this is available, type is also available, and elements is unavailable.
- elements—This is an array of the same type of object. This must be available if type and val are unavailable.

Note: For each object, you must have either name, type, and val only, or name and elements only.

The following example shows the restrictions of the uDataEx output (assuming that the name of the subdialog object is 'getdata'):

```
getdata.uDataEx[0].name=Customer_Id
getdata.uDataEx[0].type=Int
getdata.uDataEx[0].val=432
getdata.uDataEx[1].name=Customer_Name
getdata.uDataEx[1].elements[0].name=First_Name
getdata.uDataEx[1].elements[0].type=Str
getdata.uDataEx[1].elements[0].val=Joe
getdata.uDataEx[1].elements[1].name=Last_Name
getdata.uDataEx[1].elements[1].type=Str
getdata.uDataEx[1].elements[1].val=Black
getdata.uDataEx[2].name=Customer_Address
getdata.uDataEx[2].elements[0].name=Zip_Code
getdata.uDataEx[2].elements[0].type=Str
getdata.uDataEx[2].elements[0].val=10115
getdata.uDataEx[2].elements[1].name=Street_Number
getdata.uDataEx[2].elements[1].type=Int
getdata.uDataEx[2].elements[1].val=120
getdata.uDataEx[2].elements[2].name=Street_Name
getdata.uDataEx[2].elements[2].type=Str
getdata.uDataEx[2].elements[2].val=Wall St
getdata.uDataEx[2].elements[3].name=APT
getdata.uDataEx[2].elements[3].type=Str
getdata.uDataEx[2].elements[4].name=City
getdata.uDataEx[2].elements[4].type=Str
getdata.uDataEx[2].elements[4].val=New York
```

```
getdata.uDataEx[2].elements[5].name=State
getdata.uDataEx[2].elements[5].type=Str
getdata.uDataEx[2].elements[5].val=NY
```

The uDataEx output is translated in the following XML message:

```
<?xml version='1.0' encoding='iso-8859-1'?>
<!DOCTYPE GctiMsg SYSTEM 'IServer.dtd'>
<GctiMsg>
 <CallId>8997001A-0C01B54E</CallId>
 <UDataResp Result='Success'>
        <RequestId>1102015385154</RequestId>
        <UDataEx>
                <Node Name='Customer_Id' Type='Int' Val='432'/>
                <List Name='Customer_Name'>
                        <Node Name='First_Name' Type='Str' Val='Joe'/>
                        <Node Name='Last_Name' Type='Str' Val='Black'/>
                </List>
                <List Name='Customer_Address'>
                        <Node Name='Zip_Code' Type='Str' Val='10115'/>
                        <Node Name='Street_Number' Type='Int' Val='120'/>
                        <Node Name='Street_Name' Type='Str' Val='Wall St'/>
                        <Node Name='APT' Type='Str' Val=''/>
                        <Node Name='City' Type='Str' Val='New York'/>
                        <Node Name='State' Type='Str' Val='NY'/>
                </List>
        </UDataEx>
 </UDataResp>
</GctiMsg>
```

Interaction Message Types

Table 4 lists the interaction types, and describes the input variables and output properties to and from each VoiceXML application.

Table 4: Message Types

Message	Description
Login Interactions	
loginrequest	Initiates a session and authenticates user access to the IVR Server.
Anytime Interactions	
logmessage	Writes a message to a log file.
getstat	Returns a full report on the requested statistics for the

Message	Description
	specified objects, such as queue, route point, and group of queues.
General Call Interactions	
newcall	Notifies the IVR Server that the call has started.
connected	Indicates that the call has been connected to the specified destination.
getcallinfo	Returns the requested information from the call database.
routerequest	Requests to route the call to a Route Point.
endcall	Notifies the IVR Server that the call has ended.
failure	Indicates that there has been an error in delivering the call to the specified destination.
User Data Interactions	
setdata	Attaches data to a call.
getdata	Requests the value for existing user data keys.
deletedata	Deletes one or all of the user data keys.
External Routing Call Interactions	
accessnumberget	Requests to route the call to a remote site.
accessnumbercancel	Cancels the request to route the call to a remote site.
Treatment Call Interactions	
treatstatus	Indicates what the call treatment process requested by the IVR Server is doing.
cancelcompleted	Indicates that the call treatment requested by the IVR Server has been canceled.

Message	Description
Transfer Call Interactions	
makecall	Initiates a new call to the destination DN.
initiatetransfer	Requests to transfer the call to another agent.
onesteptransfer	Requests to immediately transfer the call to another agent.
completetransfer	Indicates that the transfer has been completed.
Transfer/Conference Call Interactions	
retrievecall	Requests to retrieve the original call that was placed on hold.
Conference Call Interactions	
initiateconference	Requests to conference another agent into the call.
onestepconference	Requests to immediately conference another agent into the call.
completeconference	Indicates that the conference call has been set up.
Statistics Interactions	
peekstat	Returns the current values for the requested statistics.

Note: Each subdialog points to the *same* servlet, which processes the request depending on the messagetype and other required variables.

Login Interactions

The only message in this subdialog group handles login requests.

loginrequest

This message is sent automatically by the servlet during its initialization. You can set the ClientName variable by setting the socket0.clientname variable in the genesysmessages.properties file (see Modifying Configuration Properties).

Notes: This message is handled internally and takes its information from the genesysmessages.properties file.

The reportStatus variable must always be set to true.

Anytime Interactions

The messages in this subdialog group can be used at any time, and include the following messages:

- logmessage
- getstat

logmessage

This message writes a message to a log file.

Subdialog Specification

<subdialog name="logmessage" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype msgType msg" method="post">

Table 5: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is logmessage.
msgType	The type of message being logged. Possible values are Standard, Trace, or Debug.
msg	The action of the message that is written to the log. It can be either Add or Replace.

Table 6: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.

getstat

This message returns a full report on the requested statistics for the specified objects, such as queue, route point, and group of queues.

Subdialog Specification

<subdialog name="getstat" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype requestId serverName statType objectId objectType" method="post">

Table 7: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is getstat.
requestId	The ID for the request. This variable is optional.
serverName	The name of the IVR Server.
statType	The type of statistic being requested.
objectId	The ID for the object being requested.
objectType	The type of object being requested.

Table 8: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
resultCode	This is returned when the status is S. Possible values are Success, NoSuchStat, or MiscError.
result	This is the result of the request.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

General Call Interactions

The messages in this subdialog group handle general calls, and include the following messages:

- newcall
- connected
- getcallinfo
- routerequest
- endcall
- failure

newcall

This message notifies the IVR Server that the channel has answered a call.

The newcall message must be invoked as the first operation at the beginning of the VoiceXML application. After a successful newcall, the next operation should be getcallinfo. Before making additional message calls, check the event property to ensure that a valid Established event is received. The endcall

message must be the last message on the call. A call should have only one newcall and one endcall message.

To ensure that your application has received an Established event:

1. Issue the newcall message.
2. Issue a getcallinfo call on the line, asking for the event.
3. Verify that your application receives an Established event.

The IVR Driver for VoiceGenie is ready to perform additional messages only after it has received an Established event.

Subdialog Specification

<subdialog name="newcall" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype calledNum ani dnis callControlMode uDataEx extnsEx" method="post">

Table 9: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is newcall.
calledNum	The first home location of the call.
dnis	The dialed number identification service. This variable is optional.
ani	The automatic number identification. This variable is optional.
callControlMode	The control mode of the call. Possible values are Genesys, Network, or MakeCall.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 10: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be Established. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, XferComplete, or Released.
failureCause	This is only returned if the call fails. Possible values are Busy, NoAnswer, or ConnectFailed.
endCause	This is returned when the call ends. Possible values are Normal, Abandoned, Resources, FeatureNotSupported, InvalidVersion, InvalidStateTransition, or Timeout.
uDataEx	The attached user data (see uDataEx Output). This may or may not be returned.
extnsEx	The attached T-Server Extensions. This may or may not be returned.

connected

This message indicates that the call has been connected to the specified destination.

Subdialog Specification

<subdialog name="connected" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype extnsEx_totalelements extnsEx_0name extnsEx_0type extnsEx_0val" method="post">

Table 11: Input Variables

Name	Description
callId	The ID for the call is

Name	Description
	session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is connected.
extnsEx_ totalelements	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx_0name	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx_0type	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx_0val	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 12: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.

getcallinfo

This message requests information related to an active call. If a key is given that is not in the current call information, the string default behavior is that the string NoMatch is returned.

Note: If a particular value such as ANI is not available, the default behavior is that the string NULL is returned.

Subdialog Specification

<subdialog name="getcallinfo" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype" method="post">

Table 13: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is getcallinfo.

Table 14: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
ani	The automatic number identification.
dnis	The dialed number identification service.
calledNum	The first home location of the call.
connId	The T-Server connection ID.
tsCallId	The PBX call ID.
portDn	The DN assigned to the IVR port.
portTrunk	The trunk assigned to the IVR port.
portQueue	The queue assigned to the IVR port.

Name	Description
otherDn	Use this value to receive the destination DN (if any).
otherTrunk	Use this value to receive the destination trunk (if any).
otherQueue	Use this value to receive the destination queue (if any).
lastEvent	The name of the last event received on the current channel.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

routerequest

This message is sent by the application to the IVR Server to request that the call be routed by Universal Routing Server (URS). A routerequest message has two possible responses from the IVR Server:

- routeresponse
- treatcall

The IVR Driver for VoiceGenie delivers the following output for the treatcall response.

Table 15: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
msgType	This is the type of message to be treated. The default value is treatcall.

Name	Description
Type	This is the type of treatment. The default value is PlayApplication.
parameters[]	This contains an array of the following set of variables: • parameters[0].name—the application that should be played. • Parameters[0].Type—the type of response; can be either Str (string) or int (integer). • Parameters[0].val—the value of the response. **Note:** These fields are filled by the routing strategy that is executed by URS; therefore, the content depends on URS.

The following example displays a common response for the treatcall message with a PlayApplication type:

parameters[0].name = 'APP_ID'
Parameters[0].Type = 'int'
Parameters[0].val = '0'
parameters[1].name = 'LANGUAGE'
Parameters[1].Type = 'Str'
Parameters[1].val = 'Spanish'

For more information on the treatcall response and routerequest message, refer to their call flows described in the *IVR SDK 7.2 XML Developer's Guide*.

Subdialog Specification

<subdialog name="routerequest" srcexpr ="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype routeDn ced uDataEx extnsEx" method="post">

Table 16: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is routerequest.
calledNum	The first home location of the call.

Name	Description
routeDn	For iscc, the destination DN for the requested Call Data Transfer. For urs, the route point upon which URS acts. **Note:** This variable is optional for Network T-Server configuration mode.
ced	The customer entered digits.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 17: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
routeType	This is the type of route requested. Possible values are Default, Normal, Reroute, RerouteAttended, or RerouteConferenced.
dest	This is the destination to which the request is routed.
extnsEx	The attached T-Server Extensions. This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned,

Name	Description
	even if the IVR Server returns an error message.
endCause	This is returned when the call ends. Possible values are Normal, Abandoned, Resources, FeatureNotSupported, InvalidVersion, InvalidStateTransition, or Timeout.

A route request may invoke a treatment handler if the routing strategy implements the PlayApplication action while waiting for the route request to complete. See the section on Treatment Interactions for more information.

endcall

This message notifies IVR Server that the IVR channel has disconnected the call. Call this message as the last message in the interaction. No message other than getcallinfo with the lastEvent variable is expected for the call session after this message.

Subdialog Specification

<subdialog name="endcall" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype endCause uDataEx ExtnsEx” method="post">

Table 18: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is endcall.
endCause	This is returned when the call ends. Possible values are Normal, Abandoned, Resources, FeatureNotSupported, InvalidVersion, InvalidStateTransition, or Timeout.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 19: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.

failure

This message indicates that there has been an error in delivering the call to the specified destination.

Subdialog Specification

<subdialog name="failure" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype failureCause extnsEx_totalelements extnsEx_0name extnsEx_0type extnsEx_0val" method="post">

Table 20: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is failure.
failureCause	This is only returned if the call fails. Possible values are Busy, NoAnswer, or ConnectFailed.
extnsEx_ totalelements	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx_0name	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Name	Description
extnsEx_0type	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx_0val	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 21: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.

Attached User Data Interactions

The following messages in this subdialog group handle user data stored within a Genesys T-Server call context:

- setdata
- getdata
- deletedata

setdata

This message attaches user data (one or more values) with corresponding names (or keys) to the current call. You can use this message to add one or several key-value pairs of user data to a call.

T-Server or TServer_IVR (in an IVR-In-Front configuration) uses the key-value pairs to track data with a call that is handled at the contact center.

If the key of a key-value pair (which is created when new user data is attached) duplicates a key that already exists, then the new pair replaces the existing pair.

Both the key and the user value must be present. For example:

Key = Customer ID

Value = 1672

Subdialog Specification

<subdialog name="setdata" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype requestId action uDataEx extnsExl" method="post">

Table 22: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is setdata.
requestId	The ID for the request to be added or replaced.
action	It must be either Add or Replace.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsExl	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 23: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
uDataEx	The attached user data (see uDataEx Output). This may or may not be returned.

Name	Description
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

getdata

The getdata message requests the value for existing user data keys that was previously attached to the call.

Note: If a key is given that is not in the current call information, the default behavior is that the string NoMatch is returned.

When keys is set to "userdata", to get the value for the key specified in this message, use the kvpListValue message in the messagetype class and specify this same key as the input argument.

Subdialog Specification

<subdialog name="getdata" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype requestId keys" method="post">

Table 24: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is getdata.
requestId	The ID for the request to be retrieved.
keys	The key-value pairs of the data requested. It must be either "userdata" or "callinfo".

Table 25: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
uDataEx	The attached user data (see uDataEx Output). This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

deletedata

This message deletes from the T-Server user-data table the key-value pair specified by one or all of the user data keys in this call.

Subdialog Specification

<subdialog name="deletedata" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg” namelist="callId channelId messagetype action requestId key" method="post">

Table 26: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.

Name	Description
messagetype	The type of message is deletedata.
requestId	The ID for the request to be deleted.
key	The key-value pair to be deleted.

Table 27: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
uDataEx	The attached user data (see uDataEx Output). This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

External Routing Call Interactions

The following messages in this subdialog group handle external routing calls:

- accessnumberget
- accessnumbercancel

accessnumberget

This message is sent by the application to request that the call be routed to a remote site.

Subdialog Specification

<subdialog name="accessnumberget" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype destDn location xRouteType uui_number UDataEx extnsExl" method="post">

Table 28: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is accessnumberget.
destDn	The DN to which the call is transferred.
location	The switch to which the call is transferred.
xRouteType	This is the type of routing which is processed. It should be Default, Route, Reroute, Direct, DirectAni, DirectNotTaken, DirectAniDnis, DirectUUI, DirectDigits, or DnisPool.
uui_number	The unique user identification number.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 29: Output Properties

Name	Description
status	If S, the request succeeded.

Name	Description
	If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
action	The action to be taken by this message. It must be Get.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
accessNum	The access number. This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

accessnumbercancel

This message may be used only after the accessnumberget message has been invoked.

Subdialog Specification

<subdialog name="accessnumbercancel" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype" method="post">

Table 30: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.

Name	Description
messagetype	The type of message is accessnumbercancel.

Table 31: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
action	The action to be taken by this message. It must be Cancel.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
accessNum	The access number. This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Treatment Interactions

The following messages in this subdialog group handle treatment calls:

- treatstatus
- cancelcompleted

treatstatus

This message takes the request for the next action from IVR Server. If this message is used in the URS session, it returns the IVR treatment or name of the next service (usually a number for transfer).

The following Genesys routing command is equivalent to this T-Server treatment type:

PlayApplication = TreatmentPlayApplication

Treatments work in tandem with events in URS whose status needs to be communicated to the IVR. This is performed using a VoiceXML catch handler that is designed to receive the status messages. An example of the catch handler is as follows:

```
<property name="com.voicegenie.messagehandling" value="immediate"/>

<form id="play">
   <!-- Waiting for Treatment Started response -->
   <catch event="com.voicegenie.message">
      <receive name="msg"/>
      <prompt>message received:
         Status is <value expr="msg"/>.
           Content is <value expr="msg$.msgcontent"/>.
        </prompt>

      <if cond="msg$.msgcontent != null">
            <if cond="msg$.msgcontent == 'Cancel'">
               <submit expr="serverAddress +
   '/gcti/examples/CancelCompleted.jsp'" namelist="callId channelId"
   method="post"/>
            <elseif cond="msg$.msgcontent == 'Other'"/>
               <submit expr="serverAddress +
   '/gcti/examples/TreatError.jsp'" namelist="callId channelId"
   method="post"/>
            </if>
         </if>
      <goto next="#end"/>
   </catch>
…
```

Use the catch event com.voicegenie.message to receive the throw from the IVR Driver for VoiceGenie with the message content accessed via the VoiceGenie call control extension <Receive />. The message content will either contain Cancel or Other. If Cancel is received, URS is requesting the cancellation of the treatment to take control of the call and route to target. If Other is received, another condition has occurred in URS and appropriate action should be taken. In the example above, Other is an inappropriate response and causes and error condition to be executed.

To ensure that the externally thrown event is processed as soon as the event is received, the VoiceGenie property com.voicegenie.messagehandling must be set to immediate. Failure to implement this property will cause the event to be processed at the end of script processing which would cause delays in servicing the caller.

For a detailed description of treatments and variables, see the *Genesys Universal Routing 7.5 Reference Information* document. See your vendor IVR

documentation for a description of whether and how the IVR supports these treatments.

Subdialog Specification

<subdialog name="treatstatus" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype Status ced uDataEx ExtnsEx” method="post">

Table 32: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is treatstatus.
Status	If S, the request succeeded. If F, the request failed.
ced	The customer entered digits.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 33: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
action	The action to be taken by this message. It must be Cancel.
result	If the status is S, this is the result of the request. Possible

Name	Description
	values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
accessNum	The access number. This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

cancelcompleted

This message indicates that the call treatment requested by the IVR Server has been canceled, after receiving an action of type Cancel.

Subdialog Specification

<subdialog name="cancelcompleted" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype" method="post">

Table 34: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is cancelcompleted.

Table 35: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.

Name	Description
vg_error	This is returned only if the status is F, and an error reason is available.
action	The action to be taken by this message. It must be Cancel.
result	If the status is S, this is the result of the request. Possible values are Success, NoSuchCall, NoMatch, FeatureNotSupported, or MiscError.
accessNum	The access number. This may or may not be returned.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Transfer Call Interactions

The following messages in this subdialog group handle transfer calls:

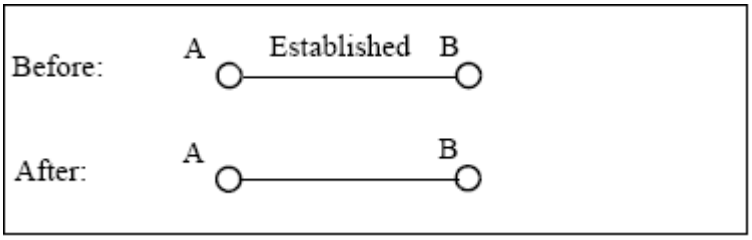
- makecall
- initiatetransfer
- onesteptransfer
- completetransfer

makecall

This message initiates a new call to the destination DN (B). See the scenario below. The makecall message works with the switch, which is monitored by T-Server.

This message is used for the IVR-Behind-Switch configuration.

Scenario



Subdialog Specification

<subdialog name="makecall" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype orgNum destNum" method="post">

Table 36: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is makecall.
orgNum	The DN from which the call is being made.
destNum	The DN to which the call is being made.

Table 37: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be Established. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, XferComplete, or Released.

Name	Description
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.
endCause	This is returned when the call ends. Possible values are Normal, Abandoned, Resources, FeatureNotSupported, InvalidVersion, InvalidStateTransition, or Timeout.

initiatetransfer

This message initiates a consulting call. For more information, see the description in the *T-Library SDK 7.2 C Developer's Guide*.

During the consulting call, this occurs:

- The original party (A) is placed on hold for the duration of the consulting call.
- The party (B) who requests the initiatetransfer message is involved in a new consulting call with a third party (C).

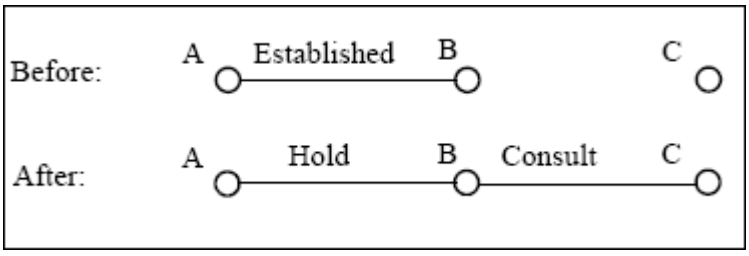
See the scenario below.

This message is used for the IVR-Behind-Switch configuration.

After this message is invoked, use the following messages to complete the operation:

- onesteptransfer
- retrievecall
- completetransfer

Scenario



Subdialog Specification

<subdialog name="initiatetransfer" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype destDn location uDataEx extnsEx" method="post">

Table 38: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is initiatetransfer.
destDn	The DN to which the call is transferred.
location	The switch to which the call is transferred. This variable is optional.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 39: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be XferComplete. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error

Name	Description
	message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

onesteptransfer

This message produces a direct transfer without an intervening conference call. For more information, see the description in the *T-Library SDK 7.2 C Developer's Guide*. This call does the following:

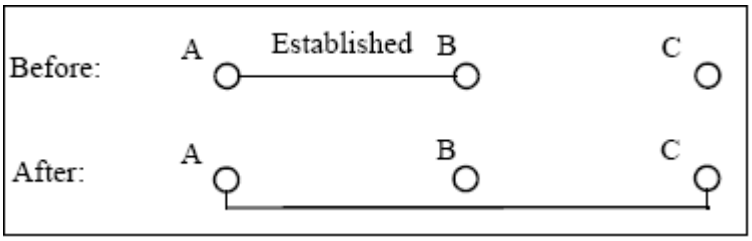
- Leaves an extension (B), where a transfer is initiated.
- Moves to a new extension (C) specified in the destination DN.

In other words, party (A), who initiates the call, is disconnected from the original DN (B) and reconnected to the destination DN (C).

See the scenario below.

This message is used only for the IVR-Behind-Switch configuration.

Scenario



Subdialog Specification

<subdialog name="onesteptransfer" srcexpr ="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype destDn location uDataEx extnsEx" method="post">

Table 40: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.

Name	Description
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is onesteptransfer.
destDn	The DN to which the call is transferred.
location	The switch to which the call is transferred.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 41: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.
event	This is the event received by CallStatus. It should only be XferComplete. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, or Released.
failureCause	This is only returned if the call fails. Possible values are Busy, NoAnswer, or ConnectFailed.

Name	Description
extnsEx	The attached T-Server Extensions (see uDataEx Output). This may or may not be returned.

completetransfer

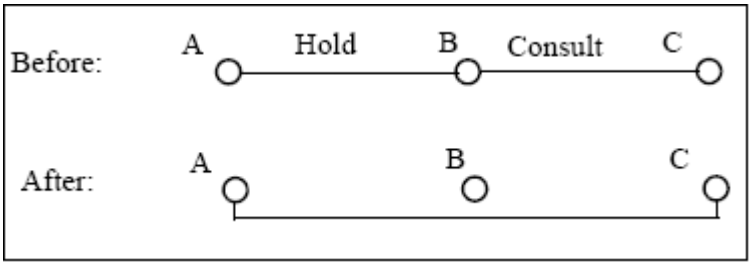
This message completes a transfer. For more information, see the description in the *T-Library SDK 7.2 C Developer's Guide*. During the transfer, this occurs:

- Caller (A) is placed on hold.
- The original recipient (B) makes a call to a new party (C).
- A consulting call is established between the original recipient (B) and the new party (C).
- The completetransfer message initiates the completion of the transfer.
- When the transfer is complete, the original recipient (B), who initiated the transfer, is dropped. The ongoing call involves only two participants—the original caller (A) and the new party (C).

See the scenario below.

This message is used for the IVR-Behind-Switch configuration.

Scenario



Subdialog Specification

<subdialog name="completetransfer" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype extnsEx" method="post">

Table 42: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.

Name	Description
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is completetransfer.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 43: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be XferComplete. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Transfer/Conference Call Interactions

The retrievecall message in this subdialog group retrieves calls for transfers and conferences.

retrievecall

This message is sent by the application to request that the original call be retrieved from being placed on hold, and complete the transfer or conference call. During the transfer or conference call, this occurs:

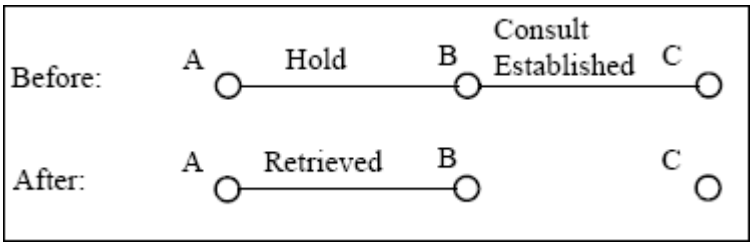
- For the duration of the transfer or conference call, the original party (A) is placed on hold.
- The second party (B) then creates a transfer or conference call to the third party (C).
- After the retrieve, the third party (C) is released, and the call between the first party (A) and second party (B) is restored.

See the scenario below.

This message is used for the IVR-Behind-Switch configuration.

Note: Use this message only after initiatetransfer or initiateconference has successfully completed.

Scenario



Subdialog Specification

<subdialog name="retrievecall" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype extnsEx" method="post">

Table 44: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is retrievecall.

Name	Description
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 45: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be XferComplete. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyAdd, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Conference Call Interactions

The following messages in this subdialog group handle conference calls:

- initiateconference
- onestepconference
- completeconference

initiateconference

This message initiates a conference call. For more information, see the description in the *T-Library SDK 7.2 C Developer's Guide*.

During the consulting call, this occurs:

- The original party (A) is placed on hold for the duration of the conference call.
- The party (B) who requests the initiateconference message is involved in a new conference call with a third party (C).

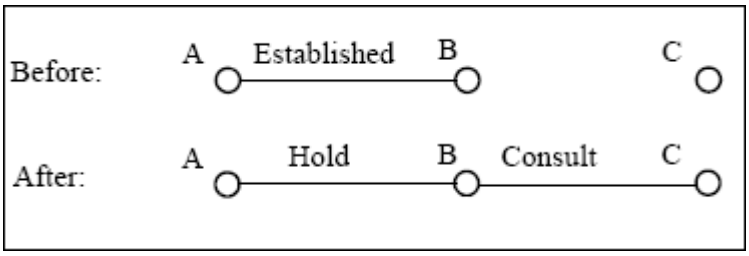
See the scenario below.

This message is used for the IVR-Behind-Switch configuration.

After this message is invoked, use one of the following messages to complete the operation:

- onestepconference
- retrievecall
- completeconference

Scenario



Subdialog Specification

<subdialog name="initiateconference" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype destDn location uDataEx extnsEx" method="post">

Table 46: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is initiateconference.
destDn	The DN to which the conference call is made.
location	The switch to which the conference call is made. This variable is optional.

Name	Description
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 47: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be ConfPartyAdd. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

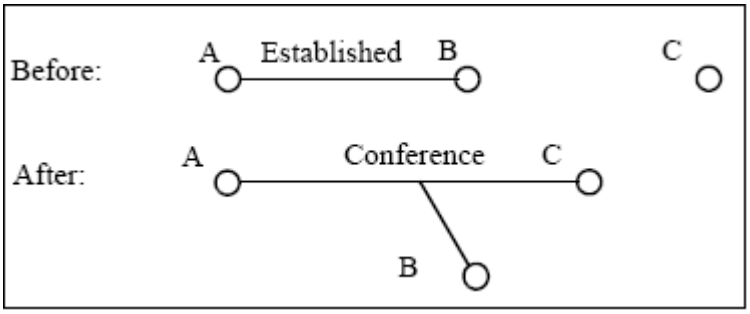
onestepconference

This message makes a conference call to a new DN by connecting the two parties (A and B) of the original conversation with an additional party (C). See the scenario below.

Using this message, three or more people can participate in the same phone conversation, talking from three or more extensions.

This message is used for the IVR-Behind-Switch configuration.

Scenario



Subdialog Specification

<subdialog name="onestepconference" srcexpr ="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype destDn location uDataEx extnsEx" method="post">

Table 48: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is onestepconference.
destDn	The DN to which the conference call is made.
location	The switch to which the conference call is made.
uDataEx	The attached user data (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 49: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.

Name	Description
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be ConfPartyAdd. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy, Held, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

completeconference

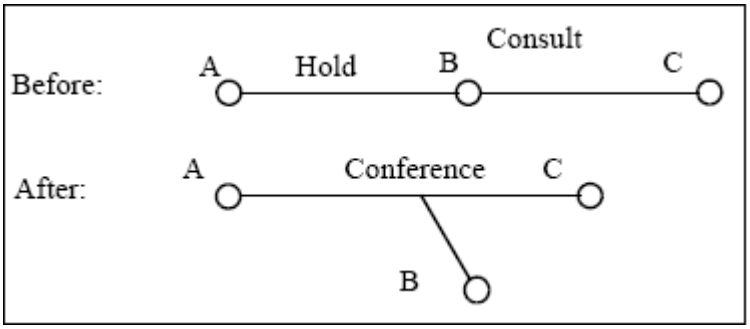
This message is sent to indicate that the conference call has been set up. It completes a consulting call by connecting the two parties (A and B) of the original conversation with an additional party (C) from the consulting call. See the scenario below.

Using this message, three or more people can participate in one phone conversation at the same time, talking from three or more extensions. For more information, see the description in the *T-Library SDK 7.2 C Developer's Guide*.

This message is used for the IVR-Behind-Switch configuration.

Note: Use this message only after initiateconference has been successfully completed.

Scenario



Subdialog Specification

<subdialog name="completeconference" srcexpr ="“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype extnsEx" method="post">

Table 50: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is completeconference.
extnsEx	The attached T-Server Extensions (see uDataEx and extnsEx Input Parameter Lists). This variable is optional.

Table 51: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be ConfPartyAdd. If the invocation was not successful, you could receive one of the following values: Retrieved, Busy,

Name	Description
	Held, ConfPartyDel, or Released.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Stat Server Interactions

The peekstat message in this subdialog group handles statistic requests.

peekstat

This message requests that Stat Server provide information about a predefined statistic. Variables of the statistic are defined in the configuration environment.

Note: The supported statistics (which must be configured in the IVR Server) are CurrNumberWaitingCalls and ExpectedWaitTime.

Subdialog Specification

<subdialog name="peekstat" srcexpr =“http://localhost:8080/gcti/ivrdriver_vg" namelist="callId channelId messagetype requestId statName" method="post">

Table 52: Input Variables

Name	Description
callId	The ID for the call is session.com.voicegenie.telephone.primarychan.
channelId	The ID for the channel is session.telephone.callidref.
messagetype	The type of message is peekstat.
statName	This string specifies the name of the required statistic. It must be CurrNumberWaitingCalls or ExpectedWaitTime.

Table 53: Output Properties

Name	Description
status	If S, the request succeeded. If F, the request failed.
vg_error	This is returned only if the status is F, and an error reason is available.
event	This is the event received by CallStatus. It should only be Retrieved. If the invocation was not successful, you could receive one of the following values: Established, Busy, Held, ConfPartyAdd, ConfPartyDel, XferComplete, or Released.
resultCode	This is returned when the status is S. Possible values are Success, NoSuchStat, or MiscError.
result	This is the result of the request.
failedReq	This is returned only if the IVR Server returns a call error message. Possible values are Unknown, NoSuchCall, OneStepXfer, OneStepConf, InitConf, CompleteConf, InitXfer, CompleteXfer, RetrieveCall, or MakeCall.
tlibErrCode	The T-Library error code. This may or may not be returned, even if the IVR Server returns an error message.

Appendix

VoiceXML Example

This appendix provides an example of VoiceXML pages that calls the subdialog interface (see Message Types).

Note: The following VoiceXML code is for illustrative purposes only. It calls the subdialog and logs the returned values. In order to make useful integration with Genesys CTI, you would have to modify the example significantly.

```
<?xml version="1.0"?>
<vxml version="2.0" xmlns=“http://www.w3.org/2001/vxml">
    <property name="metricslevel" value="3"/>
    <property name="loglevel" value="3"/>

    <script>
        <![CDATA[
        function getSIPPort(sipstr) {
            var offset;
            offset=sipstr.indexOf("@");
            return sipstr.substring(4,offset);
        }

        function getAllObject(obj, topstr) {
            var finalresult = "";
            for(var prop in obj) {
                if(prop != undefined) {
                finalresult = finalresult.concat(topstr + "." + prop +"=" + obj[prop] + "\n");
                }
            }
            return finalresult;
        }

        function getUDataEx(obj, topstr) {
            var finalresult = '';
            for(var i = 0; i < obj.length; i++) {
```

```
                    if(obj[i].name != undefined) {
                        finalresult = finalresult.concat(topstr + '[' + i + '].name=' + obj[i].name + "\n");
                    }
                    if(obj[i].type != undefined) {
                        finalresult = finalresult.concat(topstr + '[' + i + '].type=' + obj[i].type + "\n");
                    }
                    if(obj[i].val != undefined) {
                        finalresult = finalresult.concat(topstr + '[' + i + '].val=' + obj[i].val + "\n");
                    }
                    if(obj[i].elements != undefined) {
                        finalresult = finalresult.concat(getUDataEx(obj[i].elements, topstr + '[' + i +
'].elements'));
                    }
                }
                return finalresult;
            }
            ]]>
            </script>

        <var name="callId" expr="session.com.voicegenie.telephone.primarychan"/>
        <!-- SIP -->
        <!-- <var name="channelId" expr="getSIPPort(session.telephone.dnis)"/> -->
        <!-- TDM -->
        <var name="channelId" expr=" session.connection.callidref"/>
        <var name="dnis" expr="session.telephone.dnis"/>
        <var name="ani" expr="session.telephone.ani"/>
        <var name="callControlMode" expr="'Network'"/>
        <var name="messagetype"/>
        <var name="endCause" expr="'Normal'"/>

        <form id="beginningCall">
            <block>
                <log>***************</log>
                <log>CALL-ID is  <value expr="callId" /></log>
                <log>DNIS is  <value expr="dnis" /></log>
                <log>ANI is  <value expr="ani" /></log>
                <log>channelId is  <value expr="channelId" /></log>
                <log>***************</log>
            </block>
            <!-- New Call -->
            <block>
                <assign name="messagetype" expr="'newcall'"/>
            </block>
            <subdialog name="newcall" src="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId
channelId messagetype callControlMode" method="post">
                <filled>
                    <log><value expr="getAllObject(newcall, 'newcall')"/></log>
                    <!-- value expr="newcall.status"/ -->
                </filled>
            </subdialog>
```

```
<!-- Get Call Info -->
<block>
    <assign name="messagetype" expr="'getcallinfo'"/>
</block>
<subdialog name="getcallinfo" src="http://localhost:8080/gcti/ivrdriver_vg"
namelist="callId channelId messagetype" method="post">
    <filled>
        <log><value expr="getAllObject(getcallinfo,'getcallinfo')"/></log>
        <!-- value expr="getcallinfo.status"/ -->
    </filled>
</subdialog>

<!-- Set Data -->
<block>
    <assign name="messagetype" expr="'setdata'"/>
</block>
<var name="action" expr="'Add'"/>
<var name="uDataEx_totalelements" expr="2"/>
<var name="uDataEx_0name" expr="'IVRfieldnames1'"/>
<var name="uDataEx_0type" expr="'Str'"/>
<var name="uDataEx_0val" expr="'Customer_Id,Customer_Name'"/>
<var name="uDataEx_1name" expr="'IVRdata1'"/>
<var name="uDataEx_1type" expr="'Str'"/>
<var name="uDataEx_1val" expr="'432,jim'"/>

<subdialog name="setdata" src="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId
channelId messagetype action uDataEx_totalelements uDataEx_0name uDataEx_0type
uDataEx_0val uDataEx_1name uDataEx_1type uDataEx_1val" method="post">
    <filled>
        <log><value expr="getAllObject(setdata, 'setdata')"/></log>
        <!-- value expr="setdata.status"/ -->
    </filled>
</subdialog>

<!-- Get Data -->
<block>
    <assign name="messagetype" expr="'getdata'"/>
</block>
<var name="keys" expr="'IVRfieldnames1:IVRdata1'"/>
<subdialog name="getdata" src="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId
channelId messagetype keys" method="post">
    <filled>
        <log><value expr="getAllObject(getdata, 'getdata')"/></log>
        <log><value expr="getUDataEx(getdata.uDataEx,'getdata.uDataEx')"/></log>
        <!-- getdata.uDataEx=<value expr="getdata.uDataEx"/>-->
        <log><value expr="getdata.uDataEx[0].name"/>=<value
expr="getdata.uDataEx[0].val"/>,<value expr="getdata.uDataEx[1].name"/>=<value
expr="getdata.uDataEx[1].val"/></log>
        <!-- value expr="getdata.status"/ -->
```

```
            </filled>
        </subdialog>
        <!-- Peek Stat -->
        <block>
            <assign name="messagetype" expr="'peekstat'"/>
        </block>
        <var name="statName" expr="'CurrNumberWaitingCalls'"/>
        <subdialog name="peekstat" src="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId
channelId messagetype statName" method="post">
            <filled>
                <log><value expr="getAllObject(peekstat, 'peekstat')"/></log>
                <!-- value expr="peekstat.status"/ -->
            </filled>
        </subdialog>

        <!-- One Step Transfer -->
        <block>
            <assign name="messagetype" expr="'onesteptransfer'"/>
        </block>
        <var name="destDn" expr="'1771'"/>
        <subdialog name="onesteptransfer" src="http://localhost:8080/gcti/ivrdriver_vg"
namelist="callId channelId messagetype destDn" method="post">
            <filled>
                <log><value expr="getAllObject(onesteptransfer, 'onesteptransfer')"/></log>
                <!-- value expr="onesteptransfer.status"/ -->
            </filled>
        </subdialog>
        <block>
            <goto next="#endingCall"/>
        </block>
        <catch event="error telephone.disconnect">
            <goto next="#endingCall"/>
        </catch>
    </form>

    <form id="endingCall">
        <!-- End Call -->
        <block>
            <assign name="messagetype" expr="'endcall'"/>
        </block>
        <subdialog name="endcall" src="http://localhost:8080/gcti/ivrdriver_vg" namelist="callId
channelId messagetype endCause" method="post">
            <filled>
                <log><value expr="getAllObject(endcall, 'endcall')"/></log>
                <!-- value expr="endcall.status"/ -->
            </filled>
        </subdialog>
    </form>
</vxml>
```