GENIVI Alliance

GENIVI Document CS000XX

SpeechOutputService

Component Specification

Not Accepted Version 1.0.0

25-01-2017

Sponsored by:
GENIVI Alliance

true

NotAccepted

Abstract:
This document provides a component specification for the Speech Output Service.This document provides the Component Specification for the NavigationCore

Keywords:
Navigation, LocationInput, Routing, Guidance

2400 Camino Ramon, Suite 375, San Ramon, CA 94583, USA
http://www.genivi.orgtrue

GENIVI Alliance
2400 Camino Ramon, Suite 375
San Ramon, CA 94583, USA

Revision History

Document revision history

Date	Version	Author	Description
2015-01-02	*0.1*	*David Kämpf*	*Initial revision. Containing the results of the Speech F2F in Erlangen 12/2014.*
2015-01-05	*0.1.1*	*Mario Thielert*	*Minor corrections on initial revision*
2015-01-05	*0.1.2*	*David Kämpf*	*Synchronized with new UML model. Minor additions and fixes.*
2017-02-22	0.1.3	Philippe Colliot	Refine document to allow API generation from the Franca file

Table of Contents

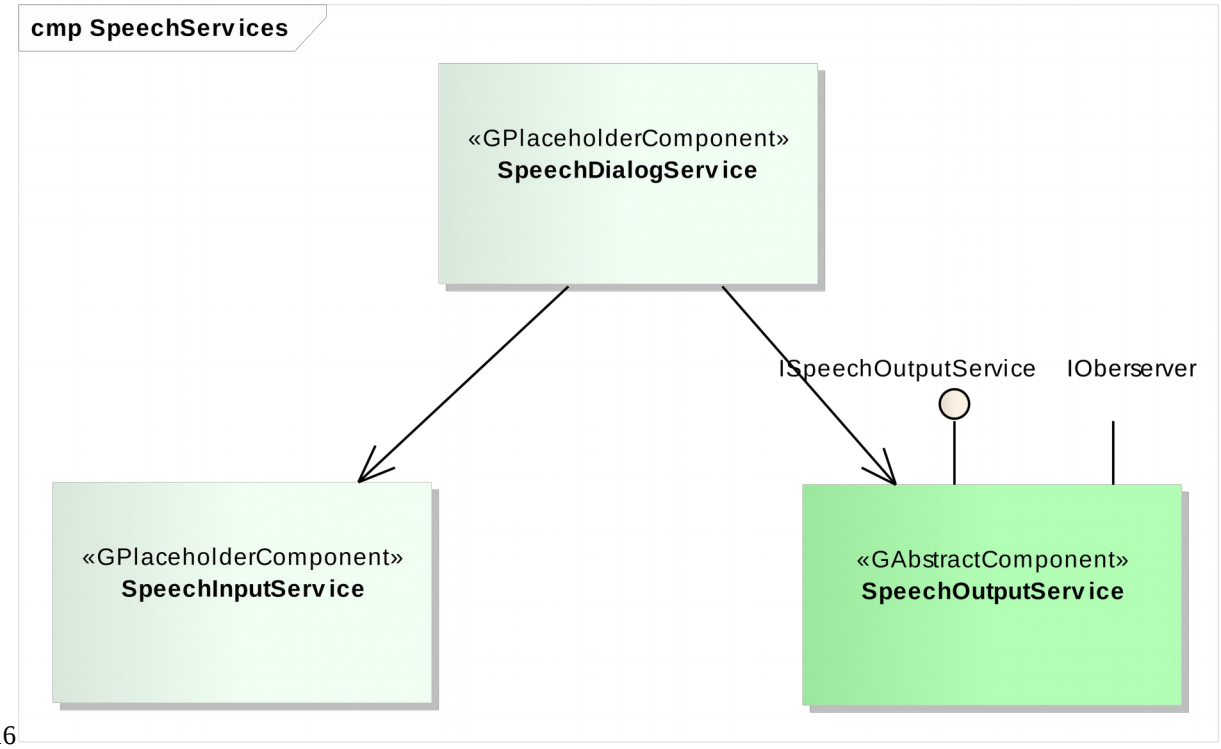
1.1 System Overview

Boiler plate, to be written, describing the overall GENIVI Software Platform.

1.2 Subsystem Speech Overview

The Speech Subsystem contains of 3 components:

- **Speech Dialog Service**
  - Modeling the User Interaction
  - Handling of resources
  - Interact with GUI
  - Interact with Business Logic
- **Speech Input Service**
  - Integration of the Voice Recognizer
  - Resource handling
- **Speech Output Service**
  - Integration of TTS Engine



1.3 Component Overview

The SpeechOutputService is encapsulating access to the TTS engine. The responsibilities of SpeechOutputService are:

- Provide access to the TTS for the SpeechDialogService
- Provide access to the TTS for applications

- Arbitrate conflicting requests to speak a prompt
- Encapsulate audio connection handling for the client application

In the current POC implementation SpeechOutputService's interface is realized as C++ interface. Due to the fact that it is strictly asynchronous other implementations are feasible as well.

1.4 Document Overview

This document is describing the SpeechOutputService abstract component version 0.1.2.

2 References

The following standards and specifications contain provisions, which through reference in this document constitute provisions of this specification. All the standards and specifications listed are normative references. At the time of publication, the editions indicated were valid. All standards and specifications are subject to revision, and parties to agreements based on this specification are encouraged to investigate the possibility of applying the most recent editions of the standards and specifications indicated below.

[1] GENIVI UML Model - https://svn.genivi.org/uml-model/genivi/trunk

1

13 Glossary

2

Acronym	Term	Definition
RDF	Resource Description Framework	
DB	Database	
NBT	Next Big Thing	
EWMH	Extended Window Manager Hints	
KDE	K Desktop (Windows) Environment	
ICCM	Inter-Client Communication Conventions Manual (of KDE)	
PDA	Personal Digital Assistant	
PIM	Personal Information Management	
UUID	Unique ID (Identification)	
CDDB	Compact Disc Database (now Gracenote)	
IDE	Integrated Drive Electronics	
SCSI	Small Computer Serial Interface	
USB	Universal Serial Bus	
UDF	Universal Disk Format (for optical media)	
VFAT	Virtual File Allocation Table (Linux support for Microsoft FAT file systems)	
TTS	Text to Speech Engine	An engine that is converting graphems (written text) into spoken output.
SPC	Speech	
SOS	Speech Output Service	
VR	Voice Recognizer	
SW	Software	
IPC	Inter Process Communication	

3Table 1 – Acronym and Term Definitions

4 Requirements

The information in this chapter is provided only for information purpose; this is not a normative part.

4.1 Functional Requirements

Requirement ID	Requirement	Priority	Rationale
SW-SPC-SOS-071	After loading of languages or voices, the TTS shall support a signal indicating the end of loading.	Medium	
SW-SPC-SOS-064	At system SW upgrade of the Speech Output component, TTS engine, languages and prerecordings shall be upgradable separately.	Medium	
SW-SPC-SOS-039	Context Dependent Disambiguation of Abbreviations: TTS shall disambiguate frequently used abbreviations based on their context. Abbreviations may have a different meaning and pronounciation based on their context. E.g. compare "St. Martin" versus "main	Medium	
SW-SPC-SOS-059	Continuity of Speech Output Within a text to be output by TTS, no unnatural pauses shall occur effected by operational constraints.	Medium	
SW-SPC-SOS-036	Control of Speaking Rate: TTS shall enable the user to adjust the average speaking rate of TTS for a specific voice according to his personal preferences. Such an adjustment is permanent and survives power-off.	Medium	
SW-SPC-SOS-037	Control of Voice Pitch: TTS shall enable the user or an application to adjust the pitch level of a voice. TTS shall accept permanent voice-specific adjustments by a user and temporary adjustments by an application.	Medium	
SW-SPC-SOS-041	Customer specific dictionary extensions shall be permanent and survive power-off.	Medium	
SW-SPC-SOS-072	During loading of languages or voices, the system shall supply an indicator for the status of the loading process, to be potentially used by the HMI to show a progress bar.	Medium	
SW-SPC-SOS-018	Identification of Foreign Words in a Text: TTS should identify the language origin of foreign language words in a native language text, and select language specific conversion rules that lead to an adequate pronounciation.	Medium	
SW-SPC-SOS-017	Identification of Text Language: TTS should identify the language in which a text is written, allowing to select an appropriate TTS module for the identified language.	Medium	
SW-SPC-SOS-055	In case of an upgrade at a major change level including modification of user-specific data, the TTS module shall provide an additional function allowing transformation of the user-specific data into a re-usable form, with an acceptable processing time an	Medium	
SW-SPC-SOS-021	In case of phonetic input, TTS shall map any phonemes from foreign languages to appropriate phonemes of the active language.	Medium	
SW-SPC-SOS-019	Language tagging: TTS should support language tags for words/phrases in the text passed to the TTS.	Medium	
SW-SPC-SOS-024	Modification of Voices at Runtime: The TTS shall support modification of the parameters of the currently used voice at runtime, to allow the adaptation of the voice (e.g. speed and pitch) to the preferences of the customer or user.	Medium	

1

SW-SPC-SOS-016	Multi-Linguality: The TTS shall be able to read text that contains different languages with adequate pronounciation.	Medium	
SW-SPC-SOS-023	Multiple Voices per Language: At least for the primary languages, TTS shall allow selection between at least one female and one male voice.	Medium	
SW-SPC-SOS-020	Pronounciation of Non-Native Phonemes: The TTS shall be able to speak a text which includes phonemes that are not part of the active language.	Medium	
SW-SPC-SOS-006	Reading of Abbreviations: The TTS module shall support pronounciation of common abbreviations.	Medium	
SW-SPC-SOS-005	Reading of Mixed Plain and Phonetic Text: The TTS module shall support mixing of text in phonetic form and in graphemic form into application-specific messages, if sufficient application-specific information is available to create an overall message inton	Medium	
SW-SPC-SOS-007	Reading of Numbers: The Speech Output component shall support tags for numbers embedded into speakble text to enforce the way numbers are spoken. E.g. Numbers can be spoken as numerical value (nine hundred eleven) or as digit sequence (nine one one).	Medium	
SW-SPC-SOS-004	Reading of Phonetic Text: The TTS module shall support reading of phonetic texts, optionally including prosody information, and shall generate output in form of an audio stream.	Medium	
SW-SPC-SOS-003	Reading of Plain Text: The TTS module shall support reading of plain (= graphemic resp. orthographic) text, generating speech output in form of an audio stream.	Medium	
SW-SPC-SOS-060	Seamless Speech Output: Concatenated prerecorded prompts shall produce a continuous voice stream, without any unintended speech gaps.This requirement has to be fulfilled during the output of a prerecorded speech segment, during concatenation at the boun	Medium	
SW-SPC-SOS-009	Speech Output component shall process application specifc tags indicating a context (like E-Mail, TTS or Navigation) in order to enable the TTS to optimize text reading.	Medium	
SW-SPC-SOS-038	Standard Dictionary for Abbreviations: TTS shall contain a standard dictionary for correct pronounciation of unusual words and for standard abbreviations.	Medium	
SW-SPC-SOS-040	TTS shall allow extension of the standard dictionaries and abbreviations by customer specific entries.	Medium	
SW-SPC-SOS-058	TTS shall be able to output user specific prompts.	Medium	
SW-SPC-SOS-057	TTS shall be able to store and access a customer or user specific exception dictionary.	Medium	
SW-SPC-SOS-035	TTS shall enable the user or an application to adjust the volume of different TTS voices in relation to each other according to his personal preferences. Such a modification shall be permanent and survives power-off.	Medium	
SW-SPC-SOS-029	TTS shall offer a mechanism to specify/mark time positions within or at the end of an utterance	Medium	
SW-SPC-SOS-043	TTS shall offer an improved pronounciation of names, if a name is explicitly marked as name.	Medium	
SW-SPC-SOS-	TTS shall offer an improved pronounciation of street	Medium	

1

044	names and location names, if the street name or location name is marked as address.		
SW-SPC-SOS-042	TTS shall offer the usual text specific preprocessing mechanisms to correctly pronounce various number formats, date, time, currencies, phone numbers etc.	Medium	
SW-SPC-SOS-046	TTS shall provide an improved pronounciation of a CD, DVD, song or movie title, if the text item is marked as a related title.	Medium	
SW-SPC-SOS-045	TTS shall provide an improved pronounciation of radio station names, if the name is marked as radio station name.	Medium	
SW-SPC-SOS-031	TTS shall support activation of a language that has been newly loaded into non-volatile memory.	Medium	
SW-SPC-SOS-062	TTS shall support activation of more than one language.	Medium	
SW-SPC-SOS-032	TTS shall support deactivation of the active language for instance for language resources exchange.	Medium	
SW-SPC-SOS-002	The G2P of the TTS module shall be aligned with the G2P used by the Speech Recognizer to pronounce the spoken text in a way that is suitable for speech recognition.	Medium	
SW-SPC-SOS-048	The TTS engine can be switched off.	Medium	
SW-SPC-SOS-010	The TTS module shall be able to correctly interpret or map the phoneme sets used by third party database providers like GraceNote or Navteq.	Medium	
SW-SPC-SOS-008	The TTS module shall support domain specific exception dictionaries. An exception dictionary bypasses the G2P processing of the passed word or abbreviation and provides the domain specific pronounciation.	Medium	
SW-SPC-SOS-012	The TTS module shall support reading of POI names provided in text form or phonetical form by navigation databases.	Medium	
SW-SPC-SOS-011	The TTS module shall support reading of location names or street names provided in text form or phonetical form by navigation databases.	Medium	
SW-SPC-SOS-013	The TTS module shall support reading of person names as stored in telephone databases or PDA contact databases.	Medium	
SW-SPC-SOS-014	The TTS module shall support reading of the contents of artist names, song titles, and album titles, as stored in the data fields of audio entertainment files like e.g. MP3-files or WMA-files.	Medium	
SW-SPC-SOS-015	The TTS module shall support reading of the contents of artist names, song titles, and album titles, using enhanced information as provided by web-based databases like e.g. Gracenote.	Medium	
SW-SPC-SOS-034	The TTS shall deliver it's status.	Medium	
SW-SPC-SOS-026	The TTS shall provide a function to stop the running prompt.	Medium	
SW-SPC-SOS-025	The TTS shall start Text to Speech conversion after receiving an explicit start command.	Medium	
SW-SPC-SOS-067	The TTS system shall support selection of a specific voice by the customer, out of a set of one or more voices per language.	Medium	
SW-SPC-SOS-056	The language loading methods shall support the exchange of application-specific voice prompt sets	Medium	

1

	without conflicts to other voice prompt sets of the used voice or voice style.		
SW-SPC-SOS-022	The mapping of phonemes of foreign languages to the active language shall be seamless, i.e. no foreign phonemes may bed dropped due to a missing mapping function.	Medium	
SW-SPC-SOS-073	The speech output component shall be compiant with GENIVI Audio architecture.	Medium	
SW-SPC-SOS-053	The speech output module shall be able to read aloud electronic mail and SMS messages.	Medium	
SW-SPC-SOS-054	The speech output module shall be able to read aloud the contents of fields in ID3 tags as provided by compressed music files.	Medium	
SW-SPC-SOS-050	The speech output module shall support output consisting of both TTS speech sequences and prompt speech sequences, mixed in arbitrary order.	Medium	
SW-SPC-SOS-051	The speech output module shall support output of announcements for navigation guidance messages.	Medium	
SW-SPC-SOS-052	The speech output module shall support output of announcements for speech dialogues.	Medium	
SW-SPC-SOS-049	The speech output module shall support output of prerecorded speech.	Medium	
SW-SPC-SOS-047	The speech output module shall support output of synthetic speech generated by TTS.	Medium	
SW-SPC-SOS-079	The speech output service shall return an error on a new prompter request while a TTS is active	Medium	
SW-SPC-SOS-076	The speech output system shall return an error if pause is called and no TTS is active	Medium	
SW-SPC-SOS-077	The speech output system shall return an error if resume is called and no TTS is active	Medium	
SW-SPC-SOS-075	The speech output system should provide the current state	Medium	
SW-SPC-SOS-069	The system shall enable the car manufacturer and the user to specify the voice of the active language.	Medium	
SW-SPC-SOS-068	The system shall enable the car manufacturer and the user to specify the voices to be loaded for each language.	Medium	
SW-SPC-SOS-033	The system shall enable the user to select the active language out of the set of loaded languages.	Medium	
SW-SPC-SOS-001	The system shall have a TTS Engine.	Medium	
SW-SPC-SOS-065	The system shall support a deletion mechanism to remove language subpackages that are not required any more.	Medium	
SW-SPC-SOS-063	The system shall support a language loading mechanism to replace the available languages.	Medium	
SW-SPC-SOS-066	The system shall support a version comparison mechanism to determine the languages that need to be updated to newer versions.	Medium	
SW-SPC-SOS-061	The system shall support output of recorded speech including signals other than speech like e.g. jingles or signal tones in a sound quality comparable to the quality of prerecorded speech.	Medium	
SW-SPC-SOS-030	Whenever TTS speech output reaches a marked position, TTS shall issue an event to the system, signalling that speech output has reached the defined time position.	Medium	
SW-SPC-SOS-070	While loading languages, voices or prompt sets, speech input and output shall be disabled for all	Medium	

	applications.		
SW-HAF-SPC-018	The system generates a spoken feedback or guidance for the user. The audio data is either already stored on the system (pre-recording) or will be generated on the fly by a TTS engine.	Medium	

4.2 Non Functional Requirements

There are currently no non-functional requirements for the SpeechOutputService.

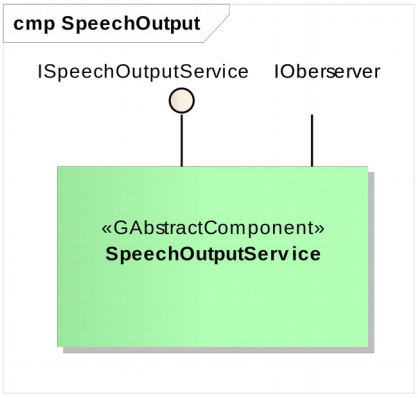
5 Constraints and Assumptions

This section shall summarize the constraints and assumptions done in the project for the component.

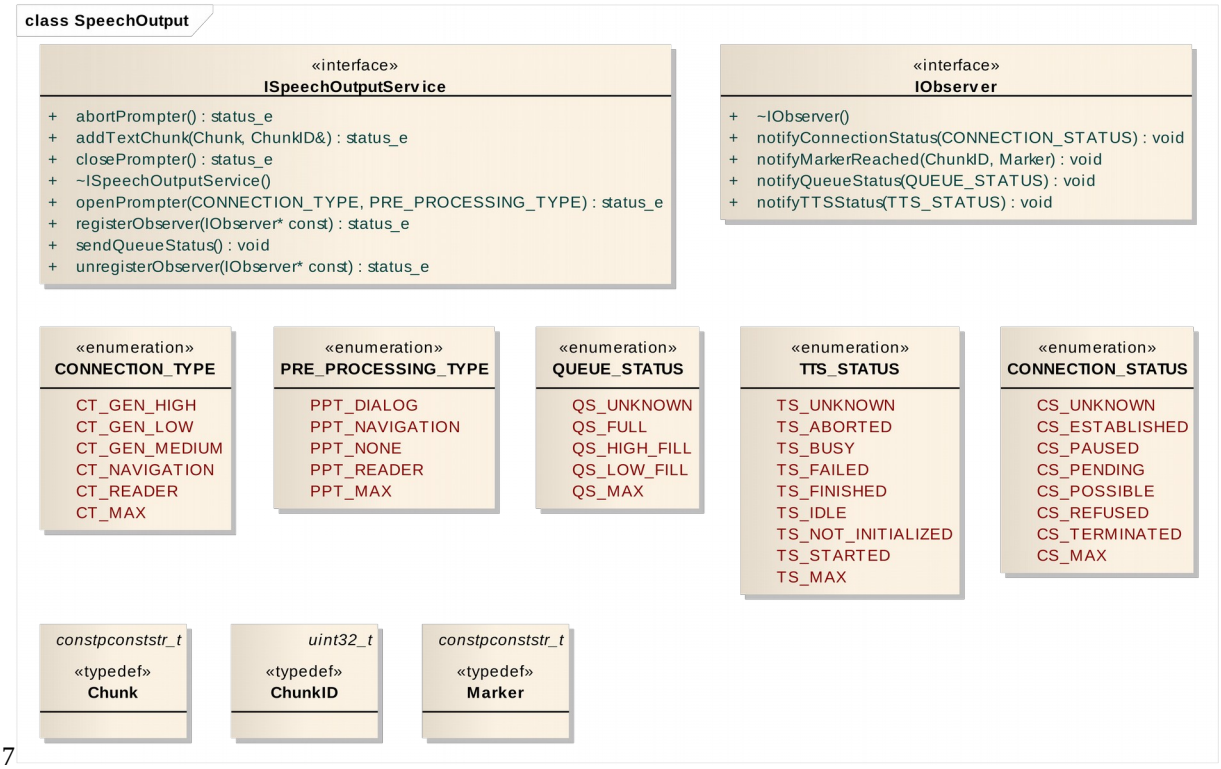
6 Architecture

The information in this chapter is provided only for information and recommendation purpose; this is not a normative part.

6.1 Architecture Overview



6.1.1 Component Interfaces



The main interface is the ISpeechOutputService which provides functions to:

- Open a prompter session
- Add text to the SpeechOutputService that will be spoken by the TTS
- Close a prompter session

In addition to that there is the IObserver interface that delivers status changes to the clients.

When opening a prompter session two paramaters will have to be provided: the connection type and the pre-processing tyoe:

- The connection type will be used to prioritize conflicting TTS requests, e.g. a Navigation prompt will not be interrupted by a Reader application trying to read out an E-Mail.

- The pre-processing type is configuring the pre-processor of the built in TTS engine in order to be optimized for a specific use case. Normally the pre-processor would select a specific set of rules that are application specific and would e.g in the case of navigation expand “in 100m turn right” to “in 100 meters turn right”.

Once the client has opened a TTS session with OpenPrompter it can speak by adding text chunks to the engine using the addTextChunk method.

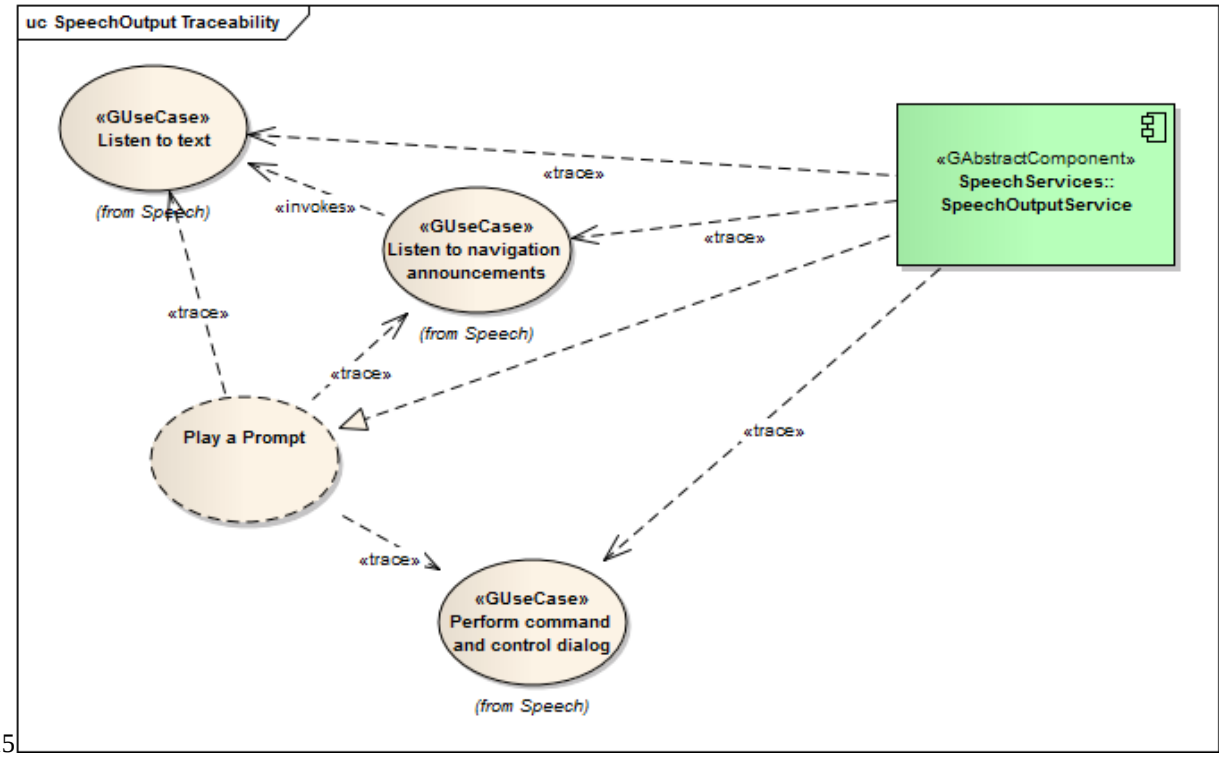
The client will receive receive status notifications concerning two major aspects:

- Status of the session (CS – Connection status), e.g. a notification the opening the prompter was successful
- Status of the TTS engine, e.g. a notfiction that the TTS engine is currently reading out text or has reached a specific position in text.

6.1.2 Component Dependencies

SpeechOutputService has currently no dependencies.

6.1.3 Component Traceability



Mainly the SpeechOutputService implements the “Play a prompt” Use Cases, be this navigation announcements, content from E-Mails or other messages and lists.

In addition to that SpeechOutputService traces all of the requirements in the section Speech Output as listed in chapter 4. Most of these requirements are satisfied by the TTS engine encapsulated by SpeechOutputService.

6.2 SpeechOutputService Details

6.2.1 Responsibility and Features

Responsibilities of the SpeechOutputService:

- Encapsulate the TTS engine and provide a vendor agnostic interface for applications
- Arbitrate concurrent access to the TTS engine from different applications

- Provide session handling to ease application development

6.2.2 Provided Interfaces

SpeechOutputService provides ISpeechOutputService which is intended to provide control over the TTS engine for applications.

ISpeechOutputService provides methods to

- Control a session (openPrompter, closePrompter)
- Add text to the TTS buffer to be spoken
- Abort a running prompt

6.2.3 Required Interfaces

SpeechOutputService in the current POC implementation requires the IObserver interface which is defining the callbacks that deliver status information to the clients.

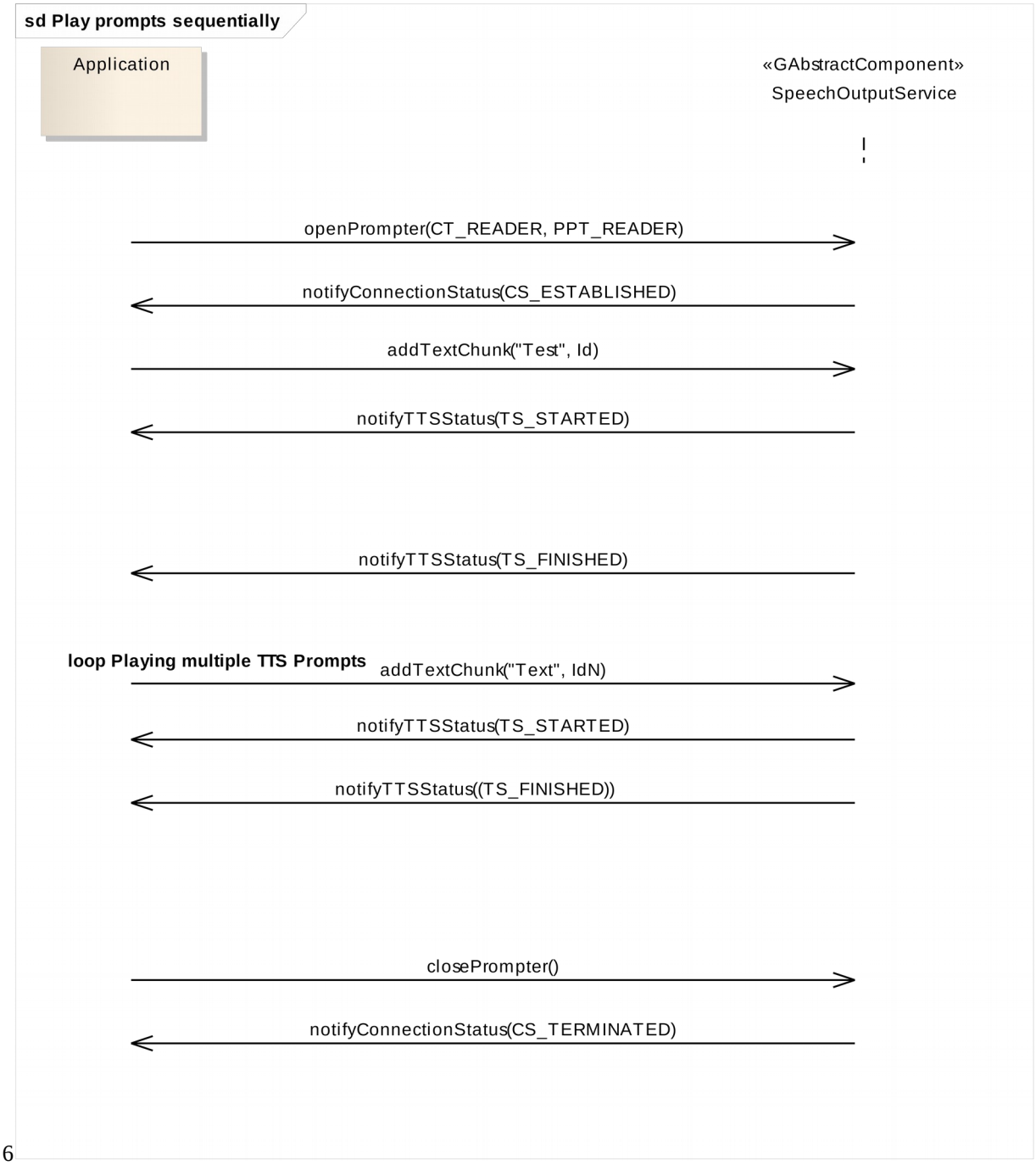
Each client can register it's callbacks at the SpeechOutputService and will be provided with information about:

- TTS status – providing information about the TTS engine, e.g. if TTS has started to put out a prompt
- Session status – providing information if the application was able to open a TTS session
- Buffer status – providing information about the text buffer

7 Collaboration

7.1 Use Case Realization: Play Prompts sequentially

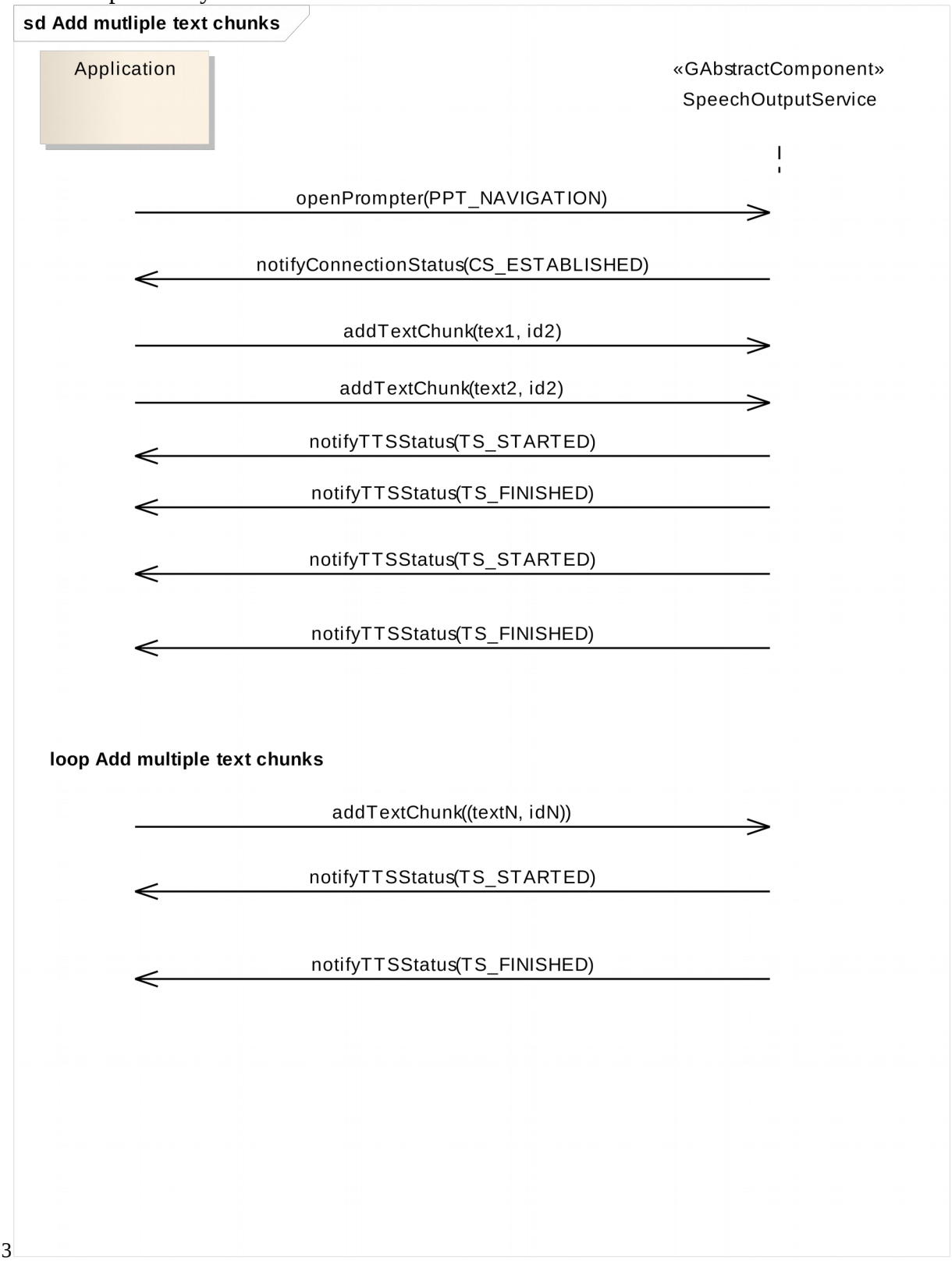
The following sequence describes the “good case” of an application trying to put out a prompt. The application opens the session successfully with openPrompter and then adds one or multiple chunks of text that get spoken by the TTS engine. After the application is finished it closes the session with closePrompter.



7.2 Use Case Realization: Add multiple Text Chunks

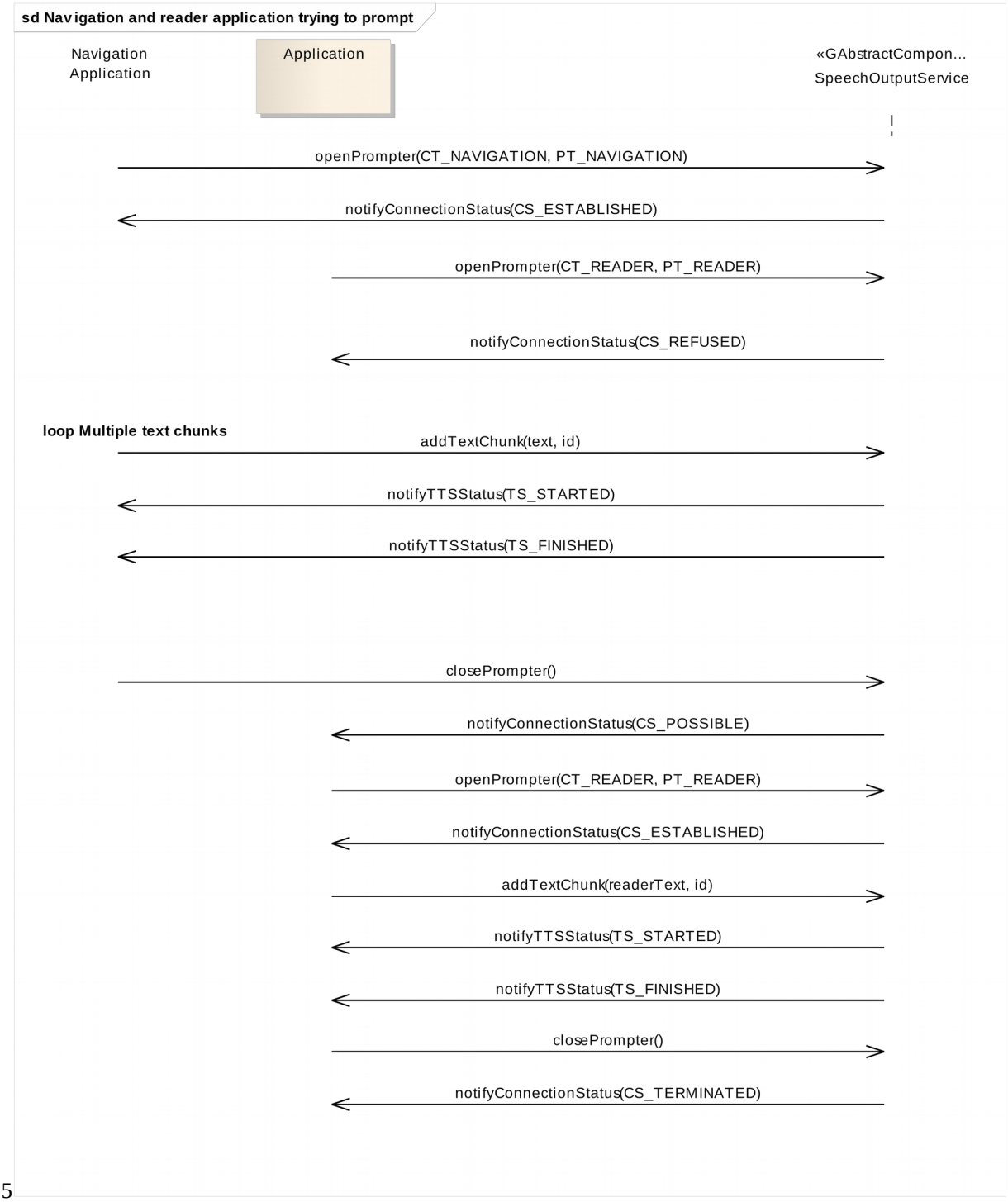
A client application can add multiple text chunks to the TTS buffer that will be spoken in this order.

It is sometimes preferable to add multiple small chunks of text instead of one big chunk in order to optimize system latencies.



7.3 Use Case Realization: Navigation and Reader Application trying to prompt

The following sequence illustrates the behavior of SpeechOutputService in case of conflicting requests. The prompter type navigation has higher priority and thus gets access to the service.



8 Interfaces

1.1 The following pages describe the interfaces of the SpeechOutputService API

SpeechOutputService

Generated by Doxygen 1.8.11

Contents

1 Namespace Documentation

1.1 org Module Reference

Modules

- genivi

1.2 org::genivi Module Reference

Modules

- hmi

1.3 org::genivi::hmi Module Reference

Modules

- module speechoutputservice

1.4 org::genivi::hmi::speechoutputservice Module Reference

Classes

- interface SpeechOutput
- interface SpeechOutput_client

2 Class Documentation

2.1 org::genivi::hmi::speechoutputservice::SpeechOutput Interface Reference

Public Types

- enum Limits { MAX_CHUNK_LENGTH = 1024 }
- enum ConnectionStatus {
  CS_UNKNOWN, CS_ESTABLISHED, CS_REFUSED, CS_POSSIBLE,
  CS_PENDING, CS_PAUSED, CS_TERMINATED, CS_MAX }
- enum ConnectionType {
  CT_NAVIGATION, CT_READER, CT_GEN_HIGH, CT_GEN_MEDIUM,
  CT_GEN_LOW, CT_MAX }
- enum PreProcessingType {
  PPT_DIALOG, PPT_NAVIGATION, PPT_NONE, PPT_READER,
  PPT_MAX }
- enum QueueStatus {
  QS_UNKNOWN, QS_FULL, QS_HIGH_FILL, QS_LOW_FILL,
  QS_MAX }
- enum TtsStatus {
  TS_UNKNOWN, TS_NOT_INITIALIZED, TS_ACTIVE, TS_ABORTED,
  TS_MARKER, TS_IDLE, TS_ENQUEUED, TS_FINISHED,
  TS_FAILED, TS_MAX }
- typedef String Chunk
- typedef UInt32 ChunkID
- typedef String Marker

Public Member Functions

- void getVersion (out Version version)
- void openPrompter (in ConnectionType connectionType, in PreProcessingType preProcessingType)
- void addTextChunk (in Chunk chunk, out ChunkID chunkID)
- void abortPrompter ()
- void closePrompter ()

Public Attributes

- const UInt32 MAX_CHUNK_LENGTH = 1024

2.1.1 Detailed Description

#comment : SpeechOutput interface. The GENIVI SpeechOutput interface allows client applications to access TTS functionality

2.1.2 Member Typedef Documentation

2.1.2.1 typedef String org::genivi::hmi::speechoutputservice::SpeechOutput::Chunk

2.1.2.2 typedef UInt32 org::genivi::hmi::speechoutputservice::SpeechOutput::ChunkID

2.1.2.3 typedef String org::genivi::hmi::speechoutputservice::SpeechOutput::Marker

2.1.3 Member Enumeration Documentation

2.1.3.1 enum org::genivi::hmi::speechoutputservice::SpeechOutput::ConnectionStatus

Enumerator

CS_UNKNOWN
CS_ESTABLISHED
CS_REFUSED
CS_POSSIBLE
CS_PENDING
CS_PAUSED
CS_TERMINATED
CS_MAX

2.1.3.2 enum org::genivi::hmi::speechoutputservice::SpeechOutput::ConnectionType

Enumerator

CT_NAVIGATION
CT_READER
CT_GEN_HIGH
CT_GEN_MEDIUM
CT_GEN_LOW
CT_MAX

2.1.3.3 enum org::genivi::hmi::speechoutputservice::SpeechOutput::Limits

Enumerator

MAX_CHUNK_LENGTH

2.1.3.4 enum org::genivi::hmi::speechoutputservice::SpeechOutput::PreProcessingType

Enumerator

PPT_DIALOG
PPT_NAVIGATION
PPT_NONE
PPT_READER
PPT_MAX

2.1.3.5 enum org::genivi::hmi::speechoutputservice::SpeechOutput::QueueStatus

Enumerator

QS_UNKNOWN
QS_FULL
QS_HIGH_FILL
QS_LOW_FILL
QS_MAX

2.1.3.6 enum org::genivi::hmi::speechoutputservice::SpeechOutput::TtsStatus

Enumerator

TS_UNKNOWN
TS_NOT_INITIALIZED
TS_ACTIVE
TS_ABORTED
TS_MARKER
TS_IDLE
TS_ENQUEUED
TS_FINISHED
TS_FAILED
TS_MAX

2.1.4 Member Function Documentation

2.1.4.1 void org::genivi::hmi::speechoutputservice::SpeechOutput::abortPrompter ( )

#comment : A prompt must be playing to perform an abort action. If no prompting operation in progress there will be no reaction of the system.

2.1.4.2 void org::genivi::hmi::speechoutputservice::SpeechOutput::addTextChunk (in **Chunk** *chunk,* out **ChunkID** *chunkID*)

#comment : The prompter must be opened to trigger the playback of the provided prompt. The prompt length must not exceed the length of a PromptChunk buffer. Synthesizes the provided text or if using the escape sequence of the engine supplier a wave file in a supported sampling rate is provided, the system will back back also wave files. The text will be normalized using the context identifier provided to openprompter. This applies to matching prerecorded files as well as the synthesis of number and words that are matched to a lexical dictionary. The synthesize will start if the prompter is idle, if the prompter is already playing the playback will be delayed until all previously added text chunks are played back. For every text chunk provided a notification will be send.

2.1.4.3 void org::genivi::hmi::speechoutputservice::SpeechOutput::closePrompter ( )

#comment : The prompter is closed after the last text chunk submitted has finished playing.

2.1.4.4 void org::genivi::hmi::speechoutputservice::SpeechOutput::getVersion (out Version *version*)

#comment : This method returns the API version implemented by the SpeechOutput.

2.1.4.5 void org::genivi::hmi::speechoutputservice::SpeechOutput::openPrompter (in **ConnectionType** *connectionType,* in **PreProcessingType** *preProcessingType*)

#comment : Must be called to open a SpeechOutputService session and to get the audio connection.

2.1.5 Member Data Documentation

2.1.5.1 const UInt32 org::genivi::hmi::speechoutputservice::SpeechOutput::MAX_CHUNK_LENGTH = 1024

#comment : Max length of a single prompt part.

The documentation for this interface was generated from the following file:

- SpeechOutput.fidl

2.2 org::genivi::hmi::speechoutputservice::SpeechOutput_client Interface Reference

Public Member Functions

- void notifyConnectionStatus (in ConnectionStatus connectionStatus)
- void notifyMarkerReached (in ChunkID chunkID, in Marker marker)
- void notifyQueueStatus (in QueueStatus queueStatus)
- void notifyTTSStatus (in TtsStatus ttsStatus)

2.2.1 Member Function Documentation

2.2.1.1 void org::genivi::hmi::speechoutputservice::SpeechOutput_client::notifyConnectionStatus (in ConnectionStatus *connectionStatus*)

broadcast #comment : Notifies the connection status

2.2.1.2 void org::genivi::hmi::speechoutputservice::SpeechOutput_client::notifyMarkerReached (in ChunkID *chunkID,* in Marker *marker*)

broadcast #comment : Notifies the last reached marker.

2.2.1.3 void org::genivi::hmi::speechoutputservice::SpeechOutput_client::notifyQueueStatus (in QueueStatus *queueStatus*)

broadcast #comment : Notifies the queue status.

2.2.1.4 void org::genivi::hmi::speechoutputservice::SpeechOutput_client::notifyTTSStatus (in TtsStatus *ttsStatus*)

broadcast #comment : Notifies the TTS engine status.

Returns

: TtsStatus TTS status information

The documentation for this interface was generated from the following file:

- SpeechOutput.fidl

3 File Documentation

3.1 SpeechOutput.fidl File Reference

Classes

- interface org::genivi::hmi::speechoutputservice::SpeechOutput
- interface org::genivi::hmi::speechoutputservice::SpeechOutput_client

Modules

- module org::genivi::hmi::speechoutputservice

Index