PureConnect®

2023 R3

Generated:

09-November-2023

Content last updated:

11-June-2019

See Change Log for summary of changes.

GENESYS™

Interaction Speech Recognition

Technical Reference

Abstract

Interaction Speech Recognition is the Automatic Speech Recognition (ASR) engine for Customer Interaction Center. This document provides an overview of the feature, the standards that it uses, and procedures for enabling and configuring the feature.

For the latest version of this document, see the PureConnect Documentation Library at: http://help.genesys.com/pureconnect.

For copyright and trademark information, see https://help.genesys.com/pureconnect/desktop/copyright_and_trademark_information.htm.

Table of Contents

Introduction to Interaction Speech Recognition

Interaction Speech Recognition is a native Automatic Speech Recognition (ASR) engine for CIC that Genesys introduced in CIC 4.0 SU4. Interaction Speech Recognition can recognize utterances (speech patterns), convert those utterances to data, and send the data to CIC. CIC can then take specific actions, such as direct a call to a specific extension or play more prompts.

Interaction Speech Recognition is an integrated feature of CIC. As such, Interaction Speech Recognition does not have an installation program nor can it reside on a separate host server.

The following PureConnect products support Interaction Speech Recognition:

- CIC server
- Interaction Media Server
- Interaction Attendant
- Interaction Administrator
- Interaction Designer (handlers)

For more information about these products, see the PureConnect Documentation Library at https://help.genesys.com/cic.

Limitations

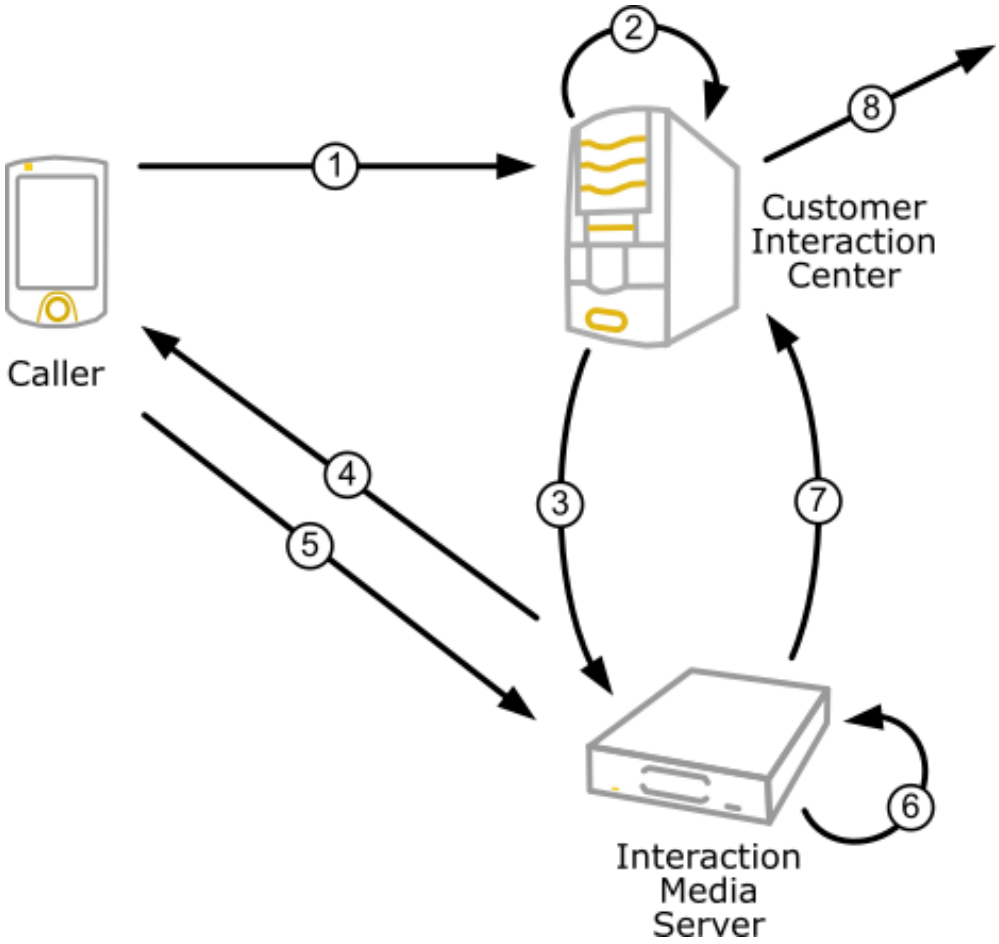
Interaction Speech Recognition currently has the following limitations:

- Limited support of the Semantic Interpretation for Speech Recognition (SISR) standard
- Diagnostic recordings for Interaction Speech Recognition don't transfer to the CIC server when the process of terminating of the recording exceeds 5 seconds.
- Interaction Speech Recognition does not currently support hotword barge-in. A hotword barge-in is an utterance that a caller speaks during the playing of a prompt that matches a defined word or phrase in a loaded grammar. When this utterance occurs, the current prompt stops and Interaction Speech Recognition returns the data associated to the match of the utterance.

Interaction Speech Recognition Requirements

Software	• CIC server 4.0 Service Update 4 or later • Interaction Media Server 4.0 Service Update 4 or later • Interaction Administrator (installed through IC Server Manager Applications 4.0 SU4 or later package)
CIC licenses	• I3_SESSION_MEDIA_SERVER_RECO • I3_FEATURE_SPEECH_RECOGNITION • I3_LICENSE_VoiceXML_SESSIONS (required only if you want to integrate Interaction Speech Recognition with VoiceXML functionality) • For any language that you want to use with Interaction Speech Recognition, purchase and install an Interaction Analyzer language license. **Note:** For more information about viewing your CIC license, including product and feature licenses, see "License Management" in the *Interaction Administrator* documentation. To purchase these licenses, contact your sales representative.
Languages	• Dutch, Netherlands (nl-NL) • English, Australia (en-AU) • English, United Kingdom (en-GB) • English, United States (en-US) • French, Canada (fr-CA) • French, France (fr-FR) • German, Germany (de-DE) • Italian, Italy (it-IT) • Japanese, Japan (ja-JP) • Mandarin Chinese, China (zh-CN) • Portuguese, Brazil (pt-BR) • Spanish, Spain (es-ES) • Spanish, United States (es-US)
Configuration	To allow automatic dialing of users through speech recognition, define one or more users in the company directory.
Standards	• Interaction Speech Recognition uses grammars that conform to the Speech Recognition Grammar Specification (SRGS) standard. • Interaction Speech Recognition has limited support for the Semantic Interpretation for Speech Recognition (SISR) v1.0 standards.

Interaction Speech Recognition Process Overview



1. CIC receives a call.
2. Interaction Attendant, VoiceXML, or a custom handler (based on information with the call) selects a prompt to play to the caller.
3. The CIC server sends grammars and prompts to use for the input operation. The CIC server then directs Interaction Media Server to play a prompt to the caller and wait for input.
4. Interaction Media Server plays the prompt to the caller and waits for a response.
5. The caller responds to the prompt through speech (an *utterance*) or by pressing keys on the telephone keypad.
6. Interaction Speech Recognition recognizes the response.
7. Interaction Speech Recognition returns the recognition data to CIC.
8. Interaction Attendant, VoiceXML, or custom handlers interpreter processes the data and proceeds accordingly. For example, plays another prompt or sends the call to a workgroup queue.

For more information about the process, see Interaction Speech Recognition Process Details.

Interaction Speech Recognition Process Details

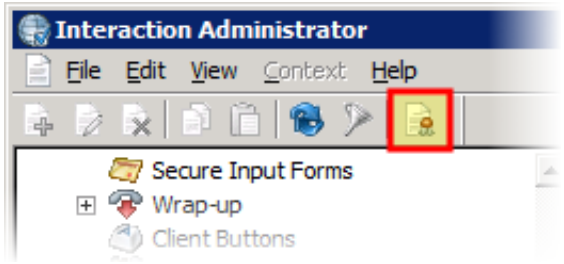
Interaction Media Server performance	An Interaction Speech Recognition session requires approximately the equivalent processing resources as a single, two-party call that the system records and transcodes.
Interaction Media Server selection	When a call requires audio, CIC selects an Interaction Media Server to provide audio based on Media Server Selection Rules and the location where the call was received or placed. If the call requires Interaction Speech Recognition, the processing for speech recognition occurs on the same Interaction Media Server that is providing audio for the call. For more information about Media Server Selection Rules, see the *Interaction Media Server Technical Reference* at https://help.genesys.com/cic/mergedProjects/wh_tr/desktop/pdfs/media_server_tr.pdf.
Grammar preloading and caching	If Interaction Media Server must download and compile large or complex grammars with hundreds or thousands of entries during a call, it can delay responsiveness. For this reason, Interaction Speech Recognition supports preloading of grammar files. When Interaction Media Server starts and connects to the CIC server, it downloads (through HTTP), compiles, and caches in memory the grammar files specified in the **Interaction Speech Recognition** object or the parent **Recognition** container in Interaction Administrator. The recognition subsystem of the CIC server also compiles and caches these grammars. You can also preload grammars in custom handlers through the **Reco Register Preloaded Grammars** tool in Interaction Designer. If you change or add a preloaded grammar through custom handlers or Interaction Administrator, Interaction Media Server downloads, compiles, and caches the new grammar automatically. Interaction Media Server caches non-preloaded grammars for Interaction Speech Recognition when used during a call.
Interoperability and customization	By creating custom handlers through Interaction Designer, you can create a solution that accomplishes your goals. For more information about Interaction Designer, see the *Interaction Designer Help* at https://help.genesys.com/cic/mergedProjects/wh_id/desktop/hid_introduction.htm. To use Interaction Speech Recognition with VoiceXML, you need a dedicated host for your VoiceXML interpreter and you need VoiceXML licenses for CIC. For more information about VoiceXML in a CIC environment, see the *VoiceXML Technical Reference* at https://help.genesys.com/cic/mergedProjects/wh_tr/desktop/pdfs/voicexml_tr.pdf.
Ports, sessions, and licensing	Genesys recommends that you purchase enough licenses for Interaction Speech Recognition ports to equal the number of licensed Interactive Voice Response (IVR) ports. This equality ensures that Interaction Speech Recognition can support all IVR sessions. The system uses a new Reco (representing the recognition subsystem of CIC) session port license each time a custom CIC handler calls a `Reco` tool step, a VoiceXML document requires Interaction Speech Recognition, or when using a default CIC handler. When a handler calls the `Reco Release` tool step, the system releases the port license.

Configure Interaction Speech Recognition

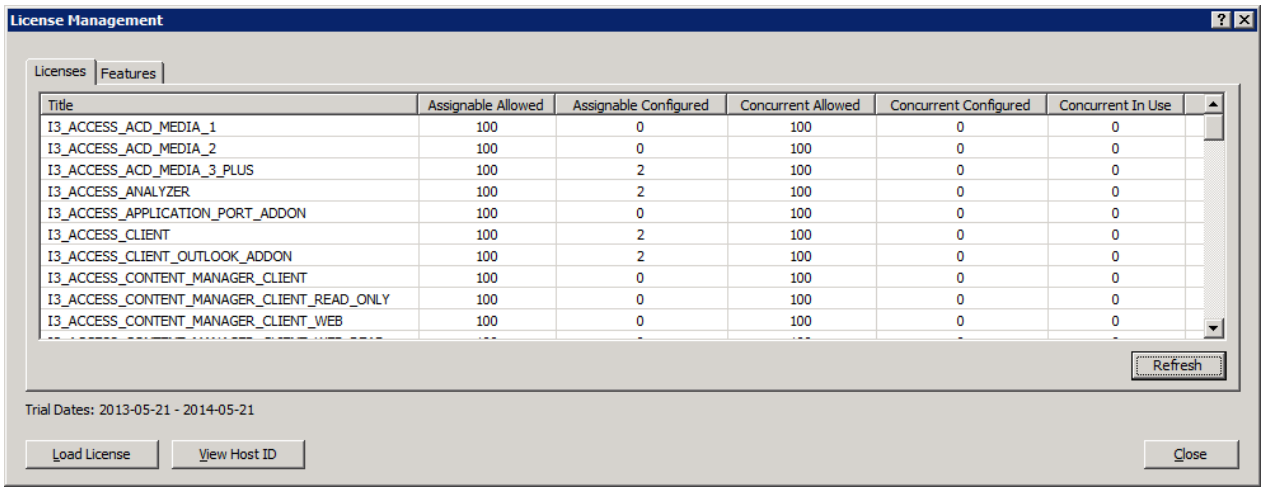
Verify Your Licenses

To verify your licenses

1. Open Interaction Administrator.



2. In the toolbar, click the **License** icon. The **License Management** dialog box appears.



3. On the **Licenses** tab, ensure that you installed the following licenses and assigned the appropriate number of sessions:
   - `I3_LICENSE_VoiceXML_SESSIONS` (if applicable)
   - `I3_SESSION_MEDIA_SERVER_RECO`
4. On the **Features** tab, ensure that you installed the following licenses and assigned the appropriate number of sessions:
   - `I3_FEATURE_ANALYZER_LANGUAGE_NN` (*NN* represents the language code of two to five characters)
   - `I3_FEATURE_SPEECH_RECOGNITION`
5. Click **Close**.

> **Note:**
> If you do not have the necessary licenses or number of ports, contact your sales representative.

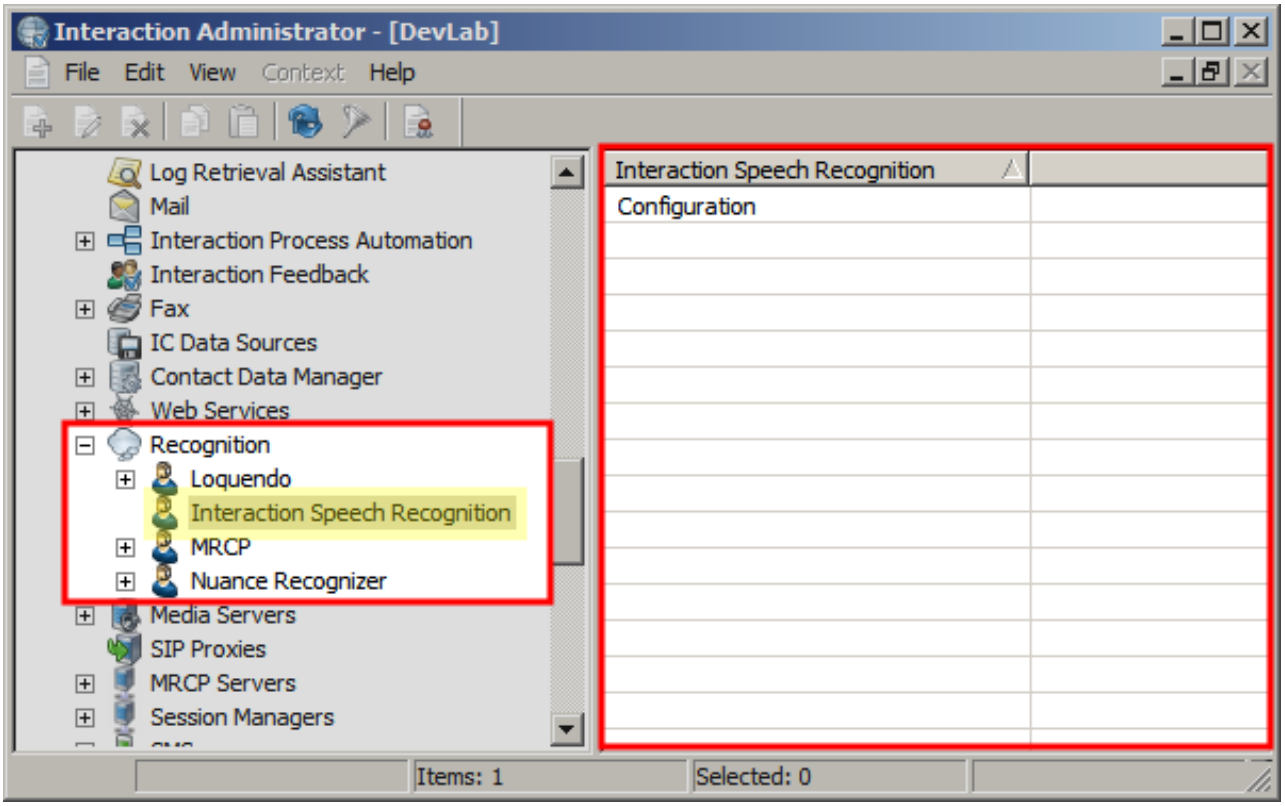
Enable Interaction Speech Recognition

After you install the Interaction Speech Recognition feature license, enable Interaction Speech Recognition.

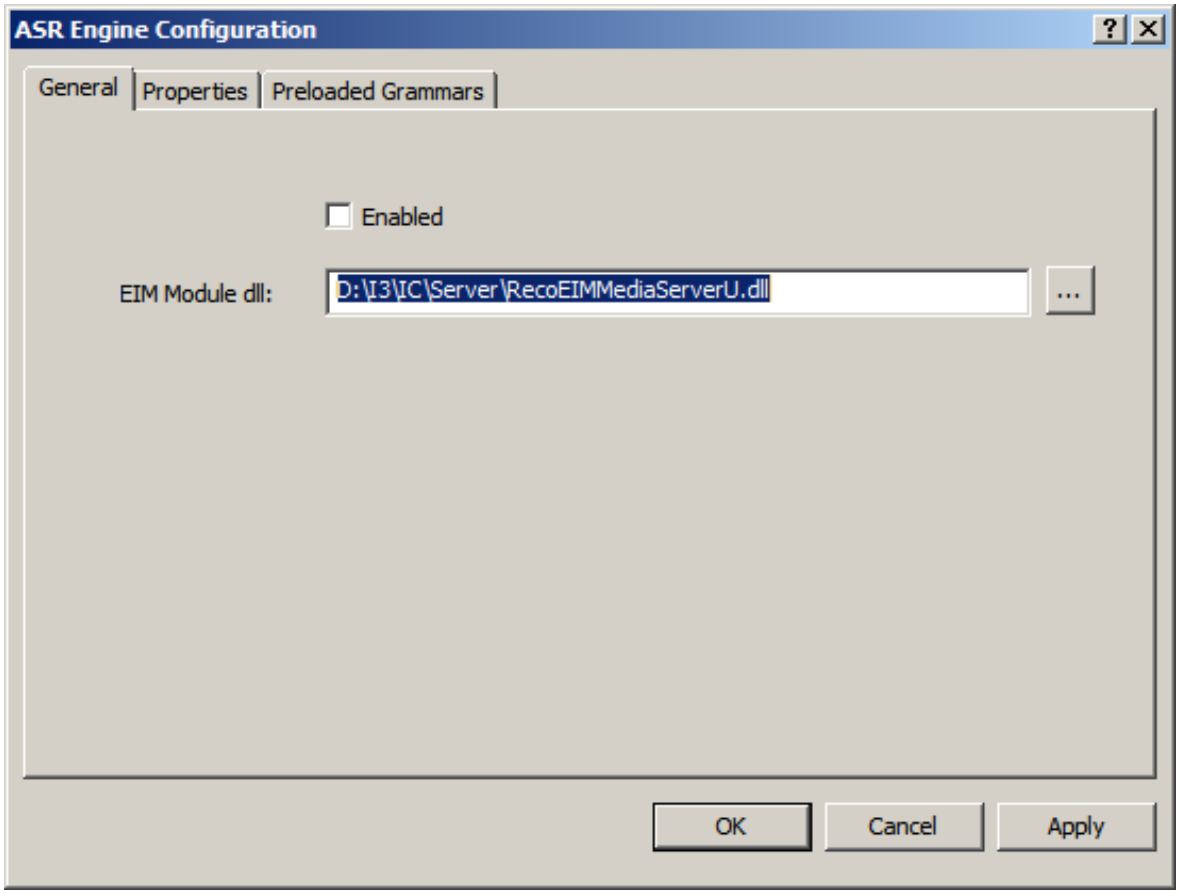
To enable Interaction Speech Recognition

1. Open Interaction Administrator.
2. In the left pane, expand the **System Configuration** container.
3. Under the System Configuration container, expand the **Recognition** container.
4. In the Recognition container, click the **Interaction Speech Recognition** object.

5. In the right pane, double-click **the Configuration** item



6. In the **ASR Engine Configuration** dialog box, select the **Enabled** check box.



7. Leave the Engine Integration Module (EIM) file specified in the **EIM Module dll** box as is and then click **OK**.

Adjust Audio Detection Sensitivity

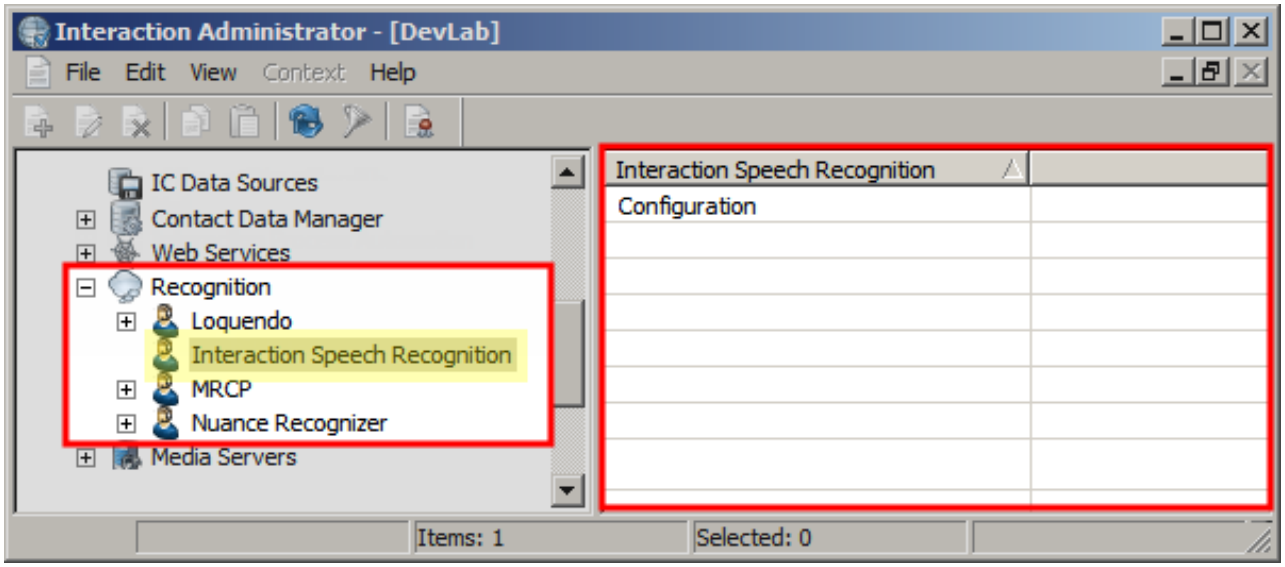
Adjust Audio Detection Sensitivity

You can adjust Interaction Speech Recognition's sensitivity to input noise by altering the value of the audio detection sensitivity property. The default value is 0.5, but you can change it to any value between 0.0 and 1.0.

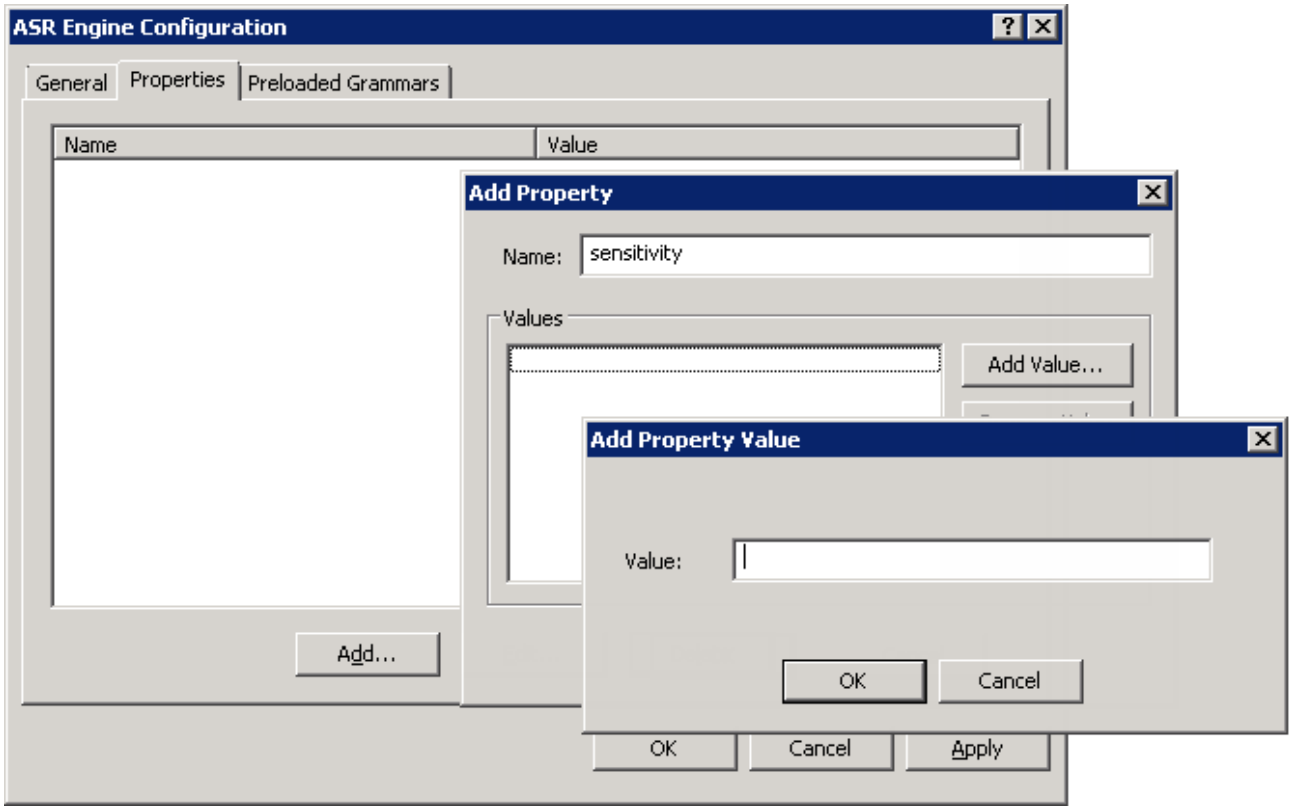
Genesys recommends using the default value of 0.5 as it is robust enough to ignore spurious noises and still trigger on true speech. A value of 0.0 configures the system to be the least sensitive to noise while a value of 1.0 makes it highly sensitive to quiet input. More specifically, a value of 0.0 causes Interaction Speech Recognition to treat spurious noise as silence and thus prevent it from triggering a speech event on such noise. At the other end of the spectrum, a value of 1.0 could possibly cause Interaction Speech Recognition to trigger a speech event on any noise.

To adjust the audio detection sensitivity

1. Open Interaction Administrator.
2. In the left pane, expand the **System Configuration** container.
3. Under the System Configuration container, expand the **Recognition** container.
4. In the Recognition container, click the **Interaction Speech Recognition** object.
5. In the right pane, double-click the **Configuration** item.



6. In the **ASR Engine Configuration** dialog box, click the **Properties** tab and then click **Add**.
7. In the **Add Property** dialog box, in the **Name** box, type *sensitivity* and then click **Add Value**.
8. In the **Add Property Value** dialog box, type a value between 0.0 and 1.0.
9. Click **OK** three times to close each of the open dialog boxes and enable the audio detection sensitivity setting.


ASR Engine Configuration
General
Properties
Preloaded Grammars
Name
Value
Add...
Add Property
Name:
sensitivity
Values
Add Value...
Add Property Value
Value:
OK
Cancel
OK
Cancel
Apply

Configure Interaction Attendant to Use Interaction Speech Recognition

You can configure Interaction Attendant, the Interactive Voice Response (IVR) product from Genesys, to work with Interaction Speech Recognition. Integrating these two products provides you with the following capabilities:

- Allow callers to speak the name of the CIC user to whom they want to communicate.
- Allow callers to speak IVR menu item selections or operations.

By default, Interaction Attendant enables the company directory speech recognition capability; however, you must configure Interaction Attendant to use Interaction Speech Recognition before the feature can function.

For more information, see the following:

- Create Users in Interaction Administrator
- Enable Speech Recognition for Company Directory in Interaction Attendant
- Use Interaction Speech Recognition for Interaction Attendant Operations and Menu
- Add Grammar Files for Preloading

Create Users in Interaction Administrator

CIC builds a grammar from the users that you define in Interaction Administrator. CIC can preload this grammar and transfer it to Interaction Media Servers where the processing for Interaction Speech Recognition occurs. This feature allows a caller to say the name of the person with whom the caller wants to communicate. CIC then transfers the caller to the recognized user.

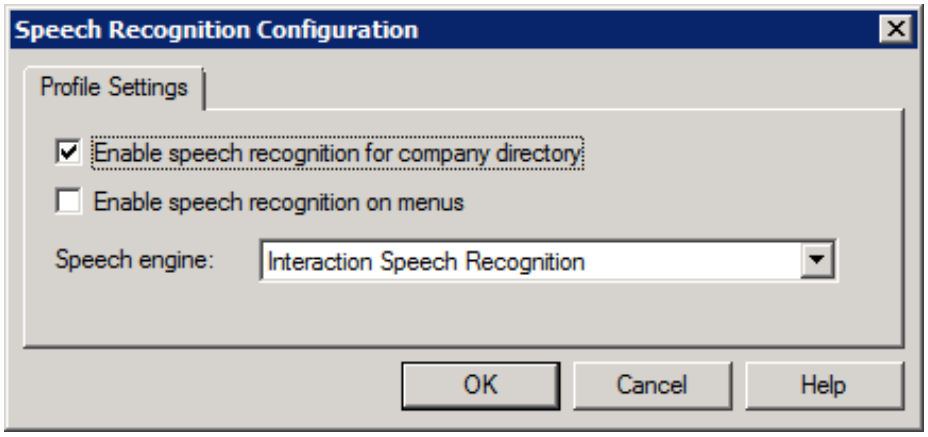
To create users in Interaction Administrator

1. Open Interaction Administrator.
2. In the left pane, expand the **People** container.
3. Under the **People** container, click **Users**.
4. In the right pane, right-click an open area and then click **New**. The **Entry Name** dialog box appears.
5. Type the name of a new CIC user. You can also click the button to the right of the **Name** box to select a user from available Microsoft Windows Server network domains.
6. Click **OK**. The **User Configuration** dialog box appears.
7. Specify the options for the new user. For more information about configuring CIC users, see "User Configuration" in the *Interaction Administrator* documentation.
8. Click **OK**. The system adds the user to the list of users.
9. Repeat this procedure for all users that you want to add to the company directory.

Enable Speech Recognition for Company Directory in Interaction Attendant

To enable speech recognition for company directory

1. Open Interaction Attendant.
2. In the left pane of the **Interaction Attendant** window, click the profile. The right pane displays the configuration interface for the selected profile.
3. In the right pane, in the **Node Characteristics** group, click **Configure Speech Recognition**. The **Speech Recognition Configuration** dialog box for the selected profile appears.



4. In the **Speech Recognition Configuration** dialog box, select the **Enable speech recognition for the company directory** check box.

   Important!
   To use Interaction Speech Recognition for other IVR operations, such as recognizing spoken menu items, you must select the **Enable speech recognition on menus** check box. However, this feature changes the default prompts associated to the **Enable speech recognition for company directory** feature. If you choose to enable speech recognition on menus and also leave it enabled for the company directory, Interaction Attendant doesn't prompt callers to speak the name of the CIC user with whom they want to communicate. To keep the prompting of the caller to supply the name of the user, modify the main greeting prompt to present the option to the caller.

5. In the **Speech engine** list box, click the **Interaction Speech Recognition** item.

   Note:
   If the **Interaction Speech Recognition** item is not present in the **Speech engine** list box, enable **Interaction Speech Recognition** in Interaction Administrator. For instructions, see Enable Interaction Speech Recognition.

6. In the **Speech Recognition Configuration** dialog box for the profile, click **OK**.
7. In the **Interaction Attendant** window menu, click **File** and then click **Publish**.

Use Interaction Speech Recognition for Interaction Attendant Operations and Menu Navigation

You can use Interaction Attendant and Interaction Speech Recognition to allow callers to select menu options and run operations using speech. By default, Interaction Attendant does not allow speech recognition for selecting menu items or for operations. Use the following procedures to enable and configure speech recognition for menu item selection and operations. These procedures assume that you enabled Interaction Speech Recognition as specified in Enable Interaction Speech Recognition.

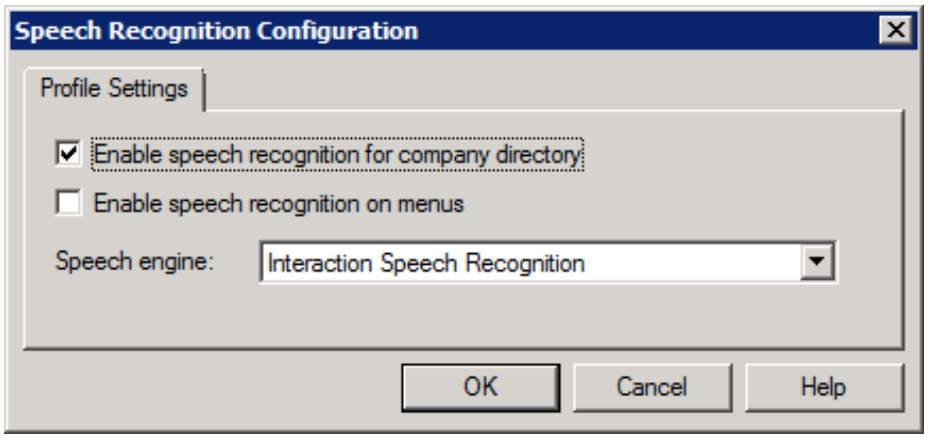
For more information, see the following:

- Enable Speech Recognition for Operations and Menu Navigation in Interaction Attendant
- Add an Operation Through Interaction Attendant
- Add Keywords and Phrases to an Interaction Attendant Operation

Enable Speech Recognition for Operations and Menu Navigation in Interaction Attendant

To enable speech recognition for operations and menu navigation

1. Open Interaction Attendant.
2. In the left pane of the **Interaction Attendant** window, click the profile.
3. In the right pane, click **Configure Speech Recognition**. The **Speech Recognition Configuration** dialog box appears.



4. Select the **Enable speech recognition on menus** check box.
5. Ensure that the value in the **Speech engine** list box is **Interaction Speech Recognition**.
6. Click **OK**.
7. (Optional) Using the **Profile Greeting** area in the right pane of the **Interaction Attendant** window, edit the main greeting prompt to inform callers that they can communicate with a CIC user by speaking the name of the user.
8. In the **Interaction Attendant** window menu, click **File** and then click **Publish**.

Add an Operation Through Interaction Attendant

You can add an operation to a schedule to allow callers to use speech to select a menu item.

To add an operation

1. Open Interaction Attendant.
2. In the left pane of the **Interaction Attendant** window, click the profile.
3. Click the schedule in the selected profile for which to allow callers to use speech to select a menu item.
4. From the menu, click **Insert** and then click **New Operation**. The system adds the operation to the schedule.

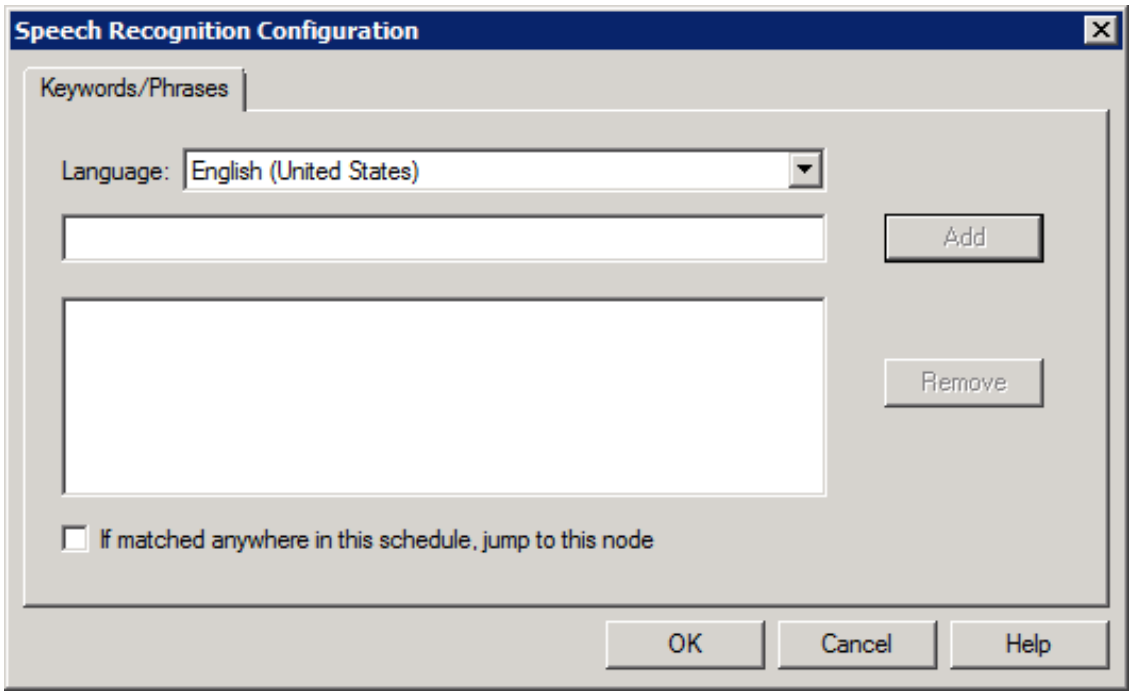
Add Keywords and Phrases to an Interaction Attendant Operation

After adding an operation to a schedule, specify the keywords and phrases that a caller can speak to cause selection of that operation.

Note:
You can add operations to Interaction Attendant schedules without assigning them to keypad digits. Then, those operations run using speech recognition only.

To add keywords and phrases

1. In the left pane of the **Interaction Attendant** window, click the operation that you added.
2. In the right pane, click **Configure Speech Recognition**. The **Speech Recognition Configuration** dialog box appears.



3. In the **Language** list box, ensure that the correct language appears.
4. In the next box, type a keyword or phrase that a caller must speak to run the operation and then click **Add**. The system adds the specified keyword or phrase to the next box.
5. Click **OK**.
6. If the selected operation requires it, provide any other configuration in the right pane of the **Interaction Attendant** window, such as selecting a workgroup queue to receive a callback request.

For more information about adding keywords and phrases for speech recognition in Interaction Attendant, see "Add Speech Recognition keywords or phrases for Inbound Call Operations" in the *Interaction Attendant Help* at https://help.genesys.com/cic/mergedProjects/wh_iat/desktop/interaction_attendant_help.htm.

Add Grammar Files for Preloading

If you already have large grammar files that you use for an Interactive Voice Response (IVR) system and they conform to the Speech Recognition Grammar Specification (SRGS) standard, you can preload those grammars for use with Interaction Speech Recognition.

For more information about speech recognition grammars, see Interaction Speech Recognition Grammars.

Note:
It is not necessary to preload small or simple grammars as they do not require significant time to transfer and compile. Adding grammars for preloading causes extra memory usage on Interaction Media Server.

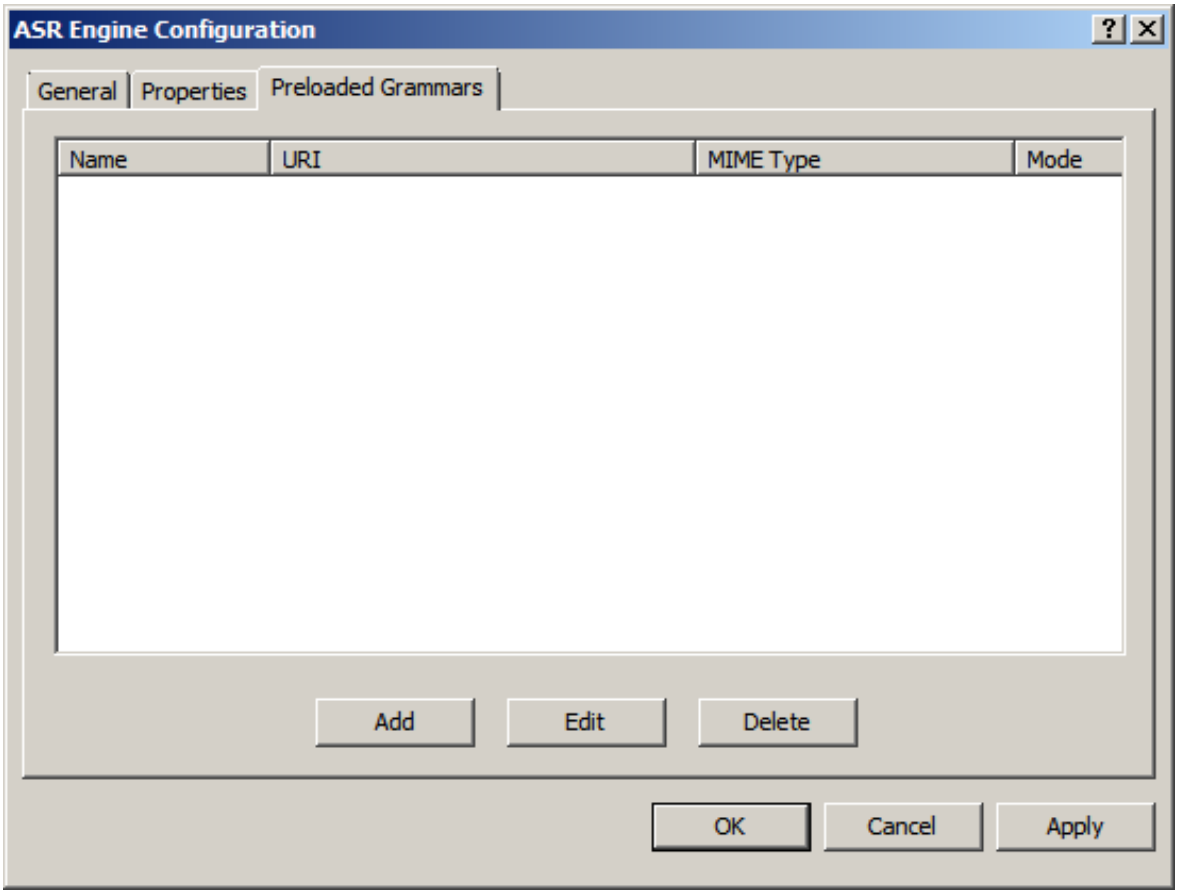
To add grammar files for preloading

1. Open Interaction Administrator.
2. In the left pane, expand the **System Configuration** container.

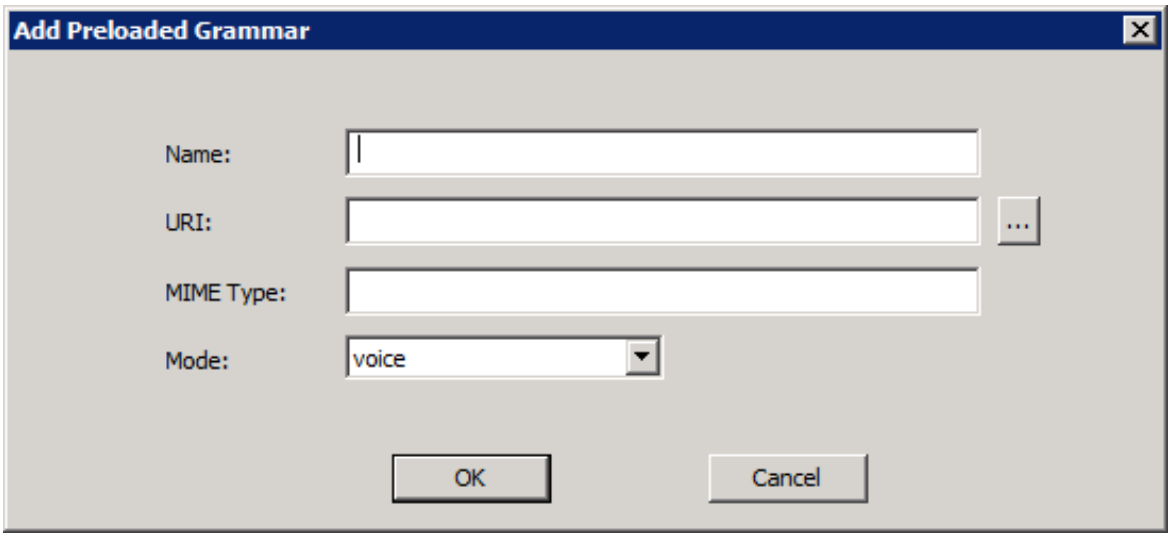
3. In the **System Configuration** container, expand the **Recognition** container.
4. Do one of the following:

- To only allow Interaction Speech Recognition to use the grammar, in the **Recognition** container, click the **Interaction Speech Recognition** object.
- To allow all speech recognition engines within CIC to use the grammar, click the **Recognition** container.

- In the right pane, double-click the **Configuration** item. The **ASR Engine Configuration** or **Recognition Configuration** dialog box appears, depending on whether you clicked **Interaction Speech Recognition** or **Recognition**.
- Click the **Preloaded Grammars** tab.



- On the **Preloaded Grammars** tab, click **Add**. The **Add Preloaded Grammar** dialog box appears.



Name: Name of the grammar. The Reco tools in handlers use this name to identify the grammar. For more information about the Reco tools in handlers, see the *Interaction Designer* documentation.

URI: Location of the grammar.

MIME Type: Multipurpose Internet Mail Extension (MIME) format of the grammar.

- For grammars in the GrXML format, use `application/srgs+xml`.
- For grammars in the ABNF format, use `application/srgs`.

Mode: Grammar mode. Select **Voice**.

8. Complete the information and then click **OK**. The new grammar appears in the list box on the **Preloaded Grammars** tab of the **ASR Engine Configuration** or **Recognition Configuration** dialog box.
9. In the **ASR Engine Configuration** or **Recognition Configuration** dialog box, click **OK**.

Interaction Speech Recognition Grammars

To recognize speech, Interaction Speech Recognition uses grammars that conform to the Speech Recognition Grammar Specification (SRGS) standard that the World Wide Web Consortium (W3C) international standards organization developed. A grammar is a text file in a special format that contains a set of words that a speech recognition engine can interpret from an utterance, such as the names of people in a company. An utterance is the speech that a caller provides in response to a prompt.

Important!
Interaction Speech Recognition supports voice grammars only. It does not support DTMF grammars. By default, the combination of Interaction Media Server and CIC provides support for recognizing DTMF tones.

For example, one built-in (default) grammar in Interaction Speech Recognition is the Boolean built-in grammar, which specifies the words associated to affirmative and negative utterances.

For more information, see the following:

- Grammar Types
- Grammar Syntax
- Best Practices
- Test Grammars
- Use Custom Grammars with Interaction Speech Recognition

Grammar Types

Following are the different types of speech recognition grammars that you can use with Interaction Speech Recognition.

Built-in grammars

Built-in grammars are a set of grammars that a speech recognition product supports by default. These grammars recognize a basic set of words that are necessary in nearly all speech recognition tasks.

The following table lists the types of words that Interaction Speech Recognition supports in its built-in grammars.

Built-in grammar type	Description
Boolean	This built-in grammar type contains the words Interaction Speech Recognition can identify for affirmative and negative utterances. For *yes* and *no* words, Interaction Speech Recognition returns the values of `true` and `false`, respectively.
Date	This built-in grammar type contains the words that Interaction Speech Recognition can identify as a calendar date, including the month, day, and year. Interaction Speech Recognition returns the date as a string of digits in the format of `yyyymmdd`. • `yyyy` represents the year (optional). • `mm` represents the month. • `dd` represents the numeric day. If a year is not specified, Interaction Speech Recognition inserts a question mark (`?`) for each digit.
Time	This built-in grammar type contains the words that Interaction Speech Recognition can identify as a specific time as it is measured by a clock, such as 10:35 AM. Interaction Speech Recognition returns the time as a five-character string in the format of `hhmmf`. • `hh` represents the hour. • `mm` represents the minute. • `f` represents the time format: • `p` = PM, such as *seven PM* • `a` = AM, such as *seven AM* • `h` = 24-hour format, such as *nineteen hundred* • `?` = Ambiguous, such as *seven o'clock*, which does not provide context

Digits	This built-in grammar type contains the words that Interaction Speech Recognition can identify as a string of individual digits. The digits grammar type is not the same as a number grammar type, where the placement of a digit has context. For example, a series of digits could represent an account number, a support case, or some other numeric identifier, such as 0045833154. *Numbers*, in the case of built-in grammars, represent amounts, such as 2,157 (two-thousand-one-hundred-fifty-seven). The digits built-in grammar type also supports the word *double* when callers reference two identical digits in succession such as *double 7* (77). **Note:** The digits built-in grammar type does not support usage of the `minlength`, `maxlength`, or `length` parameters. If you require a specific number of digits, you must add the functionality of verifying the length of the returned string of digits in your processing system, such as custom handlers or VoiceXML. Interaction Speech Recognition returns the string of digits.
Currency	This built-in grammar type contains the words that Interaction Speech Recognition can identify as being related to monetary amounts. Interaction Speech Recognition returns a string in the following ISO4217 international standard format: `UUUmmm.nn` • `UUU` = Currency type, such as USD **Note:** The en-us currency built-in grammar type of Interaction Speech Recognition recognizes only *dollars* and *cents*. • `mmm` = The number of whole units of the currency type • `nn` = The number of partial units of a whole unit
Number	This built-in grammar type contains the words that Interaction Speech Recognition can identify as a complete numeric value; not a series of digits. It also recognizes words that indicate a positive or negative value, such as *plus*, *positive*, *minus*, and *negative*. For example, if a caller says, *one-thousand-two-hundred-thirty-four*, Interaction Speech Recognition returns `1234`. If the caller says a word that indicates a positive or negative value, Interaction Speech Recognition includes an addition symbol (+) or hyphen (–), respectively.
Phone	This built-in grammar type contains the words that Interaction Speech Recognition can identify as a telephone number, including an optional extension. For example, if a caller says, *eight, eight, eight, five, five, five, one, two, three, four, extension five, six, seven,* Interaction Speech Recognition returns `8005551234x567`. A caller can also say numbers in place of digits, such as *eight hundred* (800) and *forty-seven* (47).

Custom grammars

Custom grammars are grammars that you create for one or more reasons. Such as:

- The built-in grammars do not provide enough recognition entries for all situations.
- The needs of the business require recognition of a specific set of words, such as names of products, names of subsidiary companies, industry-specific words (medical, technology, security, agriculture, government, and so on), or names of cities, states, or countries.

You create custom grammars as text files that specify the words and optional words for Interaction Speech Recognition to recognize. All custom grammars use the formats and syntax specified by the Speech Recognition Grammar Specification (SRGS) standard. For more information, see Grammar Syntax.

Tip:
When creating a custom grammar, ensure that you do not create circular references that result in infinite processing. However, when necessary, recursive grammars can provide a great deal of efficiency in your custom grammars.

Note:
Genesys recommends that you divide grammars that require more than a gigabyte of memory into smaller grammars in order to avoid running out of memory when loading.

Preloaded grammars

Sometimes, grammars are very large or complex. Downloading, compiling, and processing these large or complex grammars can impact performance, both in audio communications and with the resources of the processing system. For these reasons, you can specify that the processing systems preload those grammars. Processing systems outside an active speech recognition session download, compile, and cache preloaded grammars. When a speech recognition session then requires use of one of these grammars, the processing system already has the grammar in memory and can recognize utterances quickly.

You do not need to preload all grammars as each preloaded grammar occupies memory of the processing systems. In most situations, the grammar files are small or simple enough that downloading, compiling, and processing occurs without impacting the session or the performance of the processing system.

Genesys recommends that you preload grammars only if testing of a non-preloaded grammar causes performance problems, such as delays in responsiveness and audio quality issues.

VoiceXML grammars

VoiceXML is a W3C standard for using text files to facilitate interactive voice conversations between people and computers. VoiceXML uses text-based files with specialized eXtensible Markup Language (XML) elements that can define words for Text-to-Speech (TTS), speech recognition, grammars, conversation management, and playing recorded audio.

If you use Interaction Speech Recognition with a VoiceXML system, your VoiceXML files can include or reference grammars that Interaction Speech Recognition uses. VoiceXML grammars also use the Speech Recognition Grammar Specification (SRGS) standard.

When Interaction Speech Recognition recognizes a speech pattern based on the supplied grammar, it returns that data to the VoiceXML interpreter.

Pronunciation lexicon documents

A grammar can optionally reference one or more external pronunciation lexicon documents. Interaction Media Server supports the loading, processing, and usage of external pronunciation lexicons in Interaction Speech Recognition as required by the Speech Recognition Grammar Specification (SRGS) 1.0. For more information about SRGS requirements regarding external pronunciation lexicons, see https://www.w3.org/TR/speech-grammar/#S4.10.

User-defined dictionaries

Beginning with CIC 2019 R3, Interaction Speech Recognition supports user-defined dictionaries. This feature allows you to specify alternative pronunciations of uncommon names and words to use for speech recognition and in the playing of prompts in speech synthesis.

Pronunciation Lexicon Specification (PLS) is a definition that allows automated speech recognition and text-to-speech engines to use external dictionaries during speech recognition and speech synthesis. For more information, see https://www.w3.org/TR/pronunciation-lexicon/.

You specify the alternative pronunciations in a lexicon file and the system matches them during grammar-based speech recognition. This feature is especially useful for recognition of dialects and names where pronunciations can vary. The system will also match default pronunciations during speech recognition.

VoiceXML

The VoiceXML standard supports a feature that can use external dictionaries with speech recognition and TTS engines. For speech recognition, you specify the dictionary in the grammar file in a specially-defined lexicon element. You can specify multiple dictionaries at different points in the IVR dialog flow. For TTS prompts, you specify the dictionary in the lexicon element defined inside a prompt element. For more information about definitions and the VoiceXML standard, see https://www.w3.org/TR/voicexml20/.

In CIC, reference the VoiceXML document containing the lexicon in the **Document URI** field of a handler subroutine. For more information about handler setup, see the https://help.genesys.com/cic/mergedProjects/wh_tr/desktop/pdfs/voicexml_tr.pdf. For more information about specifying lexicons in grammar files, see https://www.w3.org/TR/speech-grammar/#S4.10.

Example of a lexicon file with an alternate pronunciation for the name Anna

```
<?xml version="1.0" encoding="UTF-8"?>
<lexicon version="1.0"
      xmlns="http://www.w3.org/2005/01/pronunciation-lexicon"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.w3.org/2005/01/pronunciation-lexicon
        http://www.w3.org/TR/2007/CR-pronunciation-lexicon-20071212/pls.xsd"
      alphabet="x-inin-arpabet" xml:lang="en-US">
  <lexeme>
    <grapheme>Anna</grapheme>
    <phoneme>ah n ah</phoneme>
  </lexeme>
</lexicon>
```

Example of lexicon usage in VoiceXML

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
ASR of name Anna followed by a greeting
-->
<vxml xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
   http://www.w3.org/TR/voicexml20/vxml.xsd"
   version="2.0">
  <form>
    <field name="name">
      <prompt>
         Please say your name
      </prompt>
      <grammar type="application/srgs+xml" src="name.grxml"/>
    </field>
    <filled>
      <prompt>
<lexicon uri="file:///path/to/file/alt_pron_for_anna.pls"/>
Hello <value expr="name"/></prompt>
      <exit/>
    </filled>
  </form>
</vxml>
```

Grammar Syntax

Speech Recognition Grammar Specification (SRGS) consists of two syntaxes for defining grammars:

- Augmented Backus-Naur Form (ABNF).
- Grammar Extensible Markup Language (GrXML)

Interaction Speech Recognition supports both syntaxes in grammar files.

SRGS uses Semantic Interpretation for Speech Recognition (SISR) v1.0 for semantic interpretation.

Example ABNF grammar file

The ABNF syntax uses symbols, words, and operators; much like a scripting or programming language.

```
// Header
#ABNF 1.0;
language en-US;
mode voice;
// Root rule
root $yesorno;
// Rules
$yes = yes | correct | yeah | affirmative;
$no = no | nope | negative;
$yesorno = $yes | $no;
```

To ensure that you can easily identify ABNF grammars when browsing a file system, use the `.gram` file name extension.

Example GrXML grammar file

The GrXML syntax uses elements and pairs of elements defined within angle bracket characters (`< >`).

```
<!--Header-->
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="yesorno"
mode="voice">
<!--Rules-->
<rule id="yesorno">
<one-of>
<item><ruleref
 uri="#yes" /></item>
<item><ruleref
 uri="#no" /></item>
</one-of>
</rule>
<rule id="yes">
<one-of>
<item>yes</item>
<item>correct</item>
<item>yeah</item>
<item>affirmative</item>
</one-of>
</rule>
<rule id="no">
<one-of>
<item>no</item>
<item>nope</item>
<item>negative</item>
</one-of>
</rule>
</grammar>
```

To ensure that you can easily identify GrXML grammars when browsing a file system, use the `.grxml` file name extension.

For more information about the SRGS specification and how to create grammars, see http://www.w3.org/TR/speech-grammar/.

Best Practices

Following are some best practices that Genesys recommends regarding the development of grammars for use with Interaction Speech Recognition:

Design Grammars and Prompts Simultaneously

Genesys recommends that you design and develop grammars and prompts simultaneously. Using this method ensures that all prompts have expected utterances and associated actions. It also ensures that you do not have rules in grammars and operations that never run.

Remove Ambiguity

In grammars, ambiguity refers to entry of a word that does not provide enough context as to its meaning. When a caller speaks the ambiguous word, the grammar does not have enough details or does not cause the system to request the necessary clarification. For example, if you ask a caller to say the name of the city from which the person is calling, ensure that you account for cities in the United States having the same name in multiple states, such as *Springfield*. Ensure that your grammars provide you with the exact information that you seek, including necessary details.

Other examples of ambiguity in grammars are homonyms and homophones. Homonyms are different words that have the same spelling and pronunciation, such as *change*. Homophones are different words that do not have the same spelling but do have the same pronunciation, such as *red* and *read*, and *sale* and *sail*.

You can find lists of homonyms and homophones on various Internet websites.

Duplicate Tokens

Ensure that your grammars do not include the same token more than once. The following examples display a token replicated in two separate rules.

ABNF example of duplicate token

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $Example;
$Example = $Yes1 | $Yes2;
$Yes1 = (yes [sir|maam] | yeah);
$Yes2 = (yes | okay);
```

GrXML example of duplicate token

In the following example, it is possible for the `yes` token to match both the `$Yes1` and `$Yes2` rules. Creating duplicate tokens can also result from merging multiple grammars into a single grammar.

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="Example"
mode="voice">
<rule id="Example">
<one-of>
<item><ruleref

uri="#Yes1"/></item>
<item><ruleref

uri="#Yes2"/></item>
</one-of>
</rule>
<rule id="Yes1">
<one-of>
<item>yes</item>
<item>
<one-of>
<item

repeat="0-1">sir</item>
<item

repeat="0-1">maam</item>
</one-of>
</item>
<item>yeah</item>
</one-of>
</rule>
<rule id="Yes2">
<one-of>
<item>yes</item>
<item>okay</item>
</one-of>
</rule>
</grammar>
```

Use SISR Tags for Operations

Semantic Interpretation for Speech Recognition (SISR) is a W3C standard for defining a result for an utterance within an SRGS grammar. The result is the data that you want Interaction Speech Recognition to return, regardless of whether callers said different words within the rule or one or more extra words.

The following examples display use of SISR tags for operations.

ABNF example of SISR tag

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $Operator;
$Operator =
[Please] [transfer [me] [to] | call | dial]

[an] [the] operator
{out.action="transfer"; out.user="operator"};
```

GrXML example of SISR tag

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="Operator"
mode="voice">
<rule id="Operator">
<item repeat="0-1">Please</item>
<one-of>
<item

repeat="0-1">transfer</item>
<item>
<one-of>
<item

repeat="0-1">me</item>
<item

repeat="0-1">to</item>
</one-of>
</item>
<item

repeat="0-1">call</item>
<item

repeat="0-1">dial</item>
</one-of>
<item repeat="0-1">an</item>
<item repeat="0-1">the</item>
<item>operator<tag>out.action="transfer";out.user="operator";</tag></item>
</rule>
</grammar>
```

In these example grammars, the `Operator` rule allows many optional words to be present in a request to speak with an operator:

- *Please transfer me to an **operator**.*
- ***Operator**.*
- *Call an **operator**.*
- *Dial the **operator**.*

All these utterances result in Interaction Speech Recognition returning data defined and populated in the last line of the example defining the rule variable, its properties (`action` and `user`), and its assigned values (*`transfer`* and *`operator`*). The receiving

system can then act based on the data that Interaction Speech Recognition returns. You can also define rule variables, properties, and default values within a separate rule in the grammar and change the values within other rules in the same grammar.

For more information about SISR with examples in ABNF and GrXML, see http://www.w3.org/TR/semantic-interpretation/.

Use Grammar Weights and Probabilities

SRGS grammars support the usage of weights to specify the preference of matching a recognition of one response over another. For example, if your grammar prompted callers to speak the city from which they are calling, those city names may exist in multiple states, regions, or countries. By assigning weights to multiple instances of a word or phrase (token), you are specifying that one of them is more likely to be spoken over another. If you do not assign weight values to multiple instances of a word or phrase within a rule, all definitions of that word or phrase share the same likelihood of occurrence.

Note:
Only add weights in your grammars when tuning efforts provided evidence that they are necessary. Entering estimations of weights can result in decreased recognition confidence levels of accurate responses.

The following examples provide recognitions for the city of *Springfield*. In these examples, *Springfield* could indicate more than one state. Using weights in your grammar enables you to prefer *Springfield, Missouri* over *Springfield, Illinois* as a match to *Springfield*.

ABNF example of weight

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $Cities;
$Cities =
(/2.0/ springfield [missouri]

{out.city="Springfield"; out.state="Missouri"})


|
(/1.2/ springfield [illinois]

{out.city="Springfield"; out.state="Illinois"})


|
dallas [texas] {out.city="Dallas";

out.state="Texas"};
```

GrXML example of weight

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="cities"
mode="voice">
<rule id="cities">
<one-of>
<item

weight="2.0">
Springfield
<item

repeat="0-1">Missouri</item>
<tag>out.city="Springfield";out.state="Missouri";</tag>
</item>
<item

weight="1.2">
Springfield
<item

repeat="0-1">Illinois</item>
<tag>out.city="Springfield";out.state="Illinois";</tag>
</item>
<item>
Dallas
<tag>out.city="Dallas";out.state="Texas";</tag>
</item>
</one-of>
</rule>
</grammar>
```

For more information about SRGS weights, see http://www.w3.org/TR/grammar-spec/#S2.4.1.

Use Filler Rules or Garbage Rules to Catch Unimportant Words in Responses

In SRGS grammars, you use *filler* rules or *garbage* rules to separate the unimportant, contextual words and phrases from the important words and phrases (*tokens*) that result in specific data and actions.

Filler rules

Filler rules in grammars are when you define the optional words and phrases that a caller might say. Since you define common unimportant words in filler rules, the confidence level of the recognition is usually higher. However, the larger a grammar increases in size, the lower the confidence levels might descend without further tuning. To use filler rules in grammars, ensure that you include only those unimportant rules that a majority of IVR callers use. Defining filler rules with unimportant words that callers say infrequently or not at all impacts the efficiency and accuracy of speech recognition.

ABNF example of filler rules

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $DepartureCity;
$DepartureCity =
$PreFiller
(springfield [missouri] {out.city="Springfield";

out.state="Missouri"} |
dallas [texas] {out.city="Dallas";

out.state="Texas"})
$PostFiller;
$PreFiller =
[ ([i would | id] like to (leave

| depart) from) | ([i am | im](leaving | departing) from) ];
$PostFiller =
[ please ];
```

GrXML example of filler rules

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="cities"
mode="voice">
<rule id="cities">
<ruleref

uri="#filler"/>
<one-of>
<item>
Springfield
<item

repeat="0-1">Missouri</item>
<tag>out.city="Springfield";out.state="Missouri";</tag>
</item>
<item>
Dallas
<item

repeat="0-1">Texas</item>
<tag>out.city="Dallas";out.state="Texas";</tag>
</item>
</one-of>
<ruleref

uri="#filler"/>
</rule>
<rule id="filler">
<one-of>
<item

repeat="0-1">I would</item>
<item

repeat="0-1">Id</item>
<item

repeat="0-1">I am</item>
<item

repeat="0-1">Im</item>
</one-of>
<item

repeat="0-1">like to</item>
<one-of>
<item

repeat="0-1">leave</item>
<item

repeat="0-1">depart</item>
<item

repeat="0-1">leaving</item>
<item

repeat="0-1">departing</item>
</one-of>
<item

repeat="0-1">from</item>
</rule>
</grammar>
```

Garbage rules

Garbage rules are a part of the SRGS special rules—along with NULL and VOID—in the SRGS specification. Garbage rules use a single entity to ignore unimportant words that may or may not occur within a grammar rule. While you can use garbage rules to account for a wide number of possible responses, garbage rules generally decrease speech recognition confidence levels as it is more difficult for the speech engine to discern what speech does not match a defined token.

Important!
While filler rules and garbage rules can provide benefits, Genesys recommends that you use these types of rules sparingly. For filler rules, define only unimportant words and phrases that occur frequently in responses to avoid needlessly inflating grammar sizes. For garbage rules, do not use them as filters for large numbers of unimportant words as recognition confidence levels decrease.

The following grammar examples define filler rules or garbage rules to use in the rule for recognizing a city of departure, as when reserving passage on a commercial flight. The important part of the grammar is the city of departure. The filler rules and garbage rules are for any other commonly spoken words that are unimportant in the response, as a caller might say in the following examples:

- "*I would like to depart from* **Springfield, Missouri**, *please*."
- "*I'm leaving from* **Springfield**."
- "*I am departing from* **Springfield**."
- "*Leave* **Springfield**."

ABNF example of garbage rules

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $DepartureCity;
$DepartureCity =
$GARBAGE
((springfield [missouri]) {out.city="Springfield";

out.state="Missouri";}) |
(dallas [texas] {out.city="Dallas";

out.state="Texas";})
$GARBAGE;
```

GrXML example of garbage rules

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="cities"
mode="voice">
<rule id="cities">
<ruleref

special="GARBAGE"/>
<one-of>
<item>
Springfield
<item

repeat="0-1">Missouri</item>
<tag>out.city="Springfield";out.state="Missouri";</tag>
</item>
<item>
Dallas
<item

repeat="0-1">Texas</item>
<tag>out.city="Dallas";out.state="Texas";</tag>
</item>
</one-of>
<ruleref

special="GARBAGE"/>
</rule>
</grammar>
```

Reference Built-in Grammars Instead of Recreating Functionality

Built-in grammars provide speech recognition in an Interactive Voice Response (IVR) application for basic responses; however, you do not have control over the data that built-in grammars return. For this purpose, you may need to create your own custom grammars to provide that information, such as specifying account numbers, identification numbers, or other specific information in a certain format with a set amount of data. In custom grammars, you can refer to built-in grammars that already provide the required speech recognition functionality.

For example, if you wanted callers to speak the number of animals seen during a recent hike, you could create a custom grammar to specify that it uses the built-in number grammar appended with any entries that you define in the custom grammar. Instead of recreating the functionality of number recognition in the custom grammar, you can reference the *number* built-in grammar, as displayed in the following example grammars.

ABNF built-in grammar reference example

```
#ABNF 1.0;
language en-us;
mode voice;
tag-format <semantics/1.0>;
root $wildlife;
$wildlife=
$number $type {out.number=rules.number;

out.type=rules.type;};
$type=
((bird | birds) {out="birds";}

|
fish {out="fish";} |
(mammal | mammals) {out="mammals";}

|
(animal | animals) {out="animals";});
$number=
$<builtin:grammar/number>;
```

GrXML built-in grammar reference example

```
<?xml version="1.0"?>
<grammar xmlns="http://www.w3.org/2001/06/grammar"
xml:lang="en-US"
root="wildlife"
mode="voice">
<rule id="wildlife">
<ruleref

uri="#number" />
<ruleref uri="#type"

/>
<tag>out.number=rules.number;</tag>
<tag>out.type=rules.type;</tag>
</rule>
<rule id="type">
<one-of>
<item>
<one-of>
<item>bird</item>
<item>birds</item>
</one-of>
<tag>out="birds";</tag>
</item>
<item>fish

<tag>out="fish";</tag></item>
<item>
<one-of>
<item>mammal</item>
<item>mammals</item>
</one-of>
<tag>out="mammals";</tag>
</item>
<item>
<one-of>
<item>animal</item>
<item>animals</item>
</one-of>
<tag>out="animals";</tag>
</item>
</one-of>
</rule>
<rule id="number">
<ruleref

uri="builtin:grammar/number" />
</rule>
</grammar>
```

Do Not Try to Address All Possible Responses

When developing custom grammars, avoid including all possible words that a caller could speak. Instead, limit the words to the ones that callers most commonly use. Small grammars provide more accuracy than large grammars. Grammars that contain all possible words do not provide any benefit if most those words are rarely or never spoken.

Following are some guidelines for determining what words to include in your custom grammars.

Ensure that prompts are guiding the caller to a limited set of responses

While this task is not related to the actual development of grammars, it is relevant as prompts influence what responses callers speak. For example, if you want to know if a caller is calling from a business or a residence (a simple choice), the prompt provides those choices, as shown in the following example:

Are you calling from a residence or a business?

For this prompt, callers likely response with *residence* or *business*.

If the prompt does not provide choices, the responses could vary drastically:

> *From where are you calling?*

Does this prompt expect a geographical location, a social location (home, office), or some other information? This vague prompt can cause any number of responses:

- *New York*
- *In my car*
- *From my house*
- *ABC Company*
- *From my couch*
- *My cell phone*
- *The corner of 10th and Main*

While you had an idea of the responses you wanted, failure to consider how callers perceive the prompt can require large, complex grammars to account for the various types of responses that they provide. Even with those large, complex grammars, you still may not receive the simple information that you seek: business or residence.

Avoid defining all possible synonyms

Synonyms are different words that have the same or similar meaning. Using the example from the previous section where the expected responses are *residence* or *business*, do not enter all of the possible words that have similar meaning as the expected responses. The following table displays some examples of synonyms for *residence* or *business*:

Residence synonyms	Business synonyms
• Home • House • Apartment • Condominium • Condo • Townhouse • Dwelling • Domicile • Estate	• Office • Headquarters • Corporation • Factory • Store • Restaurant • Firm • Hospital • School

If you cannot provide choices in your prompts, ensure that your grammars only contain the most common responses to the prompt. Do not include words that are rarely spoken in the responses to a prompt unless you are tuning grammars and are testing optional words to increase accuracy.

Focus on expected responses to the prompt

Your custom grammars should contain words that relate to a prompt only. For example, if a caller navigated to a prompt of *Please say the digits of your account number*, your grammar should not attempt to cover the rare instance of a caller saying, *I just want to talk to a live person.* If you enable the grammar to recognize these words in responses to this prompt, you would need to include those words in all responses to all prompts. This mistake increases your grammars needlessly, which results in lower overall recognition accuracy.

The purpose of your Interactive Voice Response (IVR) system is to enable callers to navigate menus by speaking responses to prompts, not to simulate an artificial intelligence that can process and act on any speech.

Identify and Fix Problems After Deployment

After an initial deployment of Interaction Speech Recognition, test your system to ensure that Interaction Speech Recognition recognizes caller responses correctly. After testing, you may need to improve your grammars to account for the following items:

- Unexpected responses, including commonly occurring optional words that you did not include originally:
  - If the unexpected responses are a new interpretation that you want to act upon, include those responses in new rules.
  - If the unexpected responses are words that you did not include for an existing rule, add those responses as extra or optional words in that rule.
- Variations or alternate pronunciations in caller responses to a prompt:
  - Ensure that the variation or alternate pronunciation is a common occurrence in caller responses. Do not add to the grammar for one instance of the problem.
  - Add the variation or alternate pronunciation to existing rules. Do not create new rules for the variation that would return the same semantic interpretation as an existing rule.
- Misrecognized or missed responses:
  - Ensure that you are allowing enough time for the caller to provide a complete response.
  - Ensure that the call had enough audio clarity to facilitate recognition of the response. A call with low audio quality significantly impacts speech recognition.
  - Adjust your systems that receive interpretations not to act when the confidence level of an interpretation is below a certain value.
  - Analyze your grammars to determine whether you can modify the rules to increase confidence levels, such as making a word of a phrase optional instead of required. For example, in a grammar in ABNF format, change `technical support representative` to `[technical] support representative` or `[technical] support ([rep] | [representative] | [person])`.
- Identify and correct ambiguity in grammars. For more information, see Remove Ambiguity.
- Poor audio quality issues based on listening to recordings:
  - Ensure that no excessive noise—background noise in the caller's environment or noise introduced in the lines of communication—is present. Excessive noise can cause missed responses, inaccurate recognitions, and low confidence levels.
  - Test your IVR grammars using codecs with higher bandwidths, such as G.711. Low-bandwidth codecs reduce dynamic audio ranges, which can result in decreased recognitions and lower confidence levels.
  - Disable silence suppression to reduce bandwidth. Silence suppression is a telephony feature where audio doesn't transmit for one party while the other party is speaking or playing audio. Silence suppression could cause clipping of audio where a party speaks and the feature doesn't deactivate quickly. This situation could result in Interaction Speech Recognition not receiving and analyzing the first portion of a response.

Test Grammars

When you create a grammar, test it to ensure that it is valid and functioning as intended. The following sections provide information on how to test grammars and the data that can help you determine its level of functionality.

Test grammar validity

You can test grammars for validity with the following command line program on the CIC server:

`D:\I3\IC\Server\RecoGrammarValidatorU.exe` (default install location)

To specify a grammar for the program to parse, supply the full path as displayed in the following example:

`D:\I3\IC\Server\RecoGrammarValidatorU.exe D:\I3\IC\Resources\< file name>.<extension>`

If the program parses the grammar successfully, the following message appears in the program output in the Command Prompt window:

```
Grammar successfully parsed:
```

> **Note:**
> The `RecoGrammarValidatorU.exe` program does not test for duplicated tokens or ambiguity.

Analyze functionality

When you deploy a grammar for testing or usage, analyze its performance to ensure that it is performing as necessary. The following list provides some guidelines to assist you in analyzing grammars:

- Collect and analyze logs and recordings each day.
- Create a process for analyzing calls logs that is repeatable. A consistent process ensures that the data you collect is valid.
- Create a spreadsheet or database for recording and tracking the speech recognition data. At a minimum, track the following items:
  - Call ID and time
  - Grammars used in each call
  - Utterance recording
  - Utterance transcription
  - Utterance decoded
  - Recognition properties, such as confidence level
  - Recognition results
- Analyze a significant number of calls and identify trends before making any changes.

Use Custom Grammars with Interaction Speech Recognition

You can use custom grammars with Interaction Speech Recognition through multiple methods:

- Preload the grammar through Interaction Administrator
- The Reco Register Grammar tool in custom handlers through Interaction Designer
- VoiceXML documents

Genesys recommends that you store your custom grammars in the `Resources` directory of the CIC server. By default, this directory uses the following path: `D:\I3\IC\Resources`.

Troubleshooting

Following are some corrective procedures for issues that you may encounter while using Interaction Speech Recognition.

Grammar not loading

If your grammar doesn't load, it may require more than the maximum amount of memory, or one or more of the entries my have too many phonemes. Genesys recommends that you divide grammars that require more than a gigabyte of memory into smaller grammars in order to avoid running out of memory when loading.

Windows Event Log contains entries for invalid grammars

If CIC preloads an invalid grammar, it creates an entry in the Windows Event Log. To ensure that a preloaded grammar is valid, run the grammar validation tool. For more information, see "Test grammar validity" in Test Grammars.

Audio problems with Interaction Speech Recognition

If you experience audio problems with Interaction Speech Recognition, you can enable diagnostic recordings that Genesys Customer Care can review to determine the cause of the problem.

There are three Interaction Media Server properties that you can use to capture diagnostic recordings for Interaction Speech Recognition:

- **RecognizerDiagnosticRecording** – This property creates `.sasf` files for each interaction. These files are only available on the Media Server – the CIC server doesn't fetch them. You can only set this property through the Media Server Web interface.

  Warning!
  - Diagnostic recordings for Interaction Speech Recognition are large `.sasf` files, which can exhaust data storage capabilities when diagnostic recordings create indefinitely. Ensure that you create diagnostic recordings for a brief period. You should enable diagnostic recordings only with direction from Genesys Customer Care.
  - If you enable diagnostic recordings for Interaction Speech Recognition through the Interaction Media Server web interface, the resulting `.sasf` files create on the data storage device of Interaction Media Server and remain there until you remove them.

- **recognizerDiagnosticLog** – This property creates `.sasf` and `.xml` files containing audio data along with the corresponding events. The CIC server fetches these files and stores them in its media storage location. You can only set this property through the ASR Engine Configuration dialog in Interaction Administrator.

  Warning!
  If you enable diagnostic logs for Interaction Speech Recognition through the Interaction Speech Recognition **ASR Engine Configuration** dialog box in Interaction Administrator, every Interaction Media Server connected to the CIC server creates diagnostic logs (`.sasf` and `.xml` files). CIC then moves all diagnostic log files to itself. Ensure that you do not continue creating diagnostic logs through this method for an extended period as it could result in significant network bandwidth usage and exhaustion of the data storage capabilities of the CIC server.

- **AsrDiagnosticRecording** – This property creates three `.sasf` files for echo analysis. These files are only available on the Media Server – the CIC server doesn't fetch them. You can only set this property through the Media Server Web interface.

  Caution!
  Enable the **AsrDiagnosticRecording** property only with direction from Genesys Customer Care. ASR Diagnostic Recordings have a large impact on the performance of Interaction Media Server. These impacts reduce the number of resources that Interaction Media Server can use to service both the number of calls and the features used in those calls.

Warning!

- Diagnostic recordings for Interaction Speech Recognition are large `.sasf` files, which can exhaust data storage capabilities when diagnostic recordings create indefinitely. Ensure that you create diagnostic recordings for a brief period. You should enable diagnostic recordings only with direction from Genesys Customer Care.
- If you enable diagnostic recordings for Interaction Speech Recognition through the Interaction Media Server web interface, the resulting `.sasf` files create on the data storage device of Interaction Media Server and remain there until you remove them.

For more information about these properties, see "Interaction Media Server Config-Properties page" in the *Interaction Media Server Technical Reference* at https://help.genesys.com/cic/mergedProjects/wh_tr/desktop/pdfs/media_server_tr.pdf.

Enable diagnostic recordings for Interaction Speech Recognition through Interaction Media Server

To enable diagnostic recordings

1. Log on to the Interaction Media Server on which you want to enable and create diagnostic recordings. The **Status-About** page of the Interaction Media Server web interface appears.
2. In the upper right corner, click the **Settings** icon.
3. On the left side of the page, click the **Properties** tab.
4. In the unlabeled box at the bottom of the left column, type the following: `RecognizerDiagnosticRecording`.
5. To the right of the box, click **Add**. The system adds the **RecognizerDiagnosticRecording** property to the list of properties.
6. To the right of the **RecognizerDiagnosticRecording** property, click **true** in the list box.
7. At the bottom of the page, click **Apply**.

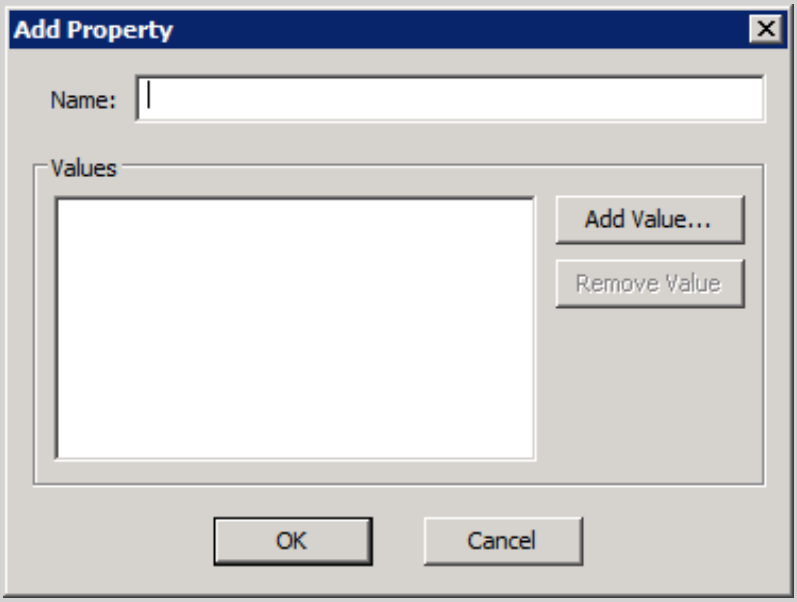
Enable diagnostic logging for Interaction Speech Recognition through Interaction Administrator

Caution!

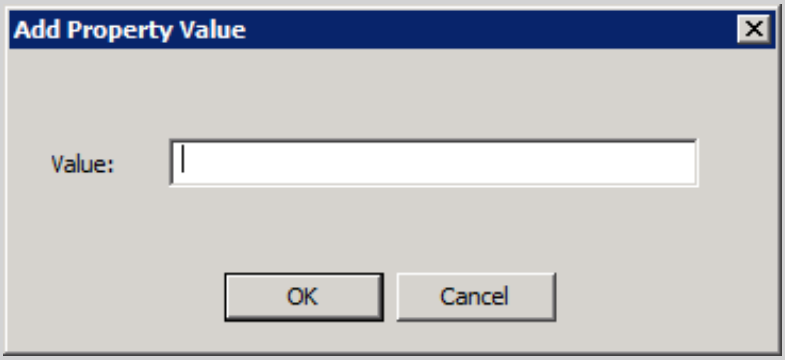
Doing this procedure enables diagnostic recordings for Interaction Speech Recognition on all Interaction Media Servers connected to this CIC server. Ensure that you have the necessary network bandwidth to accommodate the transfer of all diagnostic recording files from each Interaction Media Server to the CIC server. Ensure that the CIC server has enough free data storage space to accommodate all of the diagnostic recording files from all Interaction Media Servers.

To enable diagnostic logging

1. Log on to Interaction Administrator.
2. In the left pane, expand the **System Configuration** container and then expand the **Recognition** container.
3. In the **Recognition** container, click **Interaction Speech Recognition**.
4. In the right pane, double-click the **Configuration** item. The **ASR Engine Configuration** dialog box appears.
5. Click the **Properties** tab.
6. Click **Add**. The **Add Property** dialog box appears.



7. In the **Name** box, type the following: `recognizerDiagnosticLog`.
8. Click **Add Value**. The **Add Property Value** dialog box appears.



9. In the **Value** box, type `true` and then click **OK**.
10. In the **Add Property** dialog box, click **OK**. The **recognizerDiagnosticLog** property appears on the **Properties** tab.
11. In the **ASR Engine Configuration** dialog box, click **OK**.

Change Log

The following table lists the changes to the *Interaction Speech Recognition Technical Reference* since its initial release.

Date	Change
24-September-2013	Initial Release
04-October-2013	Removed inaccurate admonishment regarding user credentials
06-August-2014	Updated documentation to reflect changes required in the transition from version 4.0 SU# to CIC 2015 R1, such as updates to product version numbers, system requirements, installation procedures, references to PureConnect Product Information site URLs, and copyright and trademark information.
18-December-2014	Added information about the audio detection sensitivity property.
20-April-2015	• Added information about the recognizerDiagnosticLog property to the Troubleshooting section. • Updated Copyright and Trademarks page. • Updated the document to reflect the CIC 2015R3 version.
01-July-2015	• Updated cover page to reflect new color scheme and logo. • Copyright and Trademark Information • Updated the document to reflect the CIC 2015R4 version.
09-October-2015	Updated the document to reflect the CIC 2016 R1 version.
04-February-2016	• Added a link to the PureConnect Documentation Library at https://help.genesys.com/cic. • Copyright and Trademark Information • Updated the document to reflect the CIC 2016 R2 version.
02-June-2016	• Added Europe – French (fr-FR) as a supported language in Interaction Speech Recognition requirements. • Minor edits
06-December-2016	Added Europe – Spanish (es-ES) as a supported language in Interaction Speech Recognition requirements.
24-May-2017	Added language support for Mandarin Chinese and Italian.
14-December-2017	Added Pronunciation lexicon documents. Also applied Genesys branding to this document.
16-July-2018	Corrected typographical and grammatical errors.
06-March-2019	Added support for Dutch, German, Japanese, and Brazilian Portuguese.
28-May-2019	Added recommendation to avoid loading grammars with large, unpronounceable words.
06-June-2019	Reorganized the content only, which included combining some topics and deleting others that just had an introductory sentence such as, "In this section...".
11-June-2019	Added support for user-defined dictionaries.