



IBM WebSphere Voice Server with ViaVoice™ Technology Administrator's Guide

Version 1.0

Printed in the USA

Note:

Before using this information and the product it supports, read the general information in “Notices” on page 89.

First Edition (October 2000)

This edition applies to version 1, release 0, modification 0 of the IBM WebSphere Voice Server and to all subsequent releases and modifications until otherwise indicated in new editions.

©Copyright International Business Machines Corporation 2000. All Rights Reserved.
Note to U.S. Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP
Schedule Contract with IBM Corp.

Contents

About this Book	11	
Who Should Read This Book	11	
Related Publications	11	
How This Book Is Organized	12	
Document Conventions and Terminology	13	
Introduction	15	
Why Deploy Voice Applications?	15	
Typical Application Scenarios	16	
Queries	16	
Transactions	16	
What is VoiceXML?	17	
What is a Speech Server?	18	
What is a Voice Server?	18	
Contents of the Voice Server	18	
Planning Your Deployment Environment	18	
Installing the Voice Server	19	
Chapter 1	System Architecture	21
	Architectural Overview	22
	Telephony Hardware	22
	Voice Over IP (VoIP) Gateway	23
	VoIP Gateway and H.323 Gatekeeper	23
	Voice Server Components	24
	System Management (SM) Agent	24
	VoiceXML Browser	24

	Interaction with H.323 Telephony Component	25
	VoiceXML Browser Initialization	25
	Interaction with the Web Application Server	25
	IBM ViaVoice Speech Recognition Engine	26
	IBM ViaVoice Text-To-Speech Engine.	26
	H.323 Telephony Component	27
	Registration without an H.323 Gatekeeper	27
	Registration with an H.323 Gatekeeper	27
	Termination and De-Registration with the H.323 Gatekeeper	28
	Web Application Server.	29
	Enterprise Server	29
Chapter 2	Installation and Verification	31
	Configuring the Telephony Environment	31
	Providing Data to the Gateway Expert	31
	Planning and Designing the Telephony Environment	31
	Installing the Voice Server Software	32
	Starting the VoiceXML Browsers	32
	Verifying Your Installation	33
	Checking the VoiceXML Browsers Status	33
	Modifying the Configuration File, If Necessary.	33
	Running the Sample Application.	34
Chapter 3	Voice Server Configuration	35
	Understanding the Sample Configuration File	35
	Specifying Configuration File Parameters	37
	Reconfiguring the Voice Server	42
	Gathering Telephony and Application Data	42
	Modifying the Configuration File.	43
	Implementing Secure Sockets Layer Protocol (SSL)	44
	Supported Digital Certificates	45
	Validating the Configuration File	45
	Forcing a System Update	46
	Checking the VoiceXML Browser Status	46
	Running Your Web-Based Voice Applications.	46
Chapter 4	System Administration	47
	Powering Up a Speech Server	48
	Powering Down a Speech Server	49

Using the Voice Server Commands	50
System Management Naming of VoiceXML Browsers	50
Starting the VoiceXML Browsers	51
From the Command Line	51
From Tivoli GEM Console	53
Checking the VoiceXML Browsers Status	54
From the Command Line	54
From Tivoli GEM Console	55
Getting Command Help	56
From the Command Line	56
From Tivoli GEM Console	57
Stopping the VoiceXML Browsers	58
Issuing an Immediate Shutdown	58
Issuing a Graceful Shutdown	60

Chapter 5

Monitoring and Troubleshooting a Voice Server 63

System Log Files	63
Using the SM Log File Mechanism	64
Location of the SM Log File	64
Format of the SM Log File	64
Using the SM Alarm File Mechanism	65
Location of the SM Alarm File	65
Using the VoiceXML Browser Log File Mechanism	67
Location of the VoiceXML Browser Log File	67
Format of the VoiceXML Browser Log File	67
Finding a VoiceXML Browser Log File	70
Troubleshooting a Voice Server	71
VoiceXML Browser Failed to Start	71
Problem	71
Solution	71
SMConfig Did Not Show Error	71
Problem	71
Solution	72
VoiceXML Browser is Hung	72
Problem	72
Solution	72
VoiceXML Browser Busy Message	72
Problem	72
Solution	72
No Ring Before Answering a Call	73

	Problem	73
	Solution	73
	Long Silence Before Initial Greeting	73
	Problem	73
	Solution	73
Appendix A	Cisco VoIP Gateway Configuration	75
	VoIP Gateway and H.323 Gatekeeper Interaction	75
	System Administrator's Role	77
	Gateway Expert's Role	78
	VoIP Gateway and H.323 Gatekeeper Configuration	78
	Defining and Assigning Aliases	78
	Configuring for Load Distribution	79
	Specifying Load Distribution	80
	Configuring a VoIP Gateway	81
	Specifying and Configuring VoIP Dial Peers	82
Appendix B	Performance	85
	Is Your Server Suitable?	85
	Is Sufficient Memory Installed?	85
	Is the Correct Version of NT Services Installed?	86
	Is the Communications Software Installed?	86
	Do You Have All the Required HW and SW?	86
	What Factors Affect Your Performance?	86
	What About Performance Degradation?	87
Appendix C	Notices	89
	COPYRIGHT LICENSE	91
	Trademarks	92
Glossary		93

Figures

Figure 1.	Overall System Architecture	22
Figure 2.	H.323 Telephony Component Registration	28
Figure 3.	sysmgmt.properties File Example	36
Figure 4.	SM Log File Example	65
Figure 5.	SM Alarm File Example	66
Figure 6.	VoiceXML Browser Log File Example	70
Figure 7.	VoIP Gateway and Gatekeeper Interaction	76
Figure 8.	Translation Rules.	82
Figure 9.	VoIP Dial Peer Configuration Example	83

Tables

Table 1.	Telephony Configuration Parameters	37
Table 2.	General Configuration Parameters	38
Table 3.	VoiceXML Browser Parameters	40
Table 4.	Scenarios for Modifying Configuration File	44
Table 5.	vvsm System Commands	48
Table 6.	Naming Convention for VoiceXML Browsers	51
Table 7.	vvsm start Status Messages	52
Table 8.	vvsm getstatus Status Messages	54
Table 9.	vvsm stop Status Messages	60
Table 10.	System Log Files	63
Table 11.	VoiceXML Browser Log Entries	68
Table 12.	VoiceXML Browser Load Distribution Scheme	79
Table 13.	VoIP Gateway Configuration Information	80

About This Book

This book provides information on using the IBM WebSphere Voice Server with ViaVoice Technology (“Voice Server” for short) to deploy Web-based voice applications in a telephony environment. With these applications, the Voice Server is used to provide voice access to Web application data using standard telephony interfaces. Application development information is documented in the *IBM WebSphere Voice Server Software Developers Kit (SDK) Programmer’s Guide*. See “[Related Publications](#)” below.

Who Should Read This Book

Read this book if you are responsible for:

- Planning your network and telephony configurations
- Installing and configuring a Voice Server
- Deploying Web-based voice applications
- Debugging and troubleshooting a Voice Server
- Ongoing maintenance of a Voice Server

Related Publications

Information on creating Web-based voice applications is available from a variety of sources, including:

- *IBM WebSphere Voice Server Software Developers Kit (SDK) Programmer’s Guide* for information on creating and testing voice applications. This document is part of the IBM WebSphere Voice Server Software Developers Kit (SDK), which is bundled with the IBM WebSphere Voice Server and can also be downloaded from:
<http://www.ibm.com/software/voice>
- *VoiceXML 1.0 Specification* – included with the IBM WebSphere Voice Server and available from the VoiceXML Forum Web site at:
<http://www.voicexml.org>

Information on configuring your deployment environment is available from numerous sources, including:

- *Cisco Documentation* Web site at:
<http://www.cisco.com>
- HTTP 1.1 Specification Web site at:
<http://www.ietf.org/rfc/rfc2616.txt>

How This Book Is Organized

The Introduction contains an overview of the deployment of Web-based voice applications using the Voice Server.

Chapter 1 introduces the system architecture needed to support Web-based voice applications in a deployment environment.

Chapter 2 describes the steps required to plan and design the telephony environment, install the Voice Server, and verify the installation by running the sample application.

Chapter 3 describes the parameters in the **sysmgmt.properties** file and describes how to modify your configuration based on the deployment environment.

Chapter 4 describes the commands used to manage the Voice Server.

Chapter 5 describes the SM Agent log and alarm files and the VoiceXML browser log file. Troubleshooting tips are listed to help when an unexpected event or behavior occurs in the Voice Server deployment environment.

Appendix A describes the Cisco VoIP gateway models 1750/2600 and the Cisco Series 26xx H.323 gatekeepers in the Voice Server deployment environment.

Appendix B contains a list of performance issues about the Voice Server deployment environment.

Appendix C contains notices and trademark information.

Glossary identifies and defines key terminology used in this document.

Document Conventions and Terminology

The following conventions are used to present information in this document:

<i>Italic</i>	Used for emphasis, to indicate variable text, and for references to other documents.
Bold	Used for configuration parameters, file names, URLs, and user interface controls such as command buttons and menus.
Courier Regular	Used for sample code.

The following terminology is used in this document:

Speech Server	The hardware on which the IBM WebSphere Voice Server software runs.
Voice Server	The IBM WebSphere Voice Server software.



Introduction

The Voice Server enables you to deploy Web-based voice applications written in VoiceXML. Voice applications are applications where the input is through speech and/or DTMF (Dual Tone Multi Frequency) telephone keypresses and the output is through speech rather than a graphical user interface.

This chapter provides an overview on deploying Web-based voice applications using the IBM WebSphere Voice Server.

This introduction addresses the following questions:

- “Why Deploy Voice Applications?” — See [page 15](#).
- “What is VoiceXML?” — See [page 17](#).
- “What is a Speech Server?” — See [page 18](#).
- “What is a Voice Server?” — See [page 18](#).

Why Deploy Voice Applications?

Until recently, the World Wide Web has relied exclusively on visual interfaces to deliver information and services to users via computers equipped with a monitor, keyboard, and pointing device. In doing so, a huge potential customer base has been ignored: people who (due to time, location, and/or cost constraints) do not have access to a computer.

Many of these people do, however, have access to a telephone. Providing “conversational access” (that is, spoken input and audio output over a telephone) to Web-based data will permit companies to reach this untapped market. Users benefit from the convenience of using the mobile Internet for self-service transactions, while companies enjoy the Web’s relatively low transaction costs. And, unlike applications that rely on DTMF (telephone keypress) input, voice applications can be used in a hands-free or eyes-free environment, as well as by customers with rotary pulse telephone service or telephones in which the keypad is on the handset.

Typical Application Scenarios

Voice applications will typically fall into one of the following categories:

- “Queries”
- “Transactions”

Queries

In this scenario, a customer calls into a system to retrieve information from a Web-based infrastructure.

The system guides the customer through a series of menus and forms by playing instructions, prompts, and menu choices using prerecorded audio files or synthesized speech.

The customer uses spoken commands and/or DTMF (telephone keypress) input to make menu selections and fill in form fields.

Based on the customer’s input, the system locates the appropriate records in a back-end enterprise database. The system presents the desired information to the customer, either by playing back prerecorded audio files or by synthesizing speech based on the data retrieved from the database.

Examples of this type of self-service interaction include applications or voice portals providing weather reports, movie listings, stock quotes, health care provider listings, and customer service information (Web call centers).

Transactions

In this scenario, a customer calls into a system to execute specific transactions with a Web-based back-end.

The system may prompt the customer to log in using some type of ID and/or password, and then guides the customer to provide the data required for the transaction.

The system plays instructions, prompts, and menu choices using prerecorded audio files or synthesized speech, and the customer responds using spoken commands and/or DTMF (telephone keypress) input.

Based on the customer’s input, the system conducts the transaction and updates the appropriate records in a back-end enterprise database. Typically the system also reports back to the customer, either by

playing back prerecorded audio files or by synthesizing speech based on the information in the database records.

Examples of this type of self-service interaction include applications or voice portals for employee benefits, employee timecard submission, financial transactions, travel reservations, calendar appointments, customer relationship management (eRM), sales automation, and order management.

What is VoiceXML?

The Voice eXtensible Markup Language (VoiceXML) is an XML-based markup language for creating distributed voice applications, much as HTML is a markup language for creating distributed visual applications.

VoiceXML is an emerging industry standard that has been defined by the VoiceXML Forum (<http://www.voicexml.org>), of which IBM is a founding member. It has been accepted for submission by the World Wide Web Consortium (W3C) as a standard for voice markup on the Web.

The VoiceXML language enables Web developers to use a familiar markup style and Web server-side logic to deliver voice content to the Internet. The resulting VoiceXML applications can interact with your existing back-end business data and logic.

Using VoiceXML, application developers can create Web-based voice applications that users can access by telephone. VoiceXML supports dialogs that feature:

- recognition of spoken input
- DTMF (telephone key) input
- recording of spoken input
- synthesized speech output (“text-to-speech”)
- pre-recorded digitized audio output
- dialog flow control
- scoping of input

What is a Speech Server?

A speech server is a server-class system that runs the IBM WebSphere Voice Server software. See the file **readme.txt** on the IBM WebSphere Voice Server installation CD for a list of hardware and software requirements.

What is a Voice Server?

A Voice Server is the IBM WebSphere software that operates in conjunction with a Voice over IP (VoIP) gateway, H.323 gatekeeper (optional), and existing Web infrastructure to allow delivery of Web-based voice applications written in VoiceXML.

Contents of the Voice Server

The Voice Server includes:

- A speech browser that interprets VoiceXML markup.
- A connection environment that uses VoIP (H.323 V2.0) protocols to manage telephone calls and process telephony audio.
- IBM ViaVoice Speech Recognition and Text-To-Speech engines to accept voice input and generate synthesized speech output.
- Telephony-specific acoustic models for accurate speech recognition over telephone lines.
- A System Management (SM) Agent that performs basic system management functions.
- A sample VoiceXML application to test your installation and configuration.
- User documentation.

Planning Your Deployment Environment

The Voice Server represents state-of-the-art technology in voice processing. For your business to obtain maximum benefits from this product, you will need the assistance of qualified personnel to

install, configure, and service it. We recommend you establish a configuration plan when setting up your telephony and IP networks prior to installing the Voice Server.

- To setup your network, you may need the services of a Network Administrator with networking and IP communications expertise.
- To plan and design the telephony environment based on your IP network, you may need the services of a *gateway expert* with expertise in the areas of H.323 V2.0 protocol, VoIP gateways and gatekeepers, and telephony infrastructure.

Refer to the file **readme.txt** on the Voice Server Installation CD for details on the hardware requirements needed to set up your deployment environment.

Installing the Voice Server

Once your network is running, you will need to install the Voice Server. Refer to [Chapter 2, “Installation and Verification” on page 31](#) for more details regarding these steps.

1. Work closely with the gateway expert and provide information about your IP network, telephony services, Speech Servers, and VoiceXML browsers.
2. Obtain telephony information from the gateway expert to verify that the default values assumed by the Voice Server match the telephony environment.
3. Install the Sun Java Runtime Environment (Sun JRE) 1.3.0. (Included on the Voice Server Installation CD.)
4. Install the Adobe Acrobat Reader version 4.0.5 or later, (included on the Voice Server Installation CD).
5. Install the Voice Server using an Administrator ID.
6. Reboot your system.
7. Log on with an Administrator ID.
8. Start the VoiceXML browsers on the Voice Server.
9. Check the VoiceXML browser status.
10. Run the AudioSample application. (Refer to [“Running the Sample Application” on page 34](#) for details.)
11. Configure the Voice Server to run your own voice applications. Refer to [Chapter 3, “Reconfiguring the Voice Server” on page 42](#) for details.

The Voice Server deployment environment supports Web-based voice applications written in VoiceXML. This chapter describes a typical deployment configuration that may include the following components:

- “[Telephony Hardware](#)” — See [page 22](#).
- “[Voice Server Components](#)” — See [page 24](#).
- “[Web Application Server](#)” — See [page 29](#).
- “[Enterprise Server](#)” — See [page 29](#).

[Figure 1](#) shows the overall system. The Voice Server is installed in the Speech Server box. Your configuration may include one or more VoIP gateways or H.323 gatekeepers (optional) communicating with one or more Speech Servers.

Architectural Overview

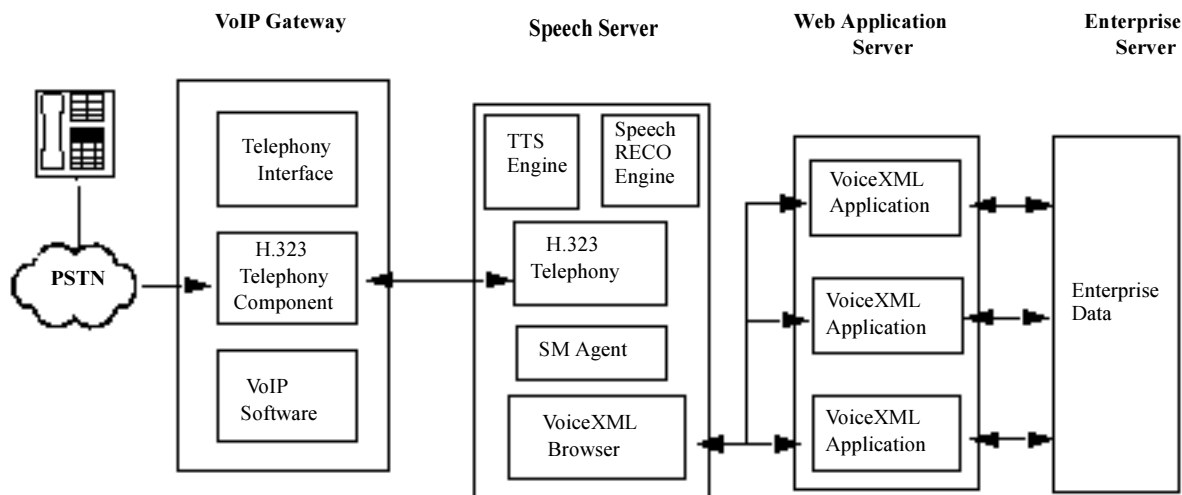


Figure 1. Overall System Architecture

Telephony Hardware

The telephony piece of the deployment environment consists of gateways that are used in conjunction with H.323 gatekeepers to provide a connection between regular telephone lines through a Public Switched Telephone Network (PSTN), Global System for Mobile Communication (GSM), or a Private Branch Exchange (PBX) network and VoiceXML browsers that reside on Voice Servers in an Internet Protocol (IP) network.

Note: If there is no H.323 gatekeeper installed, a Speech Server can support only one VoiceXML browser session.

Voice Over IP (VoIP) Gateway

The VoIP gateway provides a connection to set up calls and manage audio data flow between a telephone and a Voice Server (see [“Voice Server Components” on page 24](#)) that resides on an Internet Protocol (IP) network. The gateway includes an H.323 telephony component that communicates with the Voice Server to:

- Provide the appropriate conversion from data over the Integrated Services Digital Network (ISDN) Primary Rate Interface (PRI) connected T1 lines (E1 lines in Europe) into packetized formation IP network.
- Route a call to an available H.323 terminal endpoint in the IP network, depending upon the *call distribution scheme*.
- Detect and deliver DTMF digits and voice audio to the VoiceXML browsers.
- Provide load balancing using the VoIP gateway configuration. This includes the translation rules table and dial peers. Refer to Appendix A [“Cisco VoIP Gateway Configuration” on page 75](#).

The Voice Server requires an H.323 Version 2.0 compliant VoIP gateway. The VoIP gateway should support:

- G.711 uncompressed audio standard
- DTMF detection and extraction (in-band and out-of-band).

VoIP Gateway and H.323 Gatekeeper

In some configurations, an H.323 gatekeeper is installed and configured in conjunction with a VoIP gateway. With this configuration, you can run multiple VoiceXML browsers on a Speech Server.

The H.323 gatekeeper is responsible for routing incoming calls between the VoIP gateway and available VoiceXML browsers by providing the browser’s required IP address and port number. This information is used by the VoIP gateway to map incoming calls to the first available VoiceXML browser. (See [“Cisco VoIP Gateway Configuration” on page 75](#) for Load Distribution.)

Voice Server Components

The IBM WebSphere Voice Server runs on a Speech Server that includes the following components:

- [“System Management \(SM\) Agent”](#) — See [page 24](#).
- One or more instances of the [“VoiceXML Browser”](#) — See [page 24](#).
- [“IBM ViaVoice Speech Recognition Engine”](#) — See [page 26](#).
- [“IBM ViaVoice Text-To-Speech Engine”](#) — See [page 26](#).
- [“H.323 Telephony Component”](#) — See [page 27](#).

Your configuration may include multiple Speech Servers.

System Management (SM) Agent

The SM Agent is responsible for starting, stopping, and monitoring VoiceXML browsers. There is one SM Agent per Speech Server.

After the system reboots and the IBM WebSphere Voice Server NT service is running, you can start the Voice Server with the SM Agent. Refer to [“Starting the VoiceXML Browsers” on page 51](#) for details.

VoiceXML Browser

The VoiceXML browser is responsible for handling telephone calls and managing the dialog with the caller. It processes the calls, fetches VoiceXML documents from a location specified by the application (that is, a URI on a Web application server or local disk), and parses and interprets the documents.

You can configure the SM Agent to start multiple VoiceXML browser instances on a single Speech Server. Each VoiceXML browser processes a single telephone call. When a call is completed, the VoiceXML browser is restarted and waits for another telephone call.

The VoiceXML browser uses the VoIP protocol (H.323 Version 2.0) connection environment for managing telephone calls and processing telephony audio.

Interaction with H.323 Telephony Component

1. The actual voice from the PSTN/GSM is packetized by the VoIP gateway and sent to the H.323 telephony component of the Voice Server.
2. The VoiceXML browser communicates with the H.323 telephony component to accept or terminate telephone calls.
3. Telephone audio from the H.323 telephony component is received and streamed to the IBM ViaVoice Speech Recognition engine for processing.
4. DTMF input is received from the H.323 telephony component and is used within the VoiceXML application to fill in forms or select menu items.
5. When the VoiceXML browser encounters an element that requires speech output, it sends telephony audio to the H.323 telephony component.
6. The audio may be generated from the IBM ViaVoice Text-To-Speech engine, recorded audio stored in an external file, or audio that was recorded within the VoiceXML application itself via the VoiceXML `<record>` element.
7. The audio is packetized by the VoiceXML browser, sent to the VoIP gateway, converted to voice, and delivered to the end-user.

VoiceXML Browser Initialization

When the VoiceXML browser starts, it requests an initial VoiceXML document that specifies an interaction (or “dialog”) between the application and the end-user. The VoiceXML browser interprets and renders the document, managing the dialog with the user by playing audio prompts (using text-to-speech or recorded audio), accepting user voice and DTMF inputs, and acting on those inputs. Each of these activities changes the state of the dialog. When a dialog does not specify a transition to another dialog, the VoiceXML browser exits and the session terminates.

Refer to the *IBM WebSphere Voice Server Software Developers Kit (SDK) Programmer’s Guide* and the *VoiceXML 1.0 Specification* for detailed information.

Interaction with the Web Application Server

The VoiceXML browser uses HTTP over the Internet or an intranet to fetch VoiceXML application pages from a Web application server. The VoiceXML browser uses the local file system to cache documents fetched from the server. The cache is shared between VoiceXML browser instances running on the same Speech Server.

IBM ViaVoice Speech Recognition Engine

The VoiceXML browser uses the IBM ViaVoice Speech Recognition engine to recognize voice input from a user responding to prompts generated by a VoiceXML application. The recognition results are used within the VoiceXML application to fill in form items or select a menu item.

The IBM ViaVoice Speech Recognition engine uses telephony acoustic models for accurate speech recognition over telephone lines.

The high-level process looks like this:

1. During application development, the developer creates a series of speech recognition grammars defining the words and phrases that can be spoken by the user, and specifying where each grammar should be active within the application.
2. When you start the Voice Server, each VoiceXML browser starts an instance of the IBM ViaVoice Speech Recognition and Text-To-Speech engines.
3. When the application runs, the speech recognition engine processes the incoming audio signal and compares the sound patterns to the patterns of basic spoken sounds, trying to determine the most probable combination that represents the audio input.
4. The speech recognition engine compares the sounds to the list of words and phrases in the currently active grammar(s). Only words and phrases in the active grammars are considered as possible speech recognition candidates.
5. The speech recognition engine returns the results to the VoiceXML browser.
6. The VoiceXML browser uses the results within the VoiceXML application to fill in form fields or select menu items, thereby managing the dialog with the user.

IBM ViaVoice Text-To-Speech Engine

The IBM ViaVoice Text-To-Speech engine allows ASCII text to be converted to synthesized speech. The text source may come from prompts within a VoiceXML application or from data that is retrieved from a database.

H.323 Telephony Component

The telephony environment includes an H.323 telephony component on a Voice Server that interacts with the H.323 telephony component on the VoIP gateway (and H.323 gatekeeper, if installed). The H.323 telephony component routes call information between the telephony connection and the VoiceXML browser that is running on the Voice Server.

Registration without an H.323 Gatekeeper

Some telephony environments do not include a H.323 gatekeeper. In these gateway-only configurations, the initialization and registration process occurs when the H.323 telephony component on the VoIP gateway calls the H.323 telephony component on the Speech Server for registration of a specific call instance; the request information is pre-configured and maps to a network address of a Speech Server (IP address) where the VoiceXML browser instance is running.

Note: You can run only one VoiceXML browser on the Voice Server with this configuration.

Registration with an H.323 Gatekeeper

As part of the initialization process, the H.323 telephony component on the VoiceServer calls the H.323 telephony component on the H.323 gatekeeper to register the VoiceXML browser (H.323 terminal endpoint alias). This is done by sending a request for registration of a specific call instance; the request carries a TSAP (Telephone Service Access Point) or network address of a Speech Server (IP address) where the VoiceXML browser instance is running. The registration request allows the H.323 gatekeeper to maintain a list of available VoiceXML browsers along with their current states. The H.323 gatekeeper then routes incoming calls to the first available VoiceXML browser, according to an algorithm described in [“Specifying Load Distribution” on page 80](#).

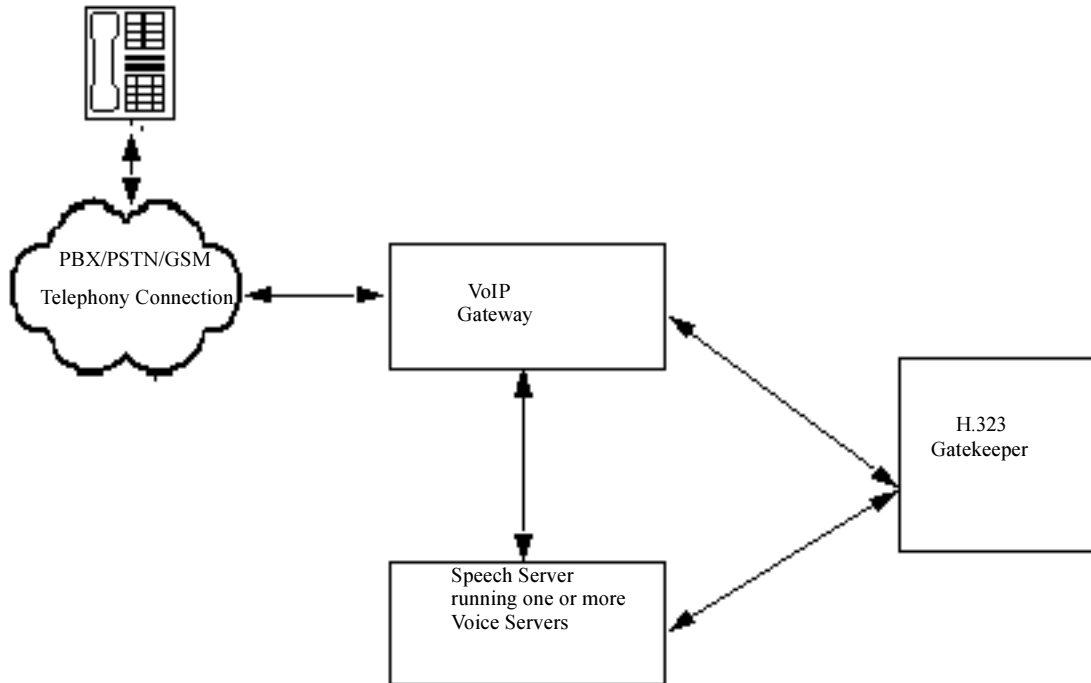


Figure 2. H.323 Telephony Component Registration

Termination and De-Registration with the H.323 Gatekeeper

As part of the termination process, the H.323 telephony component on the Speech Server calls the H.323 telephony component on the H.323 gatekeeper to de-register a specific instance of a VoiceXML browser.

Web Application Server

Based on the dialog between the user and the VoiceXML application, the VoiceXML browser proceeds with the interaction specified by the VoiceXML application. The VoiceXML browser might continue to the next dialog in the document, fetch a new VoiceXML document, or submit information to the Web application server for processing. The Web application server can use server-side programs to locate or update records in a back-end enterprise database and return the information to the VoiceXML browser. A Web-based voice application runs on top of a Web server and can be implemented with a variety of technologies such as Java servlets, JSPs, ASPs and CGI scripts.

Enterprise Server

VoiceXML applications can access existing enterprise data from your enterprise servers in the same manner as HTML applications. A requested document can be generated dynamically from data stored in the enterprise database using the same types of server-side logic (CGI scripts, Java Beans, JSPs, Java Servlets, etc.) that you use to generate HTML applications.

This chapter explains how to install the IBM WebSphere Voice Server in a deployment environment and how to verify the installation by running the sample application that is shipped with the product. A default configuration file is included on the Voice Server installation CD which is designed to run the sample application in a gateway-only environment using a single VoiceXML browser.

The following steps are required to complete the Voice Server installation:

- “[Configuring the Telephony Environment](#)” — See [page 31](#).
- “[Installing the Voice Server Software](#)” — See [page 32](#).
- “[Starting the VoiceXML Browsers](#)” — See [page 32](#).
- “[Verifying Your Installation](#)” — See [page 33](#).
- “[Running the Sample Application](#)” — See [page 34](#).

Configuring the Telephony Environment

We recommend that you work closely with a *gateway expert* to configure the telephony environment. The gateway expert should have expertise with the H.323 protocol, telephony infrastructure, VoIP gateways, and H.323 gatekeepers.

Providing Data to the Gateway Expert

You must provide specific information about your IP network to the gateway expert to enable him/her to design the call distribution for incoming calls. See “[Cisco VoIP Gateway Configuration](#)” on [page 75](#).

Planning and Designing the Telephony Environment

The gateway expert is responsible for planning and designing the telephony environment. Based on the information about your IP network, the gateway expert will assign an H.323 terminal endpoint alias for

each VoiceXML browser, and plan and configure the *dial peers* and *translation rules tables*. These tables are used to translate incoming calls to previously assigned aliases. The gateway expert will also request *channels (hunt groups)* from the telephone company and phone numbers for each application.

Note: The aliases are specified by the `INBOUND_H323_ENDPOINT_ALIASn` parameter. This parameter specifies an H.323 terminal endpoint used for gatekeeper routing. The valid values are telephone numbers based on the E164 standard. Refer to [Table 3 on page 40](#) for details.

Installing the Voice Server Software

Now that the telephony environment is properly configured and operational, you can install the IBM WebSphere Voice Server software. To do this, refer to the `readme.txt` file located on the Voice Server installation CD and follow the installation instructions.

Note: After completing the software installation, you must reboot the system.

Starting the VoiceXML Browsers

To start the VoiceXML browsers based on the pre-configured `sysmgmt.properties` file, log onto the Speech Server using an Administrator ID and run the `vvsml start` command. The VoiceXML browsers will not start automatically when the Speech Server is powered up because the `AUTO_START_ON_INIT` parameter in the `sysmgmt.properties` file included on the Voice Server installation CD is set to *false*. To automatically start the VoiceXML browser when the Speech Server is subsequently powered up, set the `AUTO_START_ON_INIT` parameter to *true*.

Verifying Your Installation

After you have installed and started the Voice Server software, complete the following steps to verify the installation.

Checking the VoiceXML Browsers Status

To check the status of the VoiceXML browsers, issue the following CLI command:

```
vvsm getstatus
```

This command checks the VoiceXML browsers that are running on the Voice Server to determine their current state. Status messages will be displayed indicating whether the VoiceXML browsers have not yet started, are in the process of starting, are operational and waiting for a call, are stopped, or are in an unknown state. See [“Checking the VoiceXML Browsers Status” on page 54](#) for command description and usage.

You can retry this command several times to observe the progress of the Voice Server while the VoiceXML browsers are starting up. If after a few minutes the “waiting for call” message is displayed, you can proceed to the next step — [“Running the Sample Application”](#).

If a problem occurs and the VoiceXML browsers do not start up, check the SM log file (**vvsmalog**) to identify the cause. See [“Using the SM Log File Mechanism” on page 64](#).

Modifying the Configuration File, If Necessary

If your telephony environment does not match the default **sysmgmt.properties** configuration file shipped on the Voice Server installation CD, you’ll need to update the file prior to running the sample application. Refer to the steps outlined in [Chapter 3 “Voice Server Configuration”](#).

Running the Sample Application

After you have verified the Voice Server installation and checked the status of the VoiceXML browser, place a call to the Voice Server sample application (**AudioSample.vxml**) to verify the integration of the telephony environment and the Voice Server software.

AudioSample.vxml (located in *<install-directory>/samples/AudioSample*) is a very simple VoiceXML application that prompts the caller to record some speech, and then plays the recorded audio back to the caller.

To call this sample application, dial the assigned Dialed Number Identification Service (DNIS) called number and wait for a response. If the sample runs without any errors, this indicates the hardware is properly connected and configured. Refer to the AudioSample **readme.txt** file located in the same directory.

This chapter explains how to configure the Voice Server to run your own VoiceXML applications. This chapter describes the following topics:

- “[Understanding the Sample Configuration File](#)” — See [page 35](#).
- “[Specifying Configuration File Parameters](#)” — See [page 37](#).
- “[Reconfiguring the Voice Server](#)” — See [page 42](#).

Understanding the Sample Configuration File

[Figure 3](#) shows the `sysmgmt.properties` file that is included on the Voice Server installation CD. It is configured to support a gateway-only environment (that is, no gatekeeper). You will notice that there are three types of configuration parameters:

- Telephony — See [Table 1 on page 37](#).
- General — See [Table 2 on page 38](#).
- VoiceXML browser — [Table 3 on page 40](#).

The configuration file documents all possible parameters in the comment sections of the file (lines whose first character is #). Only those parameters that must be explicitly assigned (no default available) are required in this file.

```
# This file contains the IBM WebSphere Voice Server with ViaVoice Technology System
# Management configuration parameters.
# The file defines 3 types of configuration parameters:
#   1. Telephony
#   2. General
#   3. VoiceXML Browser
#-----
# Telephony Configuration section
#
#   H323_GATEKEEPER_HOST
#-----
# General Configuration Parameters
#
#   ALARM_FILE_MAX_SIZE
#   AUTO_START_ON_INIT
#   BROWSER_CACHE_MAXSIZE
#   BROWSER_CACHE_PATHNAME
#   BROWSER_DIRECTORIES_KEPT
#   LOG_FILE_MAX_SIZE
#   NUMBER_INBOUND_BROWSERS
#   SYS_MGMT_OUTPUT_PATHNAME
#-----
ALARM_FILE_MAX_SIZE=1
AUTO_START_ON_INIT=false
BROWSER_CACHE_MAXSIZE=1
BROWSER_CACHE_PATHNAME=cache
LOG_FILE_MAX_SIZE=1
NUMBER_INBOUND_BROWSERS=1
SYS_MGMT_OUTPUT_PATHNAME=logs
#-----
# VoiceXML Browser Configuration Parameters
#   INBOUND_BARGE_INn
#   INBOUND_DUPLEXn
#   INBOUND_H323_ENDPOINT_ALIASn
# * INBOUND_SITEn
#   INBOUND_TIMEOUTn
# * INBOUND_URLn=file:///<install-directory>/samples/AudioSample/AudioSample.vxml
# * The value for this property must NOT contain spaces. If the directory name does
#   contain spaces, simply enter the 8.3 version of the directory name.
#
#   "n" The notation following the above INBOUND_*n parameters indicates a number from 0
#   to (NUMBER_INBOUND_BROWSERS-1) which corresponds to the System Management naming
#   convention for the VoiceXML Browsers of vvi0"n". For example, INBOUND_*0 parameters
#   correspond to VoiceXML Browser "vvi00". INBOUND_*1 parameters correspond to VoiceXML
#   Browser "vvi01", and so on.
INBOUND_URL0=file:///<install-directory>/samples/AudioSample/AudioSample.vxml
```

Figure 3. sysmgmt.properties File Example

Specifying Configuration File Parameters

Table 1. Telephony Configuration Parameters

Property	Description
H323_GATEKEEPER_HOST	Defines the H.323 gatekeeper's host name (gatekeeper discovery is not supported). If no value is assigned to this parameter, no gatekeeper will be used. Default = none Valid values = host name or IP address Required if you are using a gatekeeper.

Table 2. General Configuration Parameters

Property	Description
ALARM_FILE_MAX_SIZE	<p>Specifies the alarm file (vv_sma.alr) maximum size in megabytes.</p> <p>Note: The required disk space is twice the defined size, to allow for maintenance and backup files.</p> <p>Pre-configured value = 1 Required Valid values = decimal integer > 0</p>
AUTO_START_ON_INIT	<p>Specifies whether the VoiceXML browsers will be automatically started upon system startup.</p> <p>If set to <i>true</i>, the SM Agent starts all configured inbound VoiceXML browsers upon startup of the Speech Server.</p> <p>If set to <i>false</i>, no VoiceXML browsers will be started until the vvsm start CLI command is issued.</p> <p>Pre-configured value = false Required Valid values = true or false</p>
BROWSER_CACHE_MAXSIZE	<p>Specifies the target maximum size in megabytes of the VoiceXML browser cache after a VoiceXML browser has exited.</p> <p>Pre-configured value = 1 Required Valid values = decimal integer > = 1</p>
BROWSER_CACHE_PATHNAME	<p>Specifies the pathname where the VoiceXML browser cache is stored. The pathname must already exist. If you do not specify a valid pathname, the SM Agent will prepend the installation directory to the beginning of the pathname.</p> <p>Pre-configured value = cache Required Valid values = valid pathname</p> <p>Note: UNIX-type slashes ('/') must be used for a qualified pathname. In Windows systems you can also use double backslashes ('\\'), but it is not recommended for portability reasons.</p>

Property	Description
BROWSER_DIRECTORIES_KEPT	<p>Defines the number of VoiceXML browser trace directories that should be kept once the VoiceXML browser has stopped sending information to them. These directories are in addition to the active or running VoiceXML browser directory that is always used.</p> <p>Default = 4 Valid values = decimal integer ≥ -1 Note: -1 means to keep all directories</p>
LOG_FILE_MAX_SIZE	<p>Specifies the log file (vvvsmalog) maximum size in megabytes.</p> <p>Note: The required disk space is twice the defined size, to allow for maintenance and backup files.</p> <p>Pre-configured value = 1 Required Valid values = decimal integers > 0</p>
NUMBER_INBOUND_BROWSERS	<p>Specifies the number of inbound VoiceXML browsers the system will start for incoming calls. One set of inbound property variables must be defined for each specified browser.</p> <p>Pre-configured value = 1 Required Valid values = decimal integers > 0</p>
SYS_MGMT_OUTPUT_PATHNAME	<p>Specifies the location (directory or pathname) where the Voice Server will save log and trace files (vvvsmalog, vvvsmalr, and VoiceXML browser directories). If you do not specify a valid pathname, the SM Agent will prepend the installation directory to the beginning of the pathname. The pathname must already exist.</p> <p>Pre-configured value = logs Required Valid values = valid pathname.</p> <p>Note: UNIX-type slashes (/) must be used for a qualified pathname. In Windows systems you can also use double backslashes (\\), but it is not recommended for portability reasons.</p>

Table 3. VoiceXML Browser Parameters

Property	Description
INBOUND_BARGE_IN n	Indicates the barge-in detection method used by the VoiceXML browser. Default = recognition Valid values = energy or recognition
INBOUND_DUPLEX n	Specifies the duplex implementation. When full-duplex is specified, the user and computer can speak at the same time; when half-duplex is specified, the user should not speak while the computer is speaking, because the audio will not be received by the speech recognition engine. Default = full Valid values = full or half
INBOUND_H323_ENDPOINT_ALIAS n	An H.323 terminal endpoint identification used for routing calls from a VoIP gateway through a VoIP gatekeeper to an IP network. Note: The alias must be a telephone number based on the E164 standard. Valid values = any numeric value <u>With a VoIP gatekeeper:</u> Default = none Required Numeric - (all digits) <u>Without a VoIP gatekeeper:</u> Default = 9999
INBOUND_SITE n	Specifies the URL of a VoiceXML site customization file that is read when the VoiceXML browser starts up. The file remains loaded for the life of the VoiceXML browser instance. Default = none Valid values = existing URL
INBOUND_TIMEOUT n	Specifies the timeout value in seconds before the VoiceXML browser throws a noinput event, in the form of <i>numbers</i> . Default = 7 Valid value = decimal integers > = 2

Property	Description
INBOUND_URL <i>n</i>	Specifies the URL string for Web-based voice applications. Note: This parameter must not contain any spaces. Default = none Pre-configured value = file:///<install-directory>/samples/audiosample/ AudioSample.vxml Valid values = existing URL Required
<p>NOTES:</p> <ol style="list-style-type: none"> The "<i>n</i>" values are based on the number of VoiceXML browsers NUMBER_INBOUND_BROWSERS configured per Voice Server. Parameters with out-of-range values will be discarded. Pre-configured parameters are assigned with specific values in the sysmgmt.properties configuration file that is located on the Voice Server installation CD. Default parameters are assigned when specific values are not set (or commented out) in the sysmgmt.properties configuration file included on the Voice Server installation CD. 	

Reconfiguring the Voice Server

Based on your deployment environment, you may need to modify the configuration due to one or more of the following reasons:

- Your environment does not match the pre-configured default values assigned in the configuration file.
- You want to modify or scale your telephony environment by adding a VoIP gatekeeper and multiple VoiceXML browsers.
- You want to run your own VoiceXML application(s).

The following steps are required to configure a Voice Server:

- [“Gathering Telephony and Application Data”](#)— See [page 42](#).
- [“Modifying the Configuration File”](#) — See [page 43](#).
- (Optional) [“Implementing Secure Sockets Layer Protocol \(SSL\)”](#) — See [page 44](#).
- [“Validating the Configuration File”](#)— See [page 45](#) .
- [“Forcing a System Update”](#) — See [page 46](#).
- [“Checking the VoiceXML Browser Status”](#) — See [page 46](#).
- [“Running Your Web-Based Voice Applications”](#) — See [page 46](#).

Gathering Telephony and Application Data

You will need to obtain specific telephony information from the gateway expert. This is done to verify that the following default values assumed by the Voice Server match the telephony hardware:

- Endpoint terminal alias for each VoiceXML browser — **INBOUND_H323_ENDPOINT_ALIAS n**
- Phone numbers assigned for your VoiceXML applications, per Speech Server
- Gatekeeper host — the **H323_GATEKEEPER_HOST** parameter defines the host name of the H.323 gatekeeper and is a required parameter when a H.323 gatekeeper is installed.

Note: Each Speech Server can only run a single application, using a single DNIS. However, you can configure multiple VoiceXML browsers to run multiple instances of the application on the Speech Server.

You may also need to discuss your VoiceXML application with the application developer to determine if you will need to modify the following parameters:

- **INBOUND_BARGE_IN n**
- **INBOUND_DUPLEX n**
- **INBOUND_SITE n**
- **INBOUND_TIMEOUT n**
- **INBOUND_URL n**
- **NUMBER_INBOUND_BROWSERS**

Modifying the Configuration File

There are several types of scenarios in a deployment environment that would require configuration changes. Therefore, you will need to modify the **sysmgmt.properties** configuration file that is included on the Voice Server installation CD for each of these scenarios. The required parameters are noted in [Table 4](#) below.

Table 4. Scenarios for Modifying Configuration File

Scenario	Configuration Parameters to Change
Run Audio Sample without a gatekeeper.	None
Run Audio Sample with a gatekeeper and one VoiceXML browser	H323_GATEKEEPER_HOST INBOUND_H323_ENDPOINT_ALIAS n
Run your own VoiceXML application with no gatekeeper.	INBOUND_URL n
Run your own VoiceXML application with a gatekeeper and multiple browsers	H323_GATEKEEPER_HOST INBOUND_H323_ENDPOINT_ALIAS n INBOUND_URL n NUMBER_INBOUND_BROWSERS
Configure additional browsers for any of these scenarios	INBOUND_URL n NUMBER_INBOUND_BROWSERS

In addition, you may want to make on-going changes to the other configurable parameters in order to fine-tune or improve the system performance of your own deployment environment. See [“Specifying Configuration File Parameters” on page 37](#).

Implementing Secure Sockets Layer Protocol (SSL)

Secure Socket Layer (SSL) protocol provides authentication and data security. It encapsulates a TCP/IP socket and is used by TCP/IP applications that require secure communications. SSL is a low-level authentication and encryption service used by higher-level applications. SSL allows encrypted and secure exchange transmission between a VoiceXML browser instance and an HTTPS Web application server.

To implement secure communications within your deployment environment your Web-based voice application must point to a secure Web application server (HTTPS protocol). To do this, configure the **INBOUND_URL n** (HTTPS://...) parameter. The encryption algorithm used for the secure transmission will automatically be negotiated between the VoiceXML browser and the Web application server hosting the Web-based voice application. An optimal encryption algorithm will be chosen.

Supported Digital Certificates

Server authentication between a Web application server hosting a Web-based voice application (using HTTPS protocol) and a Voice Server VoiceXML browser is supported. This is because the Voice Server does not provide a tool to manage the X.509 digital certificates for SSL. For server authentication, the Web application server must have one of the digital certificates based on the X.509 standard below. The trusted third-parties, or signer certificates, verify the identification of a certificate holder. The certificate holders that are installed with the Voice Server include:

- Thawte Server CA
- Verisign Class 1 Public Primary Certification Authority
- Thawte Personal Basic CA
- Verisign Class 3 Public Primary Certification Authority
- Verisign Test CA Root Certificates
- Verisign Class 2 Public Primary Certification Authority
- RSA Secure Server Certification Authority
- Thawte Premium Server CA
- Thawte Personal Freemail CA
- Thawte Personal Premium CA

The digital certificate is used to authenticate the Web server to the VoiceXML browser. During the initial SSL handshake, the Web server supplies the VoiceXML browser with its X.509 certificate. If the VoiceXML browser validates the Web server's certificate, a secure encrypted communication channel is established between the Web server and the VoiceXML browser. You must make sure that the Web server hosting the VoiceXML browser application supports one of the digital certificates listed above.

Validating the Configuration File

To validate the syntax of the updated configuration file, enter **SMConfig** in a Java console window. Each time **SMConfig** detects an invalid parameter in the configuration file it displays an error message on the console and stops running. After you have corrected and saved the configuration file, you can run **SMConfig** again and repeat these steps until a clean test is performed. You will know that the configuration file has passed all syntax validation when a “No Errors” status message is returned.

Note: Parts of the configuration will not be verified until the VoiceXML browsers are started.

Forcing a System Update

After updating and validating the syntax of the configuration file, the system must be forced to re-read the file and use the updated parameter values. To do this, issue the following CLI commands. See [“Stopping the VoiceXML Browsers” on page 58](#) for command usage

```
vvsm stop
vvsm start
```

Checking the VoiceXML Browser Status

To check the status of the VoiceXML browsers, issue the following CLI command:

```
vvsm getstatus
```

This command checks the VoiceXML browsers that are running on the Voice Server to determine their current state. Status messages will be displayed indicating whether the VoiceXML browsers have not yet started, are in the process of starting, are operational and waiting for a call, are stopped, or are in an unknown state. See [“Checking the VoiceXML Browsers Status” on page 54](#) for command description and usage.

You can retry this command several times to observe the progress of a Voice Server while the VoiceXML browsers are starting up. If after a few minutes the “waiting for call” message is displayed, you can proceed to the next step — [“Running Your Web-Based Voice Applications”](#) below.

If a problem occurs and the VoiceXML browsers do not start up, check the SM log file (**vvsm.log**) to identify the cause. See [“Using the SM Log File Mechanism” on page 64](#).

Running Your Web-Based Voice Applications

Dial the assigned telephone number to run your application. See [“Cisco VoIP Gateway Configuration” on page 75](#).

This chapter defines and describes the CLI commands that can be used to manage your deployment environment. It also includes the *Tivoli Global Enterprise Manager (GEM)* console screens that are available in a Tivoli environment. For installation and configuration information for a Tivoli environment, refer to the documentation that is packaged with the Tivoli products. See the Tivoli readme file located in <install-directory>:\TivReady\TivReady_readme.txt.

Once a Speech Server is powered up, the SM Agent is started and the VoiceXML browsers are operational, you can monitor the Web-based voice applications using the CLI commands or the Tivoli GEM Console screens, depending upon your environment.

Note: You must must log on with an Administrator ID in order to use these commands.

The CLI commands are listed in [Table 5](#) and described later in this chapter. The Tivoli GEM Console screens are displayed with each associated CLI command.

Note: The “Voice Server” icon that is displayed in a Tivoli screen identifies a specific Speech Server (hardware box) that has Voice Servers running on it.

The following tasks are described in this chapter:

- “Powering Up a Speech Server” — See [page 48](#).
- “Powering Down a Speech Server” — See [page 49](#).
- “Using the Voice Server Commands” — See [page 50](#).

Table 5. vvsm System Commands

Command	Description
vvsm start	Starts the configured VoiceXML browsers. The start command forces the system to read the configuration file. This command is ignored if the VoiceXML browsers are already running.
vvsm stop	Requests all VoiceXML browsers to gracefully shut down. This request is asynchronous (that is, the CLI command ends before the VoiceXML browsers have stopped). Browsers that are in the “received call” state will not terminate until the call dialog ends, or a 10 minute timeout is reached.
vvsm stop now	Requests an immediate stop to all VoiceXML browsers. This request will not wait until the call dialog ends before shutting down the VoiceXML browsers; it will terminate an on-going call.
vvsm getstatus	Returns the status of a Voice Server which may have multiple VoiceXML browser instances running on it.
vvsm help	Displays a list of all available CLI commands with a brief description of each.

Powering Up a Speech Server

After a Speech Server is powered up, the Voice Server software installation is complete, and the system is rebooted, the Windows NT Service starts the SM Agent. The SM Agent reads the **sysmgmt.properties** configuration file for the **AUTO_START_ON_INIT** parameter which is set by default to *false*. To automatically start up upon reboot, you must change this parameter to *true* to enable the VoiceXML browsers.

Powering Down a Speech Server

Before performing an NT shutdown and powering off a Speech Server, issue the **vvs****m stop** or **vvs****m stop now** CLI command. When you issue **vvs****m stop**, end-users currently accessing the system via a telephone will not experience an abrupt cancellation in the middle of their call. The VoiceXML browsers will wait until the call dialog completes or 10 minutes elapse before shutting down. When you issue **vvs****m stop now**, all calls are terminated immediately and end-users currently accessing the system via a telephone will experience an abrupt cancellation in the middle of their call.

The **vvs****m stop** command is asynchronous and therefore returns before the stop operation has completed. To find out when all VoiceXML browsers have stopped, you may need to periodically check the VoiceXML browsers with the **vvs****m getstatus** command. Once all the VoiceXML browsers have stopped, it will be safe to perform an NT shutdown and subsequently power off the Speech Server.

Using the Voice Server Commands

System Management Naming of VoiceXML Browsers

In order to identify each VoiceXML browser started by the SM Agent, a unique logical name is assigned using the following format:

- vv** denotes a VoiceXML browser.
- c* the letter 'c' is replaced with an 'i' to indicate that the VoiceXML browser was started for inbound calls.
- dd* decimal number identifying the logical number of the VoiceXML browser.

For example, if a Voice Server is configured for 2 inbound calls, the logical names of the VoiceXML browsers on that Voice Server will be: **vvi00** and **vvi01**. This naming convention is also used in the SM log (**vvsma.log**) and alarm (**vvsm.alr**) files as shown below:

SM Log File

```
2000/Sep/22 07:40:53 vvsma smag SMAG0242E VoiceXML Browser "vvi00" ended due to an error.
```

Alarm Log File

```
2000/Sep/22 07:40:53 vvsma smag SMAG0242E VoiceXML Browser "vvi00" ended due to an error. It last reported a status of "exit" with a "ready" state of "false" and ran for approximately 20 seconds before ending with exit code of "-1". This browser was started 1 time(s). See the log files in directory "d:\VOICES~1\logs\vvi00_9999_2000Sep22_07.40.32.894\" for more information.
```

Properties for the **INBOUND_*n** parameter maps to the “*vvcd*d**” naming convention in the configuration file as well. [Table 6 on page 51](#) shows how this naming convention is assigned to two VoiceXML browsers.

Table 6. Naming Convention for VoiceXML Browsers

VoiceXML Browser vvi00	VoiceXML Browser vvi01
INBOUND_BARGE_IN0	INBOUND_BARGE_IN1
INBOUND_DUPLEX0	INBOUND_DUPLEX1
.	.
.	.
.	.
INBOUND_URL0	INBOUND_URL1

Starting the VoiceXML Browsers

This command is used to start the VoiceXML browsers after the Voice Server is installed and the system has been rebooted.

You can start the VoiceXML browsers using either of two methods:

- “[From the Command Line](#)” — See [page 51](#).
- “[From Tivoli GEM Console](#)” — See [page 53](#).

From the Command Line

To start the VoiceXML browser, issue the following CLI command:

```
vvsm start
```

where:

vvsm start launches the number of VoiceXML browsers specified by the **NUMBER_INBOUND_BROWSERS** parameter. These VoiceXML browsers will be ready to receive inbound calls.

The **vvsm start** command forces the system to read the configuration file when initially starting up or after the file has been modified.

It is also used to re-start the VoiceXML browsers if they were previously stopped.

Note: If the VoiceXML browsers are already running, the `vvsm start` command is ignored.

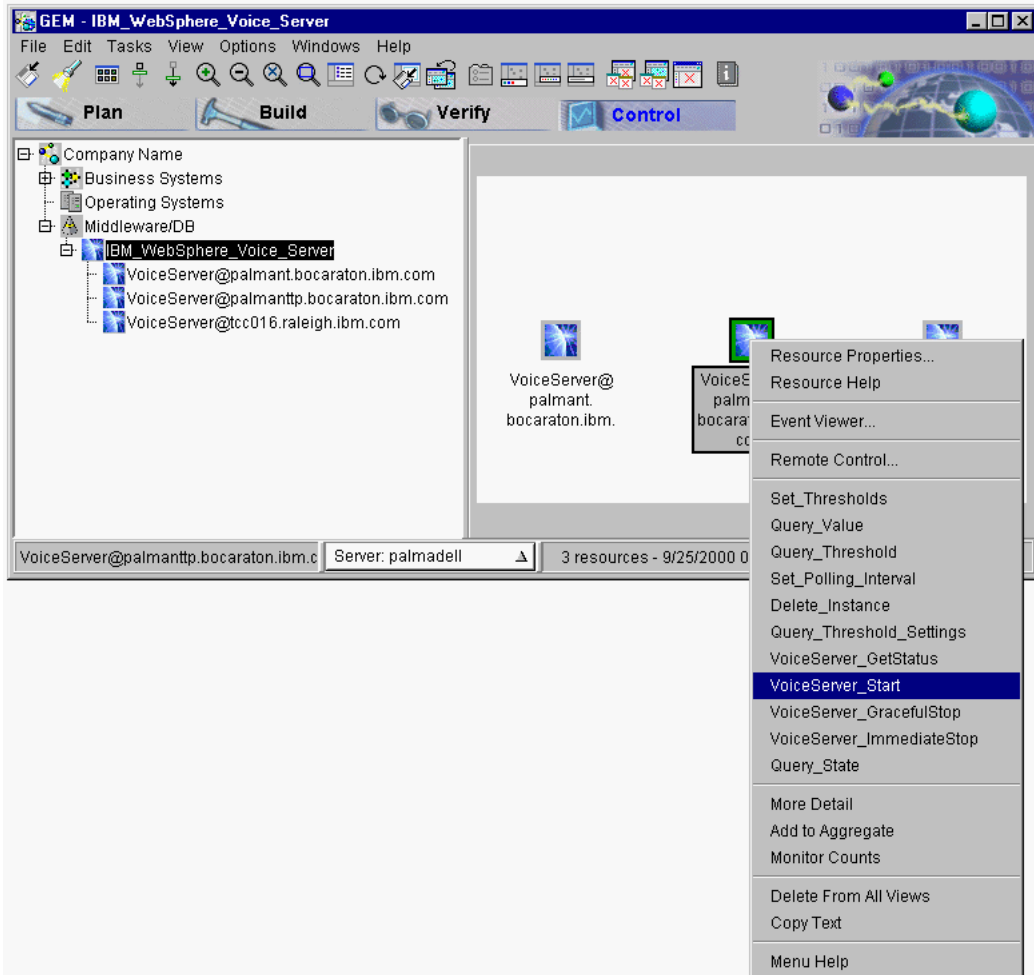
Table 7 shows status messages that are displayed for the `vvsm start` CLI command.

Table 7. vvsm start Status Messages

Message ID	Description
SMAG0012I	VoiceXML browsers are starting.
SMAG0013I	<n> inbound VoiceXML Browser(s) are started.
SMAG0200E	The VoiceXML browsers could not be started. See the SM Agent log file for more information.
SMAG0206E	The “start” operation was invoked, but no inbound VoiceXML browser(s) were able to start successfully. Examine the configuration file, correct any errors, and retry the application.
SMAG0231E	VoiceXML browsers are already started.
SMAG0232E	VoiceXML browsers are in the process of stopping from a previous request and cannot be interrupted. Please try this command again later when this operation has completed.
SMAG0240E	The VoiceXML browser {0} cannot be started due to the following exception {1}. Please examine the SM log file for more information.

From Tivoli GEM Console

You can start the system by selecting a Voice Server icon (Speech Server) and choosing **VoiceServer_Start** from the pull-down menu on the Tivoli GEM Console.



Checking the VoiceXML Browsers Status

Once you've attempted to start the system, you must make sure it is operational and that all VoiceXML browsers have been successfully launched.

You can check the VoiceXML browser status using either of two methods, depending upon your environment:

- “From the Command Line” — See [page 54](#).
- “From Tivoli GEM Console” — See [page 55](#).

You can obtain the status of your VoiceXML browsers at any time by using this command, even while the VoiceXML browsers are running. The system responds with the status message of one Voice Server. In case of an error (such as a VoiceXML browser that did not start), check the **vvsm.log** file to determine the possible cause of the problem. See [“Using the SM Log File Mechanism” on page 64](#).

From the Command Line

To check the system status, issue the following CLI command:

```
vvsm getstatus
```

where:

vvsm getstatus returns the status of a Voice Server

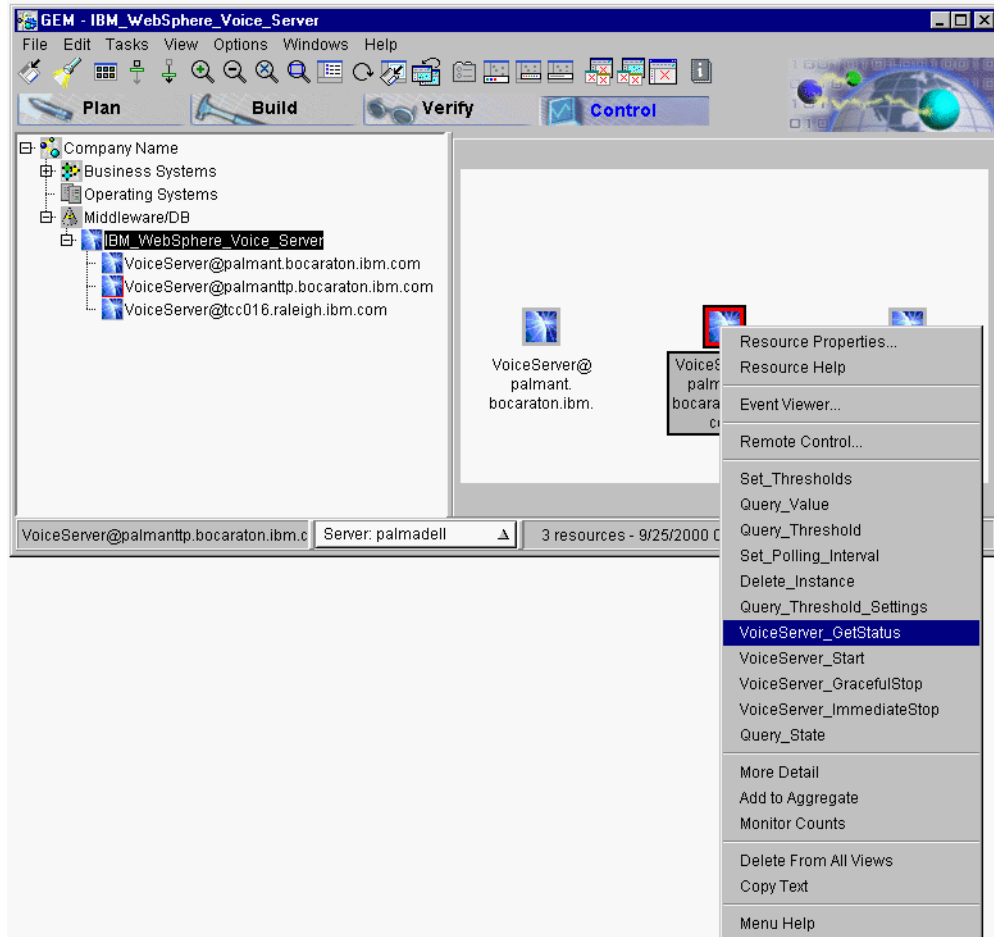
[Table 8](#) shows status messages that are displayed for the **vvsm getstatus** CLI command.

Table 8. vvsm getstatus Status Messages

Message ID	Description
SMAG0012I	VoiceXML browsers are starting.
SMAG0030I	VoiceXML Browser {vvcdd} is not started.
SMAG0031I	VoiceXML Browser {vvcdd} is operational. It reports a status of {1}.
SMAG0032I	VoiceXML Browser {vvcdd} is stopped.
SMAG0033I	VoiceXML Browser {vvcdd} is in an unknown state.
SMAG0015II	VoiceXML browsers are stopped.

From Tivoli GEM Console

You can check the system status by selecting a Voice Server icon (Speech Server) and choosing `VoiceServer_Getstatus` from the pull-down menu on the Tivoli GEM Console.



Getting Command Help

You can get Voice Server command help using either of two methods, depending upon your environment:

- “From the Command Line” — See [page 56](#).
- “From Tivoli GEM Console” — See [page 57](#).

From the Command Line

This command lists all available Voice Server commands, that is **vvsm** *command*, along with a brief description of each one. If you do not specify a parameter or if you specify an invalid parameter, the system displays an error message followed by usage information.

To get help for the **vvsm** commands, issue either of the following CLI commands:

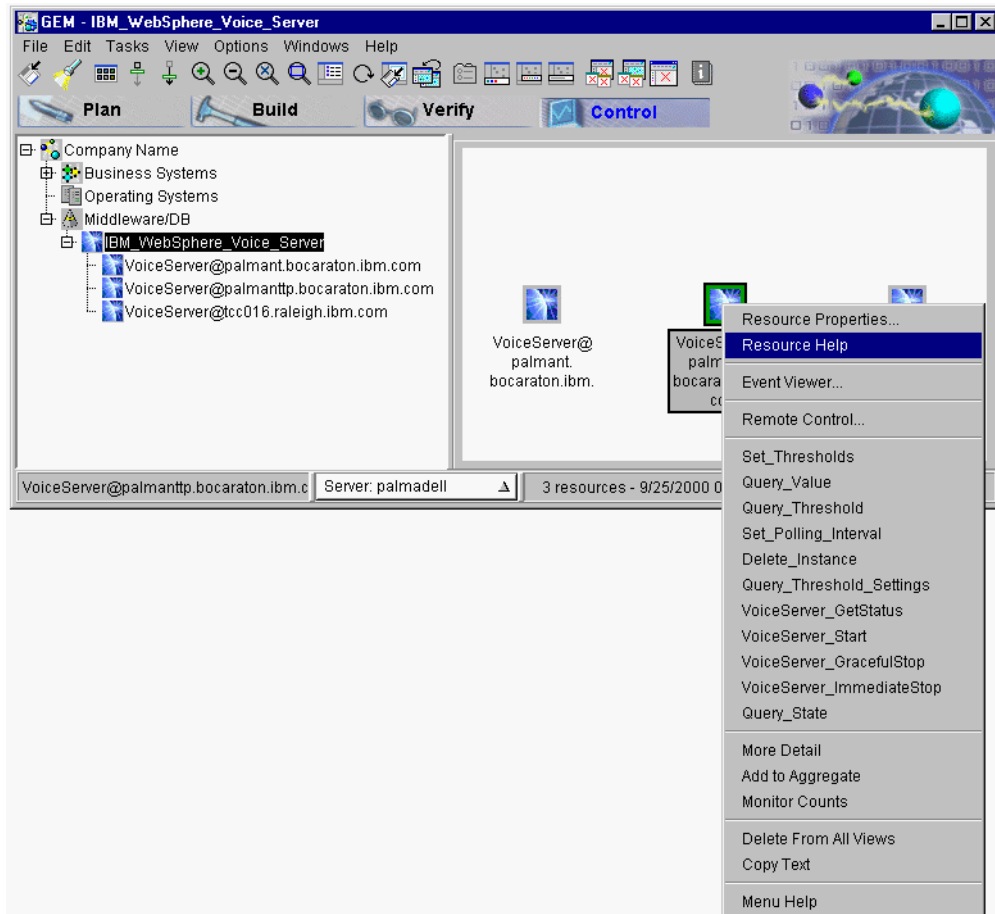
```
vvsm help
vvsm
```

where:

vvsm help returns a list of all available CLI commands

From Tivoli GEM Console

You can get system help for Voice Server commands by selecting a Voice Server icon (Speech Server) and choosing **Resource Help** from the pull-down menu on the Tivoli GEM Console.



Stopping the VoiceXML Browsers

You have the option to either shut down the VoiceXML browsers immediately or to wait until all current telephone calls complete. Log and alarm messages are written into the SM log and alarm files (**vvsm.log/vvsm.alr**) indicating the VoiceXML browsers were stopped.

Note: Only the VoiceXML browsers are stopped. The SM Agent does not shut itself down.

Issuing an Immediate Shutdown

You can immediately stop the VoiceXML browsers using either of two methods, depending upon your environment:

- “From the Command Line” — See [page 58](#).
- “From Tivoli GEM Console” — See [page 59](#).

From the Command Line

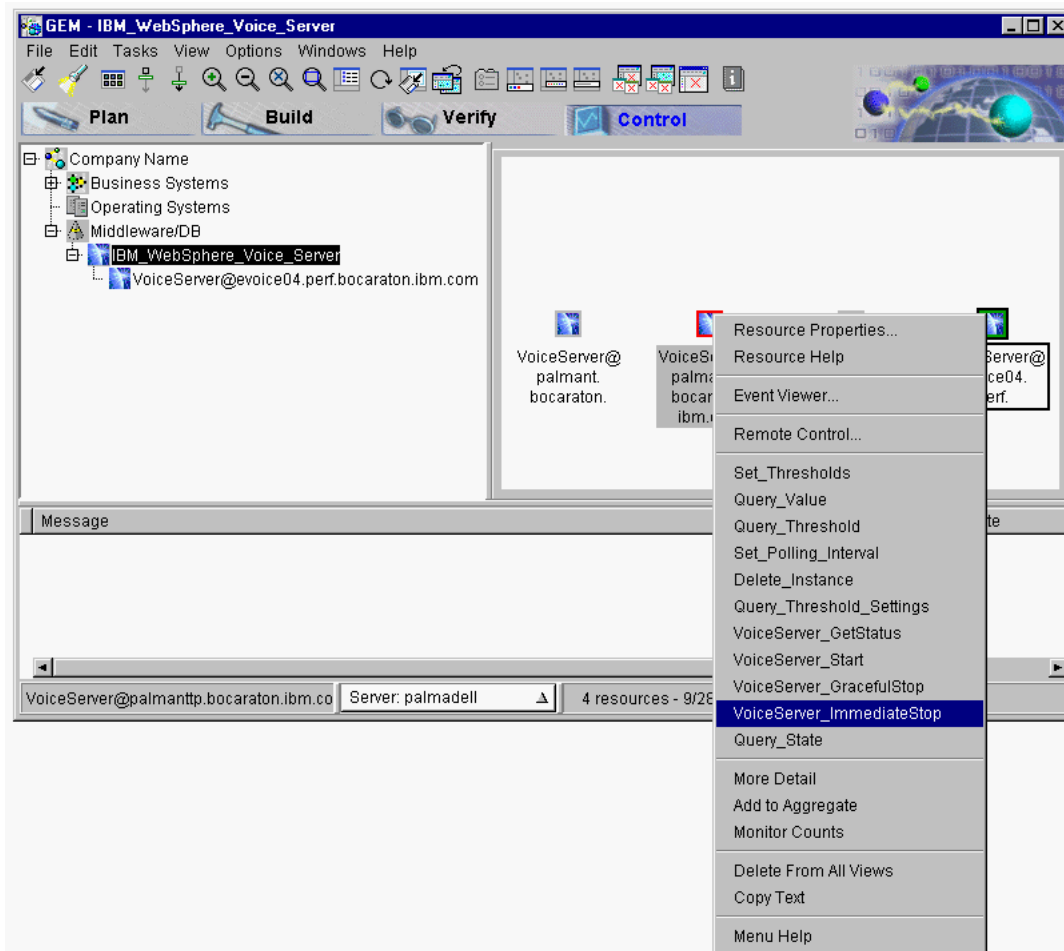
To immediately stop the VoiceXML browsers, issue the following CLI command:

```
vvsm stop now
```

vvsm stop now requests all VoiceXML browsers to immediately shut down. This command may stop a VoiceXML browser while in use and therefore calls in the “received call” state will terminate before the call dialog ends.

From Tivoli GEM Console

You can immediately stop a VoiceXML browser by selecting a Voice Server icon (Speech Server) and choosing **VoiceServer_ImmediateStop** from the pull-down menu on the Tivoli GEM Console.



Issuing a Graceful Shutdown

You can gracefully stop the VoiceXML browsers using either of two methods, depending upon your environment:

- “From the Command Line” — See [page 60](#).
- “From Tivoli GEM Console” — See [page 61](#).

However, if after issuing the graceful stop command a call is not completed within the specified time period (ten minutes), the VoiceXML browser will be forced to exit and the in-progress call is ended.

From the Command Line

To gracefully stop the VoiceXML browsers, issue the following CLI command:

```
vvsm stop
```

where:

vvsm stop requests all VoiceXML browsers to gracefully shut down. This request is asynchronous (that is, the CLI command ends before the VoiceXML browsers have stopped). Browsers that are in the “received call” state will not terminate until the call dialog ends, or a 10 minute timeout is “reached”.

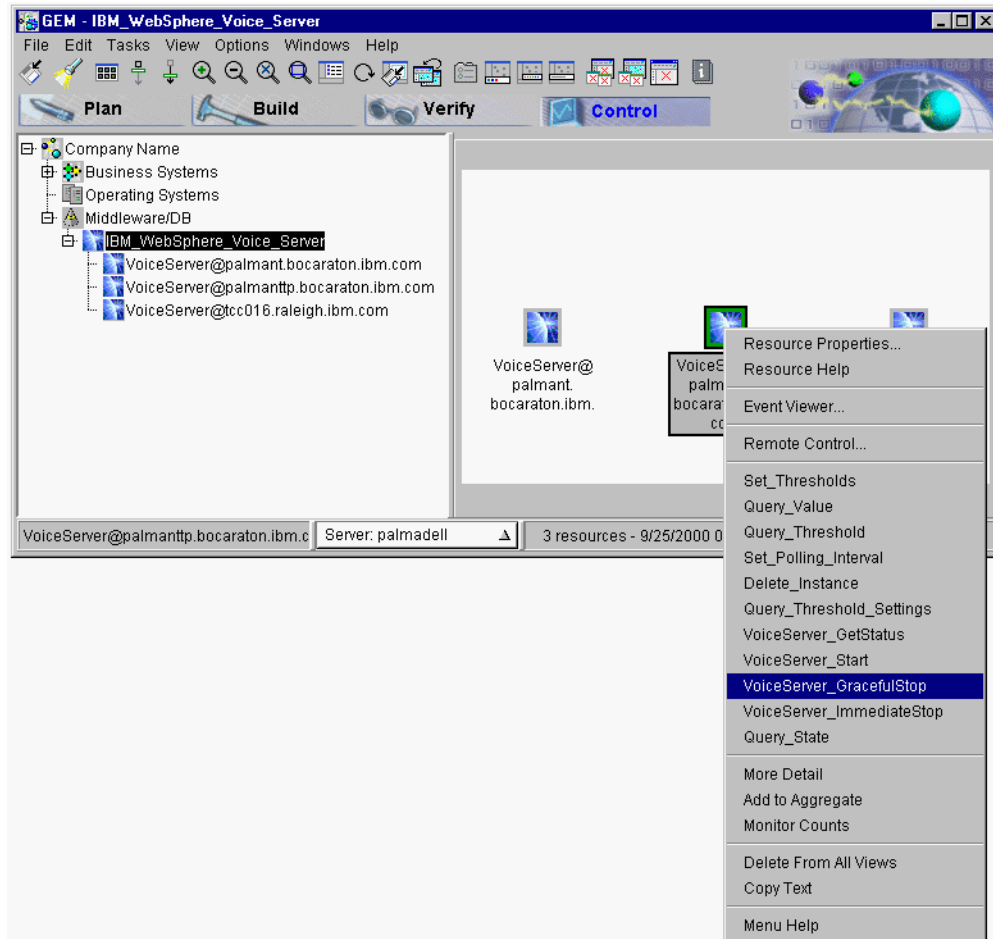
[Table 9](#) shows status messages that are displayed for the **vvsm stop** CLI command.

Table 9. vvsm stop Status Messages

Message ID	Description
SMAG0015I	VoiceXML browsers are stopped.
SMAG0230E	VoiceXML browsers are already stopped.
SMAG0233E	VoiceXML browsers are in the process of starting from a previous request and cannot be interrupted. Please try this command again later when this operation has completed.

From Tivoli GEM Console

You can gracefully stop a VoiceXML browser by selecting a Voice Server icon (Speech Server) and choosing **VoiceServer_GracefulStop** from the pull-down menu on the Tivoli GEM Console.



Monitoring and Troubleshooting a Voice Server

This chapter describes monitoring and troubleshooting information for a Voice Server. Also discussed are useful troubleshooting tips to help when an unexpected event or behavior occurs while monitoring and managing a Voice Server deployment environment.

This chapter discusses the following topics:

- “System Log Files” — See [page 63](#).
- “Troubleshooting a Voice Server” — See [page 71](#).

System Log Files

The three primary type of log files for the Voice Server include:

Table 10. System Log Files

File	Description
SM log file (vvsma.log)	The SM log file is your first source for information concerning Voice Server problems. This log indicates significant changes in the state of a Voice Server such as starting and stopping, as well as any error situation that may have been detected by the SM Agent.
SM alarm file (vvsma.alr)	The SM alarm file alerts you to an event or situation that you should be aware of.
VoiceXML browser log file (vxml.log)	The VoiceXML browser log file includes detailed information concerning the cause of an error. Normal execution tracing of events throughout the life of a call is also given.

Using the SM Log File Mechanism

All SM log file messages are written to the **vv_{sma}.log** file. The SM log file contains significant event information such as when the SM Agent attempts to start or stop a VoiceXML browser, or when errors occur. SM messages are presented in a single-line format for each event. Since the log file is an ASCII text file, you can browse it using any text editor.

Location of the SM Log File

The location of the SM Log file is specified by the **SYS_MGMT_OUTPUT_PATHNAME** parameter value which is pre-configured to *logs*. The maximum size of the file is specified by the **LOG_FILE_MAX_SIZE** parameter value which is pre-configured to *1*. When the log file reaches its defined maximum size, the SM Agent copies the messages to a backup file (**vv_{sma}bak.log**) and creates a new empty file (**vv_{sma}.log**) for logging new events. When the new log file reaches its defined maximum size, the messages are again copied to a backup file where the previous information is overwritten and therefore lost.

Format of the SM Log File

The format of the SM log file messages is: *date name component message*.

where:

<i>date</i>	The date when the log entry was created in <i>Year/Month/Day/Time</i> format.
<i>name</i>	The logical name of the component that created the entry. (vv_{sma} is the SM Agent master name.)
<i>component</i>	The name of the component that generated the log entry. (SMAG is the prefix for an SM Agent log entry.)
<i>message</i>	The message number and description indicating the nature of the problem.

Figure 4 contains an example of the **vvvsmma.log** file. In this example an error occurred in the VoiceXML application (**vxmml.log**).

```

2000/Sep/22 07:40:23 Starting the log file
2000/Sep/22 07:40:32 vvsma smag SMAG0041I The System Management "start" command was invoked.
2000/Sep/22 07:40:32 vvsma smag SMAG0012I The VoiceXML Browsers are starting.
2000/Sep/22 07:40:32 vvsma smag SMAG0013I 1 in-bound VoiceXML Browser(s) are started.
2000/Sep/22 07:40:53 vvsma smag SMAG0242E VoiceXML Browser "vvi00" ended due to an error.
It last reported a status of "exit" with a "ready" state of "false" and ran for approximately 20 seconds before ending with exit code of "-1". This browser was
started 1 time(s). See the log files in directory "d:\VOICES-1\logs\vvi00_9999_2000Sep22_07.40.32.894\" for more information.
2000/Sep/22 07:40:53 vvsma smag SMAG0254E A history of the output received from VoiceXML Browser "vvi00" before it ended in error follows.
2000/Sep/22 07:40:53 vvsma smag SMAG0255E VoiceXML Browser "vvi00" wrote line 1 of 4: "Registering
IBM engines in d:\progra-2\javasoft\jre\1.3\lib\speech.properties".
2000/Sep/22 07:40:53 vvsma smag SMAG0255E VoiceXML Browser "vvi00" wrote line 2 of 4: "S:
initializing application".
2000/Sep/22 07:40:53 vvsma smag SMAG0255E VoiceXML Browser "vvi00" wrote line 3 of 4: "E: Event error.badfetch: Unsupported XML version.".
2000/Sep/22 07:40:53 vvsma smag SMAG0255E VoiceXML Browser "vvi00" wrote line 4 of 4: "S: exit".
2000/Sep/22 07:40:53 vvsma smag SMAG0247E VoiceXML Browser "vvi00" command line launched was
"d:\progra-2\javasoft\jre\1.3\bin\javaw.exe -Dvxmml.gui=false -Dvxmml.rate=8000 -Dsm.name=vvi00
.
.
.
2000/Sep/22 07:41:02 vvsma smag SMAG0043I The System Management "stop now" command was invoked.
2000/Sep/22 07:41:02 vvsma smag SMAG0014I The VoiceXML Browsers are stopping.
2000/Sep/22 07:41:03 vvsma smag SMAG0015I The VoiceXML Browsers are stopped.

```

Figure 4. SM Log File Example

Using the SM Alarm File Mechanism

All SM alarm file messages are written to the **vvvsmma.alr** file. The information in the SM alarm file is a subset of the SM log file. This file contains a list of events that are described in fuller detail in the SM log file. After reading the SM alarm file to determine if an error occurred, you should refer to the SM log file for exact details on the error. Alarm messages are presented in a single-line format for each event. Since the log file is an ASCII text file, you can browse it using any text editor.

Location of the SM Alarm File

The location of the SM Alarm file is specified by the **SYS_MGMT_OUTPUT_PATHNAME** parameter value which is pre-configured to *logs*. The maximum size of the file is specified by the

LOG_FILE_MAX_SIZE parameter value which is pre-configured to *1*. When the alarm file reaches its defined maximum size, the SM Agent copies the messages to a backup file (**vvsmabak.log**) and creates a new empty file (**vvvma.log**) for logging new events. When the new log file reaches its defined maximum size, the messages are again copied to a backup file where the previous information is overwritten and therefore lost.

Format of the SM Alarm File

The format of this file is: *date severity hostname component message*.

where:

<i>date</i>	The date when the alarm entry was created in <i>Month/Day/Time/Year</i> .
<i>severity</i>	Letter defining the severity of the alarm. D = Disaster E = Error W = Warning I = Informational
<i>hostname</i>	The TCP/IP host name of the Speech Server.
<i>component</i>	The name of the component that generated the log entry. (SMAG is the prefix for an SM Agent log entry.)
<i>message</i>	The message number and description indicating the nature of the problem.

Figure 5 shows an example of the **vvvma.alr** SM alarm file.

```
...
2000/Sep/22 07:40:23 vvsma smag SMAG0010I The System Management Agent for the IBM WebSphere
Voice Server is starting.
2000/Sep/22 07:40:32 vvsma smag SMAG0012I The VoiceXML Browsers are starting.
2000/Sep/22 07:40:32 vvsma smag SMAG0013I 1 in-bound VoiceXML Browser(s) are started.
2000/Sep/22 07:40:53 vvsma smag SMAG0242E VoiceXML Browser "vvi00" ended due to an error. It last
reported a status of "exit" with a "ready" state of "false" and ran for approximately 20 seconds before ending
with exit code of "-1". This browser was started 1 time(s). See the log files in directory
"d:\VOICES~1\logs\vvi00_9999_2000Sep22_07.40.32.894\" for more information.
```

Figure 5. SM Alarm File Example

Using the VoiceXML Browser Log File Mechanism

All trace messages are written to the **vxml.log** file. The VoiceXML log file is in a subdirectory which is unique for each telephone call or VoiceXML browser instance.

Location of the VoiceXML Browser Log File

The directory name is: `//<install-directory> + SYS_MGMT_OUTPUT_PATHNAME + VoiceXML Browser logical name (vvcdd) + "_" + INBOUND_H323_ENDPOINT_ALIASn + "_" + date + time`

For example: D:\VoiceServer\+ logs\+ vvi00 + "_" + 9999 + "_" + 2000Sep22+ 07.40.32.894

directory name =: d:\VOICES~1\logs\vvi00_9999_2000Sep22_07.40.32.894

Note: The SM logs display directory names in Windows 8.3 format (8 character name plus 3 character extension). For example, the “VoiceServer” directory is displayed as “VOICES~1”.

Format of the VoiceXML Browser Log File

The format of each entry in the **vxml.log** file is `hh:mm:ss.sss code: message` where:

<i>hh:mm:ss.sss</i>	The timestamp in 24 hour format.
<i>code</i>	The single-character indicating the type of log entry.
<i>message</i>	The trace message.

[Table 11](#) shows all the types of VoiceXML browser log file entries.

Table 11. VoiceXML Browser Log Entries

Type	Description
A	Logged when the VoiceXML browser detects audio input, but the recognizer does not return a recognized phrase. This may be due to breathing or background noise. The message column contains the audio level messages.
C	Logged when the computer plays a prompt. The message column displays the prompt text.
E	Logged when the VoiceXML browser throws an event. The message column indicates the type of event that was thrown.
F	Logged when the speech browser fetches a resource such as a grammar file, an audio file, or a script. The message column contains the URI of the file and where the file was fetched from (either the server or cache).
H	Logged when the user responds using voice input. The message column displays the word or phrase that was recognized by the IBM ViaVoice Speech Recognition engine.
K	Provides further information about cache activity.
V	Logged when a VoiceXML browser fetches a .vxml file. The message file contains the URI of the file and where the file was fetched from (either the server or cache).
S	Indicates system information including the version of the VoiceXML browser, time, and the status of the telephony connection. A message is logged when the system is ready to receive a call, when an incoming call is ringing, when an incoming call is answered, or when a call is dropped. This information is provided only in the vxml.log file, it is not written to the Java console.
X	Specifies the version of the JVM. This information is provided only in the vxml.log file, it is not written to the Java console.
?	Logged when the IBM ViaVoice Speech Recognition engine determines that the user said something, but the confidence level is not high enough to justify using the results. In response, the speech browser throws a nomatch event. The recognized string is displayed in the message column.

Figure 6 shows an example of the VoiceXML browser log file. The messages indicate that the failure of the VoiceXML browser was caused by an error in the VoiceXML application itself. We recommend that you work closely with the application developer to correct the problem.

14:41:12.759 S: Starting V000824 at Wed Sep 06 14:41:12 EDT 2000
14:41:12.799 X: Java: IBM Corporation 1.2.2
14:42:26.465 S: initializing application
14:42:26.485 V: http://testsystem.ibm.com/voicexmldemo/main.vxml (fetch get)
14:42:27.076 K: put 000000 http://testsystem.ibm.com/voicexmldemo/main.vxml
14:42:27.397 F: http://testsystem.ibm.com/voicexmldemo/bye.gram (prefetch get)
14:42:27.607 K: put 000001 http://testsystem.ibm.com/voicexmldemo/bye.gram
14:42:31.653 C: Welcome to the I B M Web Sphere and Lotus Domino Speech Demo. At anytime you can say *Main Menu* to return here, or say *Repeat* to replay system messages.
14:42:32.113 V: #main
14:42:32.153 C: Main Menu: The Book Store dot com. Banking. Calendar.
14:42:32.203 C: or Exit
14:43:58.588 A: listening
14:44:02.844 ?: The Store (still speaking)
14:44:16.193 C: Welcome to The Bookstore dot com. Return here any time by saying: *The Book Store dot com*.
Please choose one of the following searches:
Author. Title. Best Sellers.
14:44:16.203 C: or say *Exit*.
14:44:17.054 A: Audio level (0.5)
14:44:17.275 ?: Best (still speaking)
14:44:23.674 H: Author
14:44:24.004 C: Please state the author's first and last name.
14:44:36.282 H: Tom Brokaw.
14:44:36.462 V: http://testsystem.ibm.com/servlet/VXMLByAuthorServlet?authorname=Tom+Brokaw. (fetch get)
14:44:37.393 C: Say one of these options: *Read jacket*, *Order Book*, *New Search*, or *Exit*
14:44:41.580 H: Best
14:44:41.590 V: http://testsystem.ibm.com/servlet/VXMLBestSellerServlet (fetch get)
14:44:41.590 F: send cookie sesessionid=OEVWUZAAAAAAECKTFEBAAAA for testsystem.ibm.com
14:44:41.800 C: Choose one of these five Best Sellers by saying its *number*, *title* or *author*.
 1. Nolan Ryan, Strikeout King by Nolan Ryan
 2. River's End by Nora Roberts
 3. The Long Road Home by Danielle Steele

4. The Greatest Generation by Tom Brokaw

5. The Elegant Universe by Brian Green

14:44:57.623 C: You selected Nolan Ryan, Strikeout King by Nolan Ryan. The bookstore dot com's price is \$29.00. This is a savings of \$2.00 over regular retail price.

14:44:57.663 V: http://testsystem.ibm.com/voicexmldemo/bookinfo.vxml (cache)

14:44:57.713 C: Say one of these options: *Read jacket, Order Book, New Search, or Exit*

14:45:16.209 ?: Order Book

14:45:31.662 ?: Author

14:45:31.662 C: From here, you can get information about the selected book, say *Main Menu* to return to the Main Menu, or say *The Bookstore dot com* to restart the book store application. You can leave the demo by hanging up or saying *Exit*. To hear a prompt again, say *Repeat*. To continue, please say one of the following choices: *Read Jacket, Order Book, New Search, or Exit*

14:45:37.179 S: completed call

14:45:37.510 K: clean cache 1

14:45:37.510 K: clean cache 2

14:45:37.570 K: clean cache 3

14:45:37.570 K: lock retries 0

14:45:37.570 S: exit

Figure 6. VoiceXML Browser Log File Example

Finding a VoiceXML Browser Log File

If a status message is displayed indicating that the VoiceXML browser has stopped, you can examine the VoiceXML browser log file to find out what may have happened.

The VoiceXML browser log file includes the following information:

- The cause of the error (at a system management level).
- The failing VoiceXML browser logical name (*vvi00*).
- The directory name where the failing VoiceXML browser logs are stored such as:
d:VOICES~1\logs\vvi00_9999_2000Sep22_07.40.32.894\

Troubleshooting a Voice Server

This section defines unexpected behaviors or events that may occur with the VoiceXML browsers and offers possible solutions or work arounds for handling these events.

These behaviors or events can include:

- “VoiceXML Browser Failed to Start” — See [page 71](#).
- “VoiceXML Browser is Hung” — See [page 72](#).
- “VoiceXML Browser Busy Message” — See [page 72](#).
- “No Ring Before Answering a Call” — See [page 73](#).
- “Long Silence Before Initial Greeting” — See [page 73](#).

VoiceXML Browser Failed to Start

Problem

A VoiceXML browser fails to start and the SM Agent will not attempt to restart it.

Solution

Determine the cause of the failure for the VoiceXML browser by examining the SM log file.

SMConfig Did Not Show Error

Problem

The VoiceXML browser did not start and the SMConfig utility did not show an error.

Solution

In most cases, it is an error in the file pointed to by the **INBOUND_URL*n*** parameter. Look in the **vxml.log** file for the failed VoiceXML browser to determine the cause of the problem.

VoiceXML Browser is Hung

Problem

If a VoiceXML browser is not ready to handle an incoming call within 2 minutes after the system has started it up, it is considered hung and the SM Agent will end the process.

Solution

Check the hardware requirements in the **readme.txt** to verify that you are not running more VoiceXML browsers than your CPU can support.

VoiceXML Browser Busy Message

Problem

The VoiceXML browser busy status message is “Waiting for a Call”, but the phone rings continuously or a busy signal is sent.

Solution

1. Verify the phone line is connected to the VoIP gateway and that it is active.
2. Verify the VoiceXML browser is registered with the gatekeeper (if you have one) and the terminal endpoint alias is correct.

Note: The terminal endpoint alias must be a telephone number based on the E164 standard.

3. Verify that the VoIP gateway is registered with the gatekeeper.

4. Ensure the IP name and address translation is functioning correctly by installing a DNS on your network and adding the Speech Server, VoIP gateway, and VoIP gatekeeper (if you have one) in the DNS so the name/IP resolution can occur.
5. You can add the address and name of the Speech Server, Web application server, VoIP gateway and H.323 gatekeeper in the NT HOSTS file (winnt/system32/drivers/etc/HOSTS) to provide quick resolution of the names to the required IP addresses in the same way as the DNS.
6. Check the hardware requirements to verify that you are not running more VoiceXML browsers than your CPU can support to prevent CPU overload. See the **readme.txt** file for hardware information.

No Ring Before Answering a Call

Problem

There is no ring before answering a telephone call.

Solution

This is normal with the Voice Server. The call is immediately answered following a short period of silence and the URL associated with the VoiceXML browser is started.

Long Silence Before Initial Greeting

Problem

A long silence occurs before the VoiceXML application starts playing the initial greeting.

Solution

You may want to check that the problem is not your network. Ensure the IP name and address translation is functioning correctly by installing a DNS on your network and adding the Speech Server, VoIP gateway, and VoIP gatekeeper (if you have one) in the DNS so the name/IP resolution can occur.

This appendix describes the configuration of *Cisco VoIP gateways* and *Cisco H.323 gatekeepers* in a Voice Server deployment environment. It is recommended that you work closely together with the gateway expert in order to properly configure the telephony environment and plan and configure load distribution.

Note: Refer to the Cisco documentation for information about Cisco VoIP gateway configuration commands.

The following topics are discussed in this chapter:

- “VoIP Gateway and H.323 Gatekeeper Interaction” — See [page 75](#).
- “System Administrator’s Role” — See [page 77](#).
- “Gateway Expert’s Role” — See [page 78](#).
- “VoIP Gateway and H.323 Gatekeeper Configuration” — See [page 78](#).

VoIP Gateway and H.323 Gatekeeper Interaction

Cisco VoIP gateways are used in conjunction with Cisco H.323 gatekeepers to provide a connection between a PTSN and an IP network. A VoIP gateway and H.323 gatekeeper system have the following responsibilities:

- Translate data received over ISDN PRI T1 lines (E1 lines in Europe, CAS or analog) into the appropriate packetized formation.
- Route a call to an available H.323 terminal endpoint or VoiceXML browser in the IP network using the call distribution scheme.

The VoIP gateway receives a message indicating an incoming call. This message includes the called number. The VoIP gateway uses a translation table that contains rules for the translation of the called number. The translation table also contains a specific number of VoIP dial peers that have a one-to-one correspondence with each VoiceXML browser.

When a VoIP gateway detects a new incoming call, it scans the list of defined VoIP dial peers that match the called number. The VoIP gateway continues to scroll through that list until an available matching VoIP dial peer is found. The VoIP gateway translates the called number for that dial peer into the browser's alias using the appropriate translation rule. The VoIP gateway then requests the IP address and port number of the VoiceXML browser from the H.323 gatekeeper.

The H.323 gatekeeper may reject the request for VoiceXML browser information if the VoiceXML browser is not ready to accept a call. This can occur if, for example, a VoiceXML browser is in the process of restarting after completing the previous call. When a VoIP gateway receives a rejection to its request for VoiceXML browser information, it recognizes that the VoIP dial peer is unavailable. The H.323 gateway continues to scroll through the VoIP dial peer list until it finds an available one.

Theoretically, an unlimited number of calls can be connected through the same VoIP dial peer. However, the connection should be limited to one for the Voice Server so there is a one-to-one mapping between VoIP dial peers and VoiceXML browsers. Figure 7 shows the interaction between an incoming call, VoIP gateway, and H.323 gatekeeper.

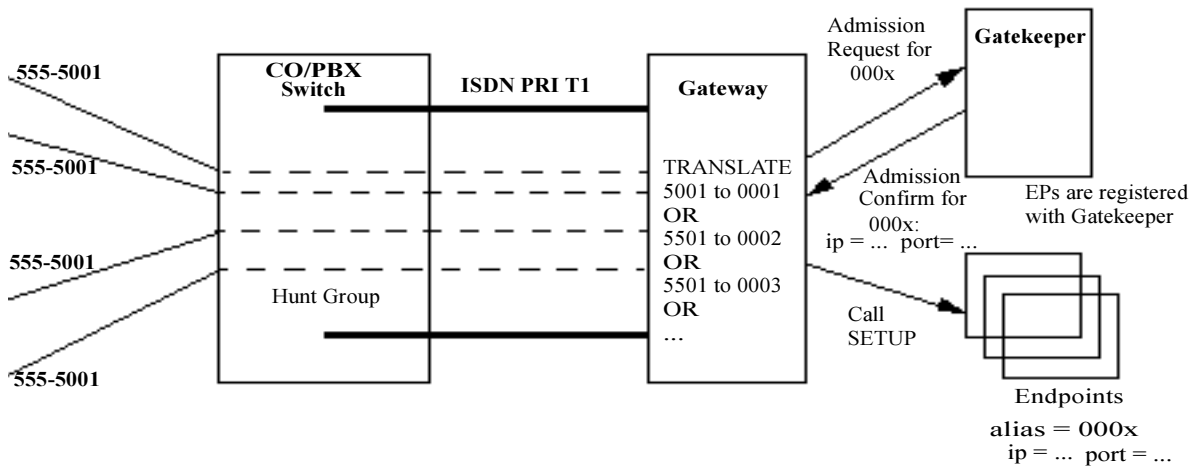


Figure 7. VoIP Gateway and Gatekeeper Interaction

System Administrator's Role

You will need to provide information to the gateway expert about your telephony services and IP network. To obtain this information you can ask the following questions:

Sample answers are based on [Table 13 on page 80](#).

- How many Speech Servers are in my IP network?
There are 3. Server1, Server2, and Server3.
- How many VoiceXML browsers per Speech Server?
There are 5 for Speech Server1, 5 for Speech Server2, and 5 for SpeechServer3.
- How many VoiceXML applications are supported?
There are 3. App1, App2, and App3.
- What VoiceXML application will run on what VoiceXML browsers?
Server1 - 5 running App1.
Server2 - 5 running App1.
Server3 - 2 running App2 and 3 running App3.
- What telephone numbers are assigned to these applications?
555-5001 for App1.
555-5002 for App2.
555-5003 for App3.

Telephony services information includes:

- Types of phone lines coming into the gateway (ISDN PRI, analog, CAS, etc.)
- Phone numbers
- Hunt groups (number of channels available per phone number).
- Which telephone numbers to assign to each specific application.

After the call distribution scheme is complete, the system administrator should receive the assigned aliases from the gateway expert and configure the VoiceXML browsers according to the call distribution scheme. Once the load balancing is completed by the gateway expert, he/she will inform you of the assigned aliases for each VoiceXML browser. See [Table 12 on page 79](#).

Gateway Expert's Role

The gateway expert uses the information you gave him/her to:

- Define the H.323 terminal endpoint aliases.
- Assign terminal endpoint aliases to the VoiceXML browsers.
- Define the call distribution scheme for load balancing configuration.
- Create the translation table.

VoIP Gateway and H.323 Gatekeeper Configuration

Defining and Assigning Aliases

The format of an H.323 alias is described in the H.323 standard. An example of an alias name is an E-mail address or a telephone number. An alias should be defined with a logical meaning within your network context. In the example below, we assigned the following aliases to App1, App2, and App3 beginning with:

- **1000** for VoiceXML browsers that run App1.
- **2000** for VoiceXML browsers that run App2.
- **3000** for VoiceXML browsers that run App3.

The results are:

- The five VoiceXML browsers running App1 on Server1 have aliases **1001** through **1005**.
- The five VoiceXML browsers running App1 on Server2 have aliases **1006** through **1010**.
- The two VoiceXML browsers running App2 on Server3 have aliases **2001** and **2002**. The three VoiceXML browsers running App3 on Server3 have aliases **3001** through **3003**.

Configuring for Load Distribution

Load distribution is implemented according to the VoIP gateway's configuration. Load distribution is only applied to calls that connect to the same application which is distributed across multiple servers. Each incoming call is routed to an available VoiceXML browser running on the Voice Server that has the least number of busy VoiceXML browsers. By using this methodology, the load required for incoming calls is kept balanced. Therefore, a VoiceXML browser will not be allocated on a server until each of the servers that participate in the load distribution have an equal load.

Load distribution can be accomplished by assigning *preferences* to the VoiceXML browsers. Theoretically, a stack of layers exist that are numbered 0 through n , where n is the maximum number of VoiceXML browsers that can reside on a participating Speech Server. Each VoiceXML browser is located on a specific layer which is spread across all participating Speech Servers. [Table 12](#) shows how each layer is spread across **Server1** and **Server2**, resulting in five layers for each Speech Server.

Table 12. VoiceXML Browser Load Distribution Scheme

Preferences	Server1	Server2
0	Browser ₁₁	Browser ₂₁
1	Browser ₁₂	Browser ₂₂
2	Browser ₁₃	Browser ₂₃
3	Browser ₁₄	Browser ₂₄
4	Browser ₁₅	Browser ₂₅

Preferences can be consecutively assigned to these layers beginning with zero (0), which has the highest priority. To achieve load distribution for App1, the next available VoiceXML browser on the current preference layer should be allocated. That is, VoiceXML browsers from the next-to-current layer are not allocated until all the VoiceXML browsers from the current layer are busy. Control over allocation of VoiceXML browsers within the same layer is not required.

[Table 13](#) summarizes all of the information needed to configure a VoIP gateway including configuration of the translation rules and VoIP dial peers.

Table 13. VoIP Gateway Configuration Information

Telephone Number Called	Runs Application	Resides On	Translated to Alias	Preference Number
555-5001	App1	Server1	1001	0
			1002	1
			1003	2
			1004	3
			1005	4
		Server2	1006	0
			1007	1
			1008	2
			1009	3
			1010	4
555-5002	App2	Server3	2001	default
555-5003	App3		2002	default
			3001	default
			3002	default
			3003	default

Specifying Load Distribution

Load distribution is provided by the static configuration of the VoIP gateway. Load distribution minimizes the number of VoiceXML browsers on the same Speech Server that are answering calls at the same time, and distributes the load amongst all available Speech Servers.

The one-to-one mapping between VoiceXML browsers and VoIP dial peers allows the VoIP gateway to determine the exact destination point when a particular VoIP dial peer is chosen. As a result, a VoiceXML browser is selected as soon as the VoIP gateway chooses the VoIP dial peer. The selection occurs even if the VoIP gateway doesn't know the actual location of the VoiceXML browser. The IP address and port number are provided by the H.323 gatekeeper in reply to the request. Therefore, the VoIP gateway should ideally be configured to choose the VoIP dial peers in a specified order.

The Cisco rotary call pattern feature provides a fail-safe environment for establishing calls during network failures. Multiple VoIP dial peers are defined to match the same telephone number. The feature can be used on the Cisco VoIP gateway to specify a VoIP dial peer selection preference. The order preference can be assigned to the VoIP dial peers, using the preference command so that destination points are tried in a predefined order.

When a call arrives, the VoIP gateway looks through the configured dial peers and creates a list, in preference order, of those that match the dialed number. Using this list, the VoIP gateway then goes from one VoIP dial peer list entry to another until it finds one that is available. Each VoIP dial peer translates the called number into a unique alias that maps to the VoiceXML browser. Because an alias is manually assigned to a VoiceXML browser, subsequent aliases may refer to VoiceXML browsers that are located on different Speech Servers.

The VoIP gateway has control over which specific VoiceXML browser answers the call if all previous VoIP dial peers and their corresponding VoiceXML browsers are busy. The VoIP gateway does not have access to the VoiceXML browser's physical location, since its IP address and port are available only after the H.323 gatekeeper confirms the request. However, the VoIP gateway does have access to the statically predefined order of VoIP dial peers and the one-to-one mapping between VoIP dial peers and VoiceXML browsers, and uses that to implement load distribution.

Configuring a VoIP Gateway

The called telephone number and translated alias are used to create the translation rules for a specific VoiceXML browser. [Figure 8](#) shows a translation rules table based on the "555-5001" number which is translated into one of ten numbers, 1001 through 1010. Refer to [Table 13 on page 80](#).

```
translation-rule 1001 Rule 1 5555001 1001
translation-rule 1002 Rule 1 5555001 1002
.
.
.
translation-rule 1010 Rule 1 5555001 1010
translation-rule 2001 Rule 1 5555002 2001
translation-rule 2002 Rule 1 5555002 2002
translation-rule 3001 Rule 1 5555003 3001
translation-rule 3002 Rule 1 5555003 3002
translation-rule 3003 Rule 1 5555003 3003
```

Figure 8. Translation Rules

Specifying and Configuring VoIP Dial Peers

The contents of the **Translated to Alias** and **Preference Number** columns in [Table 13 on page 80](#) are used to create the VoIP dial peers. The number of VoIP dial peers should match the number of VoiceXML browsers running in the system. See [Figure 9 on page 83](#).

- **dial-peer voice *number* voip**

This command defines the VoIP dial peer.

Several commands are required for each defined VoIP dial peer to ensure proper call routing. These commands include:

- **destination-pattern *string***

The *string* is used to match dialed digits to a dial peer. The dial peer is then used to complete the call.

- **session target *ras***

Used to specify a network-specific address for the dial peer. *ras* indicates that the VoIP gateway is to use the H.323 RAS protocol to obtain information from the H.323 gatekeeper to specify where the call is to be routed.

- **translate-outgoing called *number***

The called number is translated into another number by using the translation rule specified by *number*. The VoIP gateway uses the translated number when it sends admission requests to the H.323 gatekeeper when a call arrives.

- **max-conn *number***

Defines the maximum number of connections used simultaneously through a particular VoIP dial peer. This command limits the number of these type of connections to one.

- **preference *number***

Specifies the preference order for matching dial peers.

```
...
dial peer voice 1010 voip
  max-conn 1
  destination-pattern 5555001
translate-outgoing called 1
  session target ras
  codec g711ulaw
  no vad
...
```

Figure 9. VoIP Dial Peer Configuration Example



This appendix identifies performance issues for the IBM WebSphere Voice Server. This chapter discusses the following topics:

- [“Is Your Server Suitable?”](#) — See [page 85](#).
- [“What Factors Affect Your Performance?”](#) — See [page 86](#).
- [“What About Performance Degradation?”](#) — See [page 87](#).

Is Your Server Suitable?

Ensure the processor on your server meets or exceeds the following minimum requirements for your Voice Server software:

- Intel (R) Pentium (R) 550 MHz processor
- CD-Rom Drive
- 256 MB RAM
- Network hardware, as needed
- 300 MB available disk space

Is Sufficient Memory Installed?

Ensure sufficient memory is installed based on the number of phone lines, application resources, number of simultaneous VoiceXML browser sessions, communications program, protocol, etc. The amount of required memory is dependent upon your VoiceXML application and the number of simultaneous sessions you want to support.

Note: A session consists of all interactions between the VoiceXML browser, the user, and the Web application server. The session starts when the VoiceXML browser starts, continues through dialogs and the associated document transitions, and ends when the VoiceXML browser exits.

Our testing has shown that the maximum number of sessions that can be supported by a configuration

is application-dependent and limited both by available processor capacity and RAM.

Is the Correct Version of NT Services Installed?

Ensure you have installed Microsoft Windows NT Version 4 Service Pack 6a, including all the components necessary to support your proposed configuration.

Is the Communications Software Installed?

Ensure you have installed any communications software you plan to use to communicate with your Web server and/or enterprise database server (Sun JRE, HTTP, etc.).

Do You Have All the Required HW and SW?

Ensure you have all the hardware required for your proposed voice server configuration, together with any associated support software.

What Factors Affect Your Performance?

The performance of your VoiceXML application on your system will depend upon the following variables:

- Your VoiceXML application.
- The complexity of the speech grammars within the application.
- The number of simultaneously active telephony sessions.
- The average length of calls made to the system.
- The expected arrival time of calls made to the system.
- The speed and number of the CPUs in the server.
- The amount of RAM available on the server.
- The spare capacity to ensure a VoiceXML browser is always available to accept a call.

Therefore, to determine the capacity and performance of the Voice Server when running a specific application is a complex undertaking, and the previous information should only be used as a general guideline. You will need to plan and design your specific configuration to determine the actual capacity and capability of your application.

What About Performance Degradation?

If you exceed the number of sessions your configuration can support, one or several of the following conditions may occur:

- excessively long response times
- speech recognition errors
- choppy or unpleasant audio output to the end user
- prematurely terminated sessions

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
USA

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department T01B
3039 Cornwallis Road
Research Triangle Park, NC 27709-2195
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

COPYRIGHT LICENSE

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

If you are viewing this information in softcopy, the photographs and color illustrations may not appear.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

IBM

ViaVoice

WebSphere

Intel and Pentium are trademarks of Intel Corporation in the United States and other countries.

Microsoft, Windows, Windows NT, are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Java and all Java-based trademarks and logos are either registered trademarks or trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Other company, product, and service names may be trademarks or service marks of others.

Glossary

This glossary defines terms and abbreviations used in this publication. If you do not find the term you are looking for here, refer to *The IBM Dictionary of Computing*, SC20-1699, New York: McGraw-Hill, copyright 1994 by International Business Machines Corporation. Copies may be purchased from McGraw-Hill or in bookstores.

active grammar

A speech grammar that the speech recognition engine is currently listening for. One or more grammars will be active at any time, and the content of the active grammar(s) defines the user utterances that are valid in a given context.

administrator access

Super-user access abilities (required to install or uninstall the Voice Server).

application

A set of related VoiceXML documents that share the same application root document.

barge-in

A feature of full-duplex environments that allows the user to interrupt computer speech output. See also “[full-duplex](#)”.

call legs

Voice calls that are broken down into discrete segments in order to properly identify, process, and forward the calls.

channel associated signaling (CAS)

Type of T1 service.

Cisco 1750 /2600

Vendor-specific routers that function as VoIP gateways and are used in the deployment environment to route calls from the PSTN/PBX/GSM to the VoIP network.

CLI

Command Line Interface

cryptology

Science of keeping data secure. It allows you to store information or to communicate with other parties while preventing non-involved parties from understanding the stored information or communication.

deployment environment

Environment where the IBM WebSphere Voice Server is installed. A typical configuration may include a VoIP gateway, H.323 gatekeeper (optional), Speech Server, Web application server, and enterprise server.

dedicated LAN

To obtain optimal system performance, physical security, and secure transactions, all telephony applications are isolated and protected within an intranet.

dial peer

H.323 telephony component that specifies a call endpoint or destination. Each dial peer represents a discrete call leg.

dialog

The main building block for interaction between the user and the application. VoiceXML supports two types of dialogs: “form” and “menu”.

digital certificate

Allows you to use SSL for secure access to a Web site or Internet service. Contains elements such as locality, organization, common name, email address, mailing address, status and duration of validity.

DNIS

Dialed Number Identification Service. A telephony service that tells the receiving party what telephone number the caller dialed.

DTMF

Dual Tone Multiple Frequency. The tones generated by pressing keys on a telephone’s keypad.

echo cancellation

Technology that removes echo sounds from the input data stream before passing what’s left (that is, user speech) to the speech recognition engine. In a telephony environment, this is configured on the VoIP gateway; if echo cancellation is poor, you may need to turn off “[barge-in](#)” or switch to “[half-duplex](#)”.

enterprise server

Physical server containing the back-end data accessed by VoiceXML applications. Applications such as the IBM DB2 Lotus Domino database may be run.

Ethernet

A 10/100 network connection between the VoIP gateway and the Speech Server that supports VoIP.

full-duplex

Applications in which the user and computer can speak concurrently. Full-duplex applications use “[echo cancellation](#)” to subtract computer output from the incoming data to determine what was user speech. See also “[barge-in](#)”. Contrast with “[half-duplex](#)”.

gateway expert

The person responsible for the configuration of the telephony hardware (VoIP gateway and H.323 gatekeeper), with knowledge and skills in the following areas:

- H.323 protocol
- Telephony infrastructure
- VoIP gateway hardware
- H.323 gatekeeper software

grammar

A collection of rules that define the set of all user utterances that can be recognized by the speech recognition engine at a given point in time. The VoiceXML browser makes different grammars active at different points in the dialog, thereby controlling the set of valid utterances that the speech recognition engine is listening for. Grammars support word substitution and word repetition.

GSM

Global System for Mobile Communication. The cellular telephone network.

H.323 telephony component

Java application that runs in the deployment environment. The application implements H.323 v2.0; it handles call control via “[JTAPI](#)” v1.2 and audio and DTMF transport via “[JMF](#)” v2.0 to allow users access to Web content in the VoiceXML format using a telephone.

H.323 terminal endpoint alias

Terminal endpoint identifier used for H.323 gatekeeper routing of a call to the appropriate VoiceXML browser.

half-duplex

Applications in which the user should not speak while the computer is speaking, because the speech recognition engine does not receive audio while the computer is speaking. Turn-taking problems can occur when the user speaks before the computer has finished speaking; using a unique tone to indicate the end of computer output can minimize these problems by informing users when they can speak. Contrast with “[full-duplex](#)”.

HTTP

Web server protocol.

HTTPS

Secure HTTP Web server protocol.

hunt group

A group of possible routes that allow you to make a selection between available routes for a call.

IBM WebSphere Voice Server with ViaVoice Technology

Software product used to deploy VoiceXML applications in a VoIP telephony environment to provide voice access to Web-based data using standard telephony interfaces.

IBM WebSphere Voice Server Software Developer's Kit (SDK) Programmer's Guide

Document containing information on creating and testing voice applications. The document is packaged with the SDK product, which is bundled with the IBM WebSphere Voice Server or can be downloaded from:

<http://www.ibm.com/software/voicesystems>

inbound call

An incoming call from a telephone connection through the VoIP Gateway to the VoiceXML browser on a Speech Server.

IP network

Internet protocol network.

JMF

Java Media Framework.

JSP

Java Server pages. One of many server-side mechanisms for generating dynamic Web-content by transmitting data between an HTTP server and an external program. JSPs call Java programs which are executed by the HTTP server.

JTAPI

Java Telephony Application Program Interface.

JVM

Java Virtual Machine.

m-law

The compression and expansion algorithm used primarily in North America and Japan when converting from analog to digital speech data.

network administrator

A person with networking and IP communications expertise who supports the System Administrator in setting up the network prior to installing the IBM WebSphere Voice Server software.

PBX

Private Branch Exchange network.

preference value

A priority assigned to each VoiceXML browser to provide load balancing.

POTS

“Plain Old Telephone Service.”

PSTN

Public Switched Telephone Network

recognition

When utterances are known and accepted by the speech recognition engine. Only words, phrases, and DTMF key sequences in active grammars can be recognized.

SDK

IBM WebSphere Voice Server Software Developers Kit.

session

A session consists of all interactions between the VoiceXML browser, the user, and the document server. The session starts when the VoiceXML browser starts, continues through dialogs and the associated document transitions, and ends when the VoiceXML browser exits.

SM alarm file

Alerts the system administrator when important events occur.

SM log file

First source of information for finding significant changes in the state of a Voice Server (such as starting and stopping), as well as error situations that were detected by the SM Agent.

Speech Server

The hardware server used to run the IBM WebSphere Voice Server software.

speech recognition engine

Decodes the audio stream based on the current active grammar(s) and returns the recognition results to the VoiceXML browser, which uses the results to fill in forms or select menu choices or options.

SSL

Secure Sockets Layer. A protocol used by Secure HTTP (HTTPS) to implement secure transactions when accessing Web sites.

SSL light keyring database

Contains certificates for the trust-basis that should be recognized by the client.

sysmgmt.properties file

Configuration file used by the system administrator to configure the IBM WebSphere Voice Server product.

System Management (SM) Agent

The controlling program that allows a System Administrator to manage system resources. The SM Agent logs and traces messages, starts and monitors VoiceXML browsers, and recovers dead or hung processes.

text-to-speech engine

Generates computer synthesized speech output from text input.

translation rules table

Tables in the VoIP gateway used to translate incoming calls to previously assigned to an “[H.323 terminal endpoint alias](#)”.

URI

Universal Resource Indicator. The actual “[H.323 terminal endpoint alias](#)” (that is, address of a resource on the World Wide Web) for a user-dialed phone number. For example:

<http://www.ibm.com>.

URL

Universal Resource Locator. A subset of URI.

voice activity detection (VAD)

Used to disable or enable voice activity detection on a Cisco router.

voice application

An application that accepts spoken input and responds with spoken output.

VoiceXML

Voice eXtensible Markup Language. An XML-based markup language for creating distributed voice applications. Refer to the VoiceXML Forum Web site at **<http://www.voicexml.org>**.

VoiceXML browser

The VoiceXML browser fetches and processes VoiceXML documents and manages the dialog between the application and the user.

VoiceXML browser log file

Log file (**`vxml.log`**) containing all trace messages for the VoiceXML browsers that are running. It is located in a subdirectory that is unique for each telephone call or VoiceXML browser instance.

Voice Server

IBM WebSphere Voice Server software product. The components include a VoiceXML browser, the IBM ViaVoice Speech Recognition and Text-To-Speech engines, SM Agent, and H.323 telephony component.

VoIP gateway

A physical interface between a telephony connection (PSTN/GSM) and a VoIP LAN. It converts incoming analog telephony audio into digital LAN data packets. The VoIP gateway communicates with one or more H.323 telephony components on the LAN, sending and receiving audio packets and messages.

Web application server

Physical server where VoiceXML files reside.

