Internet Engineering Task Force
Internet-Draft
Intended status: Best Current Practice
Expires: September 29, 2020

G. Hellstrom
Omnitor

March 28, 2020

Real-time text solutions for multi-party sessions
draft-hellstrom-avtcore-multi-party-rtt-solutions-00

Abstract

This document specifies methods for Real-Time Text (RTT) media handling in multi-party calls. The main RTT transport is to carry Real-Time text by the RTP protocol in a time-sampled mode according to RFC 4103 [RFC4103] . The mechanisms enable the receiving application to present the received real-time text media separated per source, in different ways according to user preferences. Some presentation related features are also described explaining suitable variations of transmission and presentation of text.

Call control features are described for the SIP environment. A number of alternative methods for providing the multi-party negotiation, transmission and presentation are discussed and a recommendation for the main ones is provided. The main solution for SIP based centralized multi-party handling of real-time text is achieved through a media control unit coordinating multiple RTP text streams into one RTP stream.

Alternative methods using a single RTP stream and source identification inline in the text stream are also described, one of them being provided as a lower functionality fallback method for endpoints with no multi-party awareness for RTT.

Bridging methods where the text stream is carried without the contents being dealt with in detail by the bridge are also discussed.

Brief information is also provided for multi-party RTT in the WebRTC environment.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Copyright Notice

Table of Contents

1. Introduction

Real-time text (RTT) is a medium in real-time conversational sessions. Text entered by participants in a session is transmitted in a time-sampled fashion, so that no specific user action is needed to cause transmission. This gives a direct flow of text in the rate it is created, that is suitable in a real-time conversational setting. The real-time text medium can be combined with other media in multimedia sessions.

Media from a number of multimedia session participants can be combined in a multi-party session. The present document specifies how the real-time text streams can be handled in multi-party sessions. Recommendations are provided for preferred methods.

The description is mainly focused on the transport level, but also describes a few session and presentation level aspects.

Transport of real-time text is specified in RFC 4103 [RFC4103] RTP Payload for text conversation. It makes use of RFC 3550 [RFC3550] Real Time Protocol, for transport. Robustness against network transmission problems is normally achieved through redundant transmission based on the principle from RFC 2198 [RFC2198], with one primary and two redundant transmission of each text element. Primary and redundant transmissions are combined in packets and described by a redundancy header. This transport is usually used in the SIP Session Initiation Protocol RFC 3261 [RFC3261] environment.

A very brief overview of functions for real-time text handling in multi-party sessions is described in RFC 4597 [RFC4597] Conferencing Scenarios, sections 4.8 and 4.10. The present specification builds on that description and indicates which protocol mechanisms should be used to implement multi-party handling of real-time text.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [I-D.ietf-mmusic-t140-usage-data-channel]. Multi-party aspects for WebRTC solutions are briefly covered.

1.1. Requirements Language

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 [RFC2119].

2. Centralized conference model

In the centralized conference model for SIP, introduced in RFC 4353 [RFC4353] A Framework for Conferencing with the Session Initiation Protocol (SIP), one function co-ordinates the communication with participants in the multi-party session. This function also controls media mixer functions for the media appearing in the session. The central function is common for control of all media, while the media mixers may work differently for each media.

The central function is called the Focus UA. Many variants exist for setting up sessions including the multipoint control centre. It is not within scope of this description to describe these, but rather the media specific handling in the mixer required to handle multi-party calls with RTT.

The main principle for handling real-time text media in a centralized conference is that one RTP session for real-time text is established including the multipoint media control centre and the participating endpoints which are going to have real-time text exchange with the others.

The different possible mechanisms for mixing and transporting RTT differs in the way they multiplex the text streams and how they identify the sources of the streams. RFC 7667 [RFC7667] describes a number of possible use cases for RTP. This specification refers to different sections of RFC 7667 for further reading of the situations caused by the different possible design choices.

The recommended method for using RTT in a centralized conference model is specified in [I-D.hellstrom-avtcore-multi-party-rtt-source] based on the recommendations in the present document.

Real-time text can also be transported in the WebRTC environment, by using WebRTC data channels according to [I-D.ietf-mmusic-t140-usage-data-channel]. Ways to handle multi-party calls in that environmnent are also specified.

3. Requirements on multi-party RTT

The following requirements are placed on multi-party RTT:

> A solution shall be applicable to IMS (3GPP TS 22.173)[TS22173], SIP based VoIP and Next Generation Emergency Services (NENA i3 [NENAi3], ETSI TS 103 479 [TS103479], RFC 6443[RFC6443]).

The transmission interval for text must not be longer than 500 milliseconds when there is anything available to send. Ref ITU-T T.140 [T140].

If text loss is detected or suspected, a missing text marker shall be inserted in the text stream. Ref ITU-T T.140 Amendment 1 [T140ad1]. ETSI EN 301 549 [EN301549]

The display of text from the members of the conversation shall be arranged so that the text from each participant is clearly readable, and its source and the relative timing of entered text is visualized in the display. Mechanisms for looking back in the contents from the current session should be provided. The text should be displayed as soon as it is received. Ref ITU-T T.140 [T140]

Bridges must be multimedia capable (voice, video, text). Ref NENA i3 STA-010.2. [NENAi3]

R7: It MUST be possible to use real-time text in conferences both as a medium of discussion between individual participants (for example, for sidebar discussions in real-time text while listening to the main conference audio) and for central support of the conference with real-time text interpretation of speech. Ref RFC 5194.[RFC5194]

It should be possible to protect RTT contents with usual means for privacy and integrity.Ref RFC 6881 section 16. [RFC6881]

Conferencing procedures are documented in RFC 4579 [RFC4579]. Ref NENA i3 STA-010.2.[NENAi3]

Conferencing applies to any kind of media stream by which users may want to communicate. Ref 3GPP TS 24.147 [TS24147]

The framework for SIP conferences is specified in RFC 4353 [RFC4353]. Ref 3GPP TS 24.147 [TS24147]

The mixer performance requirements can be expressed in two figures.

1) The number of participants who can transmit simultaneously with the text not being delayed in the mixer more than 500 milliseconds. This requirement is depending on the application. Five simultaneous transmitting participants is a sufficiently high number for most situations.

2) The switching time from when the mixer is transmitting text from one participant and text arrives from another participant, until the mixer sends the text from the second participant. This time should not be more than 500 milliseconds.

4. Coordination of text RTP streams

Coordinating and sending text RTP streams in the multi-party session can be done in a number of ways. The most suitable methods are specified here with pros and cons.

A receiving and presenting endpoint MUST separate text from the different sources and identify and display them accordingly.

4.1. RTP-based solutions with a central mixer

A set of solutions can be based on the central RTP mixer. They are described here and a preferred method selected.

4.1.1. RTP Mixer using default RFC 4103 methods

Without any extra specifications, a mixer would transmit with 300 milliseconds intervals, and use RFC 4103 [RFC4103] with the default redundancy of one original and two redundant transmissions. The source of the text would be indicated by a single member in the CSRC list. Text from different sources cannot be transmitted in the same packet. Therefore, from the time when the mixer sent one piece of new text from one source, it will need to transmit that text again twice as redundant data, before it can send text from another source. The switching time will thus be 900 milliseconds. The mixer can not even send text from two simultaneous sources without introducing more than 500 milliseconds delay. This is clearly insufficient.

Pros:

Only a capability negotiation method is needed. No other update of standards are needed, just a general remark that traditional RTP-mixing is used.

Cons:

Clearly insufficient mixer switching performance.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

4.1.2. RTP Mixer using the default method but decreased transmission interval

This method makes use of the default RTP-mixing method briefly described in section Section 4.1.1. The only difference is that the transmission interval is decreased to 100 milliseconds when there is text from more than one source available for transmission. This increases the switching performance to three source switches per second. The delay of new text from a participant can be one second.

Pros:

Minor influence on standards

Cons:

Too long delay of new text from more than two simultaneous sources.

A bit complex handling of transmission when there is new text available from more than one source. The mixer needs to send two packets more with redundant text from the current source before starting to send anything from the other source.

4.1.3. RTP Mixer with frequent transmission and indicating sources in CSRC-list

An RTP media mixer combines text from participants into one RTP stream, thus all using the same destination address/port combination, the same RTP SSRC and , one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. This method is also briefly described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The sources of the text in each RTP packet are identified by the CSRC list in the RTP packets, containing the SSRC of the initial sources of text. The order of the CSRC parameters is with the SSRC of the source of the primary text first, followed by the SSRC of the first level redundancy, and then the second level.

The transmission interval should be 100 milliseconds when there is text to transmit from more than one source.

The details for application of this method together with RFC 4103 [RFC4103] are specified in [I-D.hellstrom-avtcore-multi-party-rtt-source]

The identification of the sources is made through the CSRC fields and can be made more readable through the RTCP SDES CNAME and NAME packets as described in RTP[RFC3550].

Also information provided through the notification according to RFC 4575 [RFC4575] when the participant joined the conference provides suitable information and a reference to the SSRC.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The ordered CSRC lists in the RFC 4103 [RFC4103] packets make it possible to recover from loss of one and two packets in sequence and assign the recovered text to the right source. For more loss, a marker for possible loss should be inserted or presented.

The conference server needs to have authority to decrypt the payload in the received RTP packets in order to be able to recover text from redundant data or insert the missing text marker in the stream, and repack the text in new packets.

Pros:

This method has low overhead and less complexity than the methods in sections Section 4.1.1, Section 4.1.2, Section 4.1.4 and Section 4.1.5.

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream (normally primary and two redundant levels).

This method can be implemented with most RTP implementations.

The source switching performance is sufficient for well-behaving conference participants. There can be switching between five source per second with an introduced delay of maximum 500 ms. With just two parties typing simultaneously, the delay will be a maximum of 100 ms.

Cons:

When more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the sources of the totally lost data.

The conference server needs to be allowed to decrypt/encrypt the packet payload. This is however normal for media mixers for other media.

4.1.4. RTP Mixer using timestamp to identify redundancy

This method has text only from one source per packet, as the original RFC 4103 [RFC4103] specifies. Packets with text from different sources are instead allowed to be merged. The recovery procedure in the receiver will use the RTP timestamp and timestamp offsets in the redundancy headers to evaluate if a piece of redundant data should be recovered or not in case of packet loss.

In this method, the transmission interval is 100 milliseconds when text from more than one source is available for transmission.

Pros:

The format of each packet is equal to what is specified in RFC 4103 [RFC4103].

The source switching performance is sufficient. Text from five participants can be transmitted simultaneously with 500 milliseconds interval per source.

New text from five simultaneous sources can be transmitted within 500 milliseconds. This is sufficient.

Cons:

The recovery time in case of packet loss is long. With five participants, it will be 1.5 seconds.

The recovery procedure is complex and very different from what is described in RFC 4103 [RFC4103].

It is not sure that this change can be regarded to be an update to RFC 4103.

4.1.5. RTP Mixer with multiple primary data in each packet

This method allows primary as well as redundant text from more than one source per packet. The packet payload contains an ordered set of redundant and primary data with the same number of generations of redundancy as once agreed in the SDP negotiation. The redundancy header reflects these parts of the payload. The CSRC list contains one CSRC member per source in the payload.

The maximum number of members in the CSRC-list is 16, and that is therefore the maximum number of sources that can be represented in each packet provided that all data can be fitted into the size allowable in one packet.

The transmission interval is set to 100 milliseconds when there is text from more than one source available for transmission.

Pros:

The source switching performance is good. Text from five (and in fact 16) participants can be transmitted simultaneously with 300 milliseconds interval per source.

New text from five (and in fact 16) simultaneous sources can be transmitted within 300 milliseconds. This is good performance.

Cons:

The format of each packet is very different from what is specified in RFC 4103 [RFC4103].

It is doubtful if this change can be seen as just an update to RFC 4103.

The recovery procedure is complex and very different from what is described in RFC 4103 [RFC4103].

It is doubtful if this change can be regarded to be an update to RFC 4103.

4.1.6. RTP Mixer indicating participants by a control code in the stream

Text from all participants except the receiving one is transmitted from the media mixer in the same RTP session and stream, thus all using the same destination address/port combination, the same RTP SSRC and , one sequence number series as described in Section 7.1 and 7.3 of RTP RFC 3550 [RFC3550] about the Mixer function. The sources of the text in each RTP packet are identified by a new defined T.140 control code "c" followed by a unique identification of the source in UTF-8 string format.

The receiver can use the string for presenting the source of text. This method is on the RTP level described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

The inline coding of the source of text is applied in the data stream itself, and an RTP mixer function is used for coordinating the sources of text into one RTP stream.

Information uniquely identifying each user in the multi-party session is placed as the parameter value "n" in the T.140 application

protocol function with the function code "c". The identifier shall thus be formatted like this: SOS c n ST, where SOS and ST are coded as specified in ITU-T T.140 [T140]. The "c" is the letter "c". The n parameter value is a string uniquely identifying the source. This parameter shall be kept short so that it can be repeated in the transmission without concerns for network load.

A receiving endpoint is supposed to separate text items from the different sources and identify and display them accordingly.

The conference server need to be allowed to decrypt/encrypt the packet payload in order to check the source and repack the text.

Pros:

If loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103]stream. (normally primary and two redundant levels.

This method can be implemented with most RTP implementations.

Transmitted text can also be used with other transports than RTP

Cons:

The method implies a moderate load by the need to insert the source often in the stream.

If more consecutive packet loss than the number of generations of redundant data appears, it is not possible to deduct the source of the totally lost data.

The mixer needs to be able to generate suitable and unique source identifications which are suitable as labels for the sources.

Requires an extension on the ITU-T T.140 standard, best made by the ITU.

The conference server need to be allowed to decrypt/encrypt the packet payload.

4.1.7. Mixing for multi-party unaware user agents

Multi-party real-time text contents can be transmitted to multi-party unaware user agents if source labelling and formatting of the text is performed by a mixer. This method has the limitations that the layout of the presentation and the format of source identification is

purely controlled by the mixer, and that only one source at a time is allowed to present in real-time. Other sources need to be stored temporarily waiting for an appropriate moment to switch the source of transmitted text. The mixer controls the switching of sources and inserts a source identifier in text format at the beginning of text after switch of source. The logic of the mixer to detect when a switch is appropriate should detect a number of places in text where a switch can be allowed, including new line, end of sentence, end of phrase, a period of inactivity, and a word separator after a long time of active transmission.

This method MAY be used when no support for multi-party awareness is detected in the receiving endpoint.The base for his method is described in RFC 7667, section 3.6.1 Media mixing mixer [RFC7667].

See Appendix A for an informative example of a procedure for presenting RTT to a conference-unaware UA.

Pros:

Can be transmitted to conference-unaware endpoints.

Can be used with other transports than RTP

Cons:

Does not allow full real-time presentation of more than one source at a time. Text from other sources will be delayed.

The only realistic presentation format is a style with the text from the different sources presented with a text label indicating source, and the text collected in a chat style presentation but with more frequent turn-taking.

Endpoints often have their own system for adding labels to the RTT presentation. In that case there will be two levels of labels in the presentation, one for the mixer and one for the sources.

If loss of more packets than can be recovered by the redundancy appears, it is not possible to detect which source was struck by the loss. It is also possible that a source switch occurred during the loss, and therefore a false indication of the source of text can be provided to the user after such loss.

Because of all these cons, this method is not recommended and MUST NOT be used as the main method, but only as the last resort for backwards interoperability with multi-party unaware endpoints.

The conference server need to be allowed to decrypt/encrypt the packet payload.

4.2. RTP-based bridging with RTT media contents not touched by the bridge

It may be desirable to send text in a multi-party setting in a way that allows the text stream contents to be distributed without being dealt with in detail in any central server. A number of such methods are described. However, when writing this specification, no one of these methods have a specified way of establishing the session by sdp.

4.2.1. RTP Translator sending one RTT stream per participant

Within the RTP session, text from each participant is transmitted from the RTP media translator in a separate RTP stream, thus using the same destination address/port combination, but separate RTP SSRC parameters and sequence number series as described in Section 7.1 and 7.2 of RTP RFC 3550 [RFC3550] about the Translator function. The source of the text in each RTP packet is identified by the SSRC parameter in the RTP packets, containing the SSRC of the initial source of text.

A receiving and presenting endpoint is supposed to separate text items from the different sources and identify and display them in a suitable way.

This method is described in RFC 7667, section 3.5.1 Relay-transport translator or 3.5.2 Media translator [RFC7667].

The identification of the source is made through theSSRC and the RTCP SDES CNAME and NAME packets as described in RTP[RFC3550].

Pros:

This method has moderate overhead. When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream(normally primary and two redundant levels).

More loss than what can be recovered, can be detected and the marker for text loss can be inserted in the correct stream.

It may be possible in some scenarios to keep the text encrypted through the Translator.

Cons:

There may be RTP implementations not supporting the Translator model.

This configuration is not supported by current media declarations in sdp. RFC 3264 [RFC3264]specifies in many places that one media description is supposed to describe just one RTP stream.

4.2.2. Distributing packets in an end-to-end encryption structure

In order to achieve end-to-end encryption, it is possible to let the packets from the sources just pass though a central distributor, and handle the security agreements between the participants. Specifications exist for a framework with this functionality for application on RTP based conferences in [I-D.ietf-perc-private-media-framework]. The RTP flow and mixing characteristics has similarities with the method described under "RTP Translator sending one RTT stream per participant" above. RFC 4103 RTP streams [RFC4103] would fit into the structure and it would provide a base for end-to-end encrypted rtt multi-party conferencing.

Pros:

Good security

Straightforward multi-party handling.

Cons:

Does not operate under the usual SIP central conferencing architecture.

Requires the participants to perform a lot of key handling.

Is work in progress when this is written.

4.2.3. Mesh of RTP endpoints

Text from all participants are transmitted directly to all others in one RTP session, without a central bridge. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

This method is described in RFC 7667, section 3.4 Point to multi-point using mesh [RFC7667].

Pros:

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried

in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels.

This method can be implemented with most RTP implementations.

Transmitted text can also be used with other transports than RTP

Cons:

This model is not described in IMS, NENA and EENA specifications, and does therefore not meet the requirements.

Requires a drastically increasing number of connections when the number of participants increase.

4.2.4. Multiple RTP sessions, one for each participant

Text from all participants are transmitted directly to all others in one RTP session each, without a central bridge. Each session is established with a separate media description in SDP. The sources of the text in each RTP packet are identified by the source network address and the SSRC.

Pros:

When loss of packets occur, it is possible to recover text from redundancy at loss of up to the number of redundancy levels carried in the RFC 4103 [RFC4103] stream. (normally primary and two redundant levels.

Complete loss of text can be indicated in the received stream.

This method can be implemented with most RTP implementations.

End-to-end encryption is achievable.

Cons:

This method is not described in IMS, NENA and ETSI specifications and does therefore not meet the requirements.

A lot of network resources are spent on setting up separate sessions for each participant.

4.3. RTT bridging in WebRTC

Within WebRTC, real-time text is specified to be carried in WebRTC data channels as specified in [I-D.ietf-mmusic-t140-usage-data-channel]. A few ways to handle multi-party RTT are mentioned briefly. They are repeated below.

4.3.1. RTT bridging in WebRTC with one data channel per source

A straightforward way to handle multi-party RTT is for the bridge to open one T.140 data channel per source towards the receiving participants.

The stream-id forms a unique stream identification.

The identification of the source is made through the Label property of the channel, and session information belonging to the source. The endpoint can compose a readable label for the presentation from this information.

Pros:

This is a straightforward solution.

Cons:

With a high number of participants, the overhead of establishing the high number of data channels required may be high.

4.3.2. RTT bridging in WebRTC with one common data channel

A way to handle multi-party RTT in WebRTC is for the bridge combine text from all sources into one data channel and insert the sources in the stream by a T.140 control code for source.

This method is described in a corresponding section for RTP transmission above in Section 4.1.6.

The identification of the source is made through insertion in the beginning of each text transmission from a source of a control code extension "c" followed by a string representing the source, framed by the control code start and end flags SOS and ST (See ITU-T T.140 [T140]).

A receiving endpoint is supposed to separate text items from the different sources and identify and display them in a suitable way.

The endpoint does not always display the source identification in the received text at the place where it is received, but has the information as a guide for planning the presentation of received text. A label corresponding to the source identification is presented when needed depending on the selected presentation style.

Pros:

This solution has relatively low overhead on session and network level

Cons:

This solution has higher overhead on the media contents level than the WebRTC solution above.

Standardisation of the new control code "c" in ITU-T T.140 [T140] is required.

The conference server need to be allowed to decrypt/encrypt the data channel contents.

5. Preferred multi-party RTT transport method

For RTP transport of RTT using RTP-mixer technology, one method for multi-party mixing and transport stand out as fulfilling the goals best and is therefore recommended. That is: "RTP Mixer with frequent transmission and indicating sources in the CSRC-list" in Section 4.1.3.

For RTP transport in separate streams or sessions, no current recommendation can be made. A bridging method in the process of standardisation with interesting characteristics is the end-to-end encryption model "perc" Section 4.2.2.

For WebRTC, one method is to prefer because of the simplicity. So, for WebRTC, the method to implement for multi-party RTT with multi-party aware parties when no other method is explicitly agreed between implementing parties is: "RTT bridging in WebRTC with one data channel per source" Section 4.3.1.

6. Session control of multi-party RTT sessions

General session control aspects for multi-party sessions are described in RFC 4575 [RFC4575] A Session Initiation Protocol (SIP) Event Package for Conference State, and RFC 4579 [RFC4579] Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents. The nomenclature of these specifications are used here.

The procedures for a multi-party aware model for RTT-transmission shall only be applied if a capability exchange for multi-party aware real-time text transmission has been completed and a supported method for multi-party real-time text transmission can be negotiated.

A method for detection of conference-awareness for centralized SIP conferencing in general is specified in RFC 4579 [RFC4579]. The focus sends the "isfocus" feature tag in a SIP Contact header. This causes the conference-aware endpoint to subscribe to conference notifications from the focus. The focus then sends notifications to the endpoint about entering and disappearing conference participants and their media capabilities. The information is carried XML-formatted in a 'conference-info' block in the notification according to RFC 4575 [RFC4575]. The mechanism is described in detail in RFC 4575 [RFC4575].

Before a conference media server starts sending multi-party RTT to an endpoint, a verification of its ability to handle multi-party RTT must be made. A decision on which mechanism to use for identifying text from the different participants must also be taken, implicitly or explicitly. These verifications and decisions can be done in a number of ways. The most apparent ways are specified here and their pros and cons described. One of the methods is selected to be the one to be used by implementations of the centralized conference model according to this specification.

6.1. Implicit RTT multi-party capability indication

Capability for RTT multi-party handling can be decided to be implicitly indicated by session control items.

The focus may implicitly indicate muti-party RTT capability by including the media child with value "text" in the RFC 4575 [RFC4575] conference-info provided in conference notifications.

An endpoint may implicitly indicate multi-party RTT capability by including the text media in the SDP in the session control transactions with the conference focus after the subscription to the conference has taken place.

The implicit RTT capability indication means for the focus that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

The implicit RTT capability indication means for the endpoint that it can handle multi-party RTT according to the preferred method indicated in the RTT multi-party methods section above.

If the focus detects that an endpoint implicitly declared RTT multi-party capability, it SHALL provide RTT according to the preferred method.

If the focus detects that the endpoint does not indicate any RTT multi-party capability, then it shall either provide RTT multi-party text in the way specified for conference-unaware endpoint above, or refuse to set up the session.

If the endpoint detects that the focus has implicitly declared RTT multi-party capability, it shall be prepared to present RTT in a multi-party fashion according to the preferred method.

Pros:

Acceptance of implicit multi-party capability implies that no standardisation of explicit RTT multi-party capability exchange is required.

Cons:

If other methods for multi-party RTT are to be used in the same implementation environment as the preferred ones, then capability exchange needs to be defined for them.

Cannot be used outside a strictly applied SIP central conference model.

6.2. RTT multi-party capability declared by SIP media-tags

Specifications for RTT multi-party capability declarations can be agreed for use as SIP media feature tags, to be exchanged during SIP call control operation according to the mechanisms in RFC 3840 [RFC3840] and RFC 3841 [RFC3841]. Capability for the RTT Multi-party capability is then indicated by the media feature tag "rtt-mix", with a set of possible values for the different possible methods.

The possible values in the list may for example be:

rtp-mixer

perc

rtp-mixer indicates capability for using the RTP-mixer based presentation of multi-party text.

perc indicates capability for using the perc based transmission of multi-party text.

Example: Contact: <sip:a2@beco.example.com>

;methods="INVITE,ACK,OPTIONS,BYE,CANCEL"

;+sip.rtt-mix="rtp-mixer"

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the media tag can be reduced to a single tag with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method for multi-party unaware participants can be used, or the session dropped.

If more than one text media section is included in SDP, all must be capable of using the declared RTT multi-party method.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can guide call routing to a suitable capable focus.

Cons:

Requires standardization and IANA registration.

Is not stream specific. If more than one text stream is specified, all must have the same type of multi-party capability.

Cannot be used in the WebRTC environment.

6.3. SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have the name "rtt-mix".

More than one attribute can be included in one media description.

The attribute can have a value. The value can for example be:

rtp-mixer

rtp-translator

perc

rtp-mixer indicates capability for using the RTP-mixer and CSRC-list based mixing of multi-party text.

rtp-translator indicates capability for using the RTP-translator based mixing

perc indicates capability for using the perc based transmission of multi-party text.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that endpoint.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used.

Example: a=rtt-mix:rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the attribute can be reduced to a single attribute with no list of values.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

Cannot guide SIP routing.

6.4. Simplified SDP media attribute for RTT multi-party capability indication

An attribute can be specified on media level, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The attribute can have the name "rtt-mix" with no value. It would be selected and used if only one method for multi-party rtt is brought forward from this specification, and the other suppressed or found to be possible to negotiate in another way.

An offer-answer exchange should take place and if both parties specify "rtt-mix" capability, the selected mixing method shall be used.

When no common method is declared, then only the fallback method for multi-party unaware endpoints can be used, or the session not accepted for multi-party use.

Example: a=rtt-mix

Pros:

Provides a clear decision method.

Very simple syntax and semantics.

Can be used on specific text media.

Could possibly be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

If another RTT mixing method is also specified in the future, then that method may also need to specify and register its own attribute, instead of if an attribute with a parameter value is used, when only an addition of a new possible value is needed.

Cannot guide SIP routing.

6.5. SDP format parameter for RTT multi-party capability indication

An FMTP format parameter can be specified for the RFC 4103 [RFC4103]media, to be used in text media SDP declarations for negotiating RTT multi-party capabilities. The parameter can have the name "rtt-mix", with one or more of its possible values.

The possible values in the list are:

rtp-mixer

perc

rtp-mixer indicates capability for using the RTP-mixer based mixing and presentation of multi-party text using the CSRC-list.

perc indicates capability for using the perc based transmission of multi-party text.

Example: a=fmtp 96 98/98/98 cps=30;rtt-mix=rtp-mixer

If, after evaluation of the alternatives in this specification, only one mixing method is selected to be brought to implementation, then the parameter can be reduced to a single parameter with no list of values.

An offer-answer exchange should take place and the common method selected by the answering party shall be used in the session with that UA.

When no common method is declared, then only the fallback method can be used, or the session denied.

Pros:

Provides a clear decision method.

Can be extended with new mixing methods.

Can be used on specific text media.

Can be used also for SDP-controlled WebRTC sessions with multiple streams in the same data channel.

Cons:

Requires standardization and IANA registration.

May cause interop problems with current RFC4103 [RFC4103] implementations not expecting a new fmtp-parameter.

Cannot guide SIP routing.

6.6. Preferred capability declaration method.

The preferred capability declaration method is the one with a simplified SDP attribute "a=rtt-mix" Section 6.4 because it is straightforward and partially usable also for WebRTC.

7. Identification of the source of text

The main way to identify the source of text in the RTP based solution is by the SSRC of the sending participant. In the RTP-mixer solution, this SSRC is included in the CSRC list of the transmitted packets. Further identification that may be needed for better labeling of received text may be achieved from a number of sources. It may be the RTCP SDES CNAME and NAME reports, and in the conference notification data (RFC 4575) [RFC4575].

As soon as a new member is added to the RTP session, its characteristics should be transmitted in RTCP SDES CNAME and NAME reports according to section 6.5 in RFC 3550 [RFC3550]. The information about the participant should also be included in the conference data including the text media member in a notification according to RFC 4575 [RFC4575].

The RTCP SDES report, SHOULD contain identification of the source represented by the SSRC/CSRC identifier. This identification MUST contain the CNAME field and MAY contain the NAME field and other defined fields of the SDES report.

A focus UA SHOULD primarily convey SDES information received from the sources of the session members. When such information is not available, the focus UA SHOULD compose SSRC/CSRC, CNAME and NAME information from available information from the SIP session with the participant.

8. Presentation of multi-party text

All session participants with RTP based transport MUST observe the SSRC/CSRC field of incoming text RTP packets, and make note of what source they came from in order to be able to present text in a way that makes it easy to read text from each participant in a session, and get information about the source of the text.

In the WebRTC case, the Label parameter and other provided endpoint information should be used for the same purpose.

8.1. Associating identities with text streams

A source identity SHOULD be composed from available information sources and displayed together with the text as indicated in ITU-T T.140 Appendix[T140].

The source identity should primarily be the NAME field from incoming SDES packets. If this information is not available, and the session is a two-party session, then the T.140 source identity SHOULD be composed from the SIP session participant information. For multi-party sessions the source identity may be composed by local information if sufficient information is not available in the session.

Applications may abbreviate the presented source identity to a suitable form for the available display.

Applications may also replace received source information with internally used nicknames.

8.2. Presentation details for multi-party aware endpoints.

The multi-party aware endpoint should after any action for recovery of data from lost packets, separate the incoming streams and present them according to the style that the receiving application supports and the user has selected. The decisions taken for presentation of the multi-party interchange shall be purely on the receiving side. The sending application must not insert any item in the stream to influence presentation that is not requested by the sending participant.

8.2.1. Bubble style presentation

One often used style is to present real-time text in chunks in readable bubbles identified by labels containing names of sources. Bubbles are placed in one column in the presentation area and are closed and moved upwards in the presentation area after certain items or events, when there is also newer text from another source that would go into a new bubble. The text items that allows bubble closing are any character closing a phrase or sentence followed by a space or a timeout of a suitable time (about 10 seconds).

Real-time active text sent from the local user should be presented in a separate area. When there is a reason to close a bubble from the local user, the bubble should be placed above all real-time active bubbles, so that the time order that real-time text entries were completed is visible.

Scrolling is usually provided for viewing of recent or older text. When scrolling is done to an earlier point in the text, the presentation shall not move the scroll position by new received text. It must be the decision of the local user to return to automatic viewing of latest text actions. It may be useful with an indication that there is new text to read after scrolling to an earlier position has been activated.

The presentation area may become too small to present all text in all real-time active bubbles. Various techniques can be applied to provide a good overview and good reading opportunity even in such situations. The active real-time bubble may have a limited number of lines and if their contents need more lines, then a scrolling opportunity within the real-time active bubble is provided. Another method can be to only show the label and the last line of the active real-time bubble contents, and make it possible to expand or compress the bubble presentation between full view and one line view.

Erasures require special consideration. Erasure within a real-time active bubble is straightforward. But if erasure from one participant affects the last character before a bubble, the whole previous bubble becomes the actual bubble for real-time action by that participant and is placed below all other bubbles in the presentation area. If the border between bubbles was caused by the CRLF characters (instead of the normal "Line Separator"), only one erasure action is required to erase this bubble border. When a bubble is closed, it is moved up, above all real-time active bubbles.

A three-party view is shown in this example .

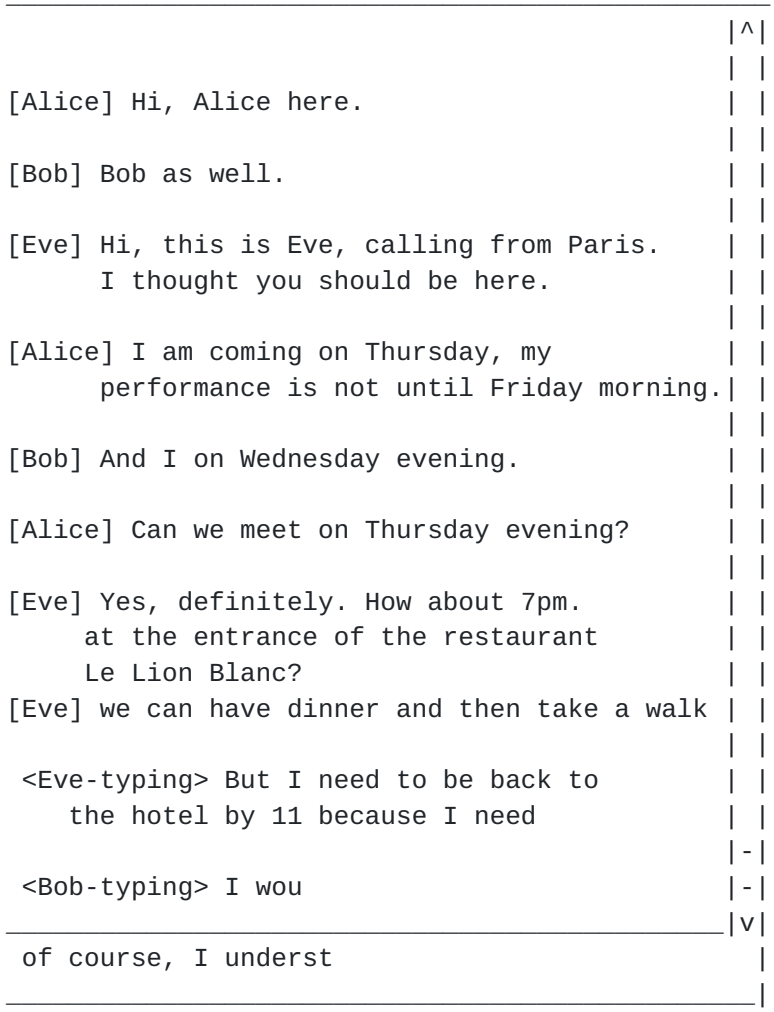

Figure 1: Example of a three-party call presented in the bubble style.

Figure 1: Three-party call with bubble style.

8.2.2. Other presentation styles

Other presentation styles than the bubble style may be arranged and appreciated by the users. In a video conference one way may be to have a real-time text area below the video view of each participant. Another view may be to provide one column in a presentation area for each participant and place the text entries in a relative vertical position corresponding to when text entry in them was completed. The labels can then be placed in the column header. The considerations for ending and moving and erasure of entered text discussed above for the bubble style are valid also for these styles.

This figure shows how a coordinated column view MAY be presented.

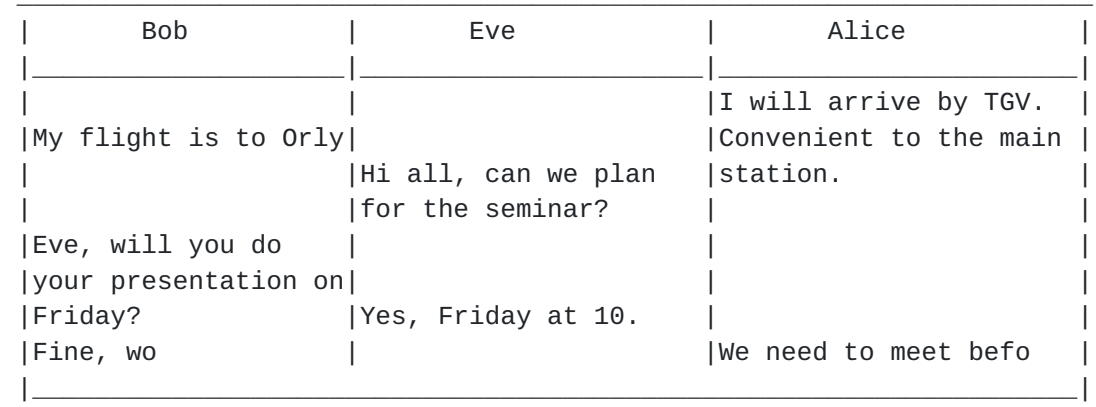

Figure 2: A coordinated column-view of a three-party session with entries ordered in approximate time-order.

9. Presentation details for multi-party unaware endpoints.

Multi-party unaware UA:s are prepared only for presentation of two sources of text, the local user and a remote user. If mixing for multi-party unaware endpoints is to be supported, in order to enable some multi-party communication with such UA, the mixer need to plan the presentation and insert labels and line breaks before lables. Many limitations appear for this presentation mode, and it must be seen as a fallback and a last resort. A realistic alternative is to not allow multi-party sessions with multi-party unaware endpoints.

See Appendix A for an informative example of a procedure for presenting RTT to a conference-unaware endpoint.

10. Security Considerations

The security considerations valid for RFC 4103 [RFC4103] and RFC 3550 [RFC3550] are valid also for the multi-party sessions with text.

11. IANA Considerations

The items for indication and negotiation of capability for multi-party rtt should be registered with IANA in the specifications where they are specified in detail.

12. Congestion considerations

The congestion considerations described in RFC 4103 [RFC4103] are valid also for the recommended RTP-based multi-party use of the real-time text transport. A risk for congestion may appear if a number of

conference participants are active transmitting text simultaneously, because the recommended RTP-based multi-party transmission method does not allow multiple sources of text to contribute to the same packet.

In situations of risk for congestion, the Focus UA MAY combine packets from the same source to increase the transmission interval per source up to one second. Local conference policy in the Focus UA may be used to decide which streams shall be selected for such transmission frequency reduction.

13. Acknowledgements

Arnoud van Wijk for contributions to an earlier, expired draft of this memo.

14. Changes

14.1. Changes from draft-hellstrom-mmusic-multi-party-rtt-02 to draft-avtcore-multi-party-rtt-solutions-00

Add discussion about switching performance, as discussed in avtcore on March 13.

Added that a decrease of transmission interval to 100 ms increases switching performance by a factor 3, but still not sufficient.

Added that the CSRC-list method also uses 100 milliseconds transmission interval.

Added the method with multiple primary text in each packet.

Added the timestamp-based method for rtp-mixing proposed by James Hamlin on March 14.

Corrected the chat style presentation example picture. Delete a few "[mix]".

14.2. Changes from version draft-hellstrom-mmusic-multi-party-rtt-01 to -02

Change from a general overview to overview with clear recommendations.

Splits text coordination methods in three groups.

Recommends rtt-mixer with sources in CSRC-list but referenes to its spec for details.

Shortened Appendix with conference-unaware example.

Cleaned up preferences.

Inserted pictures of screen-views.

15. References

15.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

15.2. Informative References

[EN301549] ETSI, "EN 301 549. Accessibility requirements for ICT products and services", November 2019.

[I-D.hellstrom-avtcore-multi-party-rtt-source] Hellstrom, G., "Indicating source of multi-party Real-time text", draft-hellstrom-avtcore-multi-party-rtt-source-02 (work in progress), March 2020.

[I-D.ietf-mmusic-t140-usage-data-channel] Holmberg, C., "T.140 Real-time Text Conversation over WebRTC Data Channels", draft-ietf-mmusic-t140-usage-data-channel-12 (work in progress), March 2020.

[I-D.ietf-perc-private-media-framework] Jones, P., Benham, D., and C. Groves, "A Solution Framework for Private Media in Privacy Enhanced RTP Conferencing (PERC)", draft-ietf-perc-private-media-framework-12 (work in progress), June 2019.

[NENAi3] NENA, "NENA-STA-010.2-2016. Detailed Functional and Interface Standards for the NENA i3 Solution", October 2016.

[RFC2198] Perkins, C., Kouvelas, I., Hodson, O., Hardman, V., Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-Parisis, "RTP Payload for Redundant Audio Data", RFC 2198, DOI 10.17487/RFC2198, September 1997, <https://www.rfc-editor.org/info/rfc2198>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <https://www.rfc-editor.org/info/rfc3264>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <https://www.rfc-editor.org/info/rfc3550>.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, DOI 10.17487/RFC3840, August 2004, <https://www.rfc-editor.org/info/rfc3840>.

[RFC3841] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Caller Preferences for the Session Initiation Protocol (SIP)", RFC 3841, DOI 10.17487/RFC3841, August 2004, <https://www.rfc-editor.org/info/rfc3841>.

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, DOI 10.17487/RFC4103, June 2005, <https://www.rfc-editor.org/info/rfc4103>.

[RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, DOI 10.17487/RFC4353, February 2006, <https://www.rfc-editor.org/info/rfc4353>.

[RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, Ed., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, DOI 10.17487/RFC4575, August 2006, <https://www.rfc-editor.org/info/rfc4575>.

[RFC4579] Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, DOI 10.17487/RFC4579, August 2006, <https://www.rfc-editor.org/info/rfc4579>.

[RFC4597] Even, R. and N. Ismail, "Conferencing Scenarios", RFC 4597, DOI 10.17487/RFC4597, August 2006, <https://www.rfc-editor.org/info/rfc4597>.

[RFC5194] van Wijk, A., Ed. and G. Gybels, Ed., "Framework for Real-Time Text over IP Using the Session Initiation Protocol (SIP)", RFC 5194, DOI 10.17487/RFC5194, June 2008, <https://www.rfc-editor.org/info/rfc5194>.

[RFC6443] Rosen, B., Schulzrinne, H., Polk, J., and A. Newton, "Framework for Emergency Calling Using Internet Multimedia", RFC 6443, DOI 10.17487/RFC6443, December 2011, <https://www.rfc-editor.org/info/rfc6443>.

[RFC6881] Rosen, B. and J. Polk, "Best Current Practice for Communications Services in Support of Emergency Calling", BCP 181, RFC 6881, DOI 10.17487/RFC6881, March 2013, <https://www.rfc-editor.org/info/rfc6881>.

[RFC7667] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 7667, DOI 10.17487/RFC7667, November 2015, <https://www.rfc-editor.org/info/rfc7667>.

[T140] ITU-T, "Recommendation ITU-T T.140 (02/1998), Protocol for multimedia application text conversation", February 1998.

[T140ad1] ITU-T, "Recommendation ITU-T.140 Addendum 1 - (02/2000), Protocol for multimedia application text conversation", February 2000.

[TS103479] ETSI, "TS 103 479. Emergency communications (EMTEL); Core elements for network independent access to emergency services", December 2019.

[TS22173] 3GPP, "IP Multimedia Core Network Subsystem (IMS) Multimedia Telephony Service and supplementary services; Stage 1", 3GPP TS 22.173 17.1.0, December 2019.

[TS24147] 3GPP, "Conferencing using the IP Multimedia (IM) Core Network (CN) subsystem; Stage 3", 3GPP TS 24.147 16.0.0, December 2019.

Appendix A. Mixing for a multi-party unaware endpoint

This informational appendix describes media mixer procedures for a multi-party conference server to format real-time text from a number of participants into one single text stream to a participant with an

endpoint that has no features for multi-party text display. The procedures are intended for implementations using ITU-T T.140 [T.140] for the real-time text coding and presentation.

A.1. Short description

The media mixer procedures described here are intended to make real-time text from a number of call participants be coordinated into one text stream to an endpoint originally intended for two-party calls. A conference server is supposed to apply the procedures.

The procedures may also be applied on an endpoint for display of multiple streams of real-time text in one area.

The intention is that text from each participant shall be displayed in suitable sections so that they are easy to read, and text from one active participant at a time is sent and displayed in real-time. The receiving terminal is assumed to have one display area for received text. The display is arranged by this procedure in a text chat style, with a name label in front of each text section where switch of source of the text has taken place.

When more than one participant transmits text at the same time, the text from only one of them is transmitted directly to the receiving terminals. Text from the other participants is stored in buffers in the conference server for transmission at a later time, when a suitable situation for switch of current transmitter can take place.

A.2. Functionality goals and drawbacks

The procedures are intended to make best efforts to present a multi-party text conversation on an endpoint that has no awareness of multi-party calls. There are some obvious drawbacks, and an endpoint designed with multi-party awareness will be able to present multi-party call contents in a more flexible way. Only two parties at a time will be allowed to display text added in real-time, while the other parties' produced text will need to be stored in the multi-party server for a moment awaiting a suitable occasion to be displayed. There are also some cases of erasure that will not be performed on the target text but only indicated in another way. Even with these drawbacks, the procedure provides an opportunity to display text from more than two parties in a smooth and readable way.

This specification does not introduce any new protocol element, and does not rely on anything else than basic two-party terminal functionality with presentation level according to ITU-T T.140 [T.140]. It is a description of a best current practice for mixing

and presentation of the real-time text component in multi-party calls with terminals without multi-party awareness.

The procedures are applicable to scenarios, when the conference focus and a User Agent have not gone through any successfully completed negotiation about multi-party awareness for the real-time text medium neither on the transport level, nor on the presentation level.

A.3. Definitions

Active participant: Any user sending text, or being in a pending period.

BOM Byte-Order-Mark, the Unicode character FEFF in UCS-16.

Buffer: A buffer intended for unsent text collected per participant.

Contributing participants: The participants selected to contribute to the text stream sent to the recipients.

By default all participants except the recipient are contributing participants for transmission to the recipient.

Current participant: The participant for whom text currently is transmitted to the recipient in real time.

Current Recipients: By default all participants.

Display Counter: A counter for the number of displayable characters in a participant's buffer or in the current entry. Used for controlling how far erasure may be performed.

Erasure replacement: A character to be displayed when an erasure was done, but the text to erase is not reachable on the multi-party display. Default 'X'.

Message delimiter: Character(s) forming the end of an imagined message. A configurable set of alternatives, consisting by default of: Line Separator, Paragraph Separator, CR, CRLF, LF.

Pending period: A configurable time period of inactivity from a participant, by default set to 7 seconds after each reception of characters from that participant, evaluated as current time minus time stamp of latest entered character.

Sentence delimiter: Characters forming end of sentence: A configurable set of alternatives, by default consisting of: dot

'.', question mark '?' and exclamation mark '!' followed by a space.

Label: A readable unique name for a participant, created by the server from a suitable source related to the participant, surrounded by the Label delimiters. The label should have a settable maximum length, with 12 being the default.

Label delimiters A configurable set of characters at the edges of the Label, by default being a left bracket [at the leading edge and a closing bracket] followed by a space at the trailing edge.

Line Separator Unicode UCS-16 2028. Used to request NewLine in Real-Time Text.

Maximum waiting time: The maximum time any participant's text shall be allowed to wait for transmission, by default set to 20 seconds.

Recipient: The terminal receiving the mixed text stream.

SGR Select Graphic Rendition, a control code to specify colours etc.

Switch Reason: A set of reasons to switch Current Participant, consisting of the following

-Waiting time higher for any other participant than the current participant combined with any of the following states:

-A message delimiter was the latest transmitted item

-A sentence delimiter was the latest transmitted item

-A Pending Period has expired and still no text has been transmitted

-The Maximum Waiting time has expired followed by a Word Delimiter or an expired Time Extension.

Waiting time: The time the first character in queue for transmission from a participant has been waiting in a buffer for transmission. The granularity shall be 0.3 Seconds or finer.

Word delimiter: Character forming end of word: space

Time extension: A configurable short extension time allowed after the Maximum waiting time during which a suitable moment for

switching Current Participant is awaited, by default set to 7 seconds.

A.4. Presentation level procedures

The conference server applies these mixing procedures to text transmitted to call participants who have not gone through a completed negotiation for conference awareness in real-time text presentation.

All the participants and the conference server use real-time text conversation presentation coding according to ITU-T T.140 [T.140]. A consequence is that real-time text transmissions are UTF-8 coded, with control codes selected from ISO 6429 [ISO 6429].

The description is from the conference server point of view.

A.4.1. Structure

The real-time text mixer structure described here is supposed to be placed in the media path so that it is implemented with one mixer per recipient. A mixer contains buffers for temporary storage of text intended for the recipient. Each mixer has one buffer for each contributing participant. A set of status variables is maintained per buffer and is used in the mixer actions. The mixer logic decides for each moment which participant's buffer content is to be sent on to the recipient. By default, the recipient does not contribute text to its own mixer. Text transmitted by a participant is usually displayed locally and it will only cause confusion if it appears also in received text.

A.4.2. Action on reception

This description of the mixer is valid per recipient.

Text from each contributing participant is checked for a set of characteristics on reception.

Delete BOM: BOM characters are deleted.

Insert in buffer: Resulting text is put into the contributing participant's buffer in the receiving participant's mixer.

Maintain a display counter: For each text character that will take a position on the receiving display, a Display Counter for each participant is increased by one.

There is one T.140 real-time text item that consists of two characters, but is regarded to be a unit and therefore increase the Display Counter with one only.That is CRLF.

Furthermore, the following control codes are regarded units that shall not take any position on the receiving display and shall therefore not increase the Display Counter:

0098 string 009C (SOS-ST strings)

ESC 0061 (INT)

009B Ps 006D (the SGR code, with special handling described below)

BEL (Alert in session)

See the section on control codes below for details.

Combination characters: Also note that it is possible to use combination characters in Unicode. Such combination characters contain more than one character part. They shall only increase the Display Counter with one. The combination characters mainly have components in the series 0300 - 0361 and 20D0 - 20E1.

Erasure: If the control code for erasure, BS, is received, the following shall be done: If the Display Counter is 0, an Erasure Replacement character, by default being "X" is inserted in the buffer instead of the erasure, to mark that erasure was intended in earlier transmitted entries. (this matches traditional habits in real-time text when participants sometimes type XXX to indicate erasure they do not bother to make explicit). If the Display Counter is >0, then the counter is reduced by one, and the erasure control code BS put into the buffer.

Initial action in the session: BOM shall be sent initially to the recipients in the beginning of the session.

Maintaining a waiting time per participant: The time that text has been in the buffer is maintained as the waiting time for each buffer. A granularity of 0.3 seconds is sufficient.

Storing time of reception for each character: Each character that is stored in a buffer shall be assigned with a time stamp indicating its time of reception. A granularity of 0.3 seconds is sufficient. This time stamp is used for calculation of idle time and waiting time in the evaluation of switch reasons.

Initial assignment of the Current Participant: The first contributing participant to send text in the session is assigned to be the Current Participant.

Actions on assignment of a Current Participant: When a participant becomes the Current Participant, the following initial actions shall be performed:

1. Scanning transmissions and timers for a Switch Reason is inactivated.

2. The Current Recipients are set so that all transmissions go to the new set of Current Recipients (See definition).

3. A Line Separator is transmitted if the switch reason was any other than a message delimiter.

4. The Label is transmitted

5. Any stored SGR code is transmitted

6. Scanning transmissions and timers for a Switch Reason is activated.

7. Text in the buffer is transmitted, recalculating and setting the waiting time for each transmitted character based on the time of reception of next character in the buffer. If a switch occurs during transmission from the buffer, the remaining buffer contents is maintained and transmission can continue next time this transmitter becomes the current participant. Any text entered into the buffer for the current participant is after that sent to the recipient until a Switch Reason occurs.

Actions on transmission and during the session: Transmissions are checked for control codes to act on at transmission as described below in the section about handling of control codes and such actions are performed. When the scanning of transmission and timers for a Switch Reason is active, the timers and the transmission to the recipient is analyzed for detection if a Switch Reason has occurred. See the definition of Switch Reasons for details.

Actions when a Switch Reason has occurred: If a Switch Reason has occurred, then the following actions shall be performed:

1. The Display Counter of the Current Participant is set to zero

2. If there is an SGR code stored for the Current Participant, a reset of SGR shall be sent by the sequence SGR 0 [009B 0000 006D].

3. A participant with the longest waiting time is assigned to be the Current Participant, and the procedure for assignment of a Current Participant described above is performed.

Handling of Control codes: The following control codes are specified by ITU-T T.140. Some of them require consideration in the conference server. Note that the codes presented here are expressed in UCS-16, while transmission is made in UTF-8 transform of these codes. Other sections specify procedures for handling of specific control codes in the conference server.

BEL 0007 Bell, provides for alerting during an active session.

BS 0008 Back Space, erases the last entered character.

NEW LINE 2028 Line separator.

CR LF 000D 000A A supported, but not preferred way of requesting a new line.

INT ESC 0061 Interrupt (used to initiate mode negotiation procedure).

SGR 009B Ps 006D Select graphic rendition. Ps is rendition parameters specified in ISO 6429.

SOS 0098 Start of string, used as a general protocol element introducer, followed by a maximum 256 bytes string.

ST 009C String terminator, end of SOS string.

ESC 001B Escape - used in control strings.

Byte order mark FEFF Zero width, no break space, used for synchronization.

Missing text mark FFFD Replacement character, marks place in stream of possible text loss.

Code for message border, useful, but not mentioned in T.140: New Message 2029 Paragraph separator

Handling of Graphic Rendition SGR: The following procedure shall be followed in order to let the participants control the graphic rendition of their entries without disturbing other participants'

graphic rendition. The text stream sent to a recipient shall be monitored for the SGR sequence. The latest conveyed SGR sequence is also stored as a status variable for the recipient. If the SGR 0 code initiated from the current participant is transmitted, the SGR storage shall be cleared.

A.5. Display examples

The following pictures are examples of the view on a participant's display.

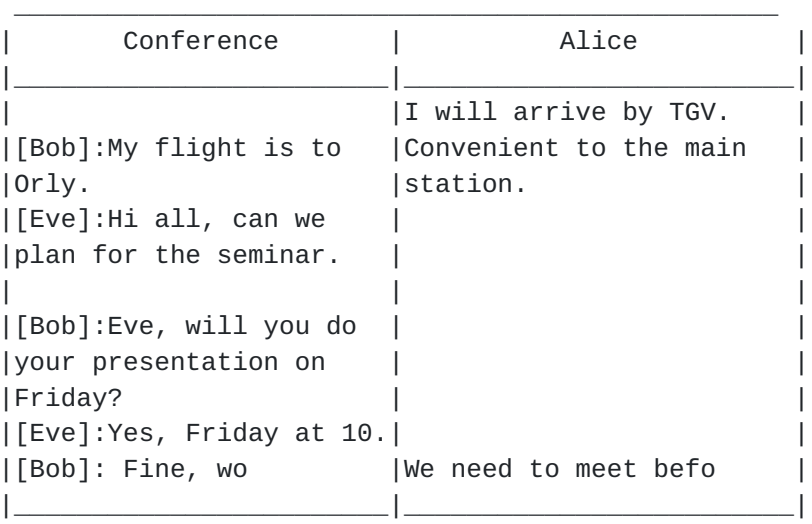

Figure A1 : Alice who has a conference-unaware client is receiving the multi-party real-time text in a single-stream. This figure shows how a coordinated column view MAY be presented on Alice's device.

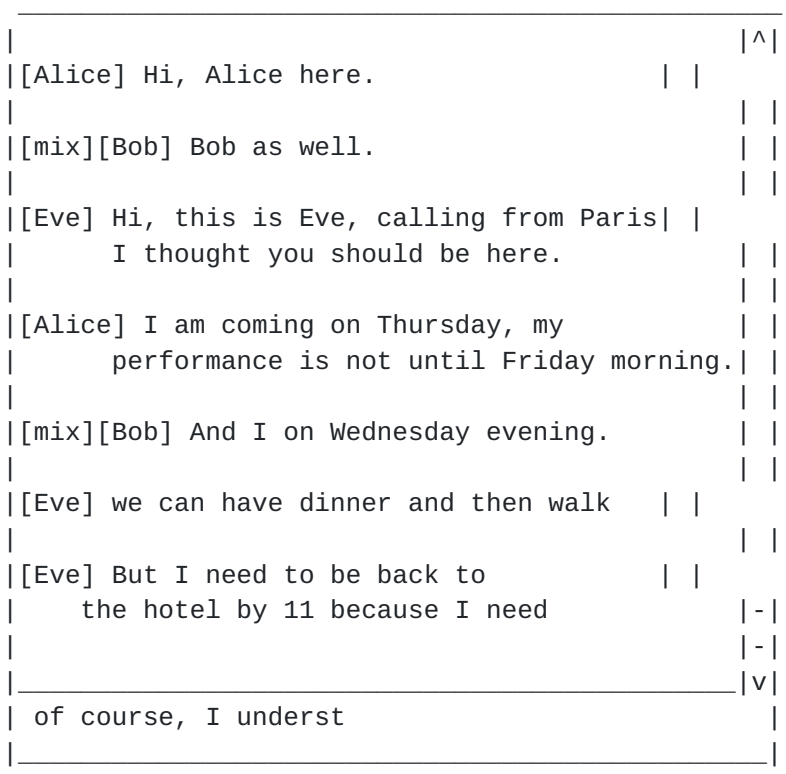

Figure A2 shows a conference with a real-time multi-party text view. Bob's text is buffered until a Current switch reason.

A.6. References for this Appendix

[T.140] ITU-T T.140 Application protocol, text conversation (including amendment 1.)

[RFC 4103] IETF RFC 4103 RTP Payload for text conversation

[RTP] IETF RFC 3550 RTP: A Transport Protocol for Real-Time Applications.

[RFC 4579] IETF RFC 4579 SIP Call Control : Conferencing for user agents.

[ISO 6429] ISO 6429 Control functions for coded character sets.

[UTF-8] IETF RFC 3629 UTF-8, a transformation format of ISO 10646

[Unicode] The Unicode Consortium, "The Unicode Standard ; Version 4.0.

[ISO 10646-1] ISO 10646 Universal multiple-octet coded character set (UCS)

[UCS-16] See ISO 10646-1

A.7. Acknowledgement for the appendix

This appendix was developed with funding in part from the National Institute on Disability and Rehabilitation Research, U.S. Department of Education, RERC on Telecommunications Access, grant # H133E090001. However, the contents do not necessarily represent the policy of the Department of Education, and you should not assume endorsement by the Federal Government.

Author's Address

Gunnar Hellstrom
Omnitor
Esplanaden 30
Vendelso SE-136 70
SE

Phone: +46 708 204 288
Email: gunnar.hellstrom@omnitor.se