Dialogic

Dialogic® Vision™ Call Server

Administration Manual

October 2011 64-0406-05

www.dialogic.com

Copyright and legal notice

Revision history

Revision	Release date	Notes
64-0406-05 Rev A	October 2011	BK, Dialogic® Vision™ 1000 Video Gateway 5.2 and Dialogic® Vision™ 1000 Programmable Media Platform 5.2.
64-0406-04 Rev A	January 2011	BK, Dialogic® Vision™ 1000 Video Gateway 5.1 and Dialogic® Vision™ 1000 Programmable Media Platform 5.1.
64-0406-03 Rev A	May 2010	BK, Dialogic® Vision™ 1000 Video Gateway 5.0 and Dialogic® Vision™ 1000 Programmable Media Platform 5.0.
64-0406-02 Rev B	December 2009	BK, Dialogic® Vision™ CX Video Gateway 4.2 and Dialogic® Vision™ VX Integrated Media Platform 4.2.
64-0406-02 Rev A	August 2009	BK, Dialogic® Vision™ CX Video Gateway 4.2 and Dialogic® Vision™ VX Integrated Media Platform 4.2.
64-0406-01 Rev A	June 2009	DEH/BK, Dialogic® Vision™ CX Video Gateway 4.1 and Dialogic® Vision™ VX Integrated Media Platform 4.1.
Last modified: 2011-10-24		

Refer to www.dialogic.com for product updates and for information about support policies, warranty coverage, and service offerings.

Table Of Contents

1. Introduction

The *Dialogic® Vision™ Call Server Administration Manual* provides detailed information about configuring the Dialogic® Vision™ Call Server for the Dialogic® Vision™ 1000 Video Gateway and the Dialogic® Vision™ 1000 Programmable Media Platform.

This manual supplements the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* and the *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual,* and assumes that you have read one of these manuals before using the current manual.

Note: Product names have been changed. The table below indicates terminology that was formerly associated with the products, as well as the new terminology by which the products are now known.

Former terminology	Current terminology
Dialogic® Vision™ CX Video Gateway	Dialogic® Vision™ 1000 Video Gateway Also referred to as "Video Gateway"
Dialogic® Vision™ VX Integrated Media Platform	Dialogic® Vision™ 1000 Programmable Media Platform Also referred to as "Programmable Media Platform"

The terms "Dialogic® Vision™ Server", "Vision Server", or "server" are used in this document to refer collectively or individually (depending on specific context) to the Dialogic® Vision™ 1000 Video Gateway and the Dialogic® Vision™ 1000 Programmable Media Platform.

2. Overview

Overview of the Dialogic Vision Call Server

The Dialogic® Vision™ Call Server is the subsystem of the Dialogic® Vision™ 1000 Video Gateway and the Dialogic® Vision™ 1000 Programmable Media Platform that manages call control capabilities. The Vision Call Server also functions as a media gateway for the Programmable Media Platform. Depending on the telecom model your Vision Server supports, the Vision Call Server can:

- Support the ISDN, ISUP, BICC, and SIP telecommunications protocols.
- Support signaling server subsystem for scalable and highly available SS7 connectivity.
- Terminate T1/E1 TDM audio trunks.
- Support 3G-324M video on TDM and IP networks.
- Leverage video transcoder resources for enhanced video gateways and applications.
- Execute CCXML applications.

Note: This manual also uses the term Call Server to refer to the Vision Call Server.

Call Server components

The Call Server is composed of three components:

- Telecom signaling layers
- Media capabilities
- CCXML Scripting Engine

These components are based on specific hardware support, including Dialogic® media boards and signaling boards.

Telecom signaling layers

The telecom signaling layers in the Call Server support the following telephony and signaling interfaces:

- ISDN protocol
- ISUP protocol
- BICC protocol
- VoIP protocols (SIP/RTP)

Media capabilities

The Call Server supports the following codecs:

Codec type	Description
Audio	• AMR (3GPP TS 26.090, 26.101, and 26.073, version 5.3.0, 2004) • G.711 A-law and mu-law • G.723.1 • G.726 • G.729 A
Video	• H.263, • H.263+ • H.264 • MPEG-4
Fax relay	T.38

For information on the codecs and the standards supported, see the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* and *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual.*

The Call Server also supports audio conferencing using CCXML, Video 3G-324M, and Video-over-IP.

CCXML scripting engine

The CCXML scripting engine enables the Call Server to execute applications written in the Call Control Extensible Markup Language (CCXML). You can use CCXML to write applications that provide call control for the duration of a phone call, including call setup, monitoring, and tear-down. You can also use CCXML to implement conferencing. The CCXML version implemented in the Call Server is based upon the W3C Working Draft of CCXML dated 29 June 2005. For information, see http://www.w3.org/TR/2005/WD-ccxml-20050629.

For general information about using CCXML, including instructions for creating a CCXML application definition file, see the *Dialogic® Vision™ CCXML Developer's Manual*. For information about configuring the CCXML engine, see CCXML scripting engine settings.

Document conventions

The Call Server software is installed in the */opt/nms/vx* directory. This manual uses the string *vx* to refer to the installation directory.

System file locations

The main Call Server system files are stored in the following default locations:

Directory	Description
vx/callserver/conf/callserver.conf	Call Server configuration file. For more information, see Call Server configuration file.
vx/callserver/conf/telecom.conf	Configuration file for signaling layers and media capabilities. For more information, see Telecom configuration file.
vx/callserver/logs	Call Server log file directory. For more information, see Call Server logging.

3. Call Server configuration file settings

Call Server configuration file

This section describes how to fine-tune the Call Server configuration by using the Call Server configuration file, *callserver.conf,* which resides in the *vx/callserver/conf* directory.

When specifying values for a particular setting, use the following syntax:

setting_name*=*setting_value

If ***setting_value*** is a long string, continue the value over several lines by placing a backslash (\) at the end of every line.

To designate time, use either seconds or milliseconds (ms), unless otherwise stated.

You can configure the following types of settings in the *callserver.conf* file:

- General
- Logging
- CCXML scripting engine
- Client-side HTTP
- VoiceXML dialog
- Fax call detection
- Call detail record generation

General settings

Use the following settings in the *callserver.conf* file to configure general settings for the Call Server:

Keyword	**Description**
ProcessName	Name of the Call Server process. Default: callserver
SNMPEnabled	Specifies whether to enable the Call Server's SNMP sub-agent. The SNMP sub-agent handles SNMP requests for getting and setting Call Server management information. It also sends SNMP notifications through the Net-SNMP master agent. Valid values: • true • false Default: true For information about using SNMP with the Call Server, see the *Dialogic® Vision™ SNMP Reference Manual.*

Keyword	Description
TelecomConfigFile	Location and name of the telecom configuration file. Default: *vx/callserver/conf/telecom.conf* For more information, see Telecom configuration file.

Logging settings

Use the following settings in the *callserver.conf* file to configure logging for the Call Server:

Keyword	Description
LogDir	Directory where Call Server system log files are stored. Default: *vx/callserver/logs*
SystemLogLevel	Severity of a log message when it is recorded in the system log file. Valid values (in order of decreasing severity and increasing verbosity): • FATAL ERROR • ERROR • WARNING • INFO1 • INFO2 • INFO3 • INFO4 • INFO5 Default: INFO1
SystemLogFileMaxNum	Maximum number of system log files kept by the Call Server. When this value is reached, the oldest system log file is deleted so that the number of log files does not exceed the specified quantity. A value of zero (0) specifies that the Call Server never deletes system log files. Valid values: 0 - 500 Default: 50

Keyword	Description
SystemLogFileMaxSize	Maximum size of the system log file. When the system log file reaches this size, the Call Server creates a new log file with an incremented file index. Include a unit identifier, such as MB or KB, with the value. The default unit identifier is MB. Valid values: 100KB - 500MB Default: 10MB
SystemLogTime	Time format for the Call Server system log. Valid values: • GMT: Greenwich Mean Time format. • LOCAL: Same time format as the local Vision Server. Default: LOCAL

For more information about logging, see Call Server logging.

CCXML scripting engine settings

Use the following settings in the *callserver.conf* file to configure the CCXML scripting engine:

Name	Description
CcxmlAppliConfig	URI of the CCXML application definition file. This file maps number ranges to CCXML applications. It also contains application settings such as dialog server addresses and outbound routes. Currently, only *file://* URIs are supported. Default: *file://vx/callserver/conf/ccxmlappcfg.xml* For more information, see the *Dialogic® Vision™ CCXML Developer's Manual*.
CcxmlAppLogLevel	Severity level of CCXML application logs. Valid values (in order of decreasing severity and increasing verbosity): • 0 (info level 1) • 1 (info level 2) • 2 (info level 3) • 3 (info level 4) • 4 (info level 5) • 5 (info level 6) • 6 (info level 7) Default: 2

Name	Description
CcxmlCacheDir	Cache directory for CCXML files. Default: *vx/callserver/data/ccxml*
CCXMLConnectionLocal	Indicates whether to populate the connection.local property field with the content of the SIP TO field. Valid values: • to: Populates the connection.local property with the content of the SIP TO field. • Any value besides *to*: Populates the connection.local property with the request-URI. Default: Blank Use this setting for backward compatibility with previous versions that only use the TO field to populate connection.local.
CcxmlDefaultAppType	Type of application used for routing inbound calls that are not matched by a CCXML application. Valid values: • CUSTOM: Call Server uses the CCXML application specified by the CcxmlInboundUri setting. By default, this application is *vx/callserver/www/ccxml/inbound.ccxml*, and it routes all inbound calls to a VoiceXML server. • GATEWAY: Call Server uses the CCXML application specified by the CcxmlGatwayUri setting, which references the information in the gateway routing table. By default, this application is called *vx/callserver/www/ccxml/gateway.ccxml.* Default: • GATEWAY, for the Video Gateway • CUSTOM, for the Programmable Media Platform
CcxmlDtdLocation	Location of the CCXML DTD file. Default: *vx/callserver/package/ccxml.dtd*
CcxmlGatewayUri	For gateway routing (CcxmlDefaultAppType = GATEWAY), URI of the CCXML application to execute when an incoming call is answered. This value can be an HTTP or file URI, and it must be a full URI. Default: *file://vx/callserver/www/ccxml/gateway.ccxml*

Name	**Description**
CcxmlGwAppliConfig	URI of the gateway route table configuration file, which is used by gateway CCXML applications. Currently, only *file://* URIs are supported. Default: *file://vx/callserver/conf/gwappcfg.xml*
CcxmlInboundUri	For custom CCXML applications (CcxmlDefaultAppType = CUSTOM), URI of the CCXML application to execute when the Vision Server answers a call whose dialed number does not match the number range for any of the configured CCXML applications. The value must be a full URI, because relative URIs are not allowed. Both HTTP and local file URIs are supported. In the latter case, the file:// protocol specifier must precede the path. Default: *vx/callserver/www/ccxml/inbound.ccxml* For information about configuring CCXML applications see the *Dialogic® Vision™ CCXML Developer's Manual*.
CcxmlNumChannels	Number of CCXML interpreter channels to be started. Each channel runs as a separate thread. The value of this field depends the number of configured CCXML ports.
CcxmlPropagateIsup	Propagate the ISUP IE inside the CCXML script when receiving ISUP events. Valid values: TRUE, FALSE. Default: TRUE.
CcxmlResManThreads	Size of the resource manager thread pool, which dictates the number of fetch requests that can be serviced simultaneously. The thread pool is shared across all CCXML interpreter channels. Default: 40

Name	Description
CcxmlSysLogLevel	Severity level of the CCXML system log. Valid values (in order of decreasing severity and increasing verbosity): • 0 (error) • 1 (warning) • 2 (info level 1) • 3 (info level 2) • 4 (info level 3) • 5 (info level 4) • 6 (info level 5) Default: 2

For information about using CCXML, see the *Dialogic® Vision™ CCXML Developer's Manual*.

Client-side HTTP settings

Use the following settings in the *callserver.conf* file to configure the Call Server client-side HTTP settings for transferring files over the internet:

Name	Description
BypassProxyAddress	Comma-separated list of IP addresses or host names for the proxy server to bypass. Default: 127.0.0.1,localhost
ProxyAddress	Name or IP address (and port) of the proxy server for the Call Server to use when fetching files. Use the following syntax to format an IP address: ***IPAddress*:*Port*** If no value is specified for this setting, the Call Server does not use a proxy server. Default: Blank
HTTPClientRepository	Location where temporary files downloaded by the Call Server are stored. Default: *vx/callserver/data/cache*

VoiceXML dialog settings

Use the DialogDisconnectMode setting in the *callserver.conf* file to specify how the Call Server operates with a VoiceXML Server when connections are disconnected; for example, when a caller hangs up.

The valid values for DialogDisconnectMode are:

Value	Description
BYE	(Default) The dialog between the Call Server and the VoiceXML Server immediately terminates.
DISCONNECT	The VoiceXML Server is notified when the call is disconnected and is then allowed to terminate the dialog gracefully. Use this mode for full W3C compliance when the Call Server expects the VoiceXML application to return exit data.

Fax call detection settings

Use the following settings in the *callserver.conf* file to enable the Call Server to detect and react to T.38 fax calls:

Name	Description
DetectFaxToneCNG	Enables or disables the CNG (calling) tone detector, which determines whether an incoming call is a fax call. Valid values: • TRUE: Enables the CNG tone detector. • FALSE: Disables the CNG tone detector. Default: FALSE
InitiateReInviteUponFaxToneCNG	Indicates whether to issue a SIP RE-INVITE with T.38 SDP when the Call Server detects a CNG tone. Valid values: • TRUE: Initiates a SIP RE-Invite with T.38 SDP • FALSE: Does not initiate a SIP RE-INVITE. Default: FALSE

By default, the fax CNG tone detector listens for tones that have a minimum amplitude of 28 decibels and a minimum duration of 300 ms. To change these specifications, see Fax tone recognition settings.

Call detail record generation settings

Use the following settings in the *callserver.conf* file to enable the Call Server to generate call detail records (CDRs).

Name	Description
BillingCDRBaseDir	Directory in which to store call detail records (CDRs). Using the tree directory structure (default structure), CDRs are written every hour to a file named ***YYYY*/*MM*/*DD*/*HH*.*cdr***, off the base directory. Using the flat directory structure, CDRs are written to a single directory. Default: *vx/callserver/cdr*
BillingEnabled	Indicates whether the Call Server generates CDRs: • true • false Default: false
BillingCDRFormat	Indicates which format to use for CDR entries: • 0: use name=value format • 1: use "|" to separate fields in the CDR entry Default: 0
BillingCDRFlatDirectoryStructure	Type of structure used for CDRs. • 0: CDR files are in multiple directories under the base directory specified by BillingCDRBaseDir. This is the tree directory structure. • 1: CDR files are in a single directory specified by BillingCDRBaseDir. This is the flat directory structure. Default: 0
BillingCDRMaxDirSize	(Flat directory structure only) When BillingCDRPurgeMode is 1, the oldest non-active files are deleted from the CDR directory when the directory reaches this size limit. The directory size is checked on every file rollover and files are deleted until the size falls below BillingCDRMaxDirSize. Include a unit identifier, such as KB or MB, with the value. The default unit is MB. Values are: 1KB to 100MB. No default value.

Name	Description
BillingCDRMaxFileAge	(Flat directory structure only) When BillingCDRPurgeMode is 3, non-active CDR files older than this age limit are deleted from the CDR directory. The file age is checked on every file rollover. Include a unit identifier, such as minute, hour, day, month, or year with the value. The default unit is minutes. Values are: 1 minute to 10 years. No default value.
BillingCDRMaxFileCount	(Flat directory structure only) When BillingCDRPurgeMode is 2, non-active CDR files that exceed this file limit are deleted from the CDR directory. The file count is checked on every file rollover. Values are: 1 to 1000. No default value.
BillingCDRPurgeMode	(Flat directory structure only) Criteria used for purging files in the CDR directory. Values are: • 0: no automatic purge • 1: purge based on directory size (BillingCDRMaxDirSize) • 2: purge based on file count (BillingCDRMaxFileCount) • 3: purge based on file age (BillingCDRMaxFileAge) Default: 0 For values 1 to 3, purging occurs only when the corresponding purge parameter is explicitly specified. For example, if BillingCDRPurgeMode is 1, purging occurs according to the value specified in BillingCDRMaxDirSize.
BillingCDRRolloverSizeLimit	(Flat directory structure only) Maximum size of an active CDR file. File rolls over when its size reaches this maximum value. Include a unit identifier such as MB or KB with the value. To distinguish active CDR files from rolled over CDR files, the active CDR file has the extension *.open*. Values are: 1KB to 500MB. Default: 0 (unlimited file size)

Name	Description
BillingCDRRolloverTimeLimit	(Flat directory structure only) Maximum time in minutes that a CDR file remains active before it is rolled over. File rolls over when it reaches this time limit. To distinguish active CDR files from rolled over CDR files, the active CDR file has the extension *.open*. Values are: 1 to 525600. Default: 60
BillingCDRStartTime	Indicates how the start time is set in the CDR entry: • ANSWER: when the call control answer event is received or sent. • H324_DONE: when the 3G-324M negotiation is complete.

CDR fields

When using the "|" character to separate fields in the CDR entry, the field names themselves are not listed in the entry; only the values separated by "|". Therefore, it is important to list the values in the expected order as shown in the table.

For more information about CDRs such as the flat directory structure and CDR file aggregation, see the *Dialogic® Vision™ 1000 Video Gateway Administration Manual.*

Index	CDR field	Description
1	Service start time	
2	Service end time	
3	Call duration	
4	Call identifier	
5	Call type	
6	Call mode	
7	Source information or ANI	
8	Destination information or DNIS	
9	Termination code	

Index	CDR field	Description
10	Call description	
11	Source type	
12	Destination type	
13	Caller identification for the parent call when the transfer occurred	
14	SIP method	Reason for generating Billing CDR; only used for cases unrelated to session.
15	Session ID	SIP request Call-ID which calling/called side network element received. If IMS domain user is the caller, the value in MGCF CDR is the Call-ID in the SIP message MGCF received. If the IMS domain user is the called party, the value in MGCF CDR is the Call-ID in the SIP message MGCF generated.
16	List of calling party address	Address of service requesting party or Session Initiation party (Public User ID or Public Service ID). Calling Party Address from the P-Asserted-Identify header; it can include SIP URL, Tel URL; if there are multiple P-Asserted-Identify, it may include a number of the AVP.
17	Service requested time stamp	Time when network elements receive the SIP request message. Used to indicate the time when service is triggered.
18	Service reason return code	Response status code of success or failure of request in SIP message.
19	Access network information	Used to determine whether the user is roaming.
20	Incoming trunk group ID	Circuit ID used by the incoming PSTN call leg. Must select when CS domain user calls IMS domain user.

Index	CDR field	Description
21	Outgoing trunk group ID	Circuit ID used by the outgoing PSTN call leg. Must select when IMS domain user calls CS domain user.
22	Role of node	Indicates the role of the MGCF network element in this session. Values are: • 0 (Originating Role) - For a call coming in to the MGCF on the IMS domain and going out to the CS domain. • 1 (Terminating Role) - For a call coming in from the CS domain and going to the MGCF on the IMS domain.
23	Originating Inter-operator identifier (Originating IOI)	Indicates the call initiator's network identity. It is set to the originating IOI value in the P-Charging-Vector header of the SIP message.
24	Terminating Inter-operator identifier (Terminating IOI)	Indicates the call destination's network identity. It is set to the terminating IOI value in the P-Charging-Vector header of the SIP message.
25	IMS charging identifier	Billing ID generated by the IMS network elements for SIP session. This ID is a globally unique value. This is set to the IMS charging identifier (ICID) in the P-Charging-Vector header of the SIP message.

4. Telecom configuration file settings

Telecom configuration file

Note: Because the Vision Server is pre-configured at the factory, you should not change the settings described in this section without first consulting Dialogic Technical Services and Support.

This section describes how to fine-tune your existing telecom configuration using the telecom configuration file, *telecom.conf,* which resides in the *vx/callserver/conf* directory.

The telecom configuration file provides information about the boards that interface with the gateway, and about logical elements such as the number of routes and the circuit selection strategy. The telecom configuration file is a text file that contains a list of keywords or attribute value pairs that the gateway interprets at start-up.

The following terms are used for describing the identifiers required by some keywords:

Term	Description
Logical board identifier	Numerical value (starting at 1) that references a signaling or media board in this configuration file, for the purpose of associating the board to other configuration elements.
Physical board index	Numerical value (generally starting at 0) that references the hardware board. For Dialogic® media boards, this is the board index in the *oamsys.cfg* configuration file. For Dialogic® signaling boards, the index can be retrieved or set using the *txcpcfg* utility. For more information about the *oamsys.cfg* file, see the *Dialogic® OAM API Developer's Manual*. For more information about the *txcpcfg* utility, see the *Dialogic® TX Series SS7 Boards TX Utilities Manual*.

When specifying values for a particular setting, use the following syntax unless specified otherwise:

setting_name setting_value

The setting name is case sensitive.

You can define the following types of settings in the *telecom.conf* file:

- ISDN
- ISUP
- BICC
- NbUP
- IPBCP
- SIP
- PSTN
- Video
- Conferencing
- DTMF

- Fax tone recognition
- Media capability
- Trunk and route
- H.100 bus

ISDN settings

Use the ISDN_board keyword or ISDN parameters in the *telecom.conf* file to define ISDN settings for the Vision Server.

ISDN_board keyword

Use the ISDN_board keyword in the *telecom.conf* file to define ISDN settings for the Vision Server.

Syntax

ISDN_board ***sig_board_num board_vendor_id ISDN_type***

Parameter	Description
sig_board_num	Logical signaling board identifier. Valid values: 1 - 8
board_vendor_id	Vendor identifier of the board.
ISDN_type	ISDN protocol variant. Valid values: • ATT_4ESS = AT&T 4ESS • ATT_5E10 = AT&T 5ESS10 • AUSTEL_1 = Australian Telecom 1 • ECMA_QSIG = ECMA 143 QSIG • ETSI = EuroISDN (ETSI ISDN) • HK_TEL = Hong Kong Telecom • KOREAN_OP = Korean operator • NI2 = US National ISDN 2 • NT_DMS = Northern Telecom DMS100 • NT_DMS250 = Northern Telecom DMS250 • NTT = Nippon Telegraph Telephone • TAIWAN_OP = Taiwanese operator • VN6 = France Telecom VN6

Example

The following example enables the EuroISDN protocol on media board 0:

```
# protocol variant = ETSI ISDN (EuroISDN)
ISDN_board 1 0 ETSI
```

Generic ISDN settings

Use the following parameters in the *telecom.conf* file to define generic ISDN settings.

Parameter	Description
isdnInboundCharTranslationTable	Maps characters in the dialed addresses that are received to what is expected in the gateway routing table or CCXML application definition page. Valid values: comma-separated parameter value pair. Only decimals are supported. For example, the following maps the ÿ (0xFF - 255) to the pound key (0x23 – 35): `isdnInboundCharTranslationTable 255=35`
isdnOutboundCharTranslationTable	Maps characters in the dialed addresses to be sent to what is expected to be sent on the ISDN network. Valid values: comma-separated parameter value pair. Only decimals are supported. For example, the following maps the # (0x23 – 35) and * (0x2A – 42) keys to the ÿ (0xFF – 255) key. `isdnOutboundCharTranslationTable 35=255,42=255`

ISUP settings

The keyword used to define ISUP settings depends on whether the Vision Server model uses native ISUP or a Vision Signaling Server for ISUP signaling.

Generic ISUP settings are described following the keyword sections.

VS_isup_server keyword

Use the VS_isup_server keyword to define signaling server settings for ISUP audio and ISUP video models that use Vision Signaling Servers.

Syntax

VS_isup_server ***sig_server_num board_vendor_id ISUP_start_circuit SS7_node_pc SS7_switch_type signaling_and_bearer_type***

Parameter	Description
sig_server_num	Logical signaling server identifier. Valid values: 1 - 2

Parameter	Description
board_vendor_id	Vendor identifier of the signaling server.
ISUP_start_circuit	Starting circuit number of the circuits that the Signaling Server will handle.
SS7_node_pc	SS7 local point code (in decimal).
SS7_switch_type	Switch type (ISUP protocol variant). Valid values: • ANS88 = ANSI 88 (ref. T1.123 – 1988) • ANS92 = ANSI 92 (ref. T1.113 – 1992) • ANS95 = ANSI 95 (ref. T1.113 – 1995) • ANSIBICC = ANSI BICC (T1.BICC.1-2000 to T1.BICC.7-2000) • ETSIV2 = ETSI v2 (ETS 300 356) • ETSIV3 = ETSI v3 (ETS 300 356-1 – 1998) • ITUBICC = ITU-T BICC (Q.1901, Q.1902-6) • ITUBLUE = ITU-T Blue Book • ITUWHITE = ITU-T White Book • ITU97 = ITU-T 1997 • JTTC = JTTC (Japan TTC) • JNTT = JNTT (Japan NTT) • Q767= ITU-T Q.767
signaling_and_bearer_type	Signaling and bearer type that the signaling server uses. Valid values are: • ISUP = ISUP • BICC+IP = BICC with IP bearer channels Default: ISUP

ISUP_board keyword

Use the ISUP_board keyword to define native ISUP settings for the Vision Server.

Syntax

ISUP_board ***sig_board_num board_vendor_id SS7_node_pc switch_type***

Parameter	Description
sig_board_num	Logical signaling board identifier. Valid values: 1 - 8
board_vendor_id	Vendor identifier of the board.
SS7_node_pc	SS7 local point code (in decimal).
switch_type	Switch type (ISUP protocol variant). Valid values: • ANS88 = ANSI 88 (ref. T1.123 – 1988) • ANS92 = ANSI 92 (ref. T1.113 – 1992) • ANS95 = ANSI 95 (ref. T1.113 – 1995) • ETSIV2 = ETSI v2 (ETS 300 356) • ETSIV3 = ETSI v3 (ETS 300 356-1 – 1998) • ITU97 = ITU-T 1997 • ITUWHITE = ITU-T White Book • ITUBLUE = ITU-T Blue Book • JTTC = JTTC (Japan TTC) • JNTT = JNTT (Japan NTT) • Q767= ITU-T Q.767 Default: None. The protocol variant specified here must match that specified in the configuration files associated with the signaling board. For basic ISUP models, the protocol variant must match the variant declared in the signaling board configuration files (*isupcp1.cfg* and *ss7load*). For more information, see the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* or the *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual*.

Example

The following example enables the ETSI v2 ISUP protocol on one signaling board (point code 8000) and the ISUP ITU White Book protocol on another signaling board (point code 9000):

```
# First ISUP board on SS7 point code 8000, linked to an ETSI v2 Switch.
ISUP_board 1 1 8000 ETSIV2 ISUP
# Second ISUP board on SS7 point code 9000, linked to an "ITU White Book" Switch
ISUP_board 2 2 9000 ITUWHITE ISUP
```

Generic ISUP settings

Use the following parameter in *telecom.conf* to define generic ISUP settings.

Name	Description
anmbackwardCallInd	When set, the Call Server initializes the backward call indicator (BCI) information element with the configured value and includes the BCI IE in all outbound ANM messages. Valid values: 0 - 65535

BICC settings

For ISUP models with Vision Signaling Servers configured for BICC, use the BICC_ip_bearer keyword to specify information about IP bearer channels and their dimensioning. The following table describes the parameters for the BICC_ip_bearer keyword that you can change if the defaults do not describe your configuration.

Syntax

BICC_ip_bearer ***sig_board_num*** ***media_board_num*** ***max_circuits*** ***trunk_type***

Parameter	Description
sig_board_num	Logical signaling board identifier, referring to a board previously declared with the VS_isup_server keyword. Valid values: 1-8.
media_board_num	Logical media board identifier, referring to a board previously declared with the Media_board keyword. Valid values: 1-8.
max_circuits	Maximum number of IP bearer channels to be created on that media board. Typically this value should not exceed the number of RAW, G.711, and AMR resources defined on that board.
trunk_type	Type of trunk. Valid values: • E1 (30 media channels) • T1 (23 media channels) • E1RAW (31 media channels) • T1RAW (24 media channels) Default: E1RAW.

NbUP settings

For ISUP models with Vision Signaling Servers configured for BICC, use the NbUP parameters to specify information about NbUP endpoints. The following table describes the NbUP parameters that you can change if the defaults do not describe your configuration.

Parameter	Description
nbupInitTimerDuration	NbUP initialization phase maximum duration in milliseconds. Default: 1000.
nbupInitRetry	Number of retries upon initialization failure. Default: 3.
nbupPayloadId	The payload id of the RTP packets to be sent and received. Default: 123.
nbupFrameDuration	The duration of the media put into each NbUP / RTP packet. Only used for G.711 or H.223 codec. This parameter is ignored for AMR streams since 20 ms frame duration is required. Default: 5.
nbupAudioPreferredCodec	The audio codec to be specified in the APP transport element of the BICC IAM message for audio calls. Valid values: • BCF_NO_CODEC • BCF_CODEC_AMR • BCF_CODEC_AMR2 • BCF_CODEC_G711 Default: BCF_NO_CODEC.
nbupPduType	Specifies whether error detection is performed on the payload. Valid values: • 0: With payload CRC (default) • 1: Without payload CRC Default: 0.
nbupMode	Specifies whether the gateway is the master or slave in the NbUP initialization. Valid values: • 0: slave • 1: master Default: 0.

IPBCP settings

For ISUP models with Vision Signaling Servers configured for BICC, use the IPBCP parameters to specify information about IPBCP negotiation. The following table describes the IPBCP parameters that you can change if the defaults do not describe your configuration.

Parameter	Description
ipbcpSetupVariant	Call flow to use for outbound audio calls. Valid values: 0 : Fast forward tunneling. 1 : Delayed forward tunneling. Default: 1.
ipbcpVideoSetupVariant	Call flow to use for outbound video calls. Valid values: 0 : Fast forward tunneling. 1 : Delayed forward tunneling. Default: 0.

SIP settings

Use the following settings in the *telecom.conf* file to configure SIP settings for the Vision Server:

Settings	Description
SIP_stack keyword	Specifies SIP interfaces and dimensioning.
SIP_config keyword	Changes configuration parameters for a declared SIP interface.
SIP_uas keyword	Defines peer SIP user agents or SIP proxy addresses that determine the platform's load balancing and failover strategy.
SIP_network keyword	Defines a SIP network.
SIP_I_module keyword	Defines settings for the SIP-I interface.
SIP header settings	Defines additional MIME headers for SIP, and includes or excludes the transport parameter from the SIP TO and FROM fields.

Settings	Description
SIP authentication settings	Defines settings for SIP authentication.
RTP-related settings	Group of settings that define RTP parameters.

SIP_stack keyword

The SIP_stack keyword specifies SIP interfaces and their dimensioning.

Syntax

SIP_stack ***interface_num DLL_name*** RTP ***max_RTP_contexts media_board_num***

Parameter	Description
interface_num	Logical interface index, numbering the SIP stack declared in the configuration and starting at 1. Currently, only one SIP stack is supported.
DLL_name	Name of the DLL providing the SIP stack. Default and only value: RV_SIP
RTP ***max_RTP_contexts vocal_board_num***	The RTP keyword specifies the maximum number of SIP contexts with on-board RTP available on the specified media board: • ***max_RTP_contexts*** = Maximum number of SIP contexts with on-board RTP. • ***vocal_board_num*** = Logical media board identifier, which refers to a media board previously declared with the Media_board keyword. Valid values are 1-8. For information about the Media_board keyword, see Media capability settings. Repeat this command sequence for each media board.

Examples

The following example uses the SIP_stack keyword to declare 120 SIP contexts and 120 RTP contexts on board 3:

```
# SIP/RTP interface with 120 SIP contexts and 120 RTP contexts as well, on board #3
SIP_stack 1     RV_SIP    RTP 120 3
```

The following example uses the SIP_stack keyword to declare 240 SIP contexts and 240 RTP contexts on boards 3 and 4. Each board has 120 ports:

```
# SIP/RTP interface with 120 SIP contexts and 120 RTP contexts as well, on board #3
# and board #4 (each having 120 ports)
SIP_stack 1     RV_SIP    RTP 120 3    RTP 120 4
```

SIP_config keyword

For each declared SIP interface, you can change several configuration parameters using the SIP_config keyword.

Syntax

SIP_config ***interface_num parameter value***

Parameter	Description
interface_num	Logical interface index, which refers to a SIP stack previously declared with the SIP_stack keyword.
parameter	(Optional) Name of an interface configuration parameter. ***parameter*** is always paired with ***value***. Repeat the ***parameter value*** sequence for each parameter whose value you want to change.
value	(Optional) Value of an interface configuration parameter specified by ***parameter***. For a description of valid values, see Valid values for parameter.

Valid values for parameter

Valid values for ***parameter*** are:

Parameter	Description
1XXNotif	Processing of SIP provisional responses. Valid values: • true: Forwards provisional SIP responses (1XX) to the Vision Server scripting engine as notifications. • false: Ignores provisional SIP responses (1XX). Default: false
ackNotif	Processing of ACK notifications. Valid values: • true: Forwards ACK messages to the Vision Server scripting engine as notifications. • false: (Default) Handles ACK messages automatically without notifying the call control scripts. Default: false
maxCallLegs	Maximum number of call legs handled in the SIP stack. Default: 512

Parameter	Description
poolNbPages	Number of pages in the memory pool used by the SIP stack. Default: 512
poolPageSize	Size in bytes for the memory pool used by the SIP stack. Default: 1024
recInfoAutoResponse	Processing of INFO messages. Valid values: • true: Sends a 200 OK reply for received SIP INFO requests. • false: Does not send a 200 OK reply for received SIP INFO requests. Default: false
sendReceiveBuffer	Maximum send/receive buffer size for SIP messages, in bytes. Valid values: Integer up to 30720 bytes (30 Kb) Default: 5120 bytes (5 Kb)
setStackToLocalAddress	Indicates whether the SIP stack listens to the local IP address. Valid values: • true: SIP stack listens to the local IP address. • false: SIP stack listens to 0.0.0.0. Default: false
sipLog	Debug traces. Valid values: • true: Generates textual SIP traces for debugging in rv*sipLog.txt*. • false: Does not generate textual SIP traces for debugging in rv*sipLog.txt*. Default: false **Note:** This parameter is intended for debugging purposes and should be used with caution. When SIP logging is enabled, many trace messages are generated, which is generally not suitable for in-service environments. When in-service, using a SIP probe or network analyzer (such as Ethereal) can provide a more suitable alternative.

Parameter	Description
T1	Value of the T1 protocol timer, in milliseconds (ms). Default: 500
tcpLocalPort	Listening port when using TCP as a transport protocol. Default: 5060
transport	Transport protocol for SIP (over TCP or UDP). Valid values: • tcp • udp Default: tcp
transportPersistency	Transport persistency level. When TCP is used as a transport layer, this option sets the transport persistency level to optimize the connections used for transactions. Valid values: • 0: TRANSPORT_PERSISTENCY_LEVEL_TRANSC Connection kept for a transaction. • 1: TRANSPORT_PERSISTENCY_LEVEL_TRANSC_USER Connection kept for all transactions from the same session. Default: 1
updLocalPort	Listening port when using UDP as a transport protocol. Default: 5060

Example

The following example uses the SIP_config keyword to change the specified SIP configuration parameters:

```
SIP_config 1  transport tcp  acknotif true  tcpLocalPort 5060  udpLocalPort 5060
```

SIP_uas keyword

The SIP_uas keyword defines peer SIP user agents or SIP proxy addresses that determine the platform's load balancing and fail-over strategy. Use this keyword for each declared SIP interface.

Syntax

SIP_uas ***interface_num media_mode IP_address***:***port[priority]***

Parameter	Description
interface_num	Logical interface index, referring to a SIP stack previously declared with the SIP_stack keyword.
media_mode	Media control mode supported by the SIP UAs. Valid value: RTP: Standard UAs using SIP/RTP
IP_address	IP address of the peer SIP UAs. The address must be given in traditional quad-dot notation (***www.xxx.yyy.zzz***), optionally specifying a port number. You can use several SIP_uas configuration lines to declare the IP addresses of the peer SIP UAs or SIP proxy servers that the platform tries to reach. If you declare several addresses, the platform uses them for its load balancing and fail-over strategy (random-robin mechanism). You can optionally specify a priority (***P***) between brackets for load-balancing and fail-over: `(0 ≤ P ≤ 65535 with priority(P1) > priority(P2) if P1 < P2)` Proxies with the same priority are subject to the same load balancing and fail-over processes.
port[priority]	(Optional) Port number for SIP on the peer SIP UA with an optional priority for load balancing or fail-over. Default port number: 5060 Valid values for priority: 0 - 65535 Default priority: 0

Example 1: Configuration with one proxy server

The following example configuration declares a SIP proxy server located at 123.123.123.201:5060:

```
# SIP proxy server
SIP_uas 1  RTP  123.123.123.201:5060
```

Example 2: Configuration with two proxy servers for fail-over

In the following example, the first line defines a SIP proxy server located at 123.123.123.201 using the default port 5060. The proxy server has priority 0 (highest priority). The second line defines a SIP proxy server located at 123.123.123.202 using port 5063, with priority 1.

With these settings, the first proxy with priority 0 is always used (no load balancing) as long as it can handle the calls. If it fails, then the second proxy with priority 1 is used instead.

```
# Main SIP proxy server
SIP_uas 1  RTP  123.123.123.201[0]
# Back-up SIP proxy server
SIP_uas 1  RTP  123.123.123.202:5063[1]
```

You can configure two proxies on one single configuration line with a comma as a separator. For example:

```
# SIP proxy servers (main and back-up)
SIP_uas 1  RTP  123.123.123.201[0],123.123.123.202:5063[1]
```

Example 3: Configuration with three proxy servers for fail-over and load balancing

The following example defines a SIP Proxy located at 123.123.123.201 with priority 0. It also defines two other proxies at 123.123.123.202 and 123.123.123.203 respectively, both with priority 1.

```
# SIP proxy servers
SIP_uas 1  RTP  123.123.123.201[0],123.123.123.202:5063[1],123.123.123.203:5063[1]
```

Proxy 123.123.123.201 is always used unless a failure occurs. In that case, the proxy servers 123.123.123.202 and 123.123.123.203 are used randomly since they have the same priority. If proxy 123.123.123.202 then fails, proxy 123.123.123.203 is used instead.

SIP_network keyword

The SIP_network keyword defines a SIP network. A SIP_network keyword is added in the configuration for each SIP network configured in the system.

Syntax

SIP_network ***sip_interface_num network_num network_name signaling_ip_addr board_interface_index media_ip_addr***

Parameter	Description
sip_interface_num	Logical interface index, referring to a SIP stack previously declared with the SIP_stack keyword. Valid values: 1
network_num	Unique logical SIP network identifier. Valid values: 1-8
network_name	Unique SIP network name. Only used for identification purposes.
signaling_ip_addr	Local IP address of the SIP network. Valid values: IP address, any.
board_interface_index	Interface index on the media board for the media used by this SIP network. Valid values: 0-127
media_ip_addr	Reserved for future use. Host IP address for the media used by this SIP network. Valid value: any.

Examples

The following example defines a single SIP network (Default) on any interface and uses interface 0 on the media board.

```
SIP_network 1 1 Default any 0 any
```

The following example defines two SIP networks on a specific IP address. The Default SIP network is bound to 192.168.0.1 host IP address and uses interface 0 on the media board for media processing. The SIP-I network is bound to 192.168.1.1 on the host and uses interface 1 on the media board for media processing.

```
SIP_network 1 1 Default 192.168.0.1 0 any
SIP_network 1 2 SIP-I 192.168.1.1 1 any
```

SIP_I_module keyword

The SIP_I_module keyword defines settings for the SIP-I interface.

Syntax

SIP_I_module 1 **enabled** [true | false] ***(variant display_name)****

where the asterisk (*) indicates that the two parameters are optional and can be repeated more than once.

Parameter	Description
enabled [true | false]	Enable SIP-I interface. Valid values: • true • false Default: false
variant	(Optional) ISUP protocol variant. Valid values: • ansi00 • ansi88 • etsi356 • itu-t88 • itu-t92+ • ttc87 • ttc93+
display_name	(Optional) ISUP protocol variant name that is included in the outbound SIP message in the Content-Type header. Typically the display name should be the same as the ISUP variant.

Example

The following example sets the display name for the itu-t92+ and ansi00 variants to ITU-T97 and ANSI-2000 respectively.

```
SIP_I_module 1 enabled true itu-t92+ ITU-T97 ansi00 ANSI-2000
```

SIP header settings

Use the following setting in the *telecom.conf* file to define additional MIME headers for SIP or to include or exclude the transport parameter from the SIP TO and FROM fields:

Name	Description
sipCustomerHeader	Defines additional MIME headers for SIP. This setting can be repeated, as shown in the following example: sipCustomHeader= Vision-ServiceNumber sipCustomHeader= Vision-Prepare-dialog sipCustomHeader= Vision-InitialURI sipCustomHeader= Vision-ParentCallID

SIP authentication settings

Use the following parameter in the *telecom.conf* file to define SIP authentication settings for the gateway.

Name	Description
sipDigestAuthentication	Defines user name and password pairs for multiple realms. Can also define a default user name and password pair. The format and examples are provided below.

The format for the SIP authentication setting is:

```
sipDigestAuthentication [realm1]:user1:password1,[realmN:userN:passwordN]*
```

where arguments in square brackets are optional and the asterisk indicates 0 or many repetitions of a field.

The syntax allows you to specify pairs of user name and password for multiple realms. It also allows you to define a default user and password. If realm is not specified for the first user name and password pair, it is considered the default user name and password. The default pair will be used when no specific user and password is found for a requested realm.

For example:

```
sipDigestAuthentication :mary:vision123
```

defines a default user *mary* with password *vision123*. These credentials will be used for all authentication requests, regardless of the realm, since the first realm argument has been omitted.

In the following example:

```
sipDigestAuthentication :mary:vision123,dialogic.com:mary:vision789
```

the same default user and password pair is defined, but a specific user and password pair (mary and vision789) is defined for realm dialogic.com.

RTP settings

Use the following settings in the *telecom.conf* file to define RTP settings for the gateway. The syntax for these settings is:

setting_name setting_value

The setting name is case sensitive.

Name	Description
mediaStreamDeactivationMode	Defines the action applied on a media stream at the end of any media over IP request (such as prompt playback). Valid values: • 0: Full release. The DS0 endpoint is returned to its pool and the media stream is disabled, disconnected, and finally destroyed. • 1: Disabled only. The media stream is disabled, but the objects are not destroyed. Voice can no longer be conveyed. • 2: No action. The media stream remains active. Default: 1
mspChnAdaptEnable	Whether the adaptive jitter mode is enabled. When the adaptive jitter mode is enabled, the jitter buffer automatically increases or decreases the number of frames in the jitter depth according to the number of frames received in the previous five seconds. Valid values: • 0: Disabled • 1: Enabled Default: 0
mspChnDecoderDtmfMode	Specifies how DTMF tones are decoded from RTP packets. The value of this parameter is a 16-bit word composed of two 8-bit bytes. The first byte is the control parameter. Valid values for the first byte: • 0: No RFC 2833 decoding. • 1: RFC 2833 decoding. • 3: RFC 2833 decoding with DTMF events sent to the application. The second byte indicates the number of decoder frames generated before stopping, when no-end-of-tone packet is received, Default value: 771 (0x0303) - RFC 2833 enabled, three frames.
mspChnDecoderGain	Decoder gain. Default: 1024

Name	Description
mspChnDecoderMode	Decoder mode. Valid values: • 0: Offline • 1: Online Default: 1
mspChnEncoderDtmfMode	Indicates how to encode DTMF tones into RTP packets. Valid values: • 0: RFC 2833 disabled. • 1: RFC 2833 enabled. DTMF tones are not transmitted as voice data. • 5: Voice enabled. DTMF tones are transmitted both as RFC 2833 packets and in-voice packets. • 9: RFC 2833 is enabled and the encoder shifts the timestamp of associated DTMF packets. Default: 1 **Note:** The payload ID for RFC 2833 compliant in-band packets is automatically defined by SIP/SDP negotiation.
mspChnEncoderGain	Encoder gain. Default: 1024
mspChnEncoderMode	Encoder mode. Valid values: • 0: Offline • 1: Online Default: 1
mspChnEncoderRate	Encoder rate applicable to G.723 only. Valid values: • 0: 6.4 kbit/s • 1: 5.3 kbit/s Default: 0

Name	Description
mspChnJitterDepth	Size in frames of the internal queue maintained by a jitter filter. A jitter filter holds frames in the queue and does not transmit them until it accumulates the number of frames specified by the jitter depth. While holding the frames, the jitter filter transfers null frames with empty payloads. Once the number of frames specified by the jitter depth has accumulated, the output function draws from the queue at the rate defined by the vocoder type. Valid values: Integer > 0 Default: 2
mspChnNotchControl	Enables, disables, or both the DTMF/CED tone suppression filters. Valid values: • 0: DTMF and CED notch filters disabled. • 1: DTMF notch filter enabled, CED notch filter disabled. • 2: DTMF notch filter disabled, CED notch filter enabled. • 3: DTMF and CED notch filters enabled. Default: 0
mspChnVadControl	Enables or disables voice activity detection (VAD) on the media stream. Valid values: • 0: Disabled. • 1: Enabled Silence packets are filtered and are not sent over the network. Comfort noise packets are sent instead. Default: 0
mspRtpTOS	IPv4 ToS (type of service) field specified in the IP header. Valid values: Integer ≥ 0 Default: 0

Name	Description
rfc2833Encoding_IPMode	In case of IP mode (no PSTN leg), set to TRUE if incoming inband DTMF must be encoded into RFC 2833 packets to the destination. Valid values: • TRUE • FALSE Default: FALSE

PSTN settings

Use the following settings in the *telecom.conf* file to define PSTN settings for the Vision Server. The syntax for these settings is:

setting_name setting_value

The setting name is case sensitive.

Setting	Description
csThreadTimeInIdleState	Time interval for which a call thread remains alive after the associated call ends. The call thread is used by another call to avoid the overhead of creating a new thread. A call thread is created for a call when needed, to manage a call's events. Default : 3600 ms (1 hour)
defaultReleaseCause	Release cause returned when no application is triggered by the CCF::IncomingCall notification. For information, see ITU-T recommendation Q.850, section 2.2.5. Default: 31.
timer_connected	Defense timer in connected, trombone, and conference states. Value in seconds or INFINITY. • ***n*** seconds • INFINITY Default: INFINITY.
timer_incomingCall	Time interval within which the gateway answers an incoming call. If the gateway cannot answer the call within the specified interval, it rejects the call. This prevents gateway timeslots from getting busy with inbound calls that did not get connected or rejected. • ***n*** seconds • INFINITY Default: 180.

Setting	Description
timer_placingCall	Waiting time of the answer to an outgoing call placed by the platform. Valid values: ***n*** seconds or INFINITY Default: 120.
timer_releaseConfirm	Waiting time of the release confirmation when a release message is sent by the platform. Valid values: • ***n*** seconds • INFINITY Default: 10

Video settings

Use the following settings in the *telecom.conf* file to define video settings for the Vision Server. The syntax for these settings is:

setting_name setting_value

The setting name is case sensitive.

Name	Description
amrModeChoice	AMR codec mode. Valid values: • 0: MR475 (4.75 kbit/s) • 1: MR515 (5.15 kbit/s) • 2: MR59 (5.90 kbit/s) • 3: MR67 (6.70 kbit/s) • 4: MR74 (7.40 kbit/s) • 5: MR795 (7.95 kbit/s) • 6: MR102 (10.20 kbit/s) • 7: MR122 (12.20 kbit/s) Default: 7

Name	Description
enableVideoTranscoding	Indicates whether video transcoder resources are available for the Vision Server. Valid values: • TRUE: If enabled, video transcoding is inserted in the video path if the Vision Server finds incompatible video codec characteristics between call legs. • FALSE: If disabled, the call is dropped if the Vision Server finds incompatible video codec characteristics. Default: FALSE
h263CapabilityMaxBitRate	Gateway terminal capability that determines the maximum bit rate for video outbound to a 3G network, in bit/s. Valid values: Integer > 0 Default: 43000
h263ModeBitRate	Gateway terminal capability that determines the bit rate for video in bit/s. Valid values: Integer > 0 Default: 43000
h324ResponseTimeout	Delay (in seconds) to wait for the H324EVN_MEDIA_SETUP_DONE event during H.324 negotiation. A video call is terminated when a timeout occurs. Valid values: Integer > 0 Default: 30
h324TraceLevel	Trace level for the Dialogic H.324 stack. The trace level is a bit mask. For more information, see the Dialogic® Video Access *3G-324M Interface Developer's Reference Manual*. Valid values include: • 0: No traces. • ALL (0xFFFFF): Full traces. • ERRORS_ONLY (0xC1084): Errors only. Default: ERRORS_ONLY

Name	Description
h324VideoOverAL2	Indicates video support for adaptation layer 2. Although most 3G terminals support video over AL2, using this parameter decreases the duration of H.245 negotiation. Valid values: • 0: false (video over AL3) • 1: true (video over AL2) Default: 1
mspAudioGwRtpDtmfControl	Indicates whether to use DTMF detection when audio transcoding. Valid values: • 0: Disabled • 3: Enabled Default: 3
mspAudioGwRtpFrameQuota	Number of frames per packet generated by the audio gateway RTP endpoint. Valid values: Integer > 0 Default: 2 Set mspAudioGwRtpFrameQuota as follows: • If the destination end point uses the AMR codec, set mspAudioGwRtpFrameQuota to 1. • If the destination end point uses the G.711 codec, set mspAudioGwRtpFrameQuota to 2.
mspVideoChnAdaptEnable	Enables or disables adaptive jitter. Valid values: • 0: Disabled • 1: Enabled Default: 0
mspVideoChnJitterDepth	Size in frames of the internal queue maintained by a jitter filter. Valid values: Integer > 0 Default: 2

Name	Description
mspVideoGwRtpFrameQuota	Number of frames per packet generated by the video gateway RTP endpoint. Valid values: Integer > 0 Default: 2
mspVideoOutOfBandDCI	Indicates what in-band DCI to replace with out-of-band DCI. Valid values: • 0 - No change. • 1 - Uses out-of-band DCI to replace only the initial DCI at the beginning of the bit stream. • 2 - Replaces all in-band DCI with the one received out-of-band. • 3 - Same as value 2, but additionally inserts one out-of-band DCI before every Iframe. Default: 2

Conferencing settings

Use the following settings in the *telecom.conf* file to define conferencing settings for all models of the Vision Server except for IP-only. The syntax for these settings is:

setting_name setting_value

The setting name is case sensitive.

Name	Description and default value
cnfActiveTalkers	Number of simultaneous active talkers. Valid values: 1 - 62 Default: 3
cnfECGain	Echo cancellation gain. Valid values: -54 - 24 Default: 0
cnfECPredelay	Echo cancellation pre-delay. Valid values: 0 - 9 Default: 0

Name	Description and default value
cnfEnterToneAmpl1	Amplitude of the first enter tone when someone joins a conference. Valid values: -54 - 3 Default: -20
cnfEnterToneAmpl2	Amplitude of the second enter tone when someone joins a conference. Valid values: -54 - 3 Default: -20
cnfEnterToneFreq1	Frequency of the first enter tone when someone joins a conference. Valid values: 200 - 3600 Default: 1000
cnfEnterToneFreq2	Frequency of the second enter tone when someone joins a conference. Valid values: • 0: Single frequency • 200 - 3600 Default: 500
cnfEnterToneIterations	Number of iterations for the enter tone. Valid values: 1..32767 Default: 2
cnfEnterToneOffTime	Amount of time in ms that the Call Server waits between generating enter tones. Valid values: 0 - 65535 Default: 200
cnfEnterToneOnTime	Duration in ms for an enter tone generated by the Call Server. Valid values: 0 - 65535 Default: 200
cnfExitToneAmpl1	Amplitude of the first exit tone when someone exits a conference. Valid values: -54 - 3 Default: -20

Name	Description and default value
cnfExitToneAmpl2	Amplitude of the second exit tone when someone exits a conference. Valid values: -54 - 3 Default: 0
cnfExitToneFreq1	Frequency of the first exit tone when someone exits a conference. Valid values: 200 - 3600 Default: 300
cnfExitToneFreq2	Frequency of the second exit tone when someone exits a conference. Valid values: • 0: Single frequency • 200 - 3600 Default: 0
cnfExitToneIterations	Number of iterations for the exit tone. Valid values: 1 - 32767 Default: 2
cnfExitToneOfftime	Duration in ms for an exit tone generated by the Call Server. Valid values: 0 - 65535 Default: 0
cnfExitToneOnTime	Minimum amount of time in ms that a tone should be on to be considered an exit tone: Valid values: 0 - 65535 Default: 200
cnfInputAGCSilenceAmpl	Silence amplitude input automatic gain control. Valid values: -45 - 0 Default: -40
cnfInputAGCTargetAmpl	Target amplitude input automatic gain control. Valid values: -45 - 0 Default: -19

Name	Description and default value
cnfInputGain	Input gain. Valid values: -12 - 12 Default: 0
cnfNumLoudest	Number of loudest speakers. Valid values: 1 - 62 Default: 3
cnfOutputAGCSilenceAmpl	Silence amplitude output automatic gain control. Valid values:-45 - 0 Default: -40
cnfOutputAGCTargetAmpl	Target amplitude output automatic gain control. Valid values:-45 - 0 Default: -19
cnfOutputGain	Output gain. Valid values: -12 - 12 Default: 0

DTMF settings

Use the following settings in the *telecom.conf* file to determine how the Call Server handles DTMF signaling:

Name	Description
acceptInboundDTMFInSIPINFO	Indicates whether the Call Server can accept an incoming SIP INFO message with DTMF content. • TRUE: Call Server can accept an incoming SIP INFO message with DTMF content when RFC 2833 is not negotiated. • FALSE: Call Server cannot accept an incoming SIP INFO message with DTMF content. It rejects the message by issuing a 415 Media not supported response. Default: FALSE

Name	Description
biccOutboundDtmf	Indicates whether the Call Server transmits DTMF out of band in voice-only calls using the BICC protocol. • INBAND: Transmit DTMF in the media stream. • OOB: Transmit DTMF out of band. • INBAND_AND_OOB: Transmit DTMF out of band and inserted in the media stream. Default: OOB
biccOutOfBandDtmfNotification	Indicates whether to request notification in out-of-band DTMF events generated by the Call Server. Only use if biccOutboundDtmf is set to OOB and if processing a voice-only BICC call. • TRUE: Signal event generated by the Call Server requests notification; that is, the action indicator in the Application Transport Message (APM) is set to *Start signal, notify*. • FALSE: Signal event generated by the Call Server does not request notification; that is, the action indicator in the APM is set to *Start signal, no notify*. Default: TRUE
biccInboundDtmf	Indicates whether the Call Server detects in-band DTMF in the BICC to SIP direction when processing voice-only BICC calls: • INBAND: Detect and process in-band DTMF only. • OOB: Detect and process out-of-band DTMF only. • INBAND_AND_OOB: Detect and process both in-band and out-of-band DTMF. Default: INBAND_AND_OOB

Name	Description
detectInbandDtmfInVideoCalls	Determines whether the Call Server detects in-band DTMF in mobile video calls. • 0: Does not detect in-band DTMF in mobile video calls. • 1: Detects in-band DTMF when audio transcoding is required. • 2: Always detects in-band DTMF. This forces audio transcoding, even if the audio codecs on both call legs are the same. Default: 1 This setting is useful for supporting handsets that do not support User Input Indication (UII). **Note:** Detecting in-band DTMF uses additional DS0 resources, because each transcoded channel requires two DS0 resources.
dtmfEventDuration	Duration of an RFC 2833 DTMF event, in ms. The Call Server generates a DTMF event after it receives an H.245 User Input Indication message containing a DTMF from a 3G network. Default: 80 for audio setup; 300 for video setup
dtmfEventNotification	Whether to notify the CCXML engine that a DTMF was received on a call leg. • TRUE: Notify the CCXML engine. • FALSE: Do not notify the CCXML engine. Default: FALSE
outboundDTMFConfig	Determines how the Call Server sends an outbound DTMF. • 0: Call Server always sends an outbound DTMF according to RFC 2833, if the other party supports RFC 2833. • 1: Call Server always sends an outbound DTMF in a SIP INFO message. • 2: Call Server sends an outbound DTMF according to RFC 2833, if the other party supports RFC 2833. If the other party does not support RFC 2833, the Call Server sends the DTMF in a SIP INFO message. Default: 2

Fax tone recognition settings

Use the following settings in the *telecom.conf* file to specify the minimum amplitude and duration of tones that the Call Server can recognize as fax CNG (calling) tones. The syntax for these settings is:

setting_name setting_value

The setting name is case sensitive.

Name	Description
faxToneCNGAmpl	Minimum amplitude of a tone in decibels that the Call Server can recognize as a fax CNG tone. Valid values: 54 to -3 Default: 28
faxToneCNGDuration	Minimum duration of a tone in ms that the Call Server can recognize as a fax CNG tone. Valid values: Any integer Default: 300

Note: To detect a fax call, the DetectFaxToneCNG setting in the *callserver.conf* file must be set to TRUE. For more information, see Fax call detection settings.

Media capability settings

Use the following keywords in the *telecom.conf* file to configure media capabilities and media streams:

Keyword	Description
Media_board	Declares a media board for media processing functions.
Resource	Declares the number and type of resources that Call Server applications can use.

Media stream channel types

A media stream can consist of one of the following channel types:

- Generic voice channel
- RTP bridge channel

Generic voice channel

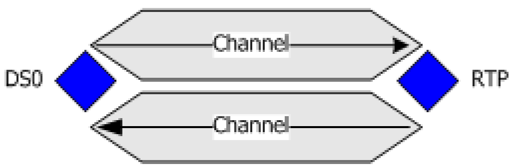
A generic voice channel encodes voice coming from a circuit-switched (PSTN) channel or from a DSP into RTP packets. It also decodes RTP packets and sends the information to the PSTN or to a DSP.

The following illustration shows a generic voice channel that connects a DS0 endpoint to an RTP endpoint:

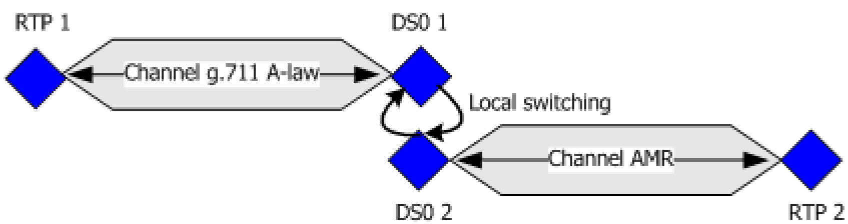
RTP bridge

An RTP bridge consists of two unidirectional channels that create an RTP connection between two RTP endpoints.

The following illustration shows an RTP bridge media stream that does not use audio transcoding:



The following illustration shows an RTP bridge media stream that transcodes the audio from AMR to G.711 A-law. In this situation, two DS0 resources must be declared:



Media_board keyword

Use media board resources for media processing functions such as audio playback and recording, 3G video capabilities, and voice codecs. To define a media resource board for the Call Server, use the Media_board keyword according to the following syntax:

Media_board ***board_num*** ***board_vendor_id*** ***port_base*** ***ip_address***

Parameters	Description
board_num	Logical media board identifier. Valid values: 1 - 8
board_vendor_id	Vendor identifier of the board.
port_base	(Optional) First port number for RTP channels. Other port values are equal to ***port_base*** + 2 x ***N***, where ***N***= 0..***nb_resources***–1). Default: 10000 For information about ***nb-resources***, see the Resource keyword.

Parameters	Description
ip_address	(Optional) IP address of the media board.

Example

The following example defines a media board that uses RTP as board number 1:

```
Media_board  1 0 2000
```

Resource keyword

After you define a media board, you can dimension resources, depending on their intended use. To specify the number and type of media resources that the Call Server uses, use the Resource keyword according to the following syntax:

Resource ***media_board_num resource_type nb_resources***

Parameter	Description
media_board_num	Logical media board identifier. Refers to a board already declared with the Media_board keyword. Valid values: 1 - 8
resource_type	Resource type on the media board that the Call Server uses. The factory sets these parameters. Do not change the settings without contacting Dialogic Technical Services and Support. Valid values: • AMR - AMR resources. • CNF - Conferencing resources. • G711 - G.711 resources. • G723 - G.723 resources. • G726 - G.726 resources. • G729 - G.729A resources. • RAW - Clear channel resource required for IP-324M. • RTP_BRIDGE - Media stream resources used in VoIP/VoIP gateways without RTP switching or with transcoding. • VIDEO_BRIDGE - Resources used for video-over-IP/video-over-IP-gateways without RTP switching or with audio transcoding. • VIDEO_GW - Resources used for video/3G processing, such as playback and recording.
nb_resources	Number of resources. The value of this field depends on the media board's specifications.

You can add multiple media resources as needed. However, the declaration must follow the circuit order, as shown in the following examples.

Example 1

The following example declares media resources for a media board with 120 G.711 ports that uses France Telecom VN6 signaling.

```
# ISDN signaling board
ISDN_board 1 0 VN6
# Media board
Media_board 2 0
# 120 voice channels
Resource 2 G711 120
.
.
.
```

Example 2

The following example declares media resources for a media board with 120 G.723 ports and 120 RTP bridge resources.

```
# Media board
Media_board 2 0
# 120 voice channels
Resource 2 G723 120
# 120 RTP switching channel
Resource 2 RTP_BRIDGE 120
.
.
.
```

Example 3

The following example declares media resources for a media board with 60 G.711 ports and 30 G.729 ports. The media board uses France Telecom VN6 signaling.

```
# ISDN Signaling board
ISDN_board 1 0 VN6
# Media board
Media_board 2 0 10000
# 60 predefined G.711 ports + 30 predefined G.729 ports
Resource 2 G711 60
Resource 2 G729 30
.
.
.
```

Trunk and route settings

A trunk is a connection port on a board that provides the physical path for transferring voice and signaling data to the PSTN network. Typical trunk types include:

Trunk type	Mode	Capacity
E1	PRI	30 voice timeslots, one signaling timeslot, and one synchronization timeslot
	RAW	31 voice timeslots and one synchronization timeslot
T1	PRI	23 voice timeslots and one signaling timeslot
	RAW	24 voice timeslots

Use the following keywords in the *telecom.conf* file to specify the PSTN trunks and routes used by the Call Server:

- Trunk
- Route

Trunk keyword

Declares the trunks used by the Call Server.

Syntax

Trunk ***trunk_num*** ***sig_board_num*** ***media_board_num*** ***trunk_vendor_id*** ***trunk_type*** ***isdn_equipment*** -r ***channel_reserved***

Parameter	Description
trunk_num	Unique logical trunk identifier. Valid values: 1 - 128
sig_board_num	Logical signaling board identifier, referring to a board previously declared with the ISDN_board or ISUP_board keywords. Valid values: 1 - 8
media_board_num	Logical media board identifier, referring to a board previously declared with the Media_board keyword. Valid values: 1 - 8
trunk_vendor_id	Vendor identifier of the trunk. Valid values: 0-15
trunk_type	(Optional) Type of trunk. Valid values: • E1 (30 media channels • T1 (23 media channels) • E1RAW (31 media channels) • T1RAW (24 media channels) Default: E1
isdn_equipment	(ISDN models only) Mandatory. Represents the type of equipment connected to the trunk. Valid values: • NT: Network equipment • TE: Terminal equipment Default: TE

Parameter	Description
-r ***channel_reserved***	(Optional) Reserved channel. Use the -r switch to declare a voice channel as unusable for audio purposes; for example, when signaling uses a voice timeslot. The value corresponds to the voice timeslot to reserve. For example, -r22 reserves voice timeslot 22 for signaling. Channels are numbered from 0 to ((maximum number of voice timeslots) - 1). The maximum number of voice timeslots depends on the type of trunk specified by the ***trunk_type*** parameter. Valid values for T1 trunks: 0 - 23 Valid values for E1 trunks: 0 - 30

Example

The following example configures four trunks. Trunks 1 and 2 are ISUP trunks in RAW mode, with channel 15 reserved for signaling. Trunks 3 and 4 are ISDN trunks in PRI mode:

```
# Trunks 1 and 2 are ISUP trunks in RAW mode
# They are linked to signaling board #1 and media board #2
# Vocal timeslot 16 (channel 15) is blocked
Trunk 1 1 2 0 E1RAW -r15
Trunk 2 1 2 1 E1RAW

# Trunks 3 and 4 are ISDN trunks in PRI mode
Trunk 3 1 2 2 E1RAW TE
Trunk 4 1 2 3 E1RAW TE
```

Route keyword

A route is a logical collection of trunks. For each trunk in the route, enter a line in the *telecom.conf* file that specifies the route for that trunk. Use the Route keyword to specify a route.

Syntax

Route ***route_num trunk_num select_strategy...trunk_num select_strategy***

Parameter	Description
route_num	Route identifier. Valid values: 1 - 16
trunk_num	Trunk identifier, referring to a previously declared trunk (using the Trunk keyword). Valid values: 1 - 128

Parameter	Description
select_strategy	(Optional) Circuit selection strategy. This parameter defines which circuits of the route are reserved for outgoing calls, and how they are selected. Circuits are identified by their circuit code identifier (CCI), an integer ranging from 1 to the highest circuit code identifier (CCI max). For example, if the route contains four E1 trunks (each containing 30 circuits), circuits are numbered from 1 to 120. The value of CCI max is 120. Valid values: • FROM_TOP • FROM_BOTTOM • DESCENDING • ASCENDING • TIMER (default) Values for ***select_strategy*** are described in the following table.

Valid values for the ***select_strategy*** parameter are:

Value	Description
FROM_TOP	Selects the first idle circuit in decreasing CCI order. This strategy always selects the highest available circuit. **Example** With this circuit selection strategy, a series of calls might be placed as follows: a. A first call is placed on the last circuit, CCI max. b. A second call is placed on circuit (CCI max – 1), because CCI max is busy processing the first call. c. The first call terminates, so CCI max becomes idle. d. A third call is placed on CCI max, because CCI max is now available. e. A fourth call is placed on (CCI max - 2), because both CCI max and (CCI max - 1) are busy processing calls 3 and 2, respectively.

Value	Description
FROM_BOTTOM	Selects the first idle circuit in increasing CCI order. This strategy always selects the lowest available circuit. **Example** With this circuit selection strategy, a series of calls might be placed as follows: a. A first call is placed on the first circuit, CCI 1. b. A second call is placed on the second circuit, CCI 2, because the first circuit is busy processing the first call. c. The first call terminates, so the first circuit becomes idle. d. A third call is placed on CCI 1, because CCI 1 is now available. e. A fourth call is placed on CCI 3, because CCI 1 and CCI 2 are busy processing calls 3 and 2, respectively.
DESCENDING	Selects a circuit by rotating circuits in decreasing CCI order, from the highest circuit (CCI max) down to the middle of the route ((CCI max / 2) + 1). If no circuit is idle on the second half of the route, a circuit on the first half of the route is selected. **Example** With this circuit selection strategy, a series of calls might be placed as follows: a. A first call is placed on the last circuit, CCI max. b. A second call is placed on (CCI max – 1). c. The first call terminates, so CCI max becomes idle. d. A third call is placed on (CCI max – 2). e. For each subsequent call, the next lower circuit is selected up to the middle of the route. When the last circuit in the half route is reached ((CCI max / 2) + 1)), the selection strategy rotates back to the last circuit CCI max, because that is the first available circuit in decreasing order of CCI.

Value	Description
ASCENDING	Selects a circuit by rotating circuits in increasing CCI order, from the first circuit (CCI 1) up to the middle of the route (CCI max / 2). If no circuit is idle on the first half of the route, a circuit on the second half is selected. **Example** With this circuit selection strategy, a series of calls might be placed as follows: a. A first call is placed on the first circuit, CCI 1. b. A second call is placed on the second circuit, CCI 2. c. The first call terminates, so the first circuit becomes idle. d. A third call is placed on CCI 3. e. For each subsequent call, the next higher circuit is selected, up to the middle of the route. When the last circuit in the half route is reached (CCI max / 2), the selection strategy rotates back to CCI 1, because that is the first available circuit in increasing order of CCI.
TIMER	(Default) The selected circuit is the one on which the inactivity timer is the most important. At the beginning, all circuits have the same inactivity timer. The circuits are selected in decreasing CCI order, starting from CCI max down to CCI 1. When all circuits have been used once, they are selected by the inactivity timer.

Example

The following example shows how to configure two ISDN routes on four trunks. One route contains trunk 2 with circuit selection strategy DESCENDING. The other route contains the three remaining trunks with the default circuit selection strategy.

```
# Four trunks (e.g. on a CG board with four E1s)
Trunk 1 1 2 0 E1 NT
Trunk 2 1 2 1 E1 NT
Trunk 3 1 2 2 E1 NT
Trunk 4 1 2 3 E1 NT
# Route 1 contains trunk 2 - selection strategy = DESCENDING
Route 1 2 DESCENDING
# Route 2 contains trunks 1, 3 and 4 - default selection strategy
Route 2 1 3 4
```

H.100 bus settings

The H.100 bus is an interoperable superset of H-MVIP and MVIP that transports telephony voice data, signaling data, and switching information across PCI boards. If your Vision Server includes SS7, then you can use an H.100 bus to connect a signaling board to a media board. You do not typically need to change settings in the Call Server configuration file to use the H.110 bus.

When a timeslot on a specific stream is used to transmit signaling data between these two boards, the timeslot must be blocked for voice data. You can accomplish this by using the H100 keyword in the *telecom.conf* file.

Note: The Vision Server includes an H.100 clock manager that synchronizes the server's boards. For more information, see the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* or the *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual*.

Syntax

H100 ***stream_1***:***slot_1*** ***stream_2***:***slot_2*** ... ***stream_n***:***slot_n***

Parameter	Description
stream:***slot***	Stream and timeslot pair on the bus. Limited by the bus capacity.

Example

The following example uses the H100 keyword to reserve timeslot 31 on streams 16 and 17 for signaling:

```
# Voice data forbidden on the following bus timeslots:
H100  16:31  17:31
```

5. Administrative tasks

Call Server logging

The Call Server creates and saves log files in the *vx/callserver/logs* directory. Each log file records information about active Call Server processes.

Call Server log files are named according to the following convention:

*callserver_**yyyy_mm_dd_**[***index***].log*

where ***index*** is an integer specifying the current incremented system log file. This value is reset daily and incremented when either the configured maximum system log file size is reached or when the Call Server is restarted. In each case, a new system log file is started.

To set the maximum system log file size, use the SystemLogFileMaxNumber log file setting. For more information, see Logging settings.

Log file format

The format of each log message is:

mm/***dd***/***yy hh***:***mm:ss.ms*** [***severity***] [***origin***:***code***] [***UID***:***threadID***] (***alarm***) [***message***]

For example, a telecom configuration error can lead to the following log file entry:

```
03/29/05 06:59:25.306 [MAJOR] [telecom.pkg:111] [-:1044] (ConfigurationError) - A trunk
is declared with an unavailable protocol, line 16.
```

The timestamp is in the same time format as the local Call Server. Use the SystemLogTime setting to change the time format to Greenwich Mean Time (GMT). For more information, see Logging settings.

The following table describes the fields in a log file:

Field	Description
severity	Severity of the log message. Valid values, in order of decreasing severity and increasing verbosity: • FATAL ERROR (a severe malfunction from which the Call Server cannot recover) • ERROR • WARNING • INFO1 • INFO2 • INFO3 • INFO4 • INFO5 (highest level of detail) Set up logging in either of the following ways: • Use the Vision Console. For information, see the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* or the *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual*. • Set the parameters in the Call Server configuration file. For information, see Logging settings.
origin	Call Server component to which the log message refers.
code	Trace identifier of the message in the Call Server component to which the log message refers.
UID	Reserved for future use. The UID is represented by a hyphen (-) in the log file.

Field	Description
alarm	Optional field that is included only when the log message refers to an alarm notification. In this situation, the field describes the general category of the alarm. Alarm categories include the following: • Started • Quiesced • Shutdown • LicenseCheck • ConfigurationError • InitializationError • SoftwareException • InternalError • ResourceLimitation • CommunicationFailure • ProcessingFailure • InvalidArgument • UnexpectedEvent • NotificationDiscarded • Watchdog • Timeout
threadID	Identifier associated with the thread that generated the message. Use this field to track the progress of a single session or call when several requests are processed simultaneously.
message	Text description of the logged occurrence.

Starting and stopping the Call Server

The Call Server starts automatically when the Vision Server starts. You do not need to start and stop the Call Server manually, unless you need to troubleshoot the Vision Server. Use the Vision Console to start and stop the Call Server manually, as described in the *Dialogic® Vision™ 1000 Video Gateway Administration Manual* and the *Dialogic® Vision™ 1000 Programmable Media Platform User's Manual*.

6. Glossary

A

ADTCP: An audio driver that provides a TCP interface to MIOSIP for rendering SSML fragments.

AMR: Adaptive multi-rate; an audio data compression scheme optimized for speech coding. This scheme was adopted by 3GPP and is used in video services.

ASR: Automatic speech recognition; ASR resources, called ASR engines in the MRCP framework, typically enable users of information systems to speak entries rather than punching numbers on a keypad. See also MRCP.

Authorization and Usage Indication interface: XML-over-HTTP mechanism that authorizes call sessions and gathers information for call detail reports.

B

blind transfer: A call transfer in which the originating caller is not announced and is connected directly to destination. In a blind transfer the Vision Server redirects the caller to the callee without remaining in the connection and does not monitor the outcome.

bridge transfer: A blind transfer in which the Vision Server redirects the caller to the callee and remains as a listener.

C

Call Server: Component of the Vision Server that manages call control and routing capabilities.

CCXML: Call Control Extensible Markup Language; a W3C Working Draft standard language for providing telephony call control support for dialog systems, gateways, and conferencing services.

CCXML application definition file: A file that maps individual CCXML applications to number ranges that trigger the execution of those applications.

clock: A periodic reference signal used for synchronization on a transmission facility, such as a telephony bus. See also clock master, clock slave, clock fallback.

clock master: A board that drives the clock signal for a system of boards connected by a bus cable. See also clock slave.

clock slave: A board that derives its clock signal from a bus cable; the clock signal is driven by the bus clock master. See also clock master.

consultation transfer: A call transfer in which the Vision Server initiates a transfer between two parties, but does not stay attached to the call once it is successfully established. The caller remains connected to the Vision Server if the transfer fails.

D

DTMF: Dual tone multi frequency; an inband signaling system that uses two simultaneous voiceband tones for dialing. Also called touchtone. Some times DMTF is used to generally describe any telephony keypad press, even if tones are not generated.

G

G.711: An ITU PCM encoder/decoder specification for mu-law and A-law encoding.

H

H.100 bus: A TDM telephony bus standard for integrating hardware from various PC board vendors. The H.100 specification defines a ribbon cable bus that transports telephony voice data and signaling data across PCI boards. The H.100 bus is an interoperable superset of the H-MVIP and MVIP-90 telephony buses.

H.223: A protocol used to multiplex control and audio and video media on and off of a single DS0 within a trunk.

H.263: An ITU video compression standard. H.263 supports CIF, QCIF, SQCIF, 4CIF and 16CIF resolutions.

H.264: An ITU and ISO video compression standard that compresses video into lower bandwidth compared to H.263 and MPEG-4. H.264 is also called MPEG-4 Part 10.

I

INAP: Intelligent Network Application Part; an SS7 protocol that facilitates building platform-independent, transport-independent, and vendor-independent applications. Such applications include service switching points (SSPs), internet protocol (IP) applications, service control points (SCPs), enhanced services platforms, service circuit nodes, and other custom applications.

ISDN: Integrated services digital network; a standard for providing voice and data telephone service with all digital transmission and message-based signaling.

ISUP: ISDN user part; the SS7 protocol layer that allows for the establishment, supervision, and clearing of circuit-switched connections between two SS7 signaling points, such as central office switches. Despite its name, the ISUP layer is not unique to interconnecting. It is used to manage all types of circuit-switched connections.

ITU: International Telecommunications Union; an international standards body for telecommunications.

IVR: Interactive voice response; a telephony application in which callers interact with programs using recorded or synthesized voice prompts, DTMF digits, or speech recognition to query or deliver information.

M

Media Resource Function: Component of the Programmable Media Platform that provides media processing including record, playback, and interfaces to speech recognition resources. The Media Resource Function is implemented by MIOSIP.

MIB: Management information base; an SNMP collection of objects that represent a managed node. Physically, a list of variables. Logically, a table with rows of variables.

MIOSIP: Implements the Media Resource Function of the Programmable Media Platform. MIOSIP provides SIP call control, media processing over RTP, DTMF generation and recognition, and an MRCP client to automatic speech recognition (ASR) resources.

MPEG-4: An ISO/IEC standard for compressing multimedia data (video, audio, and speech).

MRCP: Media Resource Control Protocol; an application protocol for implementing automatic speech recognition (ASR) and text-to-speech services (TTS). MRCP provides a distributed system of ASR and TTS engines connected over an IP network.

MTP: Message transfer part; the SS7 protocol layers responsible for the reliable, in-sequence delivery of packets between two SS7 signaling points. The MTP functions include message routing, signaling link management, signaling route management, and congestion control.

MVIP-95: Device driver specification for H-MVIP, H.100, and H.110 telephony buses.

N

NETANN: Basic Network Media Services with SIP; an interface that enables applications in a SIP network to locate and invoke basic services on a media server. These services include network announcements, user interaction, and conferencing services. Also called RFC 4240.

O

OSP: Open Settlement Protocol; a European Telecommunications Standards Institute (ESTI) protocol used to exchange authorization, accounting, and usage information for IP telephony.

P

PSTN: Public switched telephone network; a public telephone network.

R

route: A connection path. On the PSTN network, a route is a logical collection of trunks. On the IP network, a route is a destination URL.

RTP: Real time transport protocol; a layer added to the internet protocol (IP) that addressed problems caused when real-time interactive exchanges (such as audio data) are conducted over lines designed to carry packet-switched (connectionless) data.

S

SCCP: Signaling connection control part; an SS7 protocol that provides both connection-oriented and connectionless data transfer over an SS7 network. It extends the service provided by the SS7 MTP layers by adding extended addressing capabilities and multiple classes of service. The SCCP addressing capabilities allow a message to be addressed to an individual application or database within a signaling point. See also SS7.

SDP: Session description protocol, a protocol that defines a text-based format for describing streaming media sessions and multicast transmissions.

Signaling Server: An optional component of the Vision Server that provides redundant and scalable ISUP signaling.

SIP: Session initiation protocol. An IP signaling and telephony control protocol used mainly for voice over IP calls and multimedia communications. SIP relies on the session description protocol (SDP) for session description and the Real Time Transport Protocol (RTP) for actual transport.

SRGS: Speech Recognition Grammar Specification (SRGS); a syntax for representing the grammars used in speech recognition.

SS7: Signaling system 7; an out-of-band signaling system that provides fast call setup using circuit-switched connections and transaction capabilities for remote database interactions.

SSML: Speech Synthesis Markup Language; a proposed standard for enabling access to the internet using speech. SSML provides a standard way to control various aspects of speech (such as pronunciation, volume, pitch, and rate) over a variety of platforms.

SSML Processor: Component of the Programmable Media Platform that processes SSML requests for audio and text-to-speech.

T

T.38 fax: A standard for real-time fax over IP that makes it possible for fax machines from different vendors to talk to each other over IP networks. The T.38 standard defines how to conduct group 3 facsimile transmission between terminals in which a portion of the transmission path between terminals includes (besides the PSTN or ISDN) an IP network such as the internet.

TCAP: Transaction capabilities application part; an SS7 protocol that provides applications with transaction support over the SS7 network. It enables the exchange of non-circuit related data, such as database queries and responses and remote feature invocation requests between SS7 signaling points. The TCAP layer relies on both the MTP and SCCP layers for message addressing and delivery.

TDM: Time division multiplexing; a technique for transmitting a number of separate data, voice, or video signals simultaneously over one communications medium by quickly interleaving a piece of each signal one after another.

telecom configuration file: File that provides information about the resources that interface with the Call Server and about other elements, such as the number of routes and the circuit selection.

trunk: The physical interface between the telephone network and the Vision Server. In telephone networks, a trunk is a shared connection between two switches. It differs from a line in that it is not dedicated to one subscriber or extension. T1 and E1 trunks carry 24 and 31 circuits, respectively.

TTS: Text-to-speech; a system that converts written language to speech.

V

Vision Console: Web-based configuration tool that configures the Vision Server.

VoiceXML: Voice Extensible Markup Language; a language that enables users to interact with the internet through voice recognition technology.

VoiceXML application configuration file: A file that maps individual VoiceXML applications to number ranges that trigger the execution of those applications.

VoiceXML Interpreter: Component of the Programmable Media Platform that interprets VoiceXML dialogs.

VoiceXML Subsystem: Component of the Programmable Media Platform that provides media processing for VoiceXML applications. The VoiceXML Subsystem consists of the VoiceXML Interpreter, SSML Processor, and Media Resource Function.

7. Index