Usage Guide

Revision: 7
Created: July 4, 2018
Last updated: September 4, 2020
Author: Arsen Chaloyan
# Table of Contents

1 Overview ........................................................................................................................................... 4  
  1.1 Installation .................................................................................................................................. 4  
  1.2 Applicable Versions ...................................................................................................................... 4  
2 Supported Features .......................................................................................................................... 5  
  2.1 MRCP Methods ............................................................................................................................ 5  
  2.2 MRCP Events ............................................................................................................................... 5  
  2.3 MRCP Header Fields ..................................................................................................................... 5  
  2.4 Grammars ..................................................................................................................................... 6  
  2.5 Results ......................................................................................................................................... 6  
3 Configuration Format ....................................................................................................................... 7  
  3.1 Document .................................................................................................................................... 7  
  3.2 Streaming Recognition ................................................................................................................ 8  
  3.3 Speech Contexts ............................................................................................................................ 11  
  3.4 Speech Context ............................................................................................................................. 11  
  3.5 Phrase .......................................................................................................................................... 12  
  3.6 Speech and DTMF Input Detector .................................................................................................. 13  
  3.7 Utterance Manager ....................................................................................................................... 15  
  3.8 RDR Manager .............................................................................................................................. 16  
  3.9 Monitoring Agent ......................................................................................................................... 17  
  3.10 Usage Change Handler ............................................................................................................... 18  
  3.11 Usage Refresh Handler ............................................................................................................... 18  
  3.12 License Server ........................................................................................................................... 19  
4 Configuration Steps ......................................................................................................................... 21  
  4.1 Using Default Configuration ........................................................................................................ 21  
  4.2 Specifying Recognition Language .............................................................................................. 21  
  4.3 Specifying Sampling Rate ............................................................................................................ 21  
  4.4 Specifying Speech Input Parameters ............................................................................................ 21  
  4.5 Specifying DTMF Input Parameters ............................................................................................. 22  
  4.6 Specifying No-Input and Recognition Timeouts ......................................................................... 22  
  4.7 Specifying Vendor-Specific Parameters ...................................................................................... 23  
  4.8 Maintaining Utterances ............................................................................................................... 25  
  4.9 Maintaining Recognition Details Records .................................................................................... 25  
5 Recognition Grammars and Results ............................................................................................... 27  
  5.1 Using Built-in Speech Transcription ............................................................................................ 27  
  5.2 Using Built-in DTMF Grammars .................................................................................................. 28  
  5.3 Retrieving Results ....................................................................................................................... 28
6 Monitoring Usage Details ........................................................................................................29
   6.1 Log Usage .........................................................................................................................29
   6.2 Update Usage ...................................................................................................................29
   6.3 Dump Channels ...............................................................................................................30
7 Usage Examples ....................................................................................................................31
   7.1 Speech Transcription ........................................................................................................31
   7.2 DTMF Recognition .........................................................................................................32
   7.3 Speech and DTMF Recognition ......................................................................................33
8 Sequence Diagram .............................................................................................................35
9 References ..........................................................................................................................36
   9.1 IBM Watson ....................................................................................................................36
   9.2 Specifications ..................................................................................................................36
1 Overview

This guide describes how to configure and use the IBM Watson Speech Recognition (SR) plugin to the UniMRCP server. The document is intended for users having a certain knowledge of IBM Watson Speech-to-Text API and UniMRCP.

1.1 Installation

For installation instructions, use one of the guides below.

- 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 1.5.0 and above
- UniMRCP Watson SR Plugin 1.0.0 and above
2 Supported Features

This is a brief check list of the features currently supported by the UniMRCP server running with the Watson SR plugin.

2.1 MRCP Methods

✓ DEFINE-GRAMMAR
✓ RECOGNIZE
✓ START-INPUT-TIMERS
✓ STOP
✓ SET-PARAMS
✓ GET-PARAMS

2.2 MRCP Events

✓ RECOGNITION-COMPLETE
✓ START-OF-INPUT

2.3 MRCP Header Fields

✓ Input-Type
✓ No-Input-Timeout
✓ Recognition-Timeout
✓ Speech-Complete-Timeout
✓ Speech-Incomplete-Timeout
✓ Waveform-URI
✓ Media-Type
✓ Completion-Cause
✓ Confidence-Threshold
✓ Start-Input-Timers
✓ DTMF-Interdigit-Timeout
✓ DTMF-Term-Timeout
✓ DTMF-Term-Char
✓ Save-Waveform
✓ Speech-Language
✓ Cancel-If-Queue
2.4 Grammars

✓ Built-in speech transcription grammar
✓ Built-in/embedded DTMF grammar
✓ SRGS XML (limited support)

2.5 Results

✓ NLSML
3 Configuration Format

The configuration file of the Watson SR plugin is located in /opt/unimrcp/conf/umswatsonsr.xml. The configuration file is written in XML.

3.1 Document

The root element of the XML document must be <umswatsonsr>.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>license-file</td>
<td>File path</td>
<td>Specifies the license file. File name may include patterns containing '*' sign. If multiple files match the pattern, the most recent one gets used.</td>
</tr>
<tr>
<td>credentials-file</td>
<td>File path</td>
<td>Specifies the IBM Watson credentials file to use. File name may include patterns containing '*' sign. If multiple files match the pattern, the most recent one gets used.</td>
</tr>
</tbody>
</table>

Parent

None.

Children

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ws-streaming-recognition&gt;</td>
<td>String</td>
<td>Specifies parameters of streaming recognition employed via the WebSocket protocol.</td>
</tr>
<tr>
<td>&lt;speech-contexts&gt;</td>
<td>String</td>
<td>Contains a list of speech contexts. Available since WSR 1.2.0.</td>
</tr>
<tr>
<td>&lt;speech-dtmf-input-detector&gt;</td>
<td>String</td>
<td>Specifies parameters of the speech and DTMF input detector.</td>
</tr>
<tr>
<td>&lt;utterance-manager&gt;</td>
<td>String</td>
<td>Specifies parameters of the utterance manager.</td>
</tr>
<tr>
<td>&lt;rdr-manager&gt;</td>
<td>String</td>
<td>Specifies parameters of the Recognition Details Record (RDR) manager.</td>
</tr>
<tr>
<td>&lt;monitoring-agent&gt;</td>
<td>String</td>
<td>Specifies parameters of the monitoring manager.</td>
</tr>
</tbody>
</table>
<license-server> String Specifies parameters used to connect to the license server. The use of the license server is optional.

Example
This is an example of a bare document.

<umswatsons license-file="umswatsons_*.lic" credentials-file="watsonr.credentials">
</umswatsons>

3.2 Streaming Recognition
This element specifies parameters of streaming recognition employed via the WebSocket interface.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>language</td>
<td>String</td>
<td>Specifies the default language to use, if not set by the client. For a list of supported languages, visit <a href="https://console.bluemix.net/docs/services/speech-to-text/input.html#models">https://console.bluemix.net/docs/services/speech-to-text/input.html#models</a></td>
</tr>
<tr>
<td>language-customization-id</td>
<td>String</td>
<td>Specifies the custom language model id to be used for recognition, if any. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>acoustic-customization-id</td>
<td>String</td>
<td>Specifies the custom acoustic model id to be used for recognition, if any. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>base-model-version</td>
<td>String</td>
<td>Specifies the base model version to be used for recognition, if any. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>grammar-name</td>
<td>String</td>
<td>Specifies the grammar name used for speech recognition in conjunction with the customization id of the custom language model for which the grammar is defined. Can be overridden by client. Available since WSR 1.6.0.</td>
</tr>
<tr>
<td>customization-weight</td>
<td>Double</td>
<td>Specifies the weight to give to words from the custom language model compared to those from the base model, if any. Can be overridden by client. Available since WSR</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>max-alternatives</td>
<td>Integer</td>
<td>Specifies the maximum number of speech recognition result alternatives to be returned. Can be overridden by client by means of the header field N-Best-List-Length.</td>
</tr>
<tr>
<td>alternatives-below-threshold</td>
<td>Boolean</td>
<td>Specifies whether to return speech recognition result alternatives with the confidence score below the confidence threshold. Can be overridden by client. Available since WSR 1.2.0.</td>
</tr>
<tr>
<td>confidence-format</td>
<td>String</td>
<td>Specifies the format of the confidence score to be returned (use &quot;auto&quot; for a format based on protocol version, &quot;mrcpv2&quot; for a float value in the range of 0..1, &quot;mrcpv1&quot; for an integer value in the range of 0..100). Available since WSR 1.2.0.</td>
</tr>
<tr>
<td>smart-formatting</td>
<td>Boolean</td>
<td>Specifies whether the service converts dates, times, numbers, currency, and similar values into more conventional representations in the final transcript.</td>
</tr>
<tr>
<td>word-confidence</td>
<td>Boolean</td>
<td>Specifies whether the service returns a confidence for each word. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>timestamps</td>
<td>Boolean</td>
<td>Specifies whether the service returns time alignment for each word. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>speaker-labels</td>
<td>Boolean</td>
<td>Specifies whether the service returns labels identifying a speaker for each word. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>redaction</td>
<td>Boolean</td>
<td>Specifies whether the service returns masked numeric data having three or more consecutive digits. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>processing-metrics</td>
<td>Boolean</td>
<td>Specifies whether the service returns metrics on transcription of the input audio data. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>processing-metrics-interval</td>
<td>Time interval [sec]</td>
<td>Specifies an interval in seconds at which the service returns processing metrics. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>audio-metrics</td>
<td>Boolean</td>
<td>Specifies whether the service returns signal characteristics of the input audio data. Can be overridden by client. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>end-of-phrase-</td>
<td>Time interval</td>
<td>Specifies an interval between 0.0 and 12.0 seconds at which the service splits a transcript into multiple final results if it</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>start-of-input</td>
<td>String</td>
<td>Specifies the source of start of input event sent to the client (use &quot;service-originated&quot; to rely on service-originated first interim result and &quot;internal&quot; for plugin-originated event).</td>
</tr>
<tr>
<td>skip-unsupported-grammars</td>
<td>Boolean</td>
<td>Specifies whether to skip or raise an error while referencing a malformed or not supported grammar.</td>
</tr>
<tr>
<td>transcription-grammar</td>
<td>String</td>
<td>Specifies the name of the built-in speech transcription grammar. The grammar can be referenced as <code>builtin:speech/transcribe</code> or <code>builtin:grammar/transcribe</code>, where <code>transcribe</code> is the default value of this parameter.</td>
</tr>
<tr>
<td>auth-validation-period</td>
<td>Time interval</td>
<td>Specifies a period in seconds used to re-validate access token based on subscription key. Since WSR 1.4.0, if set to 0, the period is based on <code>expires_in</code> received in response from service.</td>
</tr>
<tr>
<td>http-proxy</td>
<td>String</td>
<td>Specifies the URI of HTTP proxy, if used. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>inter-result-timeout</td>
<td>Time interval</td>
<td>Specifies a timeout between interim results containing transcribed speech. If the timeout is elapsed, input is considered complete. The timeout defaults to 0 (disabled). Available since WSR 1.6.0.</td>
</tr>
</tbody>
</table>

**Parent**

`<umswatsonsr>`

**Children**

None.

**Example**

This is an example of streaming recognition element.

```xml
<ws-streaming-recognition
  language="en-US"
  max-alternatives="1"
  alternatives-below-threshold="false"
  confidence-format="auto"
  smart-formatting="true"
  start-of-input="internal"
  skip-unsupported-grammars="true"
  transcription-grammar="transcribe"
  auth-validation-period="0"
```
3.3 Speech Contexts

This element specifies a list of speech contexts.

Availability

>= WSR 1.2.0.

Attributes

None.

Parent

<umswatsonsr>

Children

<speech-context>

Example

The example below defines a speech contexts directory.

```
<speech-contexts>
  <speech-context id="directory" speech-complete="true" enable="true">
    <phrase>call Steve</phrase>
    <phrase>call John</phrase>
    <phrase>dial 5</phrase>
    <phrase>dial 6</phrase>
  </speech-context>
</speech-contexts>
```

3.4 Speech Context

This element specifies a speech context.

Availability

>= WSR 1.2.0.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
</table>
| id   | String| Specifies a unique string identifier of the speech context to
### Parent

```xml
<speech-contexts>
</speech-contexts>
```

### Children

```xml
<phrase>
</phrase>
```

### Example

This is an example of speech context element.

```xml
<speech-context id="directory" speech-complete="true" enable="true">
    <phrase>call Steve</phrase>
    <phrase>call John</phrase>
    <phrase>dial 5</phrase>
    <phrase>dial 6</phrase>
</speech-context>
```

### 3.5 Phrase

This element specifies a phrase in the speech context.

### Availability

```
>= WSR 1.2.0.
```

### Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag</td>
<td>String</td>
<td>Specifies an optional arbitrary string identifier to be returned as an instance in the NLSML result, if the</td>
</tr>
</tbody>
</table>
transcription result matches the phrase.

Parent

<speech-context>

Children

None.

This is an example of a speech context with phrases having tags specified. Available since GSR 1.9.0.

```xml
<speech-context id="boolean" speech-complete="true" scope="strict" enable="true">
  <phrase tag="true">yes</phrase>
  <phrase tag="true">sure</phrase>
  <phrase tag="true">correct</phrase>
  <phrase tag="false">no</phrase>
  <phrase tag="false">not sure</phrase>
  <phrase tag="false">incorrect</phrase>
</speech-context>
```

3.6 Speech and DTMF Input Detector

This element specifies parameters of the speech and DTMF input detector.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vad-mode</td>
<td>Integer</td>
<td>Specifies an operating mode of VAD in the range of [0 ... 3]. Default is 1.</td>
</tr>
<tr>
<td>speech-start-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait in transition mode before triggering a start of speech input event.</td>
</tr>
<tr>
<td>speech-complete-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait in transition mode before triggering an end of speech input event. The complete timeout is used when there is an interim result available.</td>
</tr>
<tr>
<td>speech-incomplete-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait in transition mode before triggering an end of speech input event. The incomplete timeout is used as long as there is no interim result available. Afterwards, the complete timeout</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------</td>
<td>-------------------------------------------------------------------</td>
</tr>
<tr>
<td>noinput-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait before triggering a no-input event.</td>
</tr>
<tr>
<td>input-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies how long to wait for input to complete.</td>
</tr>
<tr>
<td>dtmf-interdigit-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies a DTMF inter-digit timeout.</td>
</tr>
<tr>
<td>dtmf-term-timeout</td>
<td>Time interval [msec]</td>
<td>Specifies a DTMF input termination timeout.</td>
</tr>
<tr>
<td>dtmf-term-char</td>
<td>Character</td>
<td>Specifies a DTMF input termination character.</td>
</tr>
<tr>
<td>speech-leading-silence</td>
<td>Time interval [msec]</td>
<td>Specifies desired silence interval preceding spoken input.</td>
</tr>
<tr>
<td>speech-trailing-silence</td>
<td>Time interval [msec]</td>
<td>Specifies desired silence interval following spoken input.</td>
</tr>
<tr>
<td>speech-output-period</td>
<td>Time interval [msec]</td>
<td>Specifies an interval used to send speech frames to the recognizer.</td>
</tr>
</tbody>
</table>

**Parent**

 `<umswatsonsr>`

**Children**

 None.

**Example**

The example below defines a typical speech and DTMF input detector having the default parameters set.

```xml
<speech-dtmf-input-detector
  vad-mode="2"
  speech-start-timeout="300"
  speech-complete-timeout="1000"
  speech-incomplete-timeout="1000"
  noinput-timeout="5000"
  input-timeout="10000"
  dtmf-interdigit-timeout="5000"
  dtmf-term-timeout="10000"
  dtmf-term-char=""
  speech-leading-silence="300"
  speech-trailing-silence="300"
  speech-output-period="200"
```
3.7 Utterance Manager

This element specifies parameters of the utterance manager.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>save-waveforms</td>
<td>Boolean</td>
<td>Specifies whether to save waveforms or not.</td>
</tr>
<tr>
<td>purge-existing</td>
<td>Boolean</td>
<td>Specifies whether to delete existing records on start-up.</td>
</tr>
<tr>
<td>max-file-age</td>
<td>Time interval [min]</td>
<td>Specifies a time interval in minutes after expiration of which a waveform is deleted. Set 0 for infinite.</td>
</tr>
<tr>
<td>max-file-count</td>
<td>Integer</td>
<td>Specifies the max number of waveforms to store. If reached, the oldest waveform is deleted. Set 0 for infinite.</td>
</tr>
<tr>
<td>waveform-base-uri</td>
<td>String</td>
<td>Specifies the base URI used to compose an absolute waveform URI.</td>
</tr>
<tr>
<td>waveform-folder</td>
<td>Dir path</td>
<td>Specifies a folder the waveforms should be stored in.</td>
</tr>
<tr>
<td>file-prefix</td>
<td>String</td>
<td>Specifies a prefix used to compose the name of the file to be stored. Defaults to 'umswatsonsr'-, if not specified.</td>
</tr>
<tr>
<td>use-logging-tag</td>
<td>Boolean</td>
<td>Specifies whether to use the MRCP header field Logging-Tag, if present, to compose the name of the file to be stored. Available since WSR 1.6.0.</td>
</tr>
</tbody>
</table>

Parent

<umswatsonsr>

Children

None.

Example
The example below defines a typical utterance manager having the default parameters set.

```xml
<utterance-manager
    save-waveforms="false"
    purge-existing="false"
    max-file-age="60"
    max-file-count="100"
    waveform-base-uri="http://localhost/utterances/"
    waveform-folder=""
/>
```

### 3.8 RDR Manager

This element specifies parameters of the Recognition Details Record (RDR) manager.

**Attributes**

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>save-records</td>
<td>Boolean</td>
<td>Specifies whether to save recognition details records or not.</td>
</tr>
<tr>
<td>purge-existing</td>
<td>Boolean</td>
<td>Specifies whether to delete existing records on start-up.</td>
</tr>
<tr>
<td>max-file-age</td>
<td>Time interval [min]</td>
<td>Specifies a time interval in minutes after expiration of which a record is deleted. Set 0 for infinite.</td>
</tr>
<tr>
<td>max-file-count</td>
<td>Integer</td>
<td>Specifies the max number of records to store. If reached, the oldest record is deleted. Set 0 for infinite.</td>
</tr>
<tr>
<td>record-folder</td>
<td>Dir path</td>
<td>Specifies a folder to store recognition details records in. Defaults to <code>${UniMRCPInstallDir}/var</code>.</td>
</tr>
<tr>
<td>file-prefix</td>
<td>String</td>
<td>Specifies a prefix used to compose the name of the file to be stored. Defaults to 'umswatsonsr-', if not specified.</td>
</tr>
<tr>
<td>use-logging-tag</td>
<td>Boolean</td>
<td>Specifies whether to use the MRCP header field Logging-Tag, if present, to compose the name of the file to be stored. Available since WSR 1.6.0.</td>
</tr>
</tbody>
</table>
Parent

<umswatsonsr>

Children

None.

Example

The example below defines a typical utterance manager having the default parameters set.

```xml
<rdr-manager
    save-records="false"
    purge-existing="false"
    max-file-age="60"
    max-file-count="100"
    waveform-folder=""
/>
```

3.9 Monitoring Agent

This element specifies parameters of the monitoring agent.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>refresh-period</td>
<td>Time interval [sec]</td>
<td>Specifies a time interval in seconds used to periodically refresh usage details. See &lt;usage-refresh-handler&gt;.</td>
</tr>
</tbody>
</table>

Parent

<umswatsonsr>

Children

<usage-change-handler>
<usage-refresh-handler>

Example

The example below defines a monitoring agent with usage change and refresh handlers.

```xml
<monitoring-agent refresh-period="60">
    <usage-change-handler>
</monitoring-agent>
```
3.10 Usage Change Handler

This element specifies an event handler called on every usage change.

Attributes
None.

Parent
<monitoring-agent>

Children
<log-usage>
$update-usage>
<dump-channels>

Example
This is an example of the usage change event handler.

3.11 Usage Refresh Handler

This element specifies an event handler called periodically to update usage details.

Attributes
None.

Parent
<monitoring-agent>
Children

<log-usage>
<update-usage>
<dump-channels>

Example

This is an example of the usage change event handler.

```xml
<usage-refresh-handler>
  <log-usage enable="true" priority="NOTICE"/>
  <update-usage enable="false" status-file="umswatsonsr-usage.status"/>
  <dump-channels enable="false" status-file="umswatsonsr-channels.status"/>
</usage-refresh-handler>
```

3.12 License Server

This element specifies parameters used to connect to the license server.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>enable</td>
<td>Boolean</td>
<td>Specifies whether the use of license server is enabled or not. If enabled, the license-file attribute is not honored.</td>
</tr>
<tr>
<td>server-address</td>
<td>String</td>
<td>Specifies the IP address or host name of the license server.</td>
</tr>
<tr>
<td>certificate-file</td>
<td>File path</td>
<td>Specifies the client certificate used to connect to the license server. File name may include patterns containing a '*' sign. If multiple files match the pattern, the most recent one gets used.</td>
</tr>
<tr>
<td>ca-file</td>
<td>File path</td>
<td>Specifies the certificate authority used to validate the license server.</td>
</tr>
<tr>
<td>channel-count</td>
<td>Integer</td>
<td>Specifies the number of channels to check out from the license server. If not specified or set to 0, either all available channels or a pool of channels will be checked based on the configuration of the license server.</td>
</tr>
<tr>
<td>http-proxy-address</td>
<td>String</td>
<td>Specifies the IP address or host name of the</td>
</tr>
<tr>
<td>http-proxy-port</td>
<td>Integer</td>
<td>Specifies the port number of the HTTP proxy server, if used. Available since WSR 1.6.0.</td>
</tr>
</tbody>
</table>

**Parent**

<ums watson sr>

**Children**

None.

**Example**

The example below defines a typical configuration which can be used to connect to a license server located, for example, at 10.0.0.1.

```xml
<license-server
  enable="true"
  server-address="10.0.0.1"
  certificate-file="unilic_client_*.crt"
  ca-file="unilic_ca.crt"
/>```

For further reference to the license server, visit

```
http://unimrcp.org/licserver```

4 Configuration Steps

This section outlines common configuration steps.

4.1 Using Default Configuration

The default configuration should be sufficient for the general use.

4.2 Specifying Recognition Language

Recognition language can be specified by the client per MRCP session by means of the header field \textit{Speech-Language} set in a \textit{SET-PARAMS} or \textit{RECOGNIZE} request. Otherwise, the parameter \textit{language} set in the configuration file \textit{ums watsonsr.xml} is used. The parameter defaults to \textit{en-US}.

For supported languages and their corresponding codes, visit the following link.

https://console.bluemix.net/docs/services/speech-to-text/input.html#models

Since WSR 1.4.0, the recognition language can also be set by the attribute \textit{xml:lang} specified in the SRGS grammar. For example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<grammar mode="voice" root="transcribe" version="1.0"
    xml:lang="en-AU"
    xmlns="http://www.w3.org/2001/06/grammar">
    <meta name="scope" content="builtin"/>
    <rule id="transcribe"><one-of/></rule>
</grammar>
```

4.3 Specifying Sampling Rate

Sampling rate is determined based on the SDP negotiation. Refer to the configuration guide of the UniMRCP server on how to specify supported encodings and sampling rates to be used in communication between the client and server. Either 8 or 16 kHz can be used by IBM Watson Speech-to-Text API.

4.4 Specifying Speech Input Parameters

While the default parameters specified for the speech input detector are sufficient for the general use, various parameters can be adjusted to better suit a particular requirement.

- \textit{speech-start-timeout}

  This parameter is used to trigger a start of speech input. The shorter is the timeout, the sooner a \textit{START-OF-INPUT} event is delivered to the client. However, a short timeout may also lead to a false
positive. Note that if the start-of-input parameter in the ws-streaming-recognition is set to service-originated, then a START-OF-INPUT event is sent to the client at a later stage, upon reception of a first interim result.

- speech-complete-timeout

This parameter is used to trigger an end of speech input. The shorter is the timeout, the shorter is the response time. However, a short timeout may also lead to a false positive.

- vad-mode

This parameter is used to specify an operating mode of the Voice Activity Detector (VAD) within an integer range of \([0 \ldots 3]\). A higher mode is more aggressive and, as a result, is more restrictive in reporting speech. The parameter can be overridden per MRCP session by setting the header field Sensitivity-Level in a SET-PARAMS or RECOGNIZE request. The following table shows how the Sensitivity-Level is mapped to the vad-mode.

<table>
<thead>
<tr>
<th>Sensitivity-Level</th>
<th>Vad-Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>([0.00 \ldots 0.25])</td>
<td>0</td>
</tr>
<tr>
<td>([0.25 \ldots 0.50])</td>
<td>1</td>
</tr>
<tr>
<td>([0.50 \ldots 0.75])</td>
<td>2</td>
</tr>
<tr>
<td>([0.75 \ldots 1.00])</td>
<td>3</td>
</tr>
</tbody>
</table>

4.5 Specifying DTMF Input Parameters

While the default parameters specified for the DTMF input detector are sufficient for the general use, various parameters can be adjusted to better suit a particular requirement.

- dtmf-interdigit-timeout

This parameter is used to set an inter-digit timeout on DTMF input. The parameter can be overridden per MRCP session by setting the header field DTMF-Interdigit-Timeout in a SET-PARAMS or RECOGNIZE request.

- dtmf-term-timeout

This parameter is used to set a termination timeout on DTMF input and is in effect when dtmf-term-char is set and there is a match for an input grammar. The parameter can be overridden per MRCP session by setting the header field DTMF-Term-Timeout in a SET-PARAMS or RECOGNIZE request.

- dtmf-term-char

This parameter is used to set a character terminating DTMF input. The parameter can be overridden per MRCP session by setting the header field DTMF-Term-Char in a SET-PARAMS or RECOGNIZE request.

4.6 Specifying No-Input and Recognition Timeouts

- noinput-timeout
This parameter is used to trigger a no-input event. The parameter can be overridden per MRCP session by setting the header field `No-Input-Timeout` in a `SET-PARAMS` or `RECOGNIZE` request.

- **input-timeout**

This parameter is used to limit input (recognition) time. The parameter can be overridden per MRCP session by setting the header field `Recognition-Timeout` in a `SET-PARAMS` or `RECOGNIZE` request.

### 4.7 Specifying Vendor-Specific Parameters

The following parameters can optionally be specified by the MRCP client in `SET-PARAMS, DEFINE-GRAMMAR` and `RECOGNIZE` requests via the MRCP header field `Vendor-Specific-Parameters`.

<table>
<thead>
<tr>
<th>Name</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>start-of-input</td>
<td>String</td>
<td>Specifies the source of start of input event sent to the client (use &quot;service-originated&quot; for an event originated based on a first-received interim result and &quot;internal&quot; for an event determined by plugin). Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>alternatives-below-threshold</td>
<td>Boolean</td>
<td>Specifies whether to return speech recognition result alternatives with the confidence score below the confidence threshold. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>speech-start-timeout</td>
<td>Time</td>
<td>Specifies how long to wait in transition mode before triggering a start of speech input event. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td></td>
<td>interval</td>
<td>[msec]</td>
</tr>
<tr>
<td>language-customization-id</td>
<td>String</td>
<td>Specifies the custom language model id to be used for recognition, if any. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>acoustic-customization-id</td>
<td>String</td>
<td>Specifies the custom acoustic model id to be used for recognition, if any. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>base-model-version</td>
<td>String</td>
<td>Specifies the base model version to be used for recognition, if any. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>grammar-name</td>
<td>String</td>
<td>Specifies the grammar name used for speech recognition in conjunction with the customization id of the custom language model for which the grammar is defined. Available since WSR 1.6.0.</td>
</tr>
<tr>
<td>customization-weight</td>
<td>Double</td>
<td>Specifies the weight to give to words from the custom language model compared to those from the base model, if any. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>smart-formatting</td>
<td>Boolean</td>
<td>Specifies whether the service converts dates, times, numbers, currency, and similar values into more conventional representations in the final transcript.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>word-confidence</td>
<td>Boolean</td>
<td>Specifies whether the service returns a confidence for each word. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>timestamps</td>
<td>Boolean</td>
<td>Specifies whether the service returns time alignment for each word. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>speaker-labels</td>
<td>Boolean</td>
<td>Specifies whether the service returns labels identifying a speaker for each word. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>redaction</td>
<td>Boolean</td>
<td>Specifies whether the service returns masked numeric data having three or more consecutive digits. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>processing-metrics</td>
<td>Boolean</td>
<td>Specifies whether the service returns metrics on transcription of the input audio data. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>processing-metrics-interval</td>
<td>Time</td>
<td>Specifies an interval in seconds at which the service returns processing metrics. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>audio-metrics</td>
<td>Boolean</td>
<td>Specifies whether the service returns signal characteristics of the input audio data. Available since WSR 1.5.0.</td>
</tr>
<tr>
<td>end-of-phrase-silence-time</td>
<td>Time</td>
<td>Specifies an interval between 0.0 and 12.0 seconds at which the service splits a transcript into multiple final results if it encounters silence. Not used, if set to -1. Available since WSR 1.6.0.</td>
</tr>
<tr>
<td>interim-result-timeout</td>
<td>Time</td>
<td>Specifies a timeout between interim results containing transcribed speech. If the timeout is elapsed, input is considered complete. Available since WSR 1.16.0.</td>
</tr>
</tbody>
</table>

All the vendor-specific parameters can also be specified at the grammar-level via a built-in or SRGS XML grammar.

The following example demonstrates the use of a built-in grammar with the vendor-specific parameters `alternatives-below-threshold` and `speech-start-timeout` set to `true` and `100` correspondingly.

```
builtin:speech/transcribe?alternatives-below-threshold=true:speech-start-timeout=100
```

The following example demonstrates the use of an SRGS XML grammar with the vendor-specific parameters `alternatives-below-threshold` and `speech-start-timeout` set to `true` and `100` correspondingly.
4.8 Maintaining Utterances

Saving of utterances is not required for regular operation and is disabled by default. However, enabling this functionality allows to save utterances sent to the Watson Speech service and later listen to them offline.

The relevant settings can be specified via the element `utterance-manager`.

- **save-waveforms**
  Utterances can optionally be recorded and stored if the configuration parameter `save-waveforms` is set to true. The parameter can be overridden per MRCP session by setting the header field `Save-Waveforms` in a SET-PARMS or RECOGNIZE request.

- **purge-existing**
  This parameter specifies whether to delete existing waveforms on start-up.

- **max-file-age**
  This parameter specifies a time interval in minutes after expiration of which a waveform is deleted. If set to 0, there is no expiration time specified.

- **max-file-count**
  This parameter specifies the maximum number of waveforms to store. If the specified number is reached, the oldest waveform is deleted. If set to 0, there is no limit specified.

- **waveform-base-uri**
  This parameter specifies the base URI used to compose an absolute waveform URI returned in the header field `Waveform-Uri` in response to a RECOGNIZE request.

- **waveform-folder**
  This parameter specifies a path to the directory used to store waveforms in. The directory defaults to `${UniMRCPInstallDir}/var`.

4.9 Maintaining Recognition Details Records
Producing of recognition details records (RDR) is not required for regular operation and is disabled by default. However, enabling this functionality allows to store details of each recognition attempt in a separate file and analyze them later offline. The RDRs are stored in the JSON format.

The relevant settings can be specified via the element `rdr-manager`.

- **save-records**
  This parameter specifies whether to save recognition details records or not.

- **purge-existing**
  This parameter specifies whether to delete existing records on start-up.

- **max-file-age**
  This parameter specifies a time interval in minutes after expiration of which a record is deleted. If set to 0, there is no expiration time specified.

- **max-file-count**
  This parameter specifies the maximum number of records to store. If the specified number is reached, the oldest record is deleted. If set to 0, there is no limit specified.

- **record-folder**
  This parameter specifies a path to the directory used to store records in. The directory defaults to `${UniMRCPInstallDir}/var`. 
5 Recognition Grammars and Results

5.1 Using Built-in Speech Transcription

For generic speech transcription, having no speech contexts defined, a pre-set identifier `transcribe` must be used by the MRCP client in a RECOGNIZE request as follows:

```
builtin:speech/transcribe
```

The name of the identifier `transcribe` can be changed from the configuration file `umswatsonsr.xml`.

Speech contexts are defined in the configuration file `umswingsr.xml` and available since BingSR 1.5.0. A speech context is assigned a unique string identifier and holds a list of phrases.

Below is a definition of a sample speech context `directory`:

```
<speech-context id="directory" speech-complete="true">
  <phrase>call Steve</phrase>
  <phrase>call John</phrase>
  <phrase>dial 5</phrase>
  <phrase>dial 6</phrase>
</speech-context>
```

Which can be referenced in a RECOGNIZE request as follows:

```
builtin:speech/directory
```

The prefixes `builtin:speech` and `builtin:grammar` can be used interchangeably as follows:

```
builtin:grammar/directory
```

Since WSR 1.4.0, a speech context can be referenced by means metadata in SRGS XML grammar. For example, the following SRGS grammar references a built-in speech context `directory`.

```
<grammar mode="voice" root="directory" version="1.0"
  xml:lang="en-US"
  xmlns="http://www.w3.org/2001/06/grammar">
  <meta name="scope" content="builtin"/>
  <rule id="directory"><one-of/></rule>
</grammar>
```
Where the root rule name identifies a speech context.

5.2 Using Built-in DTMF Grammars

Pre-set built-in DTMF grammars can be referenced by the MRCP client in a RECOGNIZE request as follows:

```
builtin:dtmf/$id
```

Where $id is a unique string identifier of the built-in DTMF grammar.

Note that only a DTMF grammar identifier digits is currently supported.

Since WSR 1.4.0, built-in DTMF digits can also be referenced by metadata in SRGS XML grammar. The following example is equivalent to the built-in grammar above.

```
<grammar mode="dtmf" root="digits" version="1.0"
    xml:lang="en-US"
    xmlns="http://www.w3.org/2001/06/grammar">
  <meta name="scope" content="builtin"/>
  <rule id="digits"><one-of/></rule>
</grammar>
```

Where the root rule name identifies a built-in DTMF grammar.

5.3 Retrieving Results

Results received from the speech service are transformed to the NLSML format.
6 Monitoring Usage Details

The number of in-use and total licensed channels can be monitored in several alternate ways. There is a set of actions which can take place on certain events. The behavior is configurable via the element *monitoring-agent*, which contains two event handlers: *usage-change-handler* and *usage-refresh-handler*.

While the *usage-change-handler* is invoked on every acquisition and release of a licensed channel, the *usage-refresh-handler* is invoked periodically on expiration of a timeout specified by the attribute *refresh-period*.

The following actions can be specified for either of the two handlers.

6.1 Log Usage

The action *log-usage* logs the following data in the order specified.

- The number of currently in-use channels.
- The maximum number of channels used concurrently. Available since WSR 1.2.0.
- The total number of licensed channels.

The following is a sample log statement, indicating 0 in-use, 0 max-used and 2 total channels.

```
[NOTICE] WSR Usage: 0/0/2
```

6.2 Update Usage

The action *update-usage* writes the following data to a status file *umswatsonsr-usage.status*, located by default in the directory ${UniMRCPInstallDir}/var/status.

- The number of currently in-use channels.
- The maximum number of channels used concurrently. Available since WSR 1.2.0.
- The total number of licensed channels.
- The current status of the license permit.

The following is a sample content of the status file.

```
in-use channels: 0
max used channels: 0
total channels: 2
total channels: 2
```
6.3 Dump Channels

The action *dump-channels* writes the identifiers of in-use channels to a status file *umswatsonsr-channels.status*, located by default in the directory `${UniMRCPInstallDir}/var/status`. 
7 Usage Examples

7.1 Speech Transcription

This examples demonstrates how to perform speech recognition by using a RECOGNIZE request.

C->S:

```
MRCP/2.0 336 RECOGNIZE 1
Channel-Identifier: 6e1a2e4e54ae11e7@speechrecog
Content-Id: request1@form-level
Content-Type: text/uri-list
Cancel-If-Queue: false
No-Input-Timeout: 5000
Recognition-Timeout: 10000
Start-Input-Timers: true
Confidence-Threshold: 0.87
Save-Waveform: true
Content-Length: 25

builtin:speech/transcribe
```

S->C:

```
MRCP/2.0 83 1 200 IN-PROGRESS
Channel-Identifier: 6e1a2e4e54ae11e7@speechrecog
```

S->C:

```
MRCP/2.0 115 START-OF-INPUT 1 IN-PROGRESS
Channel-Identifier: 6e1a2e4e54ae11e7@speechrecog
Input-Type: speech
```

S->C:

```
MRCP/2.0 498 RECOGNITION-COMPLETE 1 COMPLETE
Channel-Identifier: 6e1a2e4e54ae11e7@speechrecog
Completion-Cause: 000 success
Waveform-Uri: <http://localhost/utterances/utter-6e1a2e4e54ae11e7-1.wav>;size=20480;duration=1280
Content-Type: application/x-nlsml
```
7.2 DTMF Recognition

This examples demonstrates how to reference a built-in DTMF grammar in a RECOGNIZE request.

C->S:

```
MRCP/2.0 266 RECOGNIZE 1
Channel-Identifier: d26bef74091a174c@speechrecog
Content-Type: text/uri-list
Cancel-If-Queue: false
Start-Input-Timers: true
Confidence-Threshold: 0.7
Speech-Language: en-US
Dtmf-Term-Char: #
Content-Length: 19

builtin:dtmf/digits
```

S->C:

```
MRCP/2.0 83 1 200 IN-PROGRESS
Channel-Identifier: d26bef74091a174c@speechrecog
```

S->C:

```
MRCP/2.0 113 START-OF-INPUT 1 IN-PROGRESS
Channel-Identifier: d26bef74091a174c@speechrecog
Input-Type: dtmf
```

S->C:
7.3 Speech and DTMF Recognition

This example demonstrates how to perform recognition by activating both speech and DTMF grammars. In this example, the user is expected to input a 4-digit pin.

C->S:

MRCP/2.0 275 RECOGNIZE 1  
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog  
Content-Type: text/uri-list  
Cancel-If-Queue: false  
Start-Input-Timers: true  
Confidence-Threshold: 0.7  
Speech-Language: en-US  
Content-Length: 47

builtin:dtmf/digits?length=4  
builtin:speech/pin

S->C:

MRCP/2.0 83 2 200 IN-PROGRESS  
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog

S->C:

MRCP/2.0 115 START-OF-INPUT 2 IN-PROGRESS  
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog  
Input-Type: speech
S->C:

MRCP/2.0 399 RECOGNITION-COMPLETE 2 COMPLETE
Channel-Identifier: 6ae0f23e1b1e3d42@speechrecog
Completion-Cause: 000 success
Content-Type: application/x-nlsml
Content-Length: 214

<?xml version="1.0"?>
<result>
  <interpretation grammar="builtin:speech/pin" confidence="1.00">
    <instance>one two three four</instance>
    <input mode="speech">one two three four</input>
  </interpretation>
</result>
8 Sequence Diagram

The following sequence diagram outlines common interactions between all the main components involved in a typical recognition session performed over MRCPv2.

![Sequence Diagram]

- IVR Platform
- UniMRCP Server
- Watson SR Plugin
- Watson Speech Service

Transactions:
- SIP INVITE
- SIP OK
- SIP ACK
- MRCPv2 RECOGNIZE
- MRCPv2 IN-PROGRESS
- MRCPv2 START-OF-INPUT
- MRCPv2 RECOGNITION-COMPLETE
- SIP BYE
- SIP OK

Processes:
- Open
- Open Done
- RECOGNIZE
- IN-PROGRESS
- START-OF-INPUT
- RECOG-COMPLETE
- Close
- Close Done
- WS Connect
- start
- listening
- results
- stop
- WS Close
9 References

9.1 IBM Watson

- Text to Speech API
- Credentials

9.2 Specifications

- Speech Recognizer Resource
- NLSML Results