Requirements for Distributed Control of ASR, SI/SV and TTS Resources

1. Abstract

This document outlines the needs and requirements for a protocol to control distributed speech processing of audio streams. By speech processing, this document specifically means automatic speech recognition, speaker recognition (which includes both speaker identification and speaker verification) and text-to-speech. Other IETF protocols, such as SIP and RTSP, address rendezvous and control for generalized media streams. However, speech processing presents additional requirements that none of the extant IETF protocols address.

Discussion of this and related documents is on the speechsc mailing list. To subscribe, send the message "subscribe speechsc" to speechsc-request@ietf.org. The public archive is at http://www.ietf.org/mail-archive/workinggroups/speechsc/current/maillist.html.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC-2119 [2]. FORMATTING NOTE: Notes, such as this one, provide additional, nonessential information that the reader may skip without missing anything essential. The primary purpose of these non-essential notes is to convey information about the rationale of this document,
or to place this document in the proper historical or evolutionary context. Readers whose sole purpose is to construct a conformant implementation may skip such information. However, it may be of use to those who wish to understand why we made certain design choices.

OPEN ISSUES: This document highlights questions that are, as yet, undecided as "OPEN ISSUES".

3. Introduction

There are multiple IETF protocols for establishment and termination of media sessions (SIP[5]), low-level media control (MGCP[6] and MEGACO[7]), and media record and playback (RTSP[8]). This document focuses on requirements for one or more protocols to support the control of network elements that perform Automated Speech Recognition (ASR), speaker identification or verification (SI/SV), and rendering text into audio, a.k.a. Text-to-Speech (TTS). Many multimedia applications can benefit from having automatic speech recognition (ASR) and text-to-speech (TTS) processing available as a distributed, network resource. This requirements document limits its focus on the distributed control of ASR, SI/SV and TTS servers.

There are a broad range of systems which can benefit from a unified approach to control of TTS, ASR, and SI/SV. These include environments such as VoIP gateways to the PSTN, IP Telephones, and wireless mobile devices who obtain speech services via servers on the network.

To date, there are a number of proprietary ASR and TTS API’s, as well as two IETF drafts that address this problem [9] [10]. However, there are serious deficiencies to the existing drafts. In particular, they mix the semantics of existing protocols yet are close enough to other protocols as to be confusing to the implementer.

This document sets forth requirements for protocols to support distributed speech processing of audio streams. For simplicity, and to remove confusion with existing protocol proposals, this document presents the requirements as being for a "new protocol" that addresses the distributed control of speech resources It refers to such a protocol as "SPEECHSC", for Speech Services Control Protocol.

4. SPEECHSC Framework

The following is the SPEECHSC framework for speech processing.

```
+-------------+    +--------+
| Application |  |  ASR   |
| Server     |
SIP or whatever /  \
/                \
+------------+    +--------+
| Media      |    | and/or |
| Processing |    |        |
```
The "Media Processing Entity" is a network element that processes media. The "Application Server" is a network element that instructs the Media Processing Entity on what transformations to make to the media stream. The "ASR and/or TTS Server" is a network element that either generates a RTP stream based on text input (TTS) or returns speech recognition results in response to an RTP stream as input (ASR). Either the Media Processing Entity or the Application Server may control the ASR or TTS Server using SPEECHSC as a control protocol.

Physical embodiments of the entities can reside in one physical instance per entity, or some combination of entities. For example, a VoiceXML [11] Gateway may combine the ASR and TTS functions on the same platform as the Media Processing Entity. Note that VoiceXML Gateways themselves are outside the scope of this protocol. Likewise, one can combine the Application Server and Media Processing Entity, as would be the case in an interactive voice response (IVR) platform.

One can also decompose the Media Processing Entity into an entity that controls media endpoints and entities that process media directly. Such would be the case with a decomposed gateway using MGCP or megaco. However, this decomposition is again orthogonal to the scope of SPEECHSC.

5. General Requirements

5.1. Reuse Existing Protocols

To the extent feasible, the SPEECHSC framework SHOULD use existing protocols.

5.2. Maintain Existing Protocol Integrity

In meeting requirement 5.1, the SPEECHSC framework MUST NOT redefine the semantics of an existing protocol. Said differently, we will not break existing protocols or cause backward compatibility problems.

5.3. Avoid Duplicating Existing Protocols

To the extent feasible, SPEECHSC SHOULD NOT duplicate the functionality of existing protocols. For example, SIP with msuri [12] and RTSP already define how to request playback of audio. The focus of SPEECHSC is new functionality not addressed by existing protocols or extending existing protocols within the strictures of requirement 5.2. Where an existing protocol can be gracefully extended to support SPEECHSC requirements, such extensions are
acceptable alternatives for meeting the requirements.

As a corollary to this, the SPEECHSC should not require a separate protocol to perform that could be easily added into the SPEECHSC protocol (like redirecting media streams, or discovering capabilities), unless it is similarly easy to embed that protocol directly into the SPEECHSC framework.

5.4. Protocol efficiency

The SPEECHSC framework SHOULD employ protocol elements known to result in efficient operation. Techniques to be considered include:
- Re-use of transport connections across sessions
- Piggybacking of responses on requests in the reverse direction
- Caching of state across requests

5.5. Explicit invocation of services

The SPEECHSC framework MUST be compliant with the IAB OPES[5] framework. The applicability of the SPEECHSC protocol will therefore be specified as occurring between clients and servers at least one of which is operating directly on behalf of the user requesting the service.

5.6. Server Location and Load Balancing

To the extent feasible, the SPEECHSC framework SHOULD exploit existing schemes for supporting service location and load balancing, such as the Service Location Protocol[13] or DNS SRV records[14]. Where such facilities are not deemed adequate, the SPEECHSC framework MAY define additional load balancing techniques.

5.7. Multiple services

The SPEECHSC framework MUST permit multiple services to operate on a single media stream so that either the same or different servers may be performing speech recognition, speaker identification or verification, etc. in parallel.

5.8. Multiple media sessions

The SPEECHSC framework MUST allow a 1:N mapping between session and RTP channels. For example, a single session may include an outbound RTP channel for TTS, an inbound for ASR and a different inbound for SI/SV (e.g. if processed by different elements on the Media Resource Element). Note: All of these can be described via SDP, so if SDP is utilized for media channel description, this requirement is met for free?
TTS Requirements

6.1. Requesting Text Playback

The SPEECHSC framework MUST allow a Media Processing Entity or Application Server, using a control protocol, to request the TTS Server to playback text as voice in an RTP stream.

6.2. Text Formats

6.2.1. Plain Text

The SPEECHSC framework MAY assume that all TTS servers are capable of reading plain text. For reading plain text, framework MUST allow the language and voicing to be indicated via session parameters. For finer control over such properties, see 6.2.2.

6.2.2. SSML

The SPEECHSC framework MUST support SSML[3] <speak> basics, and SHOULD support other SSML tags. The framework assumes all TTS servers are capable of reading SSML formatted text.

6.2.3. Text in Control Channel

The Speechsc framework assumes all TTS servers accept text over the SPEECHSC connection for reading over the RTP connection. The framework assumes the server can accept text either ?by value? (embedded in the protocol), or ?by reference? (by de-referencing a URI embedded in the protocol).

6.2.4. Document Type Indication

The SPEECHSC framework MUST be capable of explicitly indicating the document type of the text to be processed, as opposed to forcing the server to infer the content by other means.

6.3. Control Channel

The SPEECHSC framework MUST be capable of establishing the control channel between the client and server on a per-session basis, where a session is loosely defined to be associated with a single ?call? or ?dialog?.. The protocol SHOULD be capable of maintaining a long-lived control channel for multiple sessions serially, and MAY be capable of shorter time horizons as well, including as short as for the processing of a single utterance.

6.4. Media origination/termination by control elements
The SPEECHSC framework MUST NOT require the controlling element (application server, media processing entity) to accept or originate media streams. Media streams MAY source & sink from the controlled element (ASR, TTS, etc.).

6.5. Playback Controls

The Speechsc framework MUST support VCR controls?, and MUST allow for servers with varying capabilities to accommodate such controls. These capabilities include:
- The ability to jump in time to the location of a specific marker.
- The ability to jump in time, forwards or backwards, by a specified amount of time. Valid time units MUST include seconds, words, paragraphs, sentences, and markers.
- The ability to increase and decrease playout speed.
- The ability to fast-forward and fast-rewind the audio, where snippets of audio are played as the server moves forwards or backwards in time.
- The ability to pause and resume playout.
- The ability to increase and decrease playout volume.

6.6. Session Parameters

The SPEECHSC framework must support the specification of session parameters, such as language, prosody and voicing.

6.7. Speech Markers

The SPEECHSC framework MUST accommodate speech markers, with capability at least as flexible as that provided in SSML[3]. The framework MUST further provide an efficient mechanism for reporting that a marker has been reached during playout.

7. ASR Requirements

7.1. Requesting Automatic Speech Recognition

The SPEECHSC framework MUST allow a Media Processing Entity or Application Server to request the ASR Server to perform automatic speech recognition on an RTP stream, returning the results over SPEECHSC.

7.2. XML
7.3. Grammar Requirements

7.3.1. Grammar Specification

The Speechsc framework assumes all ASR servers are capable of accepting grammar specifications either "by value" (embedded in the protocol), or "by reference" (by de-referencing a URI embedded in the protocol). The latter MUST allow the indication of a grammar already known to, or otherwise "built in" to the server. The framework and protocol further SHOULD exploit the ability to store and later retrieve by reference large grammars which were originally supplied by the client.

7.3.2. Explicit Indication of Grammar Format

The SPEECHSC framework protocol MUST be able to explicitly convey the grammar format in which the grammar is encoded and MUST be extensible to allow for conveying new grammar formats as they are defined.

7.3.3. Grammar Sharing

The Speechsc framework SHOULD exploit sharing grammars across sessions for servers which are capable of doing so. This supports applications with large grammars for which it is unrealistic to dynamically load. An example is a city-country grammar for a weather service.

7.4. Session Parameters

The SPEECHSC framework MUST accommodate at a minimum all of the protocol parameters currently defined in MRCP[7]. In addition there SHOULD be a capability to reset parameters within a session.

7.5. Input Capture

The SPEECHSC framework MUST support a method directing the ASR Server to capture the input media stream for later analysis and tuning of the ASR engine.

8. Speaker Identification and Verification Requirements

8.1. Requesting SI/SV

The SPEECHSC framework MUST allow a Media Processing Entity to request the SI/SV Server to perform speaker identification or verification on an RTP stream, returning the results over SPEECHSC.
8.2. Identifiers for SI/SV

The SPEECHSC framework MUST accommodate an identifier for each verification resource and permit control of that resource by ID, because voiceprint format and contents are vendor specific.

8.3. State for multiple utterances

The SPEECHSC framework MUST work with SI/SV servers which maintain state to handle multi-utterance verification.

8.4. Input Capture

The SPEECHSC framework MUST support a method for capturing the input media stream for later analysis and tuning of the SI/SV engine. The framework may assume all servers are capable of doing so.

8.5. SI/SV functional extensibility

The SPEECHSC framework SHOULD be extensible to additional functions associated with SI/SV, such as prompting, utterance verification, and retraining.

9. Duplexing and Parallel Operation Requirements

One very important requirement for an interactive speech-driven system is that user perception of the quality of the interaction depends strongly on the ability of the user to interrupt a prompt or rendered TTS with speech. Interrupting, or barging, the speech output requires more than energy detection from the user's direction. Many advanced systems halt the media towards the user by employing the ASR engine to decide if an utterance is likely to be real speech, as opposed to a cough, for example.

9.1.1. Full Duplex operation

To achieve low latency between utterance detection and halting of playback, many implementations combine the speaking and ASR functions. The SPEECHSC framework MUST support such full-duplex implementations.

9.1.2. Multiple services in parallel

Good spoken user interfaces typically depend upon the ease with which the user can accomplish his or her task. When making use of Speaker Identification or Verification technologies, user interface improvements often come from the combination of the different technologies: simultaneous identity claim and verification (on the same utterance), simultaneous knowledge and voice verification (using ASR and verification simultaneously). Using ASR and verification on the same utterance is in fact the only way to support rolling or dynamically-generated challenge phrases (e.g.,...
"say 51723"). The SPEECHSC framework MUST support such parallel service implementations.

9.1.3. Combination of services

It is optionally of interest that the SPEECHSC framework support more complex remote combination and controls of speech engines:
- Combination in series of engines that may then act on the input or output of ASR, TTS or Speaker recognition engines. The control MAY then extend beyond such engines to include other audio input and output processing and natural language processing.
- Intermediate exchanges and coordination between engines
- Remote specification of flows between engines.

These capabilities MAY benefit from service discovery mechanisms (e.g. engines, properties and states discovery).

10. Additional Considerations (non-normative)

The framework assumes that SDP will be used to describe media sessions and streams. The framework further assumes RTP carriage of media, however since SDP can be used to describe other media transport schemes (e.g. ATM) these could be used if they provide the necessary elements (e.g. explicit timestamps).

The working group will not be defining distributed speech recognition methods (DSR), as exemplified by the ETSI Aurora project. The working group will not be recreating functionality available in other protocols, such as SIP or SDP.

TTS looks very much like playing back a file. Extending RTSP looks promising for when one requires VCR controls or markers in the text to be spoken. When one does not require VCR controls, SIP in a framework such as Network Announcements [10] works directly without modification.

ASR has an entirely different set of characteristics. For barge-in support, ASR requires real-time return of intermediate results. Barring the discovery of a good reuse model for an existing protocol, this will most likely become the focus of SPEECHSC.

11. Security Considerations

Protocols relating to speech processing must take security into account. This is particularly important as popular uses for TTS include reading financial information. Likewise, popular uses for ASR include executing financial transactions and shopping.

We envision that rather than providing application-specific security mechanisms in SPEECHSC itself, the resulting protocol will employ
security machinery of either containing protocols or the transport on which it runs. For example, we will consider solutions such as using TLS for securing the control channel, and SRTP for securing the media channel. Third-part dependencies necessitating transitive trust will be minimized or explicitly dealt with through the authentication and authorization aspects of the protocol design.

In addition to the security machinery needed by the protocol itself, there are considerations for the implementation and deployment of the clients and servers themselves. For example, speaker verification and identification employs voiceprints whose privacy and integrity must be maintained. While strictly speaking out of scope of the protocol itself, such considerations will be carefully considered and accommodated during protocol design, and will be called out as part of the applicability statement accompanying the protocol specification(s).

12. Normative References

1 Bradner, S., "The Internet Standards Process -- Revision 3", BCP 9, RFC 2026, October 1996.

2 Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997


13. Informative References


Van Dyke, J., Burger, E., Spitzer, A., O‘Connor, W., "Basic Network Media Services with SIP", draft-burger-sipping-netann-02.txt, June 2002, work in progress


Acknowledgments

Stephane Maes, Sarvi Shanmughan, Brian Eberman, Dan Burnett, and Brian Wyld all made significant contributions of requirements and proposed text for capturing them.

Author’s Addresses

Eric W. Burger
SnowShore Networks, Inc.
Chelmsford, MA
USA
Email: eburger@snowshore.com

David R. Oran
Cisco Systems, Inc.
Acton, MA
USA
Email: oran@cisco.com
16. Change Log

From version draft-burger-mrcp-reqts-00 to version draft-burger-speechsc-reqts-00:
- draft name changed per area director advice
- added speaker verification to the areas addressed, including speaker verification requirements, per Dan Burnet’s presentation at the Minneapolis BoF (see minutes).
- based on mailing list discussion, added requirement to handle both ?by value? and ?by reference? data. This is both for TTS to be played out and grammar(s) to be applied to ASR.
- Based on discussion at the BoF in Minneapolis, added a requirement concerning the use of load balancing schemes, including those based on SRVLOC, SRV.
- Added a requirement for OPES compliance, per a discussion with Sally Floyd as IAB observer for the BoF.

From version draft-burger-speechsc-reqts-00 to version draft-ietf-speechsc-reqts-00:
- Replaced SRCP with SPEECHSC everywhere
- Minor edits including mailing list name change, temporary notes removed,
- All agreements reached at the IETF 54 WG meeting, confirmed by mailing list discussion, up through 8/10/02 have been integrated
- Improved requirement on VCR controls as suggested by Dan Burnett and Sarvi Shanmughan
- Text describing dual-mode requirements for ASR and SR by Dan Burnett added.
- Suggested change to framework figure made by Rajiv Dharmadhikari incorporated
- Updated references to most recent versions

From version draft-ietf-speechsc-reqts-00 to version draft-ietf-speechsc-reqts-01.txt:
- Adopted Rajiv D.’s wording clarification to the TTS & ASR requirements to allow control to come from either a separate Application Server, or a combined server with a Media Processing entity.
- Reorganized references into separate normative and informative sections as requested by Scott Bradner

- Added numbering for requirements in sections that were not previously numbered. This necessitated a bit of text shuffling to group related requirements more closely together.

- Added a paragraph to the introduction to emphasize the wide variety of applications of the speechsc framework and explicitly call out wireless mobile devices, IP phones, and PSTN VoIP gateways.
- During WGLC, the use of the term "speaker recognition" to cover both speaker identification and speaker verification was questioned. In addition some WG participants felt that there should be separate requirements for each, while others argued that the differences, while affecting the structure of the application, did not affect the requirements for the protocol in any substantive way. There were views that the existing terminology was common in the industry and hence should not be changed, and sentiments for a variety of other solutions. The best compromise seemed to be to continue to group the requirements together, but point out where there may be subtle differences affecting applications. It also seemed prudent to keep the identification/verification distinction in the terminology, and hence the document uses the acronym SI/SV rather than SR when talking about both together.

- added a requirement in section 5 for 1:N mapping of control to media channels, but pointed out that if SDP is used this comes for free.

- Changed the input capture requirement from SHOULD to MUST, but made implementation by the server a SHOULD.

- added a SHOULD requirement for protocol efficiency with re-use of transport connections as one of a set of examples.

- noted in section 10 that while RTP is assumed, the framework applies to other media carriage schemes that can be described by SDP, as long as they have the right features.

From version draft-ietf-speechsc-reqts-01 to version draft-ietf-speechsc-reqts-02.txt:
- Reformulated requirements that applied to servers to instead apply to the framework and protocol, since the WG is defining those rather than the servers themselves.
- Added two extension requirements as suggested by Skip Cave
- Included clarifications on SSML and a few other minor things as suggested by Qiru Zhou
- Fixed typos etc. pointed out by Dan Durnett and others.

Burger & Oran   Informational? Expires August 2002
Distributed Media Control Requirements February 2002

Full Copyright Statement
Copyright (C) The Internet Society (2002). All Rights Reserved.
This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.
The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns. This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

The Internet Society currently provides funding for the RFC Editor function.