Product Description

AudioCodes Intuitive Human Communications for Chatbot Services

Voice.AI Gateway

Version 2.2

audiocodes

Table of Contents

Notice

Information contained in this document is believed to be accurate and reliable at the time of printing. However, due to ongoing product improvements and revisions, AudioCodes cannot guarantee accuracy of printed material after the Date Published nor can it accept responsibility for errors or omissions. Updates to this document can be downloaded from https://www.audiocodes.com/library/technical-documents.

This document is subject to change without notice.

Date Published: November-24-2020

WEEE EU Directive

Pursuant to the WEEE EU Directive, electronic and electrical waste must not be disposed of with unsorted waste. Please contact your local recycling authority for disposal of this product.

Customer Support

Customer technical support and services are provided by AudioCodes or by an authorized AudioCodes Service Partner. For more information on how to buy technical support for AudioCodes products and for contact information, please visit our website at https://www.audiocodes.com/services-support/maintenance-and-support.

Stay in the Loop with AudioCodes

Abbreviations and Terminology

Each abbreviation, unless widely used, is spelled out in full when first used.

Related Documentation

Document Name
Voice.AI Gateway API Reference Guide
Voice.AI Gateway Integration Guide
Voice.AI Gateway Security Guidelines

General Notes, Warnings, and Safety Information

Note: OPEN SOURCE SOFTWARE. Portions of the software may be open source software and may be governed by and distributed under open source licenses, such as the terms of the GNU General Public License (GPL), the terms of the Lesser General Public License (LGPL), BSD and LDAP, which terms are located at https://www.audiocodes.com/services-support/open-source/ and all are incorporated herein by reference. If any open source software is provided in object code, and its accompanying license requires that it be provided in source code as well, Buyer may receive such source code by contacting AudioCodes, by following the instructions available on AudioCodes website.

Document Revision Record

LTRT	Description
30902	Initial document release.
30903	Features updated; deployment options updated.
30904	Support for Google Dialogflow, Google Cloud Text-to-Speech and Speech-to-Text. Sections updated: Connect on Bot Prompt (default); Language (voice name); SSML for TTS; Punctuation of STT Transcriptions (default); Barge-In (seconds); TTS Caching (default); Call Transfer; Disconnect
30905	Updated to Ver. 1.6 Nuance STT/TTS for trial; fault management using alarms; Azure Key Vault; storing STT transcriptions; playing prompts to user upon errors; barge-in by DTMF; sending DTMF; activities upon call failure; overriding STT parameters; timeouts for user input and bot/STT/TTS response
30906	Updated to Ver. 1.8; Agent Assist; forwarding metadata from bot
30907	Updated to Ver. 2.0. ELK; dynamic parameter settings using placeholders; sharing configuration between bots; Dialogflow support for agent assist; forwarding metadata over HTTP; DTMF timeout.
30908	Updated to Ver. 2.2. Dialogflow CX added; Google speech recognition using class tokens;Google speech adaption using boosts; audio logging by Azure STT; DTMF collection; starting and stopping voice recording activities by bot; asynchronous bot activities to user; OAuth 2.0; dialout (outbound routing)
30909	Resources for deployment updated.

Documentation Feedback

AudioCodes continually strives to produce high quality documentation. If you have any comments (suggestions or errors) regarding this document, please fill out the Documentation Feedback form on our website at https://online.audiocodes.com/documentation-feedback.

1 Introduction

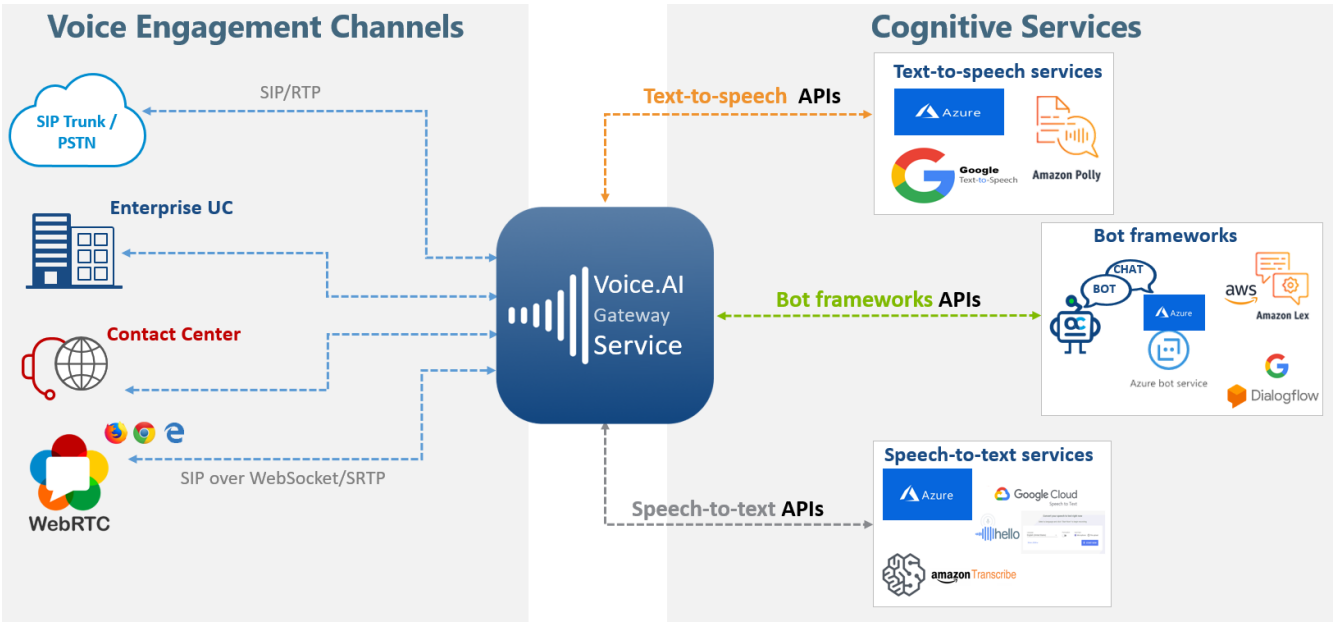
AudioCodes Voice.AI Gateway enhances chatbot functionality by allowing human communication with chatbots through **voice** (voicebot), offering an audio-centric user experience. Employing AudioCodes' Voice.AI Gateway service provides you with a single-vendor solution, assisting you in migrating your text-based chatbot experience into a voice-based chatbot.

The Voice.AI Gateway also allows you to provide this enhanced chatbot service to a wider audience that uses various voice engagement channels from where users initiate calls with chatbots:

- SIP Trunk
- PSTN or cellular
- Enterprise Unified Communications (UC) and IP-PBXs
- Contact Centers
- WebRTC

AudioCodes' field-proven and sophisticated voice communications technology embedded in the Voice.AI Gateway allows seamless integration into any existing voice network. The Voice.AI Gateway can connect to and integrate with any third-party cognitive service – bot frameworks, speech-to-text (STT) engines, and text-to-speech (TTS) engines. It can also operate with TTS and STT engines for multiple languages.

Figure 1-1: Voice Engagement Channel and Cognitive Services of Voice.AI Gateway



The Voice.AI Gateway provides advanced call management capabilities, for example, call disconnect, call transfer to a human agent, and call recording.

As such, the Voice.AI Gateway provides and uses various APIs for different purposes:

- SIP, RTP and WebRTC APIs for communicating with voice engagement channels
- HTTP-based APIs for interfacing with third-party voice cognitive services that convert voice to text (STT) and text to voice (TTS)
- HTTP-based APIs for connecting to third-party bot frameworks such as Azure, Google, and AWS

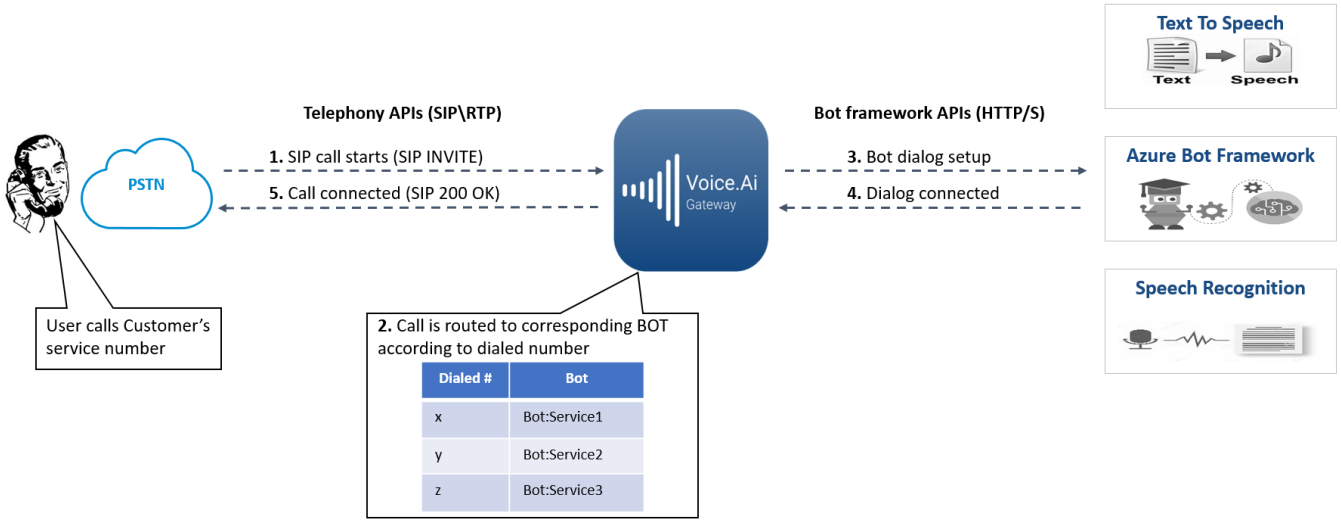
2 Conversation Flow

This section describes the call flow between the voice engagement channel (telephony) and the cognitive chatbot services, through AudioCodes Voice.AI Gateway.

2.1 Conversation Setup

The call setup flow between the voice engagement channel (telephony) and the cognitive chatbot services through AudioCodes Voice.AI Gateway is shown and explained below:

Figure 2-1: Dialog Flow Between Voice Channel and Chatbot Cognitive Services

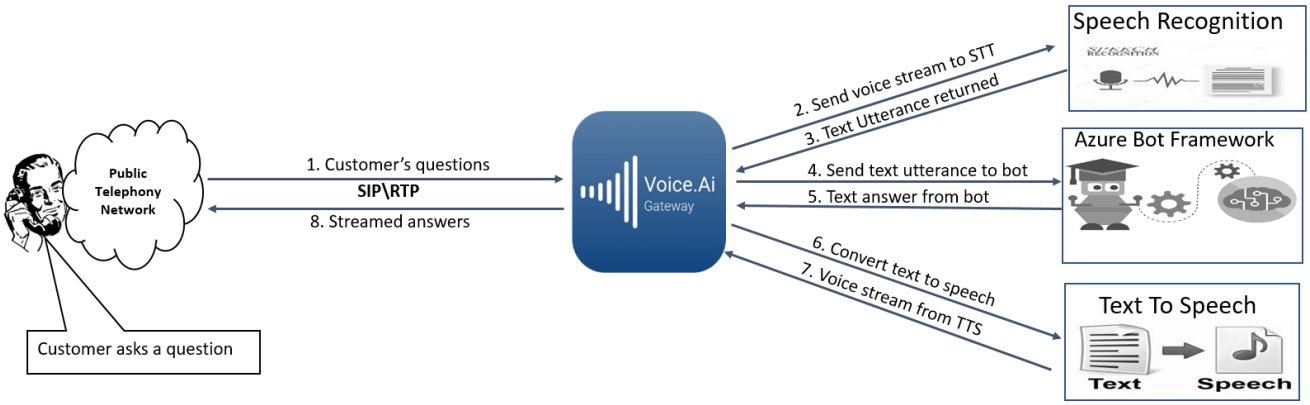


1. The user initiates a call with the chatbot, sending a SIP INVITE message to the Voice.AI Gateway.
2. The Voice.AI Gateway processes the call by employing its inherent session border controller (SBC) functionality. This includes basic call handling stages (call classification, message manipulation and routing) and optionally, other feature-rich SBC capabilities. The device searches its IP-to-IP Routing table for a matching routing rule (typically based on dialed number), which determines the route to a specific bot framework (destination).
3. The device initiates an HTTP-based dialog setup with a specific bot framework, according to the destination of the matched routing rule.
4. An HTTP-based connection is established between the Voice.AI Gateway and the bot framework.
5. The Voice.AI Gateway sends a SIP 200 OK response to the user, establishing the media session between it and the user.

2.2 Speech Flow

The speech flow between the voice engagement channel (telephony) and the cognitive chatbot services via AudioCodes Voice.AI Gateway is shown and explained below:

Figure 2-2: Speech Flow Between Voice Channel and Chatbot Cognitive Services



1. The user asks the bot a question, which is sent to the Voice.AI Gateway over RTP.
2. The Voice.AI Gateway sends the voice stream to the third-party speech-to-text (STT) engine.
3. The STT engine converts the spoken utterance into text, and then sends it to the Voice.AI Gateway.
4. The Voice.AI Gateway sends the text transcription to the bot.
5. The bot employs artificial intelligence and searches its database for the most appropriate answer, and then sends it in text format to the Voice.AI Gateway.
6. The Voice.AI Gateway sends the text-based answer to a text-to-speech (TTS) engine.
7. The TTS engine converts the text into speech and then sends the voice stream to the Voice.AI Gateway.
8. The Voice.AI Gateway plays the audio-based answer to the user.

3 Main Components of Voice.AI Gateway

The Voice.AI Gateway solution is comprised of the following main components:

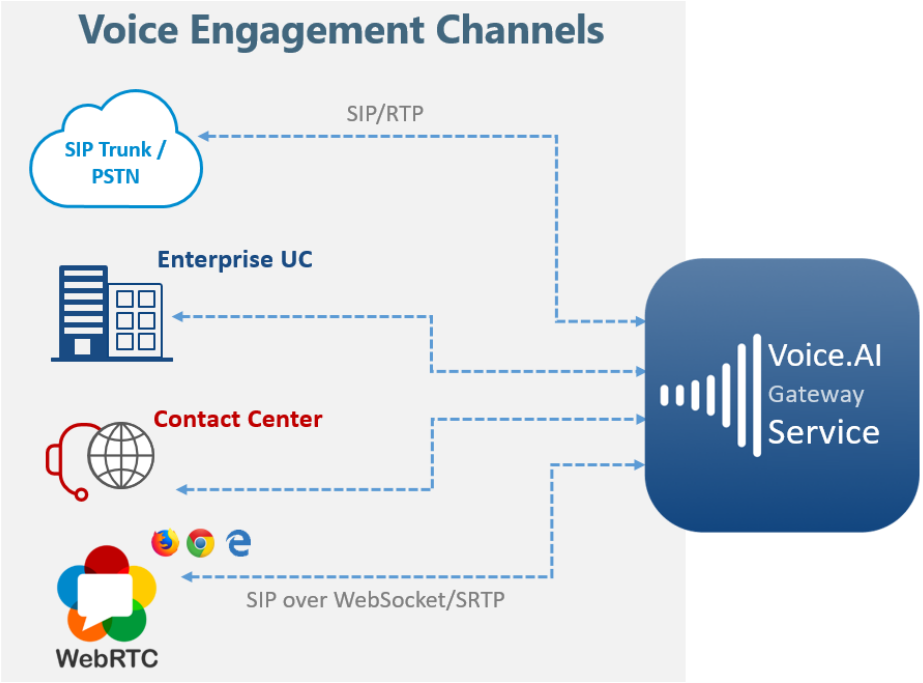
- Voice Engagement Channel Component
- Cognitive Services Component

3.1 Voice Engagement Channel Component

The Voice Engagement Channel component is based on the Voice.AI Gateway's embedded and feature-rich session border controller (SBC) module. It is responsible for interfacing with almost any voice engagement channel, as shown in the figure below. This component inherits all the SBC capabilities, for example, SIP interoperability, media handling (including translation), security, high availability, and scalability.

The Voice Engagement Channel component processes the SIP signaling and media (RTP) traffic and then converts the traffic into HTTP, which it routes to specific bots residing on specific bot frameworks (discussed in the next section).

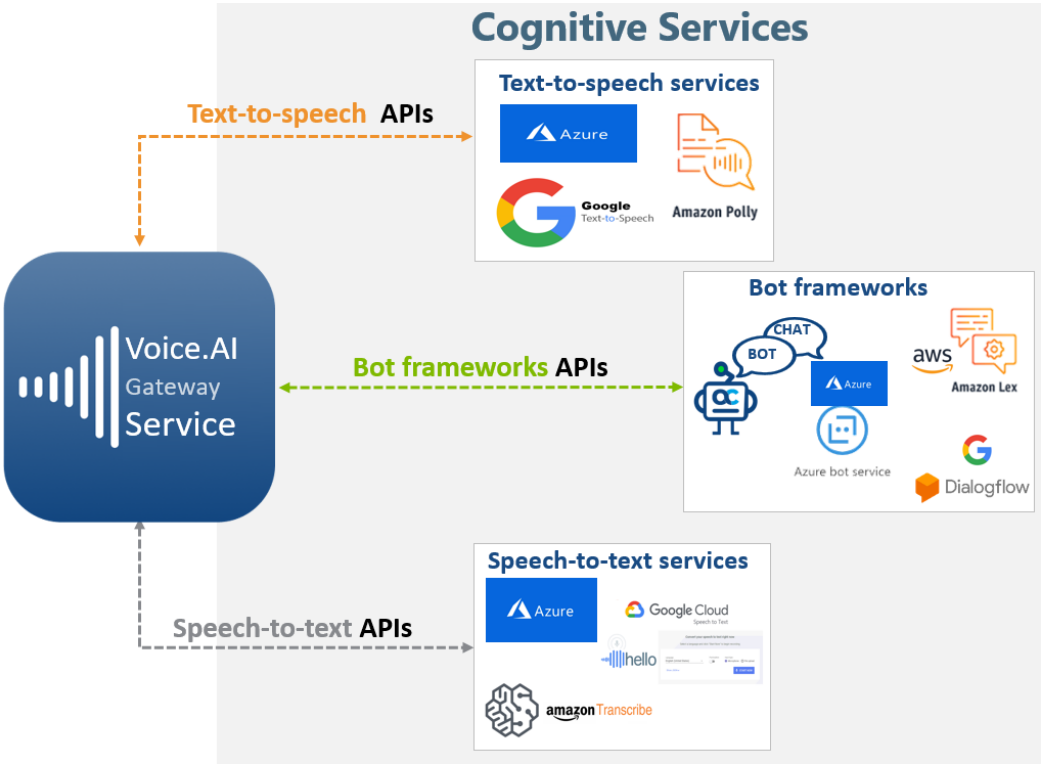
Figure 3-1: Voice Engagement Channel Component of Voice.AI Gateway



3.2 Cognitive Services Component

The Voice.AI Gateway's Cognitive Service component interfaces with third-party, cloud-based chatbot services, which include bot frameworks, speech-to-text (STT) engines, and text-to-speech (TTS) engines. The Voice.AI Gateway implements a multi-chatbot platform approach, enabling the use of different chatbot services based on dialed (called) number. Integration and support for these multi-bot frameworks and speech services is done through HTTP-based APIs, which convert any SIP event into the bot framework API, and vice versa. The Voice.AI Gateway's Cognitive Service component thus orchestrates the stream flow between the bot framework, STT engine and TTS engine.

Figure 3-2: Cognitive Services Component of Voice.AI Gateway



3.2.1 Text-to-Speech Services

The Voice.AI Gateway uses third-party text-to-speech (TTS) engines to convert text, sent from the bot into speech, which it then plays to the user.

The Voice.AI Gateway supports the following third-party TTS services:

- Azure Speech Services
- Amazon Web Services (AWS) Amazon Polly[1]
- Google Cloud Text-to-Speech
- Nuance Text-to-Speech[1]
- Almagu[1]
- Yandex[1]

Note: The Voice.AI Gateway interfaces with the TTS engine, using the Customer's account of the TTS provider.

3.2.2 Speech-to-Text Services

The Voice.AI Gateway uses third-party speech-to-text (STT) engines to convert audio (speech) spoken by the chatbot user, into text, which it sends to the bot.

The Voice.AI Gateway supports the following third-party STT service providers:

- Azure Speech Services
- Google Cloud Speech-to-Text
- Nuance[2]
- Yandex[3]

The Voice.AI Gateway uses the streaming APIs of the speech services to reduce delays, resulting in a better user experience.

Note: The Voice.AI Gateway interfaces with the STT engine, using the Customer's account of the STT provider.

[1] Currently, this TTS service with Voice.AI Gateway has not been released, but can be offered as a trial.
[2] Currently, this STT service with Voice.AI Gateway has not been released, but can be offered as a trial.
[3] Currently, this STT service with Voice.AI Gateway has not been released, but can be offered as a trial.

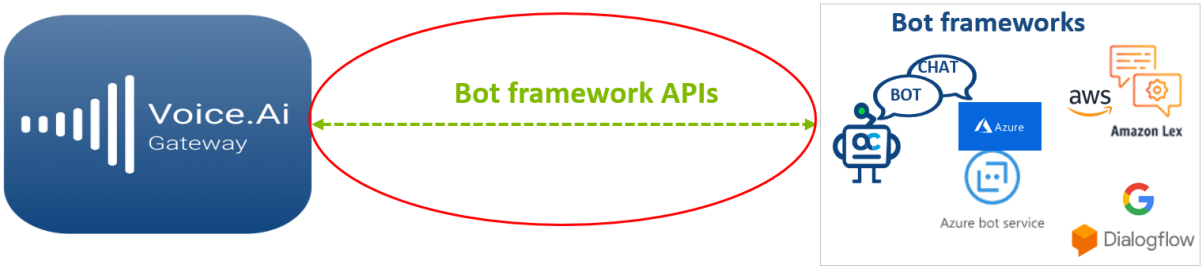
3.2.3 Bot Frameworks

The Voice.AI Gateway connects to bots through one of the following third-party chatbot frameworks:

- Microsoft Azure Bot Framework (using Direct Line 3.0 API)
- Google Dialogflow ES and Dialogflow CX
- Amazon Lex[4]
- Rasa
- Cognigy

The Voice.AI Gateway adapts to the above-mentioned bot framework's APIs. In addition, the Voice.AI Gateway provides a proprietary exposed HTTP-based APIs that allows **any bot frameworks or middlewares** to connect with it over HTTP/S. For a detailed description on the Voice.AI Gateway's APIs, contact AudioCodes.

Figure 3-3: Bot Frameworks



These APIs are used to initiate text-based dialogs to a specific bot implemented by the bot developer on a specific bot framework. Basic dialog initiation, authentication and text exchange is done according to the specifications of the chosen bot framework.

The Voice.AI Gateway also sends telephony metadata and notifications, as described in the document *Voice.AI Gateway Integration Guide*.

The bot developer can send specific actions and instructions to the Voice.AI Gateway, as described in the document *Voice.AI Gateway Integration Guide*.

Dialog initiation requires credentials to authenticate the specific bot according to the specific bot framework. These credentials must be configured on the Voice.AI Gateway per bot.

[4] Currently, this bot framework has not been tested with Voice.AI Gateway, but can be offered as a trial.

4 Deployment Options

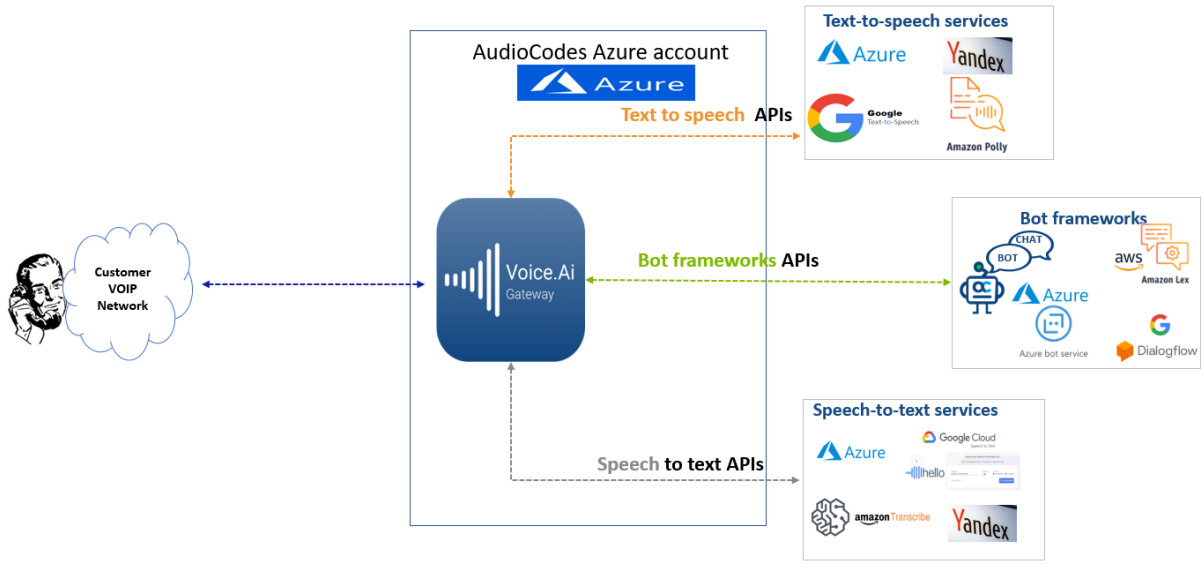
The Voice.AI Gateway offers multiple deployment options, which are all offered as a managed service by AudioCodes. This managed service includes installation, initial configuration, integration, as well as maintenance, active monitoring and fault management.

Note: For initial deployment, AudioCodes requires various information on the third-party STT, TTS, and bot frameworks used by the Customer, as well as bot-specific configuration. For more information, refer to the *Voice.AI Gateway Integration Guide*.

4.1 AudioCodes Cloud Account

The Voice.AI Gateway can be deployed and managed in AudioCodes cloud account. This deployment option uses AudioCodes infrastructure and resources required for the Voice.AI Gateway functionality. For more information on this managed service, please contact AudioCodes Support.

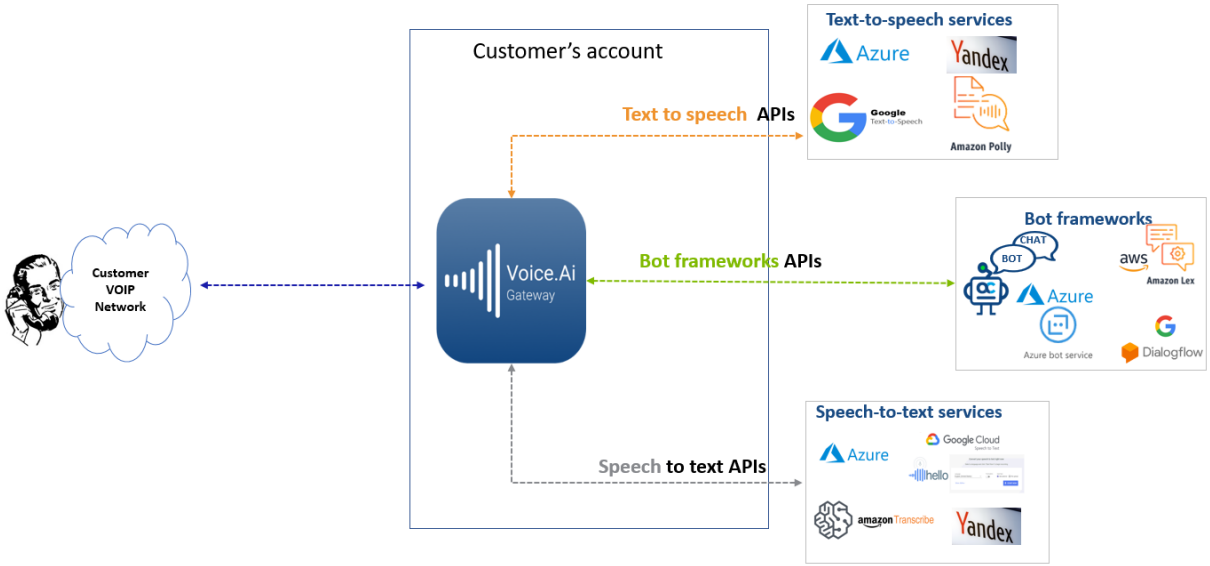
Figure: Voice.AI Gateway Deployed in AudioCodes Cloud



4.2 Customer's Data Center

The Voice.AI Gateway can be deployed in the Customer's data center, but managed by AudioCodes. For more information on this managed service, please contact AudioCodes Support.

Figure: Voice.AI Gateway Deployed in Customers Data Center

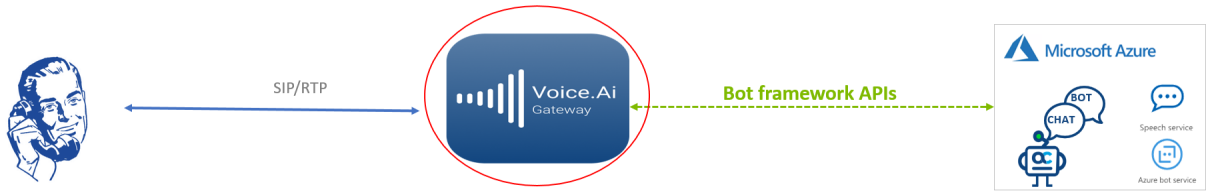


4.2.1 Required Resources

The required resources when Voice.AI Gateway is deployed in the Customer's data center includes the following:

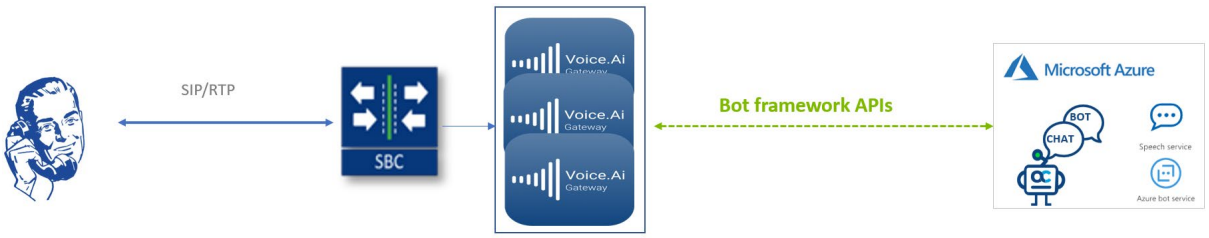
- Single Voice.AI Gateway components (for Azure):
  - VoIP-facing component: 4 vCPU, 16 Gigabyte RAM (Azure DS3_v2)
  - Connector: 2 vCPU; 8 Gigabyte RAM; 30 Giga disk (Azure D2s_v3)
  - Overall capacity: 300 sessions

Figure 4-1: Single Voice.AI Gateway



- For resiliency and scalability (thousands of sessions), more resources are required, by implementing 1+1 AudioCodes Session Border Controllers (SBC) that load-balances traffic to multiple Voice.AI Gateways:
  - 2 x SBC load balancers (1+1): 2 vCPU, 8 Gigabyte RAM (Azure B2ms)
  - N x Voice.AI Gateways, where each can scale up to 300 sessions and requires the following:
    - VoIP-facing component: 4 vCPU, 16 Gigabyte RAM (Azure DS3_v2)
    - Connector: 2 vCPU; 8 Gigabyte RAM; 30 Giga disk (Azure D2s_v3)

Figure 4-2: Voice.AI Gateways with Multiple SBCs



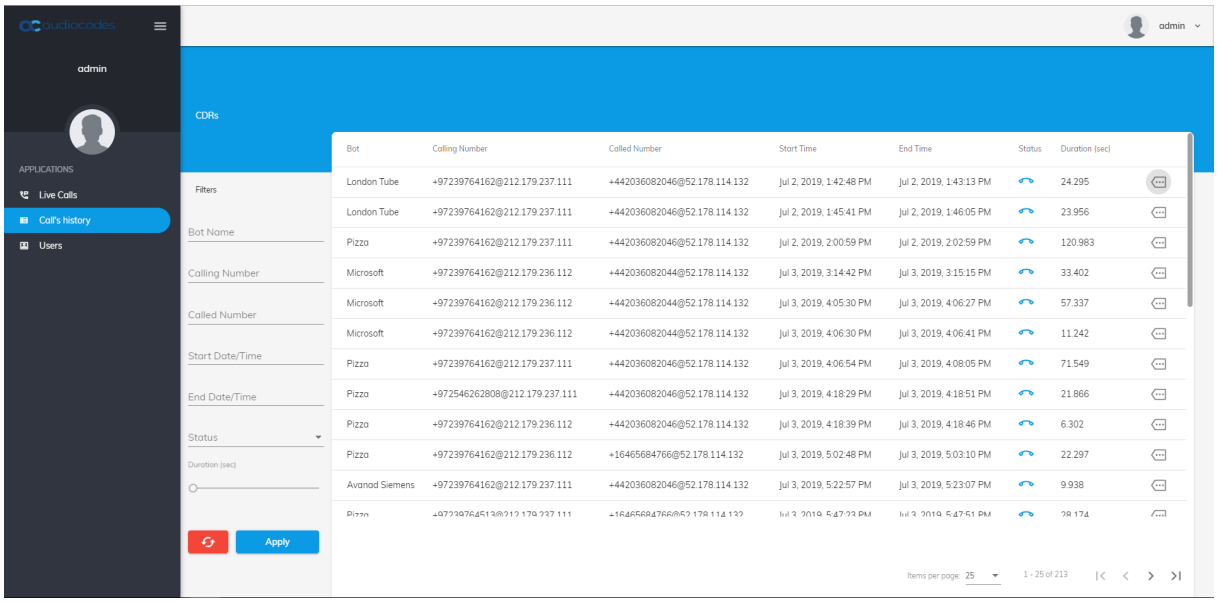
This page is intentionally left blank.

5 Performance Monitoring and Diagnosis

5.1 CDRs and Call Transcripts

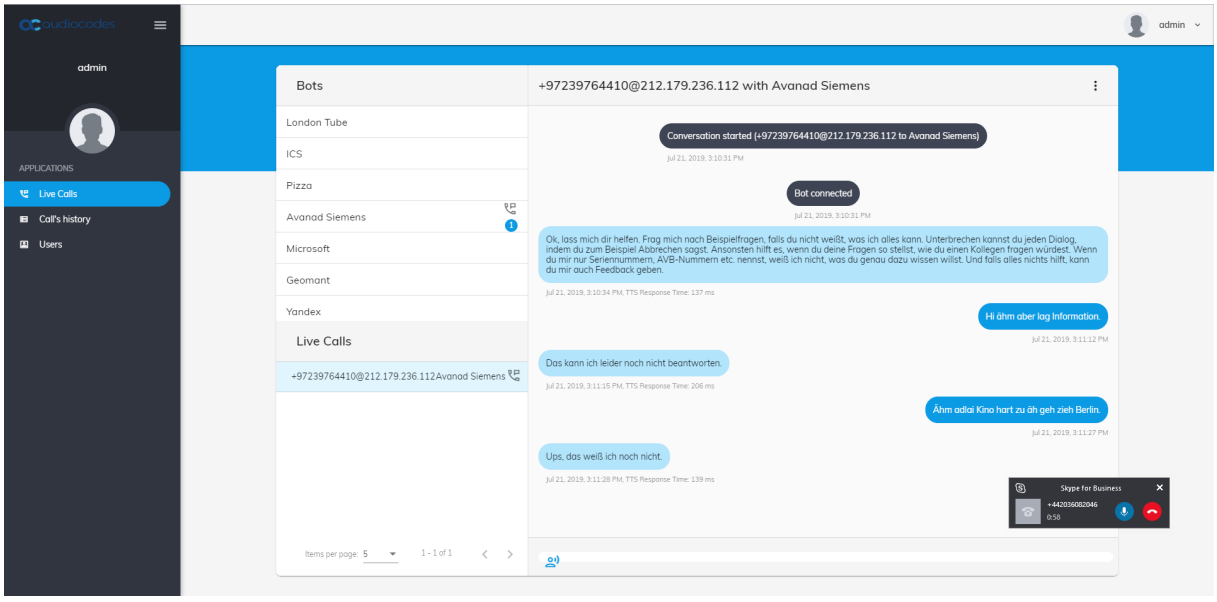
The Voice.AI Gateway can be enabled (default) to store call detail records (CDRs) of live and history chatbot calls on a local database. They can be viewed by connecting to the Voice.AI Gateway's Web-based management interface. Below shows an example of CDRs for historical chatbot call.

Figure 5-1: CDR Display in Voice.AI Gateway Web GUI



The Voice.AI Gateway can also be enabled (default) to store chatbot call transcripts (speech-to-text) on a local database. You can view the transcripts by connecting to the Voice.AI Gateway's Web-based management interface. Below shows an example of the transcript of a live chatbot call.

Figure 5-2: Call Transcript Display in Voice.AI Gateway Web GUI



5.2 Alarms and Fault Management

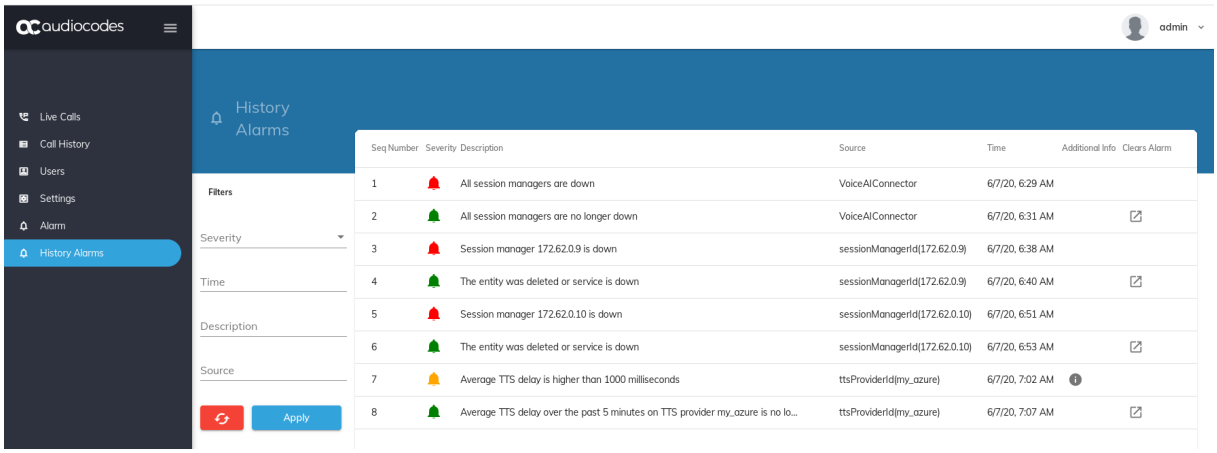
Fault management can be done using the Voice.AI Gateway SNMPv2 MIB alarms. Alarms can be sent to multiple remote management tools (e.g., OVOC, an NMS, and Voice.AI Connector's Web interface) through SNMP. Destinations are configurable.

An alarm is sent with one of the six severity levels (from Clear to Critical).When a problem occurs, the Voice.AI Gateway sends an alarm. If the problem is resolved, the alarm is cleared by sending the same alarm with a Clear severity level. When the severity of an already sent alarm changes, the same alarm is sent again with the new severity.

Alarms are displayed in an Active Alarms table. When an alarm is cleared, it's removed from the table. All alarms are also stored in a History Alarms table. Stored alarms are removed from the History Alarms table when the table reaches maximum capacity.

The Voice.AI Connector component allows you to customize alarms. This includes disabling alarms so that they are never sent or enabling alarms only for specific entities (whitelist), for example, a specific bot provider or STT provider. In addition, each alarm's properties can be customized, for example, severity level and displayed text.

Figure 5-3: Alarm Display (e.g., History) in Voice.AI Connector Web GUI



5.3 Troubleshooting with ELK

The Voice.AI Gateway supports ELK, which is an effective tool for troubleshooting. ELK is installed with Docker and composed of the following open-source tools:

- **E**lasticsearch - a powerful search and analytics engine used for full-text search and for analyzing logs and metrics.
- **L**ogstash - ingests data from multiple sources and then transforms it into logs and events
- **K**ibana – lets users view logs and events with charts and graphs.

ELK provides the following benefits:

- Aggregates logs from the entire Voice.AI Gateway system (e.g., SBC, Session Manager, Services Manager)
- Logs are available in one click
- Improves troubleshooting by visualizing the logs
- Fast search
- Monitors and alerts

.

6 Voice.AI Gateway Features

This section describes features supported by the Voice.AI Gateway.

Some features affect the entire conversation and are configured by the Administrator. For example, audio caching of received text-to-speech results.

Some features can be applied dynamically during the conversation and are controlled by the bot using the specific bot framework's APIs.

In addition, the Voice.AI Gateway sends at various stages of the conversation events with their relevant metadata to the bot. For example, when a conversation starts, the Voice.AI Gateway sends an event message that includes the phone number of the caller and called party.

Features that can be configured by the Administrator is done by AudioCodes according to the Customer's requirements. Features that are controlled by the bot are activated by the Bot Developer using the specific bot framework's APIs (as described in the *Voice.AI Gateway Integration Guide*).

6.1 Bot Frameworks

The Voice.AI Gateway can be configured to operate with specific third-party bot frameworks per bot. For bot frameworks supported by the Voice.AI Gateway, see Section Bot Frameworks.

This feature is controlled by the Administrator.

6.2 Conversation Initiation Features

This section describes conversation initiation features.

6.2.1 Initial Activity to Bot

When the user starts a conversation, the Voice.AI Gateway sends an initial activity message to the bot, by default. This message includes metadata such as caller and callee number and host, and caller display name.

The Voice.AI Gateway can be configured to override the content of this activity message with a specific JSON object, or to disable this activity entirely.

This feature is controlled by the Administrator.

6.2.1.1 Passing SIP Headers to Bot

In some setups, it is important to forward SIP header values, extracted from the SIP INVITE message received by the Voice.AI Gateway, to the bot. For example, in contact centers, an ID is associated with the call which is usually sent in a SIP header, and may be required by the bot application.

The Voice.AI Gateway can be configured to extract values from the SIP INVITE message and then send them in the initial message to the bot. The values are sent as additional attributes in the detail of the initial message.

For more information, please contact AudioCodes support.

6.2.2 Connect on Bot Prompt

By default, when the user starts a conversation the Voice.AI Gateway waits for the first bot message from the bot framework before connecting the call. It is possible to configure the Voice.AI Gateway to connect immediately without waiting for the first bot message.

This feature is controlled by the Administrator.

6.2.3 Initial Message to Bot

The Voice.AI Gateway can be configured to send a textual message to the bot when the user starts a conversation. This is typically used for bots that require a specific "hello" message to start a conversation.

This feature is controlled by the Administrator.

6.2.4 Welcome Message

The Voice.AI Gateway can be configured to send an initial "welcome" text message to the user when conversation starts. This is typically required in scenarios where the bot doesn't send a welcome text prompt.

This feature is controlled by the Administrator.

6.2.5 Dynamic Parameter Settings using Placeholders

The Voice.AI Gateway can populate a parameter according to the conversation's start attributes (e.g., SIP header values), using placeholders. The Voice.AI Gateway handles placeholder parameters as JavaScript template literals (string), allowing placeholders to be specified using the "${script}" syntax. The placeholders are resolved when the conversation starts (after the Voice.AI Gateway runs the manipulation.startToBot hook).

For example, the placeholder below is used to obtain the value (URL) for the sendMetaDataUrl parameter:

```
"sendMetaDataUrl": "https://example.com/update/${getSIPHeader('X-Conversation-ID')}"
```

Currently, this feature is supported only for sending metadata over HTTP, enabling the use of a different URL per conversation.

6.3 Speech Features

This section describes speech-related features.

6.3.1 STT and TTS Providers

The Voice.AI Gateway can be configured to operate with specific third-party STT and TTS providers per bot. For TTS and STT providers supported by the Voice.AI Gateway, see sections Text-to-Speech Services and

Speech-to-Text Services, respectively.

This feature is controlled by the Administrator.

6.3.2 Language

The language (e.g., English US, English UK, or German) is required by the TTS and STT engines for interfacing with the user. In addition, the TTS engine can be configured with a specific TTS voice (e.g., female or male).

By default, the language and voice are configured by the Administrator per bot. In addition, the bot can dynamically change the language and voice during conversation.

6.3.3 Custom Language and STT Context

Certain STT engines (e.g., Azure) allows the user to build and use its own customized language that adapts to the specific use case jargon. Other STT engines (e.g., Google) allow the user to dynamically associate STT contexts with each STT query. These methods can be used to optimize STT accuracy.

The Voice.AI Gateway can be configured to use a specific custom language (Azure). It also allows the Bot Developer to add dynamic context (Google) or change the custom speech at any stage of the conversation.

6.3.3.1 Speech Recognition using Google Class Tokens

Google's *classes* represent common concepts that occur in natural language, such as monetary units and calendar dates. A class allows you to improve transcription accuracy for large groups of words that map to a common concept, but that don't always include identical words or phrases. For example, the audio data may include recordings of people saying their street address. One may say "my house is 123 Main Street, the fourth house on the left." In this case, you want Speech-to-Text to recognize the first sequence of numerals ("123") as an address rather than as an ordinal ("one-hundred twenty-third"). However, not all people live at "123 Main Street" and it's impractical to list every possible street address in a Speech Context object. Instead, you can use a *class token* (e.g., "$ADDRESSNUM") in the phrases field of the Speech Context object to indicate that a street number should be recognized no matter what the number actually is. For more information, go to https://cloud.google.com/speech-to-text/docs/speech-adaptation.

The Voice.AI Gateway Administrator can configure multiple phrases for speech contexts using these Google's class tokens.

6.3.3.2 Speech Adaption using Boosts

Google's speech adaptation boost feature allows you to increase the recognition model bias, by assigning more weight to some phrases than others. For example, most of the bot users in a deployment may be saying the phrase "fair" more often than "fare". To increase the probability of the STT engine in recognizing the word as "fair", the Voice.AI Gateway Administrator can apply a higher boost value (or weight) to the word "fair" in the configured speech context phrase.

6.3.4 SSML for TTS

Speech Synthesis Markup Language (SSML) can be sent in Text-to-Speech requests to allow for more customization in the audio response, by providing details on pauses, and audio formatting for acronyms, dates, times, abbreviations, or prosody.

The bot can use SSML when sending text messages to the Voice.AI Gateway. The Voice.AI Gateway forwards the SSML-based requests to the TTS engine.

Note: SSML support depends on the specific TTS provider used in the chatbot environment.

6.3.5 Continuous ASR

By default, the STT engine recognizes the user's end of utterance according to the duration of detected audio silence (or by other means). Each recognized utterance is sent by the Voice.AI Gateway to the bot as a separate textual message.

Sometimes, the detection of end of utterance occurs too quickly and the user is cut off while speaking. For example, when the user replies with a long description that is comprised of several sentences. In such cases, all the utterances should be sent together to the bot as one long textual message.

Continuous automatic speech recognition enables the Voice.AI Gateway to collect all the user's utterances. When it detects silence for a specified duration or a configured DTMF key (e.g., # key) is pressed by the user, it concatenates the multiple STT detected utterances, and then sends them as a single textual message to the bot. In this way, the user can configure a longer silence timeout.

This feature is controlled by the Administrator. In addition, the bot can dynamically control this mode during the conversation.

6.3.6 Overriding STT Parameters for Activation and Streaming

The Administrator can configure the Voice.AI Gateway to override parameters that it uses to activate the STT request. For example, to improve speech recognition accuracy, it may help to increase the sample rate (in Hz) of the sent audio.

In addition, the Voice.AI Gateway can also override streaming-related parameters when it activates the STT request. For example, to optimize the recognition of short utterances and also minimizes latency, the single_utterance parameter can be set true.

This feature is currently only supported for specific STT providers (Google).

6.3.7 Stored STT Transcriptions

The Voice.AI Gateway can be configured to store transcripts received from the STT engine. The Administrator can view these transcripts in the Voice.AI Connector Web-based management tool per bot.

6.3.8 Punctuation of STT Transcriptions

By default, the Voice.AI Gateway enables punctuation detection (e.g., periods, commas and question marks) for the STT engine's transcription and therefore, messages sent by Voice.AI Gateway to the bot includes punctuation. The Voice.AI Gateway can be configured to disable punctuation detection by the STT engine.

Note that punctuation detection is supported only by specific STT providers.

This feature is controlled by the Administrator. In addition, the bot can dynamically control this feature during the conversation.

6.3.9 Barge-In by Speech or DTMF Input

The Barge-In feature controls the Voice.AI Gateway's behavior in scenarios where the user starts speaking or dials DTMF digits while the bot is playing its response to the user. In other words, the user interrupts ("barges-in") the bot.

When these two features are disabled (default), the Voice.AI Gateway ignores user speech input, from the detection of end of utterance until the bot has finished[5] playing its response (or responses if the bot sends multiple consecutive response messages). Only then does the

[5] The Voice.AI Gateway actually starts detecting speech a few tenths of a second before the bot finishes playing its response. This ensures that the first words of the user aren't skipped.

Voice.AI Gateway expect user speech or DTMF input. However, if no bot response arrives within a user-defined timeout, triggered from the detection of end of utterance, speech-to-text recognition is re-activated and the user can speak or send DTMF digits again.

When the Barge-In is enabled, detection of user speech input or DTMF is always active. If the user starts to speak or presses DTMF digits while the bot is playing its response, the Voice.AI Gateway detects this speech or DTMF and immediately stops the bot response playback and sends the detected user utterances or DTMF to the bot. If there are additional queued text messages from the bot, they are purged from the queue.

These two feature are controlled separately by the Administrator. In addition, the bot can dynamically control this feature during the conversation.

6.3.10 TTS Caching

By default, the Voice.AI Gateway caches TTS results from the TTS engine. This offers various benefits such as the following:

- Speeding up service by eliminating TTS round-trip delay
- Reducing TTS service costs by avoiding unneeded TTS transactions.

The default lifetime of the TTS cache is 24 hours (configurable). The TTS cache can also be cleared by the Administrator.

The Voice.AI Gateway can be configured to disable caching of TTS results.

This feature is controlled by the Administrator and can be disabled by the Bot Developer per text message.

6.3.11 Audio Logging by STT Engine

Microsoft Azure Speech to Text service provides an audio logging feature that records and logs endpoint user utterances received for speech-to-text transcription. The Voice.AI Gateway Administrator can instruct the STT engine to enable or disable this feature for specific endpoints. Note that recording is done by the Azure STT and stored on the STT engine.

6.4 Sending User DTMF Digits to Bot

By default, DTMF digits entered by the user are not sent to the bot. However, the Administrator can enable the Voice.AI Gateway to accept DTMF input from the user and send them to the bot. This feature can also be combined with the Barge-In feature, whereby if the user presses DTMF digits while the bot is playing its response, it interrupts and stops the bot response, as described in Barge-In by Speech or DTMF Input.

By default, the Voice.AI Gateway sends each collected digit to the bot only after the bot acknowledges receipt of the previous digit or a user-defined timeout expires (whichever occurs first). In other words, when the Voice.AI Gateway receives a DTMF digit entered by the user, it sends that single digit to the bot. However, the Administrator can configure the Voice.AI Gateway to first collect all the DTMF digits entered by the user, and only then send them all together to the bot. With this DTMF handling method, the following can be configured:

- Maximum digits to collect before sending them to the bot.
- Maximum timeout between each entered digit. If the timeout expires since the last digit entered by the user, the Voice.AI Gateway sends all the currently collected digits to the bot.
- Special "submit" digit (e.g., # key) that if entered by the user indicates to the Voice.AI Gateway that it has received all the required digits and can now send them to the bot.

6.5 Activities upon Failure During Call

The Administrator can configure the Voice.AI Gateway to perform a specific action if an error occurs during an established call. The error can be with the bot framework, STT provider, TTS provider, or SBC. For example, if an error occurs, a prompt (audio or text) can be played to the user or the call can be transferred to a live agent.

6.6 Timeouts for User Input and Bot/STT/TTS Responses

The Administrator can configure timeouts within which input from the user or response from the bot, STT or TTS must occur. These timeouts are configured separately for each of these entities.

If the timeout expires without any user input or bot response, the Voice.AI Gateway can be configured to play a prompt (audio or text) to the user, asking the user to say something. If there is still no input from the user or response from the bot, the Voice.AI Gateway can be configured to prompt the user again (number of times to prompt is configurable).

If still no input or bot response is received after the timeout expires, the Voice.AI Gateway disconnects the call. In this failure scenario, the Voice.AI Gateway can also be configured to perform a specific activity, for example, play a prompt to the user or transfer the call (see Activities upon Failure During Call).

6.7 Voice Recording Activities by Bot

The Voice.AI Gateway supports activities sent by the bot that start or stop voice recordings of the conversation, at any stage of the call. Recording is done by the session border controller (SBC) using the SIPRec protocol, whereby the SBC acts as the session recording client (SRC) while the session recording server (SRS) can be AudioCodes SmartTAP recording tool or any other third-party SRS. The Voice.AI Gateway communicates with the bot over REST, and with the SBC over a WebSocket tunnel.

The activity received from the bot indicates the destination of the SRS (i.e., IP Group as configured on the SBC). Optionally, it also, conveys the recording session ID that can be used by the Administrator after recording, to retrieve the recorded files stored on the SRS of the specific bot session.

6.8 Playing Prompts to User for Error Handling

The Voice.AI Gateway can be configured by the Administrator to play a prompt (audio or text) to the user in the following scenarios:

- Playing a prompt activity during failover activities execution (e.g., Bot framework, STT, TTS, or SBC error).
- Playing a prompt when there's no input from the bot for a user-defined time.
- Playing a prompt when there's no input from the user for a user-defined time.

6.9 Sharing Configuration between Bots

Bots often share the same configuration settings. Instead of duplicating configuration for each bot, the Voice.AI Gateway can be configured with a "base" bot, whose configuration settings can be referenced by any other bot requiring the same configuration. In other words, these bots inherit the configuration of the "base" bot. Note that the "base" bot is not a real bot; but used only as a configuration template.

6.10 Asynchronous Bot Activities to User

Typically, bots based on AudioCodes Bot APIs operate by request-response communication with users. The user's input is conveyed to the bot in the request and the bot's immediate response is conveyed in the response. The asynchronous API allows bots to also send any messages (activities) asynchronously to users through the Voice.AI Gateway (i.e., without requiring a request).

An example of a scenario where asynchronous messaging could be implemented is when the bot needs to perform a time consuming action such as retrieving information from a database. In this scenario, the bot may first send a reply of "please wait" to the user, and then once the information is retrieved, it sends a message to the user with the information.

Asynchronous messages are sent through a WebSocket connection (over HTTPS). As soon as a conversation is initiated, the Voice.AI Gateway optionally opens a WebSocket connection with the bot (acting as the server). This connection is used by the bot to send activities whenever it wants. The bot is the entity that decides if it wants to use the WebSocket for sending activities, or use the normal method for sending activities as responses to the Voice.AI Gateway requests. The Voice.AI Gateway only uses the WebSocket connection to receive activities from the bot; it doesn't send any messages through this connection. The Voice.AI Gateway closes the WebSocket connection upon the end of the conversation between the bot and user. The WebSocket connection can be secured using HTTPS and OAuth 2.0 authorization.

6.11 Security

This section describes the main security features of Voice.AI Gateway.

6.11.1 Azure Key Vault for Secure Secrets Storage

Typically, secrets or keys for accessing the Customer's bot framework provider are configured on the Voice.AI Gateway in plain text. However, if the Customer is using Microsoft's Azure Key Vault to securely store its secret keys, the secrets can be configured on the Voice.AI Gateway using the Key Vaults URLs. Therefore, the secrets will not be exposed to potential attackers. Note that Azure's Key Vault can be used to store secrets of any provider (e.g., Google).

6.11.2 Client OAuth 2.0 Authentication

The Voice.AI Gateway supports OAuth 2.0 for client authentication and authorization. Before the client can perform actions (limited) on the Voice.AI Gateway, the client must send its OAuth access token (shared secret) to the Voice.AI Gateway (according to Section 4.4.2 of RFC 6749). Only if the access token is correct, does the Voice.AI Gateway grant the client access, with permissions according to the client's OAuth scope. The scope can grant permission to perform any or all of the following:

- Read operations (HTTP GET requests) on bot configuration
- Read operations (HTTP GET requests) on provider configuration
- Write operations (HTTP POST, PUT or DELETE requests) on bot configuration
- Write operations (HTTP POST, PUT or DELETE requests) on provider configuration
- Outbound Calling ("dialout") operations (as described in Section 6.13.4, Triggering Outbound Calls to SBC and Bot)

The access token and scope per client (user) is generated and configured respectively, by the Administrator using the Voice.AI Gateway's Web-based management interface. Once generated, AudioCodes needs to provide the client with the access token.

6.11.3 OAuth 2.0 Authentication for Accessing AudioCodes Bot APIs

OAuth 2.0 authorization standard can be used to authenticate the Voice.AI Gateway with a bot's service implementing AudioCodes Bot API. If your setup does not require OAuth 2.0 authentication, you can use a simpler authentication method that uses a permanent token.

Upon initial communication, the Voice.AI Gateway acting as a client, requests an access token from a third-party OAuth 2.0 server (determined by the Customer). The authorization server identifies the Voice.AI Gateway by a shared secret key, client ID and optionally, scope (provided to AudioCodes by the Customer). Upon receipt of the access token, the Voice.AI Gateway sends this token (in the HTTP Authorization request header) to the AudioCodes Bot API that needs to be accessed at the provider.

The provider's bot uses this access token for all sessions, until the token expires. Prior to expiry, the Voice.AI Gateway requests a new access token from the authorization server.

6.12 Agent Assist

The Voice.AI Gateway enables an Agent Assist bot to receive a transcript of an ongoing conversation between a customer and a human agent. An Agent Assist bot can be implemented for different reasons, for example:

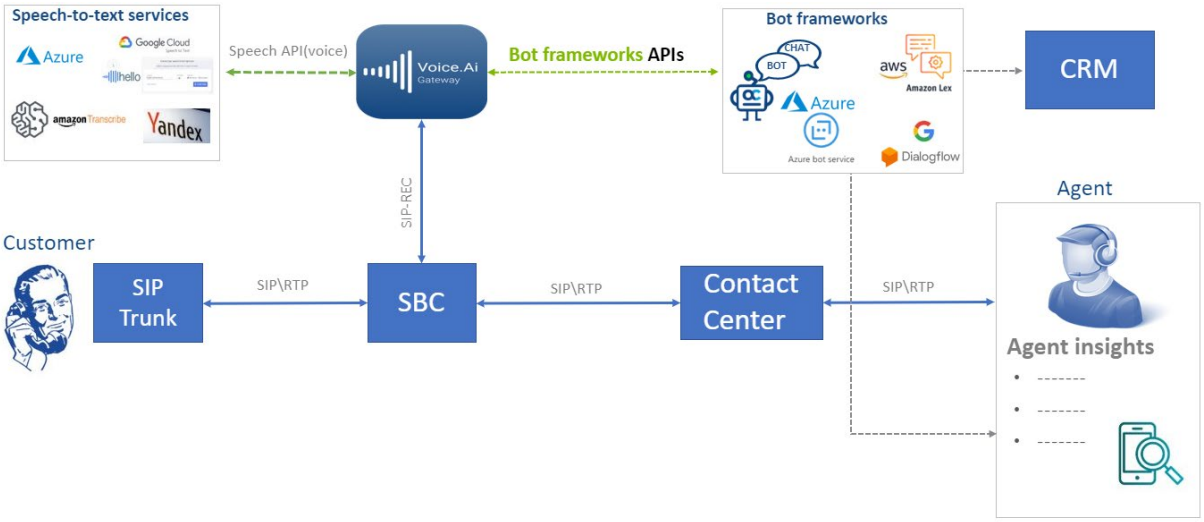
- To provide assistance to a human agent as need arises. The bot can provide information and suggestions (insights) to the human agent. For example, a customer may ask for bank account information from the human agent, and the bot can provide this information to the human agent. In this way, the bot can optimize the human agent's productivity, reducing handling time of customer inquiries.
- To monitor the conversation for key words, for example, foul language. In such a scenario, the bot can notify the human agent's supervisor.

To provide support for the Agent Assist feature, the following functional components are required:

- SIPRec client (SRC) – typically, a session border controller (SBC)
- Voice.AI Gateway serving as a SIPRec server (SRS) and providing connectivity with the Agent Assist bot
- Third-party speech-to-text service provider used by the Voice.AI Gateway to obtain real-time transcription of the conversation

The following illustration provides an example of an implementation of an Agent Assist bot with the Voice.AI Gateway:

Figure 6-1: Example of Agent Assist Bot with Voice.AI Gateway



The flow of the above example is as follows:

1. A VoIP call is established between the customer and human agent.
2. The SIPRec client (e.g., SBC) issues a SIPRec session towards the Voice.AI Gateway, recording the conversation.
3. The Voice.AI Gateway starts a conversation with the Agent Assist bot, providing the bot with details of the participants (customer and human agent).
4. The bot can request to receive conversation text streams of the customer, agent, or both.
5. The SIPRec client forwards the participants' voice streams to the Voice.AI Gateway.
6. The Voice.AI Gateway uses a speech-to-text service provider to obtain real-time transcription of the conversation.
7. The Voice.AI Gateway forwards the online text streams of customer and/or human agent conversation to the bot.
8. At this stage, the Agent Assist bot can send real-time insights to the human agent, human agent's supervisor, or a Customer Relationship Management (CRM) system. These insights can be sent using the Voice.AI Gateway (SIP INFO messages), or using any other method not related to the Voice.AI Gateway.

Note: Agent Assist functionality is supported by Azure, Google Dialogflow, and AudioCodes API.

6.13 Call Control

This section describes call control-related features.

6.13.1 Call Transfer

A common scenario, especially when chatbots are used in contact center environments, is the need to transfer the call to a human agent. This typically occurs when the bot cannot provide an adequate response (answer) to the user's question, or if the user explicitly requests to speak to someone. In such cases, the Voice.AI Gateway provides the option to transfer the call to either a live agent or a different bot.

The transfer can be initiated by the Bot Developer, by issuing the specific transfer activity through the bot framework's APIs. The transfer activity has multiple attributes:

- Transfer target URI, which defines the new destination of the call (SIP-based live agent or a different bot). A default transfer target (URI) can be configured (by the Administrator), which is used when the bot requests a transfer without indicating a target.
- Textual description of the reason for the transfer. This is used for logging and CDRs.

There are two options to implement the transfer by the Voice.AI Gateway:

- Sending a new SIP INVITE message to the transfer target URI. (In this option, the Voice.AI Gateway remains part of the path for the call, even after the call is successfully transferred.)
- Sending a SIP REFER message to the SIP peer (e.g., contact center or SIP service provider) who initiated the original call. This REFER message causes the SIP peer to perform the transfer. In this option, the Voice.AI Gateway leaves the signaling path after the call is successfully transferred. This requires support for REFER messages by the SIP peer.

After the call is transferred, the Voice.AI Gateway disconnects the bot from the conversation.

This feature is controlled by the Bot Developer at any stage of the conversation and is done by sending the `transfer` activity, as described in the *Voice.AI Gateway Integration Guide*.

6.13.1.1 Adding SIP Headers on Call Transfer

When the bot performs a call transfer, the bot can optionally add data that is sent as SIP headers in the SIP message (REFER or INVITE) generated by the Voice.AI Gateway. This can be set by the Bot Developer as an attribute in the transfer activity, which lists names and their values that are converted into SIP headers (e.g., "My-Header: my-value") by the Voice.AI Gateway.

6.13.2 Forwarding Metadata from Bot

The Voice.AI Gateway can receive metadata from the bot for a specific conversation. The Voice.AI Gateway can be configured to forward this metadata to an entity (e.g., a contact center) to which the Voice.AI Gateway is connected, using one of the following methods:

- SIP INFO messages to a SIP server
- HTTP POST requests to an HTTP server

6.13.3 Disconnect

At any stage of the conversation, the bot can disconnect the conversation by initiating a disconnect activity (*hangup*). The conversation is disconnected by sending a SIP BYE message by the Voice.AI Gateway to the SIP peer.

- The disconnect activity has an optional attribute that describes the reason for the disconnect.

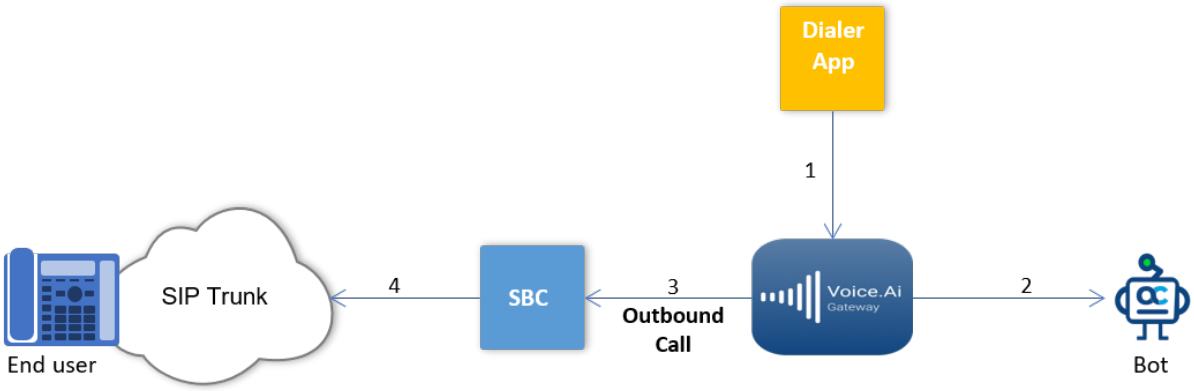
6.13.4 Triggering Outbound Calls to SBC and Bot

In a typical bot deployment, the Voice.AI Gateway first receives the call from the SBC and then sends it to the bot. However, the Voice.AI Gateway can also be triggered by a third-party dialer application to initiate the call instead of the SBC. This setup is often used by companies for outbound dialing campaigns, whereby the dialer automatically initiates calls with potential customers.

The basic call flow for outbound dialing is as follows:

1. The Voice.AI Gateway receives a message over HTTP (POST) from an outbound dialer application, which triggers it to make an outbound call.
2. The Voice.AI Gateway establishes a connection with the bot.
3. The Voice.AI Gateway initiates a call (*outbound*) to the end user (e.g., PSTN number) through the SBC.
4. When the end user answers the call, STT begins and the user's first utterance (e.g., "Hi") is sent to the bot to trigger its logic (e.g., "Hi there, we are calling from…").

This feature also allows the Voice.AI Gateway to send the dialer application asynchronous status notifications of the call (e.g., "call answered" or "call failed").



International Headquarters
1 Hayarden Street,
Airport City
Lod 7019900, Israel
Tel: +972-3-976-4000
Fax: +972-3-976-4040

AudioCodes Inc.
200 Cottontail Lane
Suite A101E
Somerset NJ 08873
Tel: +1-732-469-0880
Fax: +1-732-469-2298

Contact us: https://www.audiocodes.com/corporate/offices-worldwide
Website: https://www.audiocodes.com/

Document #: LTRT-30909