PureConnect®

2020 R4

Generated:

09-November-2020

Content last updated:

21-June-2019

See Change Log for summary of changes.

GENESYS™

CIC Speech Recognition Overview

Technical Reference

Abstract

This document covers the speech recognition subsystem, a powerful component of Customer Interaction Center (CIC) that gives customers the flexibility in defining voice and DTMF inputs for their Interactive Voice Response (IVR) system. Speech recognition functions are accessible using handler tools that provide building blocks for the speech application. These recognition tools allow a handler developer to take advantage of simple yes/no voice responses or develop something more complex using standard based SRGS grammars.

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

Table of Contents

Introduction to CIC Speech Recognition

A large part of any "Interactive Voice Response" (IVR) application is input recognition. The application has to be able to process a user's response to a specific prompt intelligently. PureConnect's recognition (Reco) subsystem interprets user input accurately, whether it's DTMF or voice. The Reco subsystem has a built-in DTMF recognizer that detects accurately the frequencies or tone of any key on a touchtone keypad or telephone. It also wraps engines from leading automatic speech recognition (ASR) vendors to take advantage of their advance voice processing capabilities such as endpointing, noise cancellation, and acoustic-phonetic analysis.

Another quality of the Reco subsystem is its advance grammar support and semantic interpretation. It uses W3C's SRGS standard for specifying grammars and supports SISR for semantic interpretation of recognition results. Grammar registration and result analysis are simplified using a wide array of recognition tools.

The Reco subsystem supports other features such as load balancing, intelligent grammar caching, built-in grammar support, and web-based administration.

The *CIC Speech Recognition Overview Technical Reference* describes how to get started using recognition tools in handlers to create simple applications. One example walks you through prompting and processing a simple yes or no response from a user. Another example shows how to register a grammar and use the **Reco Input** tool to use that grammar.

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

Prerequisites

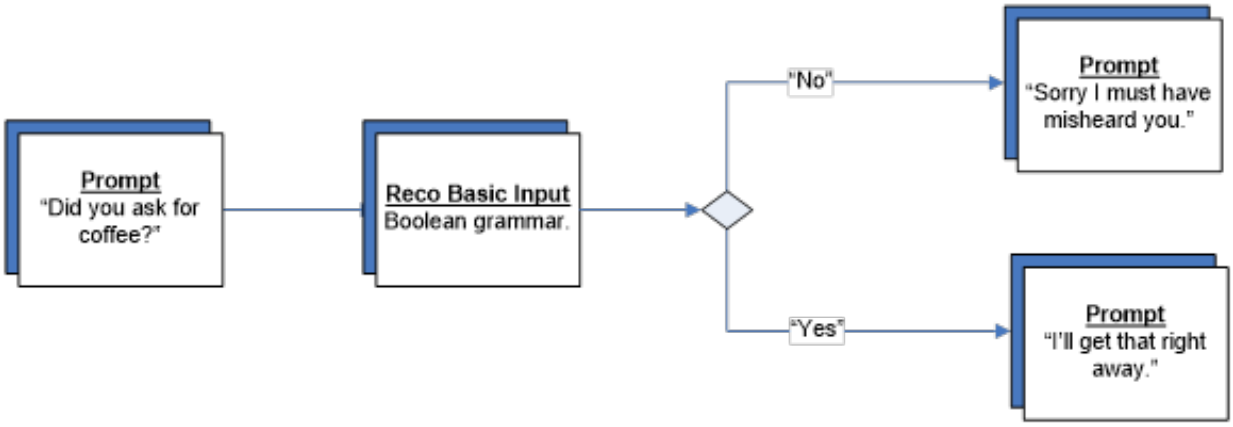
Before you begin, ensure that you meet the following prerequisites:

- Your organization purchased a valid Speech Recognition license for the CIC server.
- Your organization purchased at least one Continuous Speech Processing (CSP)/Speech resource from the Telephony vendor.
- Your organization installed at least one ASR Engine with valid licenses.
- Your organization configured the ASR Server to work with the CIC server using the web configuration of the ASR Server.

Process Basic Inputs

Typically, the system only requires simple responses from a user. For example, when the system requests confirmation ("Did you say ____?") or the system request a quantity ("How many ____ do you want?"). In these cases, the **Reco Basic Input** tool is sufficient for capturing input from the user. It is always a good idea to plan your call flow.

Example call flow



For more information, see the following:

- Start a recognition session
- Add a prompt
- Configure Reco Basic Input
- Analyze the result
- Release the Session
- Reco Basic Input Example

Start a recognition session

The first handler step is to start a new recognition session using the **Reco Initialize** tool. You can also define other session parameters with the **Reco Initialize** tool, such as which ASR engine to use, which language to use, or which input modes to use. The only required parameter is the interaction ID, which is a requirement for most of the tool steps. For this example, we accept the default parameters of the session.

Add a prompt

To get input from a user, configure the application to request it. Setting up a prompt requires much thought to ensure that the application is user friendly.

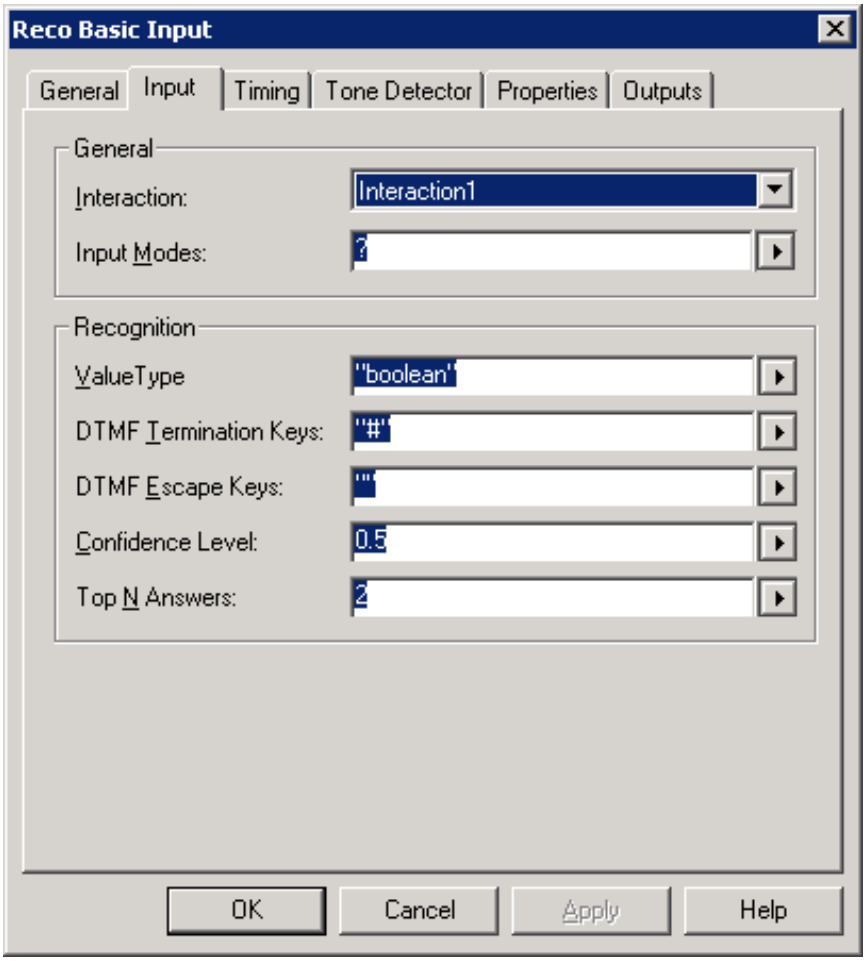
Consider the following:

- Accuracy of the prompts based on the grammars used.
- Whether the prompts are for DTMF and voice.
- Whether the prompts sound natural.

You can use prompts that are either pre-recorded or synthesized dynamically. Pre-recorded prompts usually sound more natural, although sometimes you can get equal quality voices from modern Text-to-Speech engines such as ITTS or Vocalizer. For this example, using synthesized text prompts using the **Play String** tool under the telephony tools is sufficient.

Configure Reco Basic Input

Use the **Reco Basic Input** tool to recognize simple inputs against a built-in grammar. Depending on the ASR Engine installed, the following grammars may be available: Boolean, digits, date, currency, number, phone, and time. Use the **Value Type** list box to specify which grammar to use. These grammar types correspond to the VoiceXML grammars.



Analyze the result

There are several tools that allow you to analyze the result of a Reco Input. The results are in an XML format that is similar to W3C's *Natural Language Semantics Markup Language* (NLSML). This format ensures that the results returned are independent of the engine used. The results are in the following format:

```
<result>
   <hypothesis mode="voice|dtmf" conf="confidence" grammar="grammar-id">
       <utterance> recognized words or tokens if available </utterance>
       <slot name="slot-name1" conf="confidence1"> slot value1 </slot>
       <slot name="slot-name2" conf="confidence2"> slot value2 </slot>
       …
       <slot name="slot-nameN" conf="confidenceN"> slot valueN </slot>
       Custom Elements (engine specific)
   </hypothesis>
   <hypothesis … >
      …
   </hypothesis>
   Custom Elements (engine specific)
</result>
```

Result

The <result> element is the root of the recognition result. It has no attributes. The <hypothesis> child elements are in decreasing order of confidence. Thus, the highest scoring hypothesis is first.

Hypothesis

The <hypothesis> element represents how a recognition engine may have interpreted what the user said (also called utterance). Therefore, a result can contain several <hypothesis> elements with varying utterances and confidences. You can use the Top N Answers property to limit the number of hypotheses returned. A hypothesis has the following attributes:

mode

This attribute specifies whether the system obtained this hypothesis through speech or DTMF input. Possible values are:

- voice - The hypothesis is based on speech input.
- dtmf - The hypothesis is based on DTMF input.

conf

Confidence score of this hypothesis. The value ranges from 0.0 to 1.0.

grammar

This attribute contains the Grammar ID of the grammar that accepted this hypothesis.

Utterance

The <utterance> element contains the transcription of the recognized utterances, either as words or tokens. The exact representation in this string is engine-dependent and may contain special filler tokens (such as from garbage rules). Basically, this element contains the tokens that caused the grammar match. The value of this element is not for automatic processing but rather for debugging and logging purposes. For DTMF input, it contains the pressed keys without the termination character. This element is empty for engines that don't supply the raw utterance/tokens.

Slot

Each recognition hypothesis may contain one or more slots that the grammar fills. The value of the element represents the value of the slot. Some engines support nested interpretation results and thus slot elements may be nested. The <slot> element has the following attributes:

name

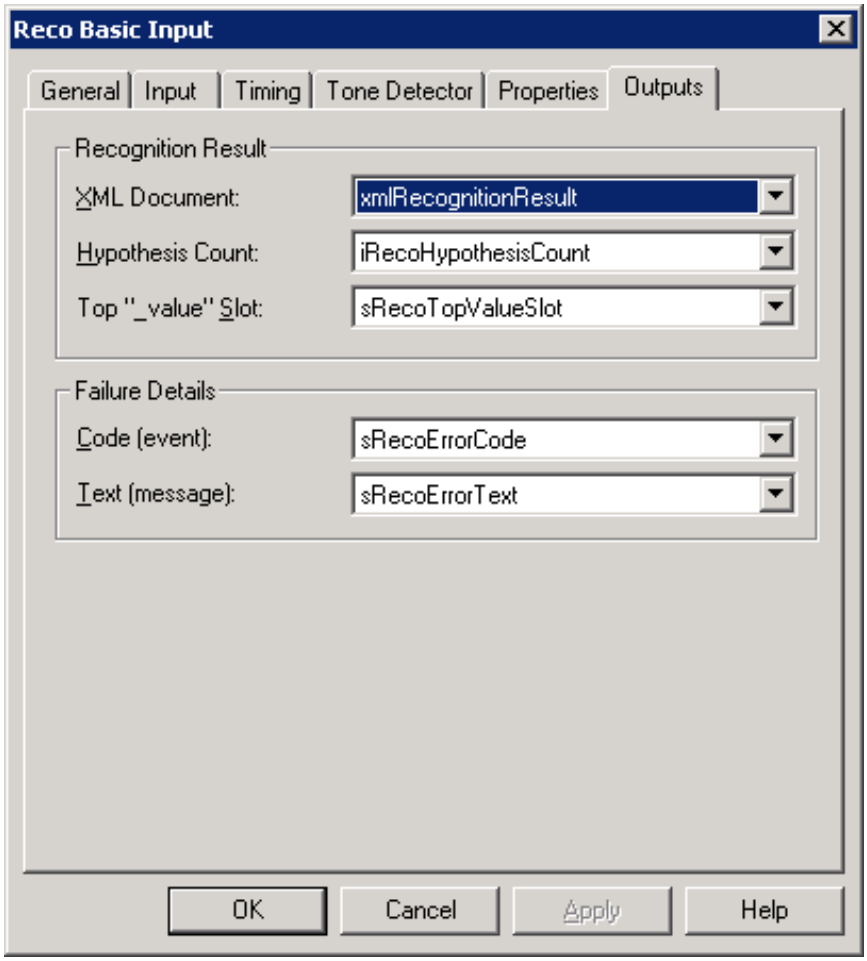
The name of the slot. This attribute is optional, as some engines may not provide names for the slots. The slots are in the order that the ASR engine provides (usually, order of tags in grammar).

conf

Confidence score of this slot. This attribute is optional and only present for engines that support this feature. If the attribute is not present, clients can assume that the confidence is 1.0.

Custom elements

Both <hypothesis> and <result> elements may have custom elements. The engine integration module creating the recognition result appends these elements to the end of the corresponding element. They provide an engine integration module with the ability to return any engine-specific data in the recognition result (for example, more statistics). The Reco subsystem or tools do not interpret these custom elements.



The Reco tools that are available to analyze results allow an application to iterate through the XML results and view the properties for each hypothesis and slot. For built-in grammars (like the example), the recognition results have a special slot named `_value` that contains the normalized string representation of the data type that the built-in grammar represents. For example, the Boolean built-in grammar has a result value of `true` or `false`.

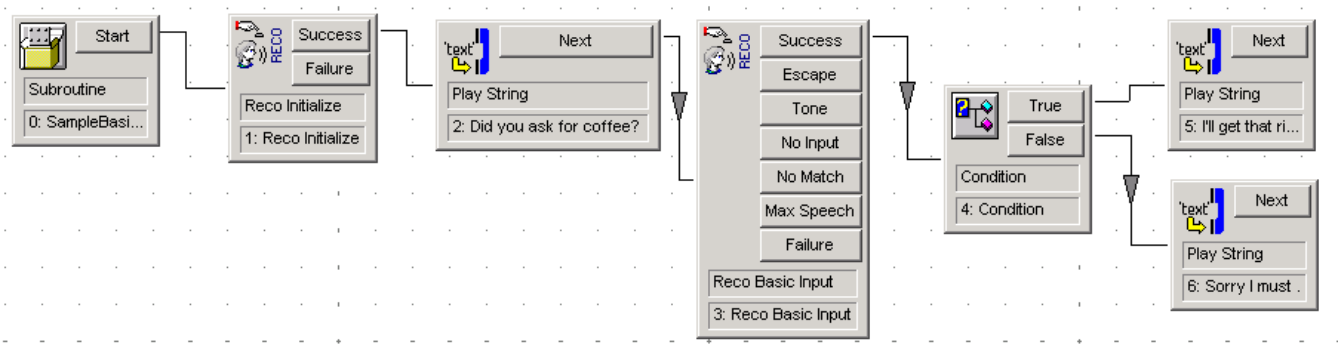
The **Reco Basic Input** tool exposes this value using the **Top "_value" Slot** output parameter that's in a string variable. In the example, it compares the variable to `true` or `false`.

Release the Session

A handler can explicitly use the **Reco Release** tool to indicate that the interaction is finished with the recognition session. Use with care because all context information for the session is lost. Also, it can be expensive to set up a new session so an interaction has to be certain that it is finished with recognition before calling **Reco Release**.

If you don't use the **Reco Release** tool, the session releases automatically when the interaction disconnects.

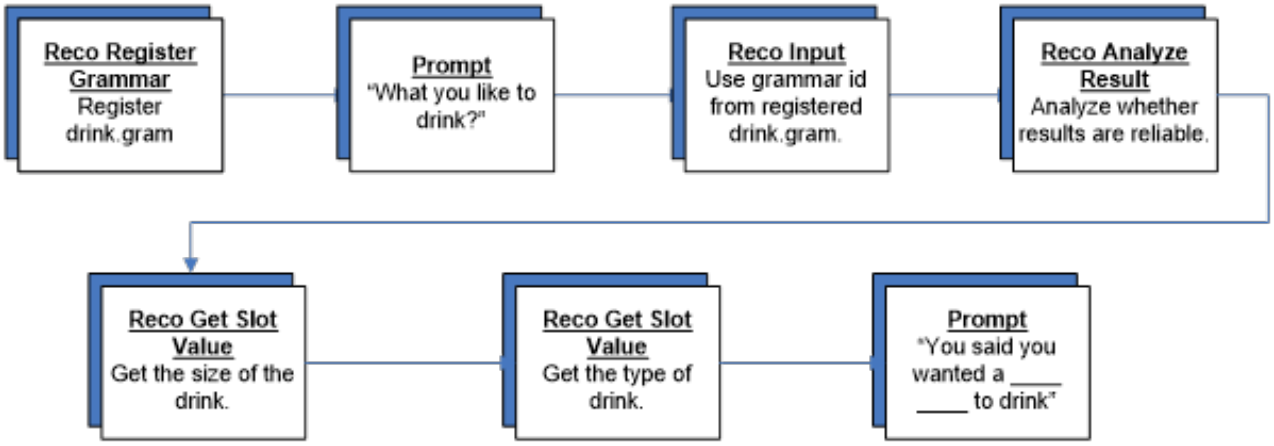
Reco Basic Input Example



Process Grammar Referenced Inputs

You can handle more advance inputs using the **Reco Input** tool. The **Reco Input** tool takes in a list of grammars to do recognition against. Ensure that you registered the grammars previously using the **Reco Register Grammar** or **Reco Register Grammar String** tool.

Example call flow



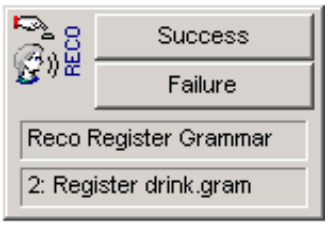
For more information, see the following:

- Register a Grammar
- Configure Reco Input
- Analyze Reco Results
- Reco Get Slot Value/Reco Bind Slot Values
- Reco Input Example
- Grammar Example

Register a Grammar

The first step is to register a grammar to use during the **Reco Input**. The Reco subsystem supports W3C's SRGS format, which provides a standard way to specify grammars across all engines. It also supports engine-specific grammars when needed, such as Nuance's GSL format and Open Speech Recognizer's binary grammar format. To register a grammar, a handler can use the **Reco Register Grammar** tool and specify the path to the grammar file. Or, it can use the **Reco Register Grammar String** and specify the source of the grammar directly in the tool step.

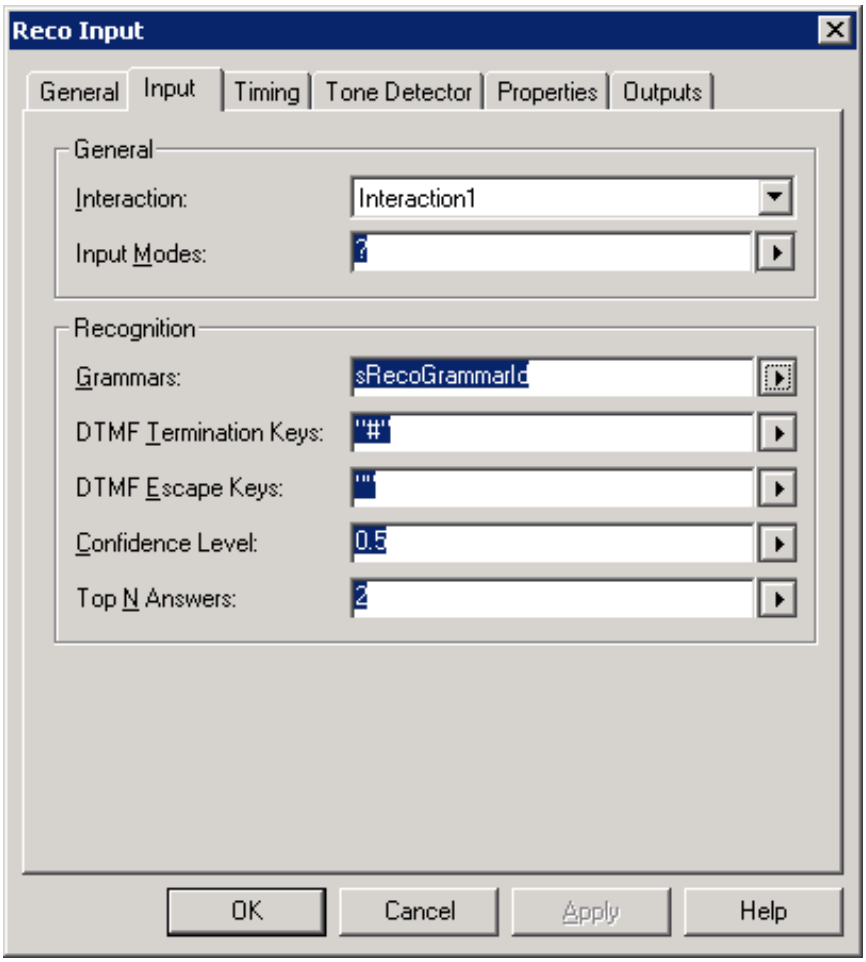
In the following example, we are registering an SRGS grammar file in ABNF format called `drink.gram`.



Configure Reco Input

Calling the **Reco Input** tool starts the recognition and sends the request to the ASR Server. The tool step takes in a space-delimited list of grammars to use during the recognition. All other parameters are optional. Like the **Reco Basic Input** tool, this tool returns the recognition result in XML format.

In the following example, we are using the grammar ID that the **Reco Register Grammar** tool returned.



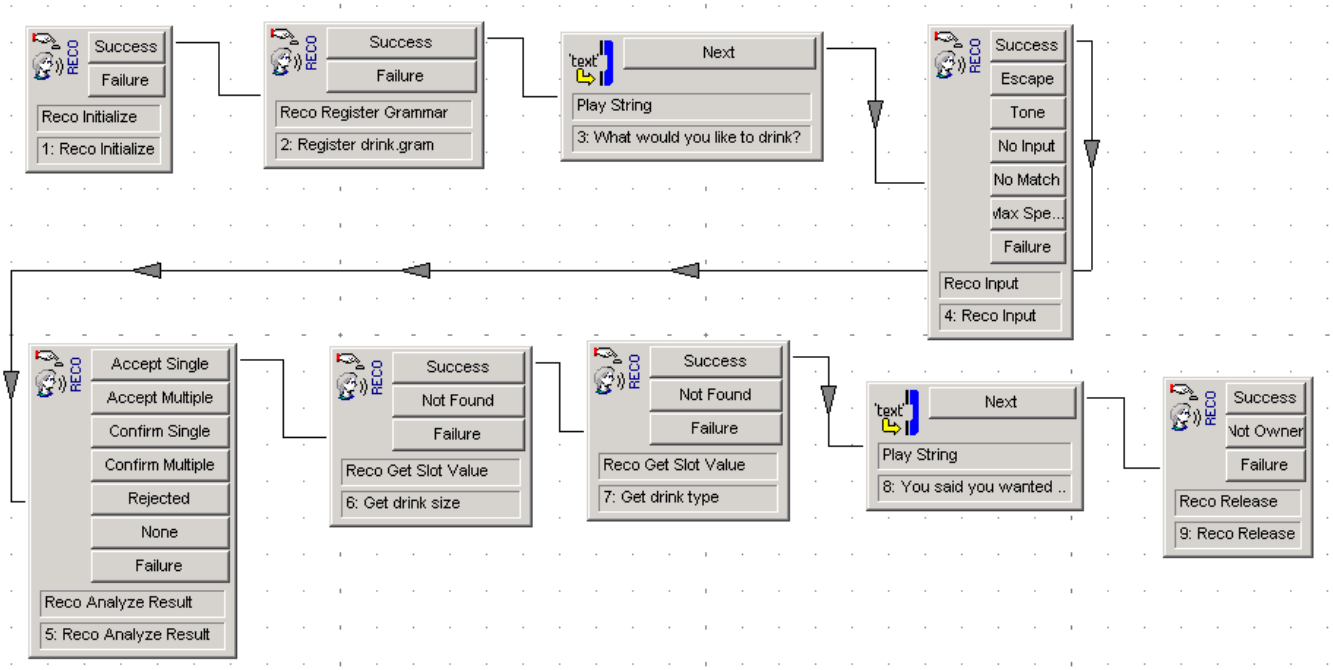
Analyze Reco Results

The **Reco Analyze Results** tool is useful for analyzing whether the recognition results are reliable enough to use, or whether multiple results need disambiguated. If you have a grammar with several items that sound alike and can be mistaken for one another, we recommend that you include this step.

Reco Get Slot Value/Reco Bind Slot Values

Results contain slots that define the semantic meaning of an utterance. This tool extracts the values of those slots from a specific hypothesis. In our example, the slots contain the size and type of drink requested. You can also use the **Reco Bind Slot Values** toolstep to extract several slot values at once and bind them to existing variables.

Reco Input Example



Grammar Example

The following example is of the SRGS grammar file used in the previous example.

```
   #ABNF 1.0;
   language en-us;
   mode voice;
   tag-format <semantics/1.0>;
   root $root;

   public $root = [I would | "I'ld" ] [like]

[a | an | one ]
                $size

{out.size=rules.latest();}
                $drink

{out.drink=rules.latest();}
                [please];

   private $size = small {out="small";}|

medium {out="medium";}| large {out="large";}|
extra
large {out="extra

   large";};

   private $drink = [diet {out="diet ";}]

(pepsi {out=out+"pepsi";}| (coke | coca cola) {out=out+"coke";}

| sprite

   {out=out+"sprite";}| mountain dew {out=out+"mountain

dew";});
```

Support Other Languages

By default, the Recognition subsystem uses the default language configured for CIC. You can view the default language in the System Configuration container in Interaction Administrator. To use a different language for a recognition session, specify an ISO language code during the **Reco Initialize** tool step. Also, you have to ensure that all grammars used for that session are in the language specified since most recognition engines only support a single language at a time using a specific set of acoustic models for that language. Each ASR engine has its own setup for different languages so it is best to consult the ASR engine's documentation for instructions.

Error Handling

Most Reco Tools return errors in the form of an error code and description. The following table provides a summary of those error codes.

Code	Description
connection.disconnect	Call disconnected during the operation.
noinput	No input provided during the timeout period.
nomatch	Input provided didn't match any grammars.
maxspeechtimeout	Maximum speech timeout exceeded.
com.inin.input.escape	Escape key pressed.
com.inin.input.tone	Fax tone detected.
error	Unspecified error occurred
error.unsupported.format	Specified (media) type for the resource not supported.
error.unsupported.language	Specified language not supported.
error.unsupported.builtin	Specified built-in grammar not supported.
error.noresource	Operation failed because the resource limit was reached.
error.noresource.license	Operation failed because a license was unavailable. For example, ASR port license.
error.noresource.cpustarvation	Recognition stopped because of excessive CPU load.
error.noauthorization	Operation failed because user didn't have the appropriate permissions (does not include file access errors).
error.semantic	Runtime error occurred. For example, attempt to divide by 0.
error.com.inin	PureConnect-specific error prefix.
error.com.inin.interaction_id	Specified interaction ID is invalid (not a known interaction).
error.com.inin.interaction_type	Specified interaction type is invalid (not a call).
error.com.inin.inputmodes	Specified input modes are invalid (mask to degenerate case).
error.com.inin.ownership	Specified ownership token does not represent the current owner of the interaction or the ownership was lost.
error.com.inin.parameter[.name]	Specified parameter is invalid.
error.com.inin.mode	Error related to the modes.
error.com.inin.mode.invalid	Specified mode is invalid.
error.com.inin.type	Media (MIME) type error occurred.
error.com.inin.type.invalid	Specified media type is invalid.
error.com.inin.unsupported	Operation or function not supported.
error.com.inin.timeout	Operation timed out. **Note:** This error is *not* an input timeout (which returns "throughnoinput"). For Notifier operations, it corresponds to `MSG_TIMEDOUT`.

error.com.inin.win32.<xxx>	Common Win32 error. <xxx> is the numeric error code.
error.com.inin.cancelled	Operation canceled for some reason.
error.com.inin.internal	Unspecified internal error occurred (see trace log).
error.com.inin.shutdown	A system is in the process of shutting down or is shut down already.
error.com.inin.notifier	Notifier error prefix.
error.com.inin.notifier.rejected	Notifier request rejected (subsystem down?). Corresponds to MSG_REJECTED.
error.com.inin.notifier.noconn	No Notifier connection exists. Corresponds to MSG_NOCONNECTION.
error.com.inin.notifier.cancelled	Notifier operation canceled. Corresponds to MSG_CANCELLED.
error.com.inin.notifier.badconn	Notifier connection failed or shut down. Corresponds to MSG_BADCONNECTION.
error.com.inin.notifier.wokeup	Notification loop woke up. Corresponds to MSG_WOKEUP.
error.com.inin.reco.session.inactive	Operation is only valid when a Reco session was created (and operation doesn't create a session automatically).
error.com.inin.reco.session.tied	Cannot change the ASR engine as the session already has an engine and the engine disallows changes.
error.com.inin.reco.asr	ASR server or engine integration error.
error.com.inin.reco.asr.engine	ASR engine error.
error.com.inin.reco.asr.engine.unknown	Specified ASR engine not supported.
error.com.inin.reco.asr.engine.parameter	Specified ASR engine parameters are invalid.
error.com.inin.reco.asr.engine.error	Error occurred starting the ASR engine.
error.com.inin.reco.feature	Specified feature not supported.
error.com.inin.reco.busy	Recognition is active; cannot execute the operation.
error.com.inin.reco.property.unknown	Attempt to access an unknown property.
error.com.inin.reco.property.read_only	Attempt to modify a read-only property.
error.com.inin.reco.property.invalid_value	Attempt to set an invalid property value.
error.com.inin.reco.customop.unknown	Attempt to invoke a custom operation that's unknown or not supported.
error.com.inin.reco.verifier	Speaker training, verification, or identification error.
error.com.inin.reco.verifier.key	Key specified for verification or identification not found.
error.com.inin.grammar	Unregistered Grammar ID or undefined value type specified.
error.com.inin.grammar.unknown	Unknown or invalid grammar specified for input or unregistering.
error.com.inin.grammar.id.duplicate	Specified Grammar ID in use already.
error.com.inin.grammar.id.invalid	Specified Grammar ID is invalid. For example, format or syntax.
error.com.inin.grammar.rendering	Failed to render the grammar into an ASR engine-specific grammar.
error.com.inin.grammar.language	Language error occurred. Most likely, the grammar contains an unsupported language or engine doesn't support language attachments.

error.com.inin.grammar.tags	Error related to the semantic interpretation tags of the grammar occurred. For example, engine integration doesn't support global SI tags.
error.com.inin.grammar.runtime.sisr	Runtime error occurred interpreting the tags as SISR scripts.
error.badfetch	Fetching the grammar data or compilation failed.
error.badfetch.uri	URI used to designate the grammar is invalid.
error.badfetch.builtin	Built-in grammar URI is invalid (mode or type).
error.badfetch.encoding	Invalid character encoding specified.
error.badfetch.grammar	Compilation of the grammar failed.
error.badfetch.grammar.syntax	Compilation of the grammar failed because of a syntax error.
error.badfetch.grammar.syntax.sisr	Compilation of the grammar failed because the content of a tag is not a valid SISR script.
error.badfetch.grammar.mode	Grammar has a different mode than specified in the tool.
error.badfetch.grammar.type	Media type of the grammar not supported, invalid, or doesn't match the data.
error.badfetch.grammar.size	Grammar data size is excessive.
error.badfetch.file.notfound	File not found.
error.badfetch.file.accessdenied	Access to the file denied.
error.badfetch.file.sharingviolation	Sharing violation occurred.
error.badfetch.file.win32.<xxx>	Some other error occurred. <xxx> is Win32 error code.

Change Log

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

Date	Change
25-February-2013	Updated Copyright and Trademarks for 2013
01-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 Interactive Intelligence Product Information site URLs, and copyright and trademark information.
01-July-2015	Updated cover page to reflect new color scheme and logo. Updated copyright and trademark information.
09-October-2015	Updated the document to reflect the CIC 2016 R1 version.
04-February-2016	• Updated Copyright and Trademarks for 2016. • Updated the document to reflect the CIC 2016 R2 version. • Added a link to the CIC Documentation Library at help.genesys.com.
24-October-2017	• Rebranded this document to apply Genesys styles and terminology. • Updated Copyrights and Trademarks • ISR Confidence Scoring: The accuracy of Interaction Speech Recognition (ISR) confidence scoring has been improved for short phrases in selected languages.
04-January-2018	Corrected several minor typographical errors.
13-July-2018	Corrected typographical and grammatical errors.
21-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...".