FS – Media Translation Manager

Content

History

Version	Date	Adjustments
A	2006-10-06	First revision. (MBEME)

1 Introduction

This document specifies the functionality of the Media Translation Manager.

1.1 How to read this document

This document uses the standards for streaming media such as RTP, RTSP, MRCPv1 and variations thereof and it is highly recommended that these standards are understood before reading this document.

It is also recommended that the reader has knowledge of ASR and TTS engines.

2 Definitions

2.1 ASR

With ASR in this context it means the ability to recognize human speech and produce tokens recognized by the application.

Typically the input to an ASR engine is an RTP stream, grammar and the output is recognition messages in the form of MRCP.

2.2 TTS

With TTS in this context it means the ability to produce speech based on a textual representation of the speech.

Typically a TTS engine produces an RTP stream containing the speech based on a text.

3 Function Requirements (Commercial)

The following commercial requirements have been identified.

1. Support for TTS engines using RTSP and MRCPv1.
2. Support for ASR engines using RTSP and MRCPv1.

4 Function Specification (Design Related)

4.1 Introduction

The Media Translation component handles the translation of media objects from one format to another, currently the objective is to translate from text to speech (TTS) and recognize speech (ASR).

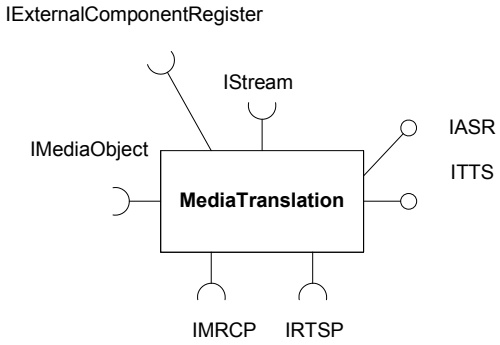

Figure 1: The exported and imported interfaces

The translation or recognition is typically done by an external component which is communicated with using RTSP/MRCP.

4.2 Exported Interfaces

4.2.1 The text to speech interface (ITTS)

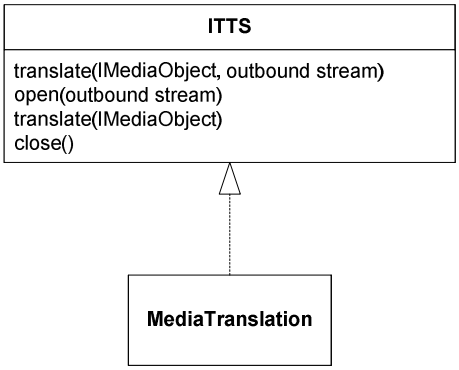

Figure 2: The text to speech interface

This interface is used for text to speech conversions and control of the conversion process. This interface is session specific.

4.2.1.1 Functions

4.2.1.1.1 translate

translate (IMediaObject, outbound stream)

This function is called from a media object that does not have media of the same properties as the outbound stream it is supposed to play the media on. This is only for media objects of the text type (text/plain and application/ssml).

translate (IMediaObject)

This function initiates a translation process utilizing a previously opened streaming session. The type constraints are the same as for the translate function above.

This method is used in conjunction with open.

4.2.1.1.2 open

open (outbound stream)

This function initializes a TTS streaming session. The speech is to be streamed through the outbound stream.

4.2.1.1.3 close

close()

This function closes a previously opened streaming session.

4.2.1.2 Parameter Type Description

4.2.1.2.1 IMediaObject

IMediaObject (see [9]) contains the "text" to be translated. The text can be in one of the following mime formats: text/html, text/plain and application/ssml. Depending on which format that is supported by the TTS engine the text is translated into the supported format.

4.2.1.2.2 Outbound stream

The speech is sent to this stream.

4.2.2 The automatic speech recognition interface (IASR)

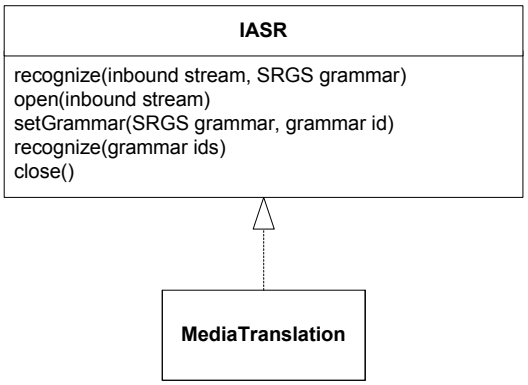

Figure 3: The automatic speech recognition interface

This interface is to start and control the speech recognition. This interface is session specific.

4.2.2.1 Functions

4.2.2.1.1 recognize

recognize (inbound stream, SRGS Document)

This function is used to initiate a speech recognition based on a grammar and recognition properties and an inbound stream. The recognition properties and SRGS supported is dependant on the ASR engine.

recognize (grammar ids)

This function is used to initiate a speech recognition based upon a set of predefined grammars (represented by grammar ids).

4.2.2.1.2 open

open (inbound stream)

This function initializes an ASR session. The speech is to be streamed through the inbound stream.

4.2.2.1.3 setGrammar

setGrammar (SRGS Document, grammar id)

This function deploys an SRGS grammar to the ASR engine through a previously opened ASR session. The grammar is referred to through a grammar id defined by the caller.

4.2.2.1.4 close

close

This function closes an open ASR session.

4.2.2.1.5 control

control (action, data)

This function controls the recognition. Only supported action is "stop" and data is currently not used.

4.2.2.2 Parameter Type Description

4.2.2.2.1 SRGS Document

The recognition grammar that is used by the ASR engine when listening to the RTP stream, see [5] for further details.

Example of an SRGS document:

```
<?xml version="1.0"?>

<!-- the default grammar language is US English -->
<grammar xml:lang="en-US" version="1.0">

<!-- single language attachment to tokens -->
        <rule id="yes">
                <one-of>
                        <item xml:lang="fr-CA">oui</item>
                        <item xml:lang="en-US">yes</item>
                </one-of>
        </rule>

<!-- single language attachment to a rule expansion -->
        <rule id="request">
                may I speak to
                <one-of xml:lang="fr-CA">
                        <item>Michel Tremblay</item>
                        <item>Andre Roy</item>
                </one-of>
        </rule>
</grammar>
```

4.2.2.2.2 Recognition properties

Properties on how the recognition is to be handled for instance confidence threshold, sensitivity level, number of returned matches, timeout values etc.

4.2.2.2.3 Inbound stream

The speech, which is to be recognized/translated, is received through this stream.

4.2.2.2.4 Action

This action can currently only be "stop".

4.3 Imported Interfaces

4.3.1 IExternalComponentRegister

The external component register is used to retrieve information about the external ASR/TTS engines (e.g. IP-address and port number).

4.3.2 IMediaObject

The media object contains the media, the text to be spoken by the TTS engine.

4.3.3 IStream

Booth inbound and outbound streams are used by the media translation manager. In specific the outbound stream is utilized for issuing play finished/failed events by the TTS handling.

4.3.4 IMRCP

The media translation manager communicates with external ASR/TTS engines over the MRCP protocol.

4.3.5 IRTSP

The MRCP requests are carried over RTSP.

4.4 Events

4.4.1 Generated

4.4.1.1 Recognition complete

This event is sent when the ASR engine has recognized a spoken sentence or word according to the specified SRGS and recognition properties. It will contain completion cause according to MRCP and the NLSML document with the match, see [3] and [6] for further detail.

Example of a result in NLSML:

```
<?xml version="1.0"?>
<result x-model="http://IdentityModel"
        xmlns:xf="http://www.w3.org/2000/xforms"
        grammar="session:request1@form-level.store>
        <interpretation>
                <xf:instance name="Person">
                        <Person>
                                <Name> Andre Roy </Name>
                        </Person>
                </xf:instance>
                <input>   may I speak to Andre Roy </input>
```

```
        </interpretation>
</result>
```

4.4.1.2 Recognition No Match

This event is sent when the ASR engine has failed to match the speech with the defined grammar.

4.4.1.3 Recognition No Input

This event is sent when the ASR engine has failed due to no speech.

4.4.1.4 Recognition Failed

This event is sent when the ASR engine has failed in one way or the other. The event will contain the MTM cause. The MTM cause is mapped from.

4.5 Function Specification

This FS specifies the following scenarios:

1. Initiate a text to speech translation
2. Control a text to speech translation
3. Initiate a speech recognition
4. Control a speech recognition
5. Load balancing media resources
6. Wrong media properties on inbound stream for ASR
7. Wrong media properties on outbound stream for TTS

4.5.1 RTSP/MRCP Support

The media translation component supports the RTSP/MRCPv1 combination for external TTS or ASR engines. See [2] and [3] for further details.

The handling/utilization of RTSP/MRCPv1 is encapsulated so that the implementation of media translation component does not depend upon the implementation of protocol implementations.

4.5.3 Initiate a text to speech translation to external TTS

When a text to speech translation to an external TTS engine is used the following scenario happens.

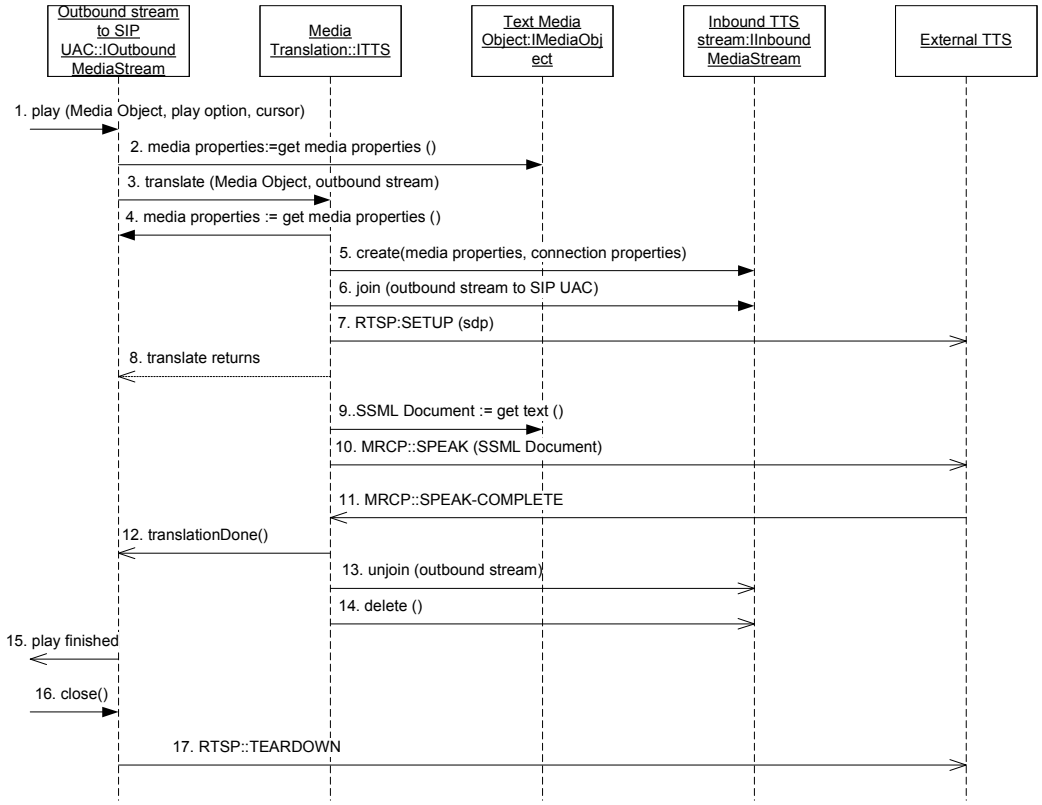

Figure 4: Play of text

It starts when a media object of type text is played:

1. The execution engine for example starts the play on the media object on the outbound media stream.
2. The stream checks the media properties of the media object and discovers that it is a text media object.
3. The outbound stream initiates the text to speech translation by calling the translate function on the media translation object.
4. The media translation object retrieves the outbound streams media properties.
5. The inbound stream from the TTS engine is created with the same media properties as the outbound stream.
6. The media translate object joins the streams.
7. The media translate sets up the RTSP session towards the TTS engine returned by the MCR towards the IExternalComponentRegister interface, see [7]. The service searched for is "TTS".
8. The translate returns to signal that the translation has been initiated to the media stream.
9. The media translator gets the text from the media object.

10. The media translator translates the text into the format that is supported by the TTS engine (see 4.5.3.1) and sends it to the external TTS engine. The engine starts streaming on the inbound stream. The streams are joined so the inbound RTP packets are streamed to the outbound RTP packets.
11. The external TTS engine signals the end of the speech.
12. The media translator signals that the translation has finished.
13. The media translator un-joins the streams.
14. The inbound TTS stream is deleted.
15. The outbound stream signals that the streaming has ended.
16. Close is called.
17. The TTS session is terminated

4.5.3.1 Translation of text

The text format which is supported by the TTS engine is defined in a configuration parameter. (Currently the format can be either HTML or SSML.) This means that the Media Translation Manager is responsible for converting (if necessary) the text, which is retrieved from IMediaObject, to a format that is supported by the TTS engine.

4.5.7 Initiate a speech recognition

When ASR is required media translation sets up the ASR and waits for recognition events from the ASR engine.

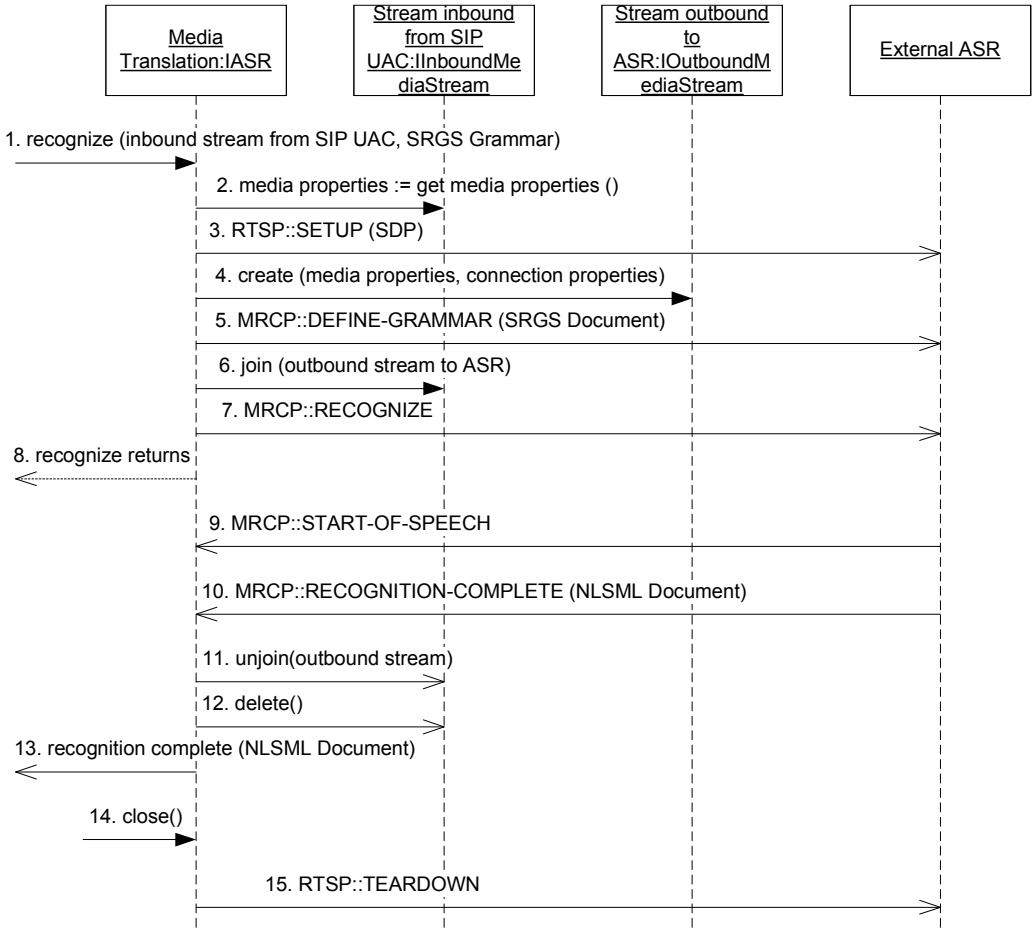

Figure 7: Initiation and recognition from the ASR

The recognition is initiated by a call to media translation through the recognize function.

1. Any component calls the recognize function giving the inbound stream where the incoming speech is and the SRGS grammar used for recognition.
2. The media properties from the inbound stream are retrieved to be able to create the outbound stream to the ASR engine.
3. The external ASR is setup with an RTSP message. The media properties are negotiated here.
4. The outbound stream to the ASR engine is created.
5. The grammar is sent to the ASR engine through an MRCP message.
6. The two streams are joined.
7. The recognition is started on the ASR engine.
8. The recognize function returns from the media translator to signal that the translation has started.

9. The ASR engine signals that the speech has arrived to the ASR engine.
10. The ASR engine signals that it has recognized speech that fits the grammar and responds with a NLSML document with the matches.
11. The two streams are un-joined.
12. The outbound stream is deleted.
13. The media translation component signals that the recognition is complete.
14. Close is called.
15. The ASR session is terminated.

4.5.8 Control a speech recognition

Whenever recognition has started it continues until stopped. Stopping recognition is done using the control function of the media translation component.

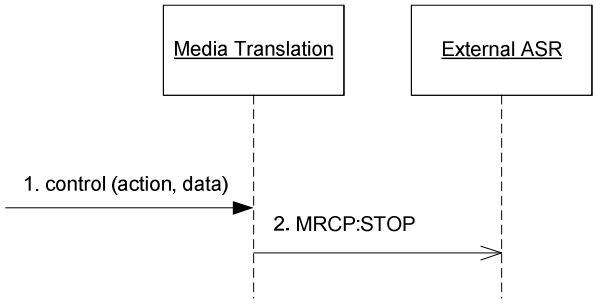

Figure 8: Control a speech recognition task

The scenario starts with a control function call:

1. Any component that wants to stop the current recognition sends a control call with "stop" as the control action, no data is needed.
2. The media translation component sends the external ASR an MRCP stop message.

4.5.9 Failing to initialize ASR

If the media translation manager fails to create and join an outbound stream or fails to set up recognition session no ASR can be started.

The inbound stream must provide PCMU audio in order to stream audio to the ASR engine.

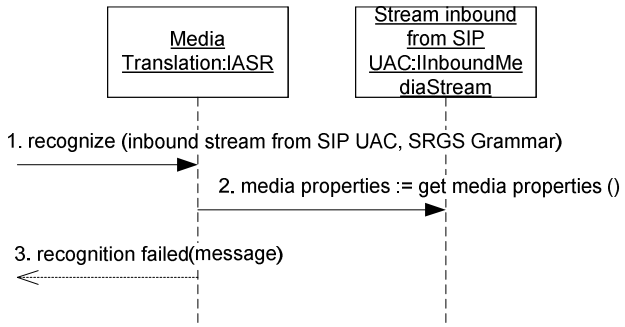

Figure 9: ASR initialization fail

The recognize starts as usual, with a recognize call to the media translation object:

1. The recognize method is called with the inbound stream and the SRGS grammar.
2. The media translation object fails to create and join an outbound stream or fails to create a recognition session.
3. The media translation object issues a recognition failed event.

4.5.10 Failing to intialize TTS

If the media translation manager fails to create and join an inbound stream or fails to set up text to speech session no TTS can be started.

The outbound stream must support PCMU audio in order to stream the translated text audio (from the TTS engine).

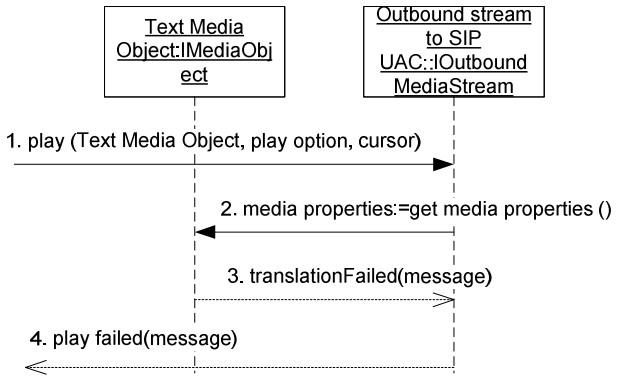

Figure 10: TTS initialization fail

The play starts as usual, with a play call to the outbound stream:

1. The play is called with media object.
2. The media translation object fails to create and join an inbound stream or fails to create a text to speech session.

3. The media translation object notifies the outbound stream that translation failed.
4. The outbound stream issues a play failed event.

4.5.11 Configuration

The following configuration is needed by the media translation manager:

1. Which protocol used when communicating with the TTS engine (currently *none* or *mrcp*).
2. Which protocol used when communicating with the ASR engine (currently *none* or *mrcp*).

The configuration values are validated. Configuration errors, such as missing and invalid values, are logged.

4.5.12 Logging

When communication with external host is lost it is only logged once. There is only one log entry when the communication is lost and only one entry when the communication is re-established. External host are the ASR and TTS engines.

4.5.13 Load balancing

Since there can be more than one TTS/ASR host there must be some kind of load balancing. The load balancing is handled by the TTS/ASR hosts.

5 References

1. IWD Extensible Message Protocol
   2/155 19-1/HDB 101 02 Uen
2. RTSP, Real Time Streaming Protocol
   RFC 2326
3. MRCP, Media Resource Control Protocol
   RFC 4463
4. SSML Speech Synthesis Markup Language Version 1, Recommendation 7
   http://www.w3.org/TR/speech-synthesis/
5. SRGS Speech Recognition Grammar Specification Version 1, Recommendation 16
   http://www.w3.org/TR/speech-grammar/
6. NLSML Natural Language Semantics Markup Language for the Speech Interface Framework, Draft 20
   http://www.w3.org/TR/nl-spec/
7. FS – External Component Register
   20/FS-MAS0001 Uen
8. FS - Stream
   11/FS-MAS0001 Uen

9. FS – Media Object
   13/FS-MAS0001 Uen