Powered by Universal Speech Solutions LLC

UniMRCP

Asterisk Modules

Usage Guide

Revision: 48

Last Updated: June 21, 2017

Created by: Arsen Chaloyan

Table of Contents

1 Overview

UniMRCP integration allows Asterisk to facilitate the use of the industry standard media resource servers for implementation of speech-enabled applications in accordance with the MRCP specifications. UniMRCP supports both version 1 and 2 of the MRCP protocol.

This guide describes how to configure and use the UniMRCP modules for Asterisk.

1.1 Installation

For installation instructions, start with one of the guides below:

- Installation from Source
- RPM Package Installation (Red Hat / Cent OS)
- Deb Package Installation (Debian / Ubuntu)

1.2 Applicable Versions

Instructions provided in this guide are applicable to the following versions.

UniMRCP modules for Asterisk 1.3.0 and above

1.3 Supported MRCP Servers

The UniMRCP modules for Asterisk are known to work with the following MRCP servers.

MRCP Server	ASR	TTS
UniMRCP Server	✓	✓
LumenVox Media Server	✓	✓
Nuance Speech Server	✓	✓

Note: installation of the MRCP server is not covered in this document.

1.4 Asterisk Modules

There are two Asterisk modules provided:

- res_speech_unimrcp (Generic Speech Recognition API)

- app_unimrcp (Suite of UniMRCP Applications)

1.5 Licensing

Since Asterisk is distributed under the GPLv2 license, and the UniMRCP modules are loaded by and directly interface with Asterisk, the GPLv2 license applies to the UniMRCP modules too.
See the following link for more information regarding the GPLv2 license.

GNU General Public License, version 2

2 Generic Speech Recognition API

2.1 Overview

The module *res_speech_unimrcp.so* is an implementation of the Generic Speech Recognition API of Asterisk, based on the UniMRCP client library.

2.2 Configuration

The module *res_speech_unimrcp.so* loads the UniMRCP client library by means of the XML configuration format defined in the Client Configuration Guide. The configuration file is located in the directory */usr/local/unimrcp/unimrcpclient.xml* by default.
Module-specific configuration parameters are specified in the file *res-speech-unimrcp.conf*, located in the directory */etc/asterisk* by default. This configuration file uses the Asterisk native configuration format, where a configuration file is broken into various sections with the section name surrounded by square brackets.

General Settings

The section *general* contains global settings of the module.

Name
[general]

Settings

Name	Description
unimrcp-profile	References a profile defined in the UniMRCP client configuration. The profiles are located in the directory */usr/local/unimrcp/client-profiles* by default.
log-level	Specifies the log level, one of: • EMERGENCY • ALERT • CRITICAL • ERROR • WARNING • NOTICE • INFO

	• DEBUG.

Example

The section *general* contains the parameter *unimrcp-profile* set to *uni2* and the parameter *log-level* set to *DEBUG*.

```
[general]
unimrcp-profile = uni2
log-level = DEBUG
```

Common Grammars

The section *grammars* optionally specifies a set of grammars, which are applicable to and need to be loaded for all the MRCP sessions established from the client to the server. The grammars are specified by means of name-value pairs, where the name is an arbitrary but unique string and the value specifies a grammar file path.

Name

[grammars]

Settings

Name	Description
"grammar-name"	Maps the grammar name to the grammar file path.

Example

The section *grammar* specifies two grammars, which are supposed to be implicitly used with all the MRCP sessions.

```
[grammars]
yes-no = /usr/local/unimrcp/data/yes-no.xml
exit = /usr/local/unimrcp/data/exit.xml
```

MRCPv2 Properties

The section *mrcpv2-properties* specifies a set of header fields to be used for all the MRCPv2 sessions established from the client to the server. The applicable header fields are defined in the section

Recognizer Header Fields.

Name

[mrcpv2-properties]

Settings

Name	Description
"header-field-name"	Maps the header field name to the header field value.

Example

The section *mrcpv2-properties* specifies the recognition and no-input timeouts.

```
[mrcpv2-properties]
Recognition-Timeout = 20000
No-Input-Timeout = 15000
```

MRCPv1 Properties

The section *mrcpv1-properties* specifies a set of header fields to be used for all the MRCPv1 sessions established from the client to the server. The applicable header fields are defined in the section Recognizer Header Fields.

Name

[mrcpv1-properties]

Settings

Name	Description
"header-field-name"	Maps the header field name to the header field value.

Example

The section *mrcpv1-properties* specifies the recognition and no-input timeouts.

```
[mrcpv1-properties]
Recognition-Timeout = 20000
```

```
No-Input-Timeout = 15000
```

2.3 Dialplan Applications

SpeechCreate

Synopsis

Creates a recognition channel.

Syntax

SpeechCreate (Engine Name)

Input parameters

- Engine Name - The engine name is *unimrcp*. If not specified, default engine will be used

SpeechLoadGrammar

Synopsis

Loads grammar on a channel.

Syntax

SpeechLoadGrammar (Grammar Name | [Content-Type:] Path)

Input parameters

- Grammar Name - The name of the grammar to load
- Path - The path to grammar file

SpeechLoadGrammar (grammar, path) - Loads a grammar from the specified location. Grammar type will be implicitly determined.
SpeechLoadGrammar (grammar, content-type:path) - Loads a grammar from the specified location. Grammar type is explicitly specified.
SpeechLoadGrammar (grammar, builtin:grammar/digits) - Specifies a built-in grammar.

SpeechUnloadGrammar

Synopsis

Unloads grammar on a channel.

Syntax

SpeechUnloadGrammar (Grammar Name)

Input parameters

- Grammar Name - The name of the grammar to unload

SpeechActivateGrammar

Synopsis
Activates the specified grammar on a channel.

Syntax
SpeechActivateGrammar (Grammar Name)

Input parameters

- Grammar Name - The name of the grammar to activate

SpeechStart

Synopsis
Starts speech recognition on a channel.

Syntax
SpeechStart ()

SpeechBackground

Synopsis
Plays a file and waits for input.

Syntax
SpeechBackground (Sound File | Timeout)

Input parameters

- Sound File - The sound file to play
- Timeout - Timeout to wait

Output results

- ${SPEECH(status)}
- ${SPEECH(spoke)}
- ${SPEECH(results)}

- ${SPEECH_SCORE(result number)}
- ${SPEECH_TEXT(result number)}
- ${SPEECH_GRAMMAR(result number)}

SpeechDeactivateGrammar

Synopsis
Deactivates the specified grammar on a channel.

Syntax
SpeechDeactivateGrammar (Grammar Name)

Input parameters

- Grammar Name - The name of the grammar to deactivate

SpeechProcessingSound

Synopsis
Changes the file that SpeechBackground is playing.

Syntax
SpeechProcessingSound (Sound File)

Input parameters

- Sound File - The sound file to play

SpeechDestroy

Synopsis
Destroys channel.

Syntax
SpeechDestroy ()

3 Suite of UniMRCP Applications

3.1 Overview

The module *app_unimrcp.so* is a suite of speech recognition and synthesis applications for Asterisk.

3.2 Configuration

The module *app_unimrcp.so* uses the Asterisk native configuration format, where a configuration file is broken into various sections with the section name surrounded by square brackets. The configuration file *mrcp.conf* is located in */etc/asterisk* by default and consists of general settings and MRCP profiles.

General Settings

The section *general* contains global settings of the module.

Name

[general]

Settings

Name	Description
default-asr-profile	Specifies the default profile to be used for speech recognition. More than one profile can be defined in the same configuration file.
default-tts-profile	Specifies the default profile to be used for speech synthesis. More than one profile can be defined in the same configuration file.
log-level	Specifies the log level, one of: • EMERGENCY • ALERT • CRITICAL • ERROR • WARNING • NOTICE • INFO • DEBUG

max-connection-count	Specifies the maximum number of TCP/MRCPv2 connections.
offer-new-connection	Specifies whether to offer the server to establish a new connection or re-use the existing one.
rx-buffer-size	The size of the buffer which temporarily holds the data received from the network socket.
tx-buffer-size	The size of the buffer which temporarily holds the data to be sent via the network socket.
request-timeout	Specifies the network timeout used for outgoing MRCPv2 and RTSP (MRCPv1) requests.

Example

The section *general* specifies commonly used parameters.

```
[general]
default-asr-profile = unimrcpv2
default-tts-profile = unimrcpv2
log-level = DEBUG
max-connection-count = 100
offer-new-connection = 1
```

MRCP Profiles

One or more profiles can be specified. Each section in the configuration corresponds to a particular profile, where the section name specifies the name of the profile and the settings are profile parameters.

Name

["unique-profile-name"]

Settings

Name	Description
version	Specifies the MRCP version: 1 or 2.
server-ip	Specifies the IP address of the server.
server-port	Specifies the SIP (MRCPv2) or RTSP (MRCPv1) port on the

	server.
client-ip [MRCPv2 only]	Specifies the IP address of the client (Asterisk) to be used for SIP signaling.
client-port [MRCPv2 only]	Specifies the local SIP port of the client.
sip-transport [MRCPv2 only]	Specifies the SIP transport: UDP, TCP
resource-location [MRCPv1 only]	Specifies the MRCP resource location on the server.
speechsynth [MRCPv1 only]	Specifies the name of the speech synthesizer resource.
speechrecog [MRCPv1 only]	Specifies the name of the speech recognizer resource.
rtp-ip	Specifies the client (Asterisk) IP address to be used for RTP streaming.
rtp-port-min	Specifies the first (inclusive) port number in the RTP port range.
rtp-port-max	Specifies the last (exclusive) port number in the RTP range.
playout-delay	Specifies the initial playout delay in the jitter buffer.
min-playout-delay	Specifies the min playout delay in the jitter buffer.
max-playout-delay	Specifies the max playout delay in the jitter buffer.
ptime	Specifies the RTP packetization time.
codecs	Specifies the RTP supported codecs.
rtcp	Specifies whether RTCP is enabled or not.

Example 1

This is a typical MRCPv1 profile, where the client (Asterisk) IP address is 192.168.0.29 and the server IP address is 192.168.0.30.

```
[unimrcpv1]
; MRCP settings
version = 1
;
; RTSP settings
server-ip = 192.168.0.30
server-port = 1554
```

```
resource-location = media
speechsynth = speechsynthesizer
speechrecog = speechrecognizer;
;
; RTP factory
rtp-ip = 192.168.0.29
rtp-port-min = 28000
rtp-port-max = 29000
;
; Jitter buffer settings
playout-delay = 50
max-playout-delay = 200
; RTP settings
ptime = 20
codecs = PCMU PCMA L16/96/8000 telephone-event/101/8000
; RTCP settings
rtcp = 0
```

Example 2

This is a typical MRCPv2 profile, where the client (Asterisk) IP address is 192.168.0.29 and the server IP address is 192.168.0.30.

```
[unimrcpv2]
; MRCP settings
version = 2
;
; SIP settings
server-ip = 192.168.0.30
server-port = 8060
; SIP user agent
client-ip = 192.168.0.29
client-port = 25097
sip-transport = udp
;
; RTP factory
rtp-ip = 192.168.0.29
rtp-port-min = 28000
rtp-port-max = 29000
;
; Jitter buffer settings
playout-delay = 50
max-playout-delay = 200
```

```
; RTP settings
ptime = 20
codecs = PCMU PCMA L16/96/8000 telephone-event/101/8000
; RTCP settings
rtcp = 0
```

3.3 Dialplan Applications

SynthAndRecog

Synopsis

Plays a synthesized prompt and waits for speech to be recognized.

Syntax

SynthAndRecog (prompt, grammar, options)

Description

This application establishes two MRCP sessions: one for speech synthesis and the other for speech recognition. Once the user starts speaking (barge-in occurred), the synthesis session is stopped, and the recognition engine starts processing the input. Once recognition completes, the application exits and returns results to the Asterisk dialplan.

Input parameters

- prompt [required]

  A prompt specified as a plain text, an SSML content, or by means of a file or URI reference.

- grammar [required]

  An inline or URI grammar to be used for recognition.

- options

  Additional parameters such as an MRCP profile, synthesizer and recognizer header fields to be used.

Name	Description
p	profile to use in mrcp.conf
t	recognition timeout (msec)
b	barge-in value (no barge-in allowed=0, barge-in allowed=1)
gd	grammar delimiters
ct	confidence threshold (0.0 - 1.0)
sl	sensitivity level (0.0 - 1.0)
sva	speed versus accuracy (0.0 - 1.0)
nb	n-best list length
nit	no input timeout (msec)

sct	speech complete timeout (msec)
sint	speech incomplete timeout (msec)
dit	DTMF interdigit timeout (msec)
dtt	DTMF terminate timeout (msec)
dttc	DTMF terminate characters
sw	save waveform (true/false)
nac	new audio channel (true/false)
spl	speech language (en-US/en-GB/etc.)
rm	recognition mode (normal/hotword)
hmaxd	hotword max duration (msec)
hmind	hotword min duration (msec)
cdb	clear DTMF buffer (true/false)
enm	early no match (true/false)
iwu	input waveform URI
mt	media type
pv	prosody volume (silent/x-soft/soft/medium/load/x-loud/default)
pr	prosody rate (x-slow/slow/medium/fast/x-fast/default)
vn	voice name to use (e.g. "Daniel", "Karin", etc.)
vg	voice gender to use (e.g. "male", "female")
vv	voice variant
a	voice age to use
uer	URI-encoded results (1: URI-encode NLMSL results, 0: do not encode)
od	Output (prompt) delimiters.
sit	Start input timers value (0: no, 1: yes [start with RECOGNIZE], 2: auto [start when prompt is finished])

Return parameters

- RECOG_STATUS

  If recognition completed, the variable ${RECOG_STATUS} is set to *OK*. Otherwise, if recognition could not be started, the variable ${RECOG_STATUS} is set to *ERROR*. If the caller hung up while recognition was still in-progress, the variable ${RECOG_STATUS} is set to *INTERRUPTED*.

- RECOG_COMPLETION_CAUSE

  The variable ${RECOG_COMPLETION_CAUSE} indicates whether recognition completed successfully with a match or an error occurred.
  ("000" - success, "001" - nomatch, "002" - noinput)

- RECOG_RESULT

  If recognition completed successfully, the variable ${RECOG_RESULT} is set to an NLSML result received from the MRCP server. Alternatively, the recognition result data can be retrieved by using the following dialplan functions RECOG_CONFIDENCE(), RECOG_GRAMMAR(), RECOG_INPUT(), and RECOG_INSTANCE().

Example 1

This context demonstrates how to use the application SynthAndRecog() with a plain-text prompt and a built-in speech grammar.

```
[synthandrecog-app1]
exten => s,1,Answer
exten => s,n,SynthAndRecog(Please say a
number,builtin:grammar/number,t=5000&b=1&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 2

This context demonstrates how to use the application SynthAndRecog() with a plain-text prompt and a built-in DTMF grammar.

```
[synthandrecog-app2]
exten => s,1,Answer
exten => s,n,SynthAndRecog(Please input a
number,builtin:dtmf/number,t=5000&b=1&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 3

This context demonstrates how to use the application SynthAndRecog() with an SSML prompt and an inline SRGS XML speech grammar.

```
[synthandrecog-app3]
exten => s,1,Answer
exten => s,n,SynthAndRecog(<?xml version=\"1.0\"?><speak version=\"1.0\"
xmlns=\"http://www.w3.org/2001/10/synthesis\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-
instance\" xsi:schemaLocation=\"http://www.w3.org/2001/10/synthesis http://www.w3.org/TR/speech-
synthesis/synthesis.xsd\" xml:lang=\"en-US\">Please pick a color: red <break/> green <break/> or blue
</speak>,<?xml version=\"1.0\"?><grammar xmlns=\"http://www.w3.org/2001/06/grammar\"
xml:lang=\"en-US\" version=\"1.0\" mode=\"voice\" tag-format=\"semantics/1.0-literals\"
root=\"color\"><rule id=\"color\"><one-
of><item>red</item><item>green</item><item>blue</item></one-
of></rule></grammar>,t=5000&b=1&ct=0.7)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
```

```
exten => s,n,Hangup
```

Example 4

This context demonstrates how to use the application SynthAndRecog() with an SSML prompt and an inline SRGS XML DTMF grammar.

```
[synthandrecog-app4]

exten => s,1,Answer
exten => s,n,SynthAndRecog(<?xml version=\"1.0\"?><speak version=\"1.0\"
xmlns=\"http://www.w3.org/2001/10/synthesis\" xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-
instance\" xsi:schemaLocation=\"http://www.w3.org/2001/10/synthesis http://www.w3.org/TR/speech-
synthesis/synthesis.xsd\" xml:lang=\"en-US\">Please pick a color: for red press 1 <break/> for green
press 2 <break/> for blue press 3 </speak>,<?xml version=\"1.0\"?><grammar
xmlns=\"http://www.w3.org/2001/06/grammar\" version=\"1.0\" mode=\"dtmf\" tag-
format=\"semantics/1.0-literals\" root=\"color\"><rule id=\"color\"><one-
of><item>1<tag>red</tag></item><item>2<tag>green</tag></item><item>3<tag>blue</tag></item>
</one-of></rule></grammar>,t=5000&b=1&ct=0.7)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 5

This context demonstrates how to use the application SynthAndRecog() with a non-bargeinable prompt.

```
[synthandrecog-app5]

exten => s,1,Answer
exten => s,n,SynthAndRecog("This is a non-bargeinable prompt. When the prompt is finished, say a
number",builtin:grammar/number,t=5000&b=0&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 6

This context demonstrates how to use the application SynthAndRecog() with a prompt and a grammar referenced from files.

```
[synthandrecog-app6]

exten => s,1,Answer
```

```
exten =>
s,n,SynthAndRecog(/usr/local/unimrcp/data/speak.xml,/usr/local/unimrcp/data/grammar.xml,t=5000&
b=1&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 7

This context demonstrates how to use the application SynthAndRecog() with a grammar referenced by an HTTP URI.

```
[synthandrecog-app7]
exten => s,1,Answer
exten => s,n,SynthAndRecog(Please say a
digit,http://unimrcp.net/data/grammar.xml,t=5000&b=1&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

Example 8

This context demonstrates how to use the application SynthAndRecog() with multiple grammars specified using a comma-separated list.

```
[synthandrecog-app8]
exten => s,1,Answer
exten => s,n,SynthAndRecog(Please say a
number,"builtin:grammar/number,builtin:dtmf/number",t=5000&b=1&ct=0.7&spl=en-US)
exten => s,n,Verbose(1, ${RECOG_STATUS}, ${RECOG_COMPLETION_CAUSE},
${RECOG_RESULT})
exten => s,n,Hangup
```

MRCPRecog

Synopsis

Performs MRCP recognition.

Syntax

MRCPRecog (grammar, options)

Description

This application establishes an MRCP session for speech recognition and optionally plays a prompt file to the user. Once recognition completes, the application exits and returns results to the Asterisk dialplan.

Input parameters

- grammar [required]

  An inline or URI grammar to be used for recognition.

- options

  Additional parameters such as an MRCP profile, recognizer header fields to be used.

Name	Description
p	profile to use in mrcp.conf
i	digits to allow recognition to be interrupted with (by default DTMFs are sent to the MRCP server to recognize; otherwise, if "any" or other digits are specified, recognition will be interrupted)
f	filename on play (if empty or not specified, no file is played)
t	recognition timeout (msec)
b	barge-in value (no barge-in allowed=0, barge-in allowed=1)
gd	grammar delimiters
ct	confidence threshold (0.0 - 1.0)
sl	sensitivity level (0.0 - 1.0)
sva	speed versus accuracy (0.0 - 1.0)
nb	n-best list length
nit	no input timeout (msec)
sct	speech complete timeout (msec)
sint	speech incomplete timeout (msec)
dit	DTMF interdigit timeout (msec)
dtt	DTMF terminate timeout (msec)
dttc	DTMF terminate characters
sw	save waveform (true/false)
nac	new audio channel (true/false)
spl	speech language (en-US/en-GB/etc.)
rm	recognition mode (normal/hotword)
hmaxd	hotword max duration (msec)
hmind	hotword min duration (msec)
cdb	clear DTMF buffer (true/false)
enm	early no match (true/false)
iwu	input waveform URI
mt	media type
epe	exit on play error (1: terminate recognition on file play error, 0: continue even if file play fails)
uer	URI-encoded results (1: URI-encode NLMSL results, 0: do not encode)
od	Output (prompt) delimiters.
sit	Start input timers value (0: no, 1: yes [start with RECOGNIZE], 2: auto [start when prompt is finished])

Return parameters

- RECOGSTATUS

  If recognition completed, the variable ${RECOGSTATUS} is set to *OK*. Otherwise, if recognition could not be started, the variable ${RECOGSTATUS} is set to *ERROR*. If the caller hung up while recognition was still in-progress, the variable ${RECOGSTATUS} is set to *INTERRUPTED*.

- RECOG_COMPLETION_CAUSE

  The variable ${RECOG_COMPLETION_CAUSE} indicates whether recognition completed successfully with a match or an error occurred.
  ("000" - success, "001" - nomatch, "002" - noinput)

- RECOG_RESULT

  If recognition completed successfully, the variable ${RECOG_RESULT} is set to an NLSML result received from the MRCP server. Alternatively, the recognition result data can be retrieved by using the following dialplan functions RECOG_CONFIDENCE(), RECOG_GRAMMAR(), RECOG_INPUT(), and RECOG_INSTANCE().

Example

This context demonstrates how to use the application MRCPRecog() with a pre-recorded prompt file and a built-in speech grammar.

```
[mrcprecog-app1]
exten => s,1,Answer
exten => s,n,MRCPRecog(builtin:grammar/number,p=default&t=5000&b=1&ct=0.7&spl=en-US&f=hello-world)
exten => s,n,Verbose(1, ${RECOGSTATUS}, ${RECOG_COMPLETION_CAUSE}, ${RECOG_RESULT})
exten => s,n,Hangup
```

MRCPSynth

Synopsis

Performs MRCP synthesis.

Syntax

MRCPSynth (prompt, grammar, options)

Description

This application establishes an MRCP session for speech synthesis.

Input parameters

- prompt [required]

  A prompt specified as a plain text, an SSML content, or by means of a file or URI reference.

- options

  Additional parameters such as an MRCP profile, synthesizer header fields to be used.

Name	Description
p	profile to use in mrcp.conf
i	digits to allow the TTS to be interrupted with
f	filename on disk to store audio to (audio not stored if not specified or empty)
l	language to use (e.g. "en-GB", "en-US", "en-AU", etc.)
v	voice name to use (e.g. "Daniel", "Karin", etc.)
g	voice gender to use (e.g. "male", "female")
a	voice age to use
pv	prosody volume (silent/x-soft/soft/medium/load/x-loud/default)
pr	prosody rate (x-slow/slow/medium/fast/x-fast/default)
ll	load lexicon (true/false)
vv	voice variant

Return parameters

- SYNTHSTATUS

  If synthesis completed successfully, the variable ${SYNTHSTATUS} is set to *OK*; otherwise, if an error occurred, the variable ${SYNTHSTATUS} is set to *ERROR*. If the caller hung up while the synthesis was in-progress, the variable ${SYNTHSTATUS} is set to *INTERRUPTED*.

Example

This context demonstrates how to use the application MRCPSynth() with a plain-text prompt.

```
[mrcpsynth-app1]
exten => s,1,Answer
exten => s,n,MRCPSynth(Hello world!,p=default)
exten => s,n,Verbose(1, ${SYNTHSTATUS})
exten => s,n,Hangup
```

3.4 Dialplan Functions

RECOG_CONFIDENCE

Synopsis

Get the confidence score of an interpretation.

Syntax

RECOG_CONFIDENCE (nbest_number)

Description

This function returns the confidence score of the specified interpretation.

Input parameters

- nbest_number

  The parameter *nbest_number* specifies the index in the list of interpretations sorted best-first. This parameter defaults to 0, if not specified.

RECOG_GRAMMAR

Synopsis

Gets the matched grammar of an interpretation.

Syntax

RECOG_GRAMMAR (nbest_number)

Description

This function returns the matched grammar of the specified interpretation.

Input parameters

- nbest_number

  The parameter *nbest_number* specifies the index in the list of interpretations sorted best-first. This parameter defaults to 0, if not specified.

RECOG_INPUT

Synopsis

Gets the spoken input.

Syntax

RECOG_INPUT (nbest_number)

Description

This function returns the spoken input.

Input parameters

- nbest_number

  The parameter *nbest_number* specifies the index in the list of interpretations sorted best-first. This parameter defaults to 0, if not specified.

RECOG_INSTANCE

Synopsis

Gets the interpreted instance.

Syntax

RECOG_INSTANCE (nbest_number / instance_number)

Description

This function returns the interpreted instance.

Input parameters

- nbest_number

  The parameter *nbest_number* specifies the index in the list of interpretations sorted best-first. This parameter defaults to 0, if not specified.

- instance_number

  The parameter *instance_number* specifies the index in the list of instances for a particular interpretation. This parameter defaults to 0, if not specified.