Stream: Internet Engineering Task Force (IETF)
RFC: 9071
Updates: 4103
Category: Standards Track
Published: July 2021
ISSN: 2070-1721
Author: G. Hellström
GHAccess

RFC 9071
RTP-Mixer Formatting of Multiparty Real-Time Text

Abstract

This document provides enhancements of real-time text (as specified in RFC 4103) suitable for mixing in a centralized conference model, enabling source identification and rapidly interleaved transmission of text from different sources. The intended use is for real-time text mixers and participant endpoints capable of providing an efficient presentation or other treatment of a multiparty real-time text session. The specified mechanism builds on the standard use of the Contributing Source (CSRC) list in the Real-time Transport Protocol (RTP) packet for source identification. The method makes use of the same "text/t140" and "text/red" formats as for two-party sessions.

Solutions using multiple RTP streams in the same RTP session are briefly mentioned, as they could have some benefits over the RTP-mixer model. The RTP-mixer model was selected to be used for the fully specified solution in this document because it can be applied to a wide range of existing RTP implementations.

A capability exchange is specified so that it can be verified that a mixer and a participant can handle the multiparty-coded real-time text stream using the RTP-mixer method. The capability is indicated by the use of a Session Description Protocol (SDP) (RFC 8866) media attribute, "rtt-mixer".

This document updates RFC 4103 ("RTP Payload for Text Conversation").

A specification for how a mixer can format text for the case when the endpoint is not multiparty aware is also provided.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9071.

Copyright Notice

Table of Contents

1. Introduction

"RTP Payload for Text Conversation" [RFC4103] specifies the use of the Real-time Transport Protocol (RTP) [RFC3550] for transmission of real-time text (often called RTT) and the "text/t140" format. It also specifies a redundancy format, "text/red", for increased robustness. The "text/red" format is registered in [RFC4102].

Real-time text is usually provided together with audio and sometimes with video in conversational sessions.

A requirement related to multiparty sessions from the presentation-level standard T.140 [T140] for real-time text is as follows:

> The display of text from the members of the conversation should 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.

Another requirement is that the mixing procedure must not introduce delays in the text streams that could be perceived as disruptive to the real-time experience of the receiving users.

The use of real-time text is increasing, and specifically, use in emergency calls is increasing. Emergency call use requires multiparty mixing, because it is common that one agent needs to transfer the call to another specialized agent but is obliged to stay on the call to at least verify that the transfer was successful. Mixer implementations for RFC 4103 ("RTP Payload for Text Conversation") can use traditional RTP functions (RFC 3550) for mixing and source identification, but the performance of the mixer when giving turns for the different sources to transmit is limited when using the default transmission characteristics with redundancy.

The redundancy scheme described in [RFC4103] enables efficient transmission of earlier transmitted redundant text in packets together with new text. However, the redundancy header format has no source indicators for the redundant transmissions. The redundant parts in a packet must therefore be from the same source as the new text. The recommended transmission is one new and two redundant generations of text (T140blocks) in each packet, and the recommended transmission interval for two-party use is 300 ms.

Real-time text mixers for multiparty sessions need to include the source with each transmitted group of text from a conference participant so that the text can be transmitted interleaved with text groups from different sources at the rate at which they are created. This enables the text groups to be presented by endpoints in a suitable grouping with other text from the same source.

The presentation can then be arranged so that text from different sources can be presented in real time and easily read. At the same time, it is possible for a reading user to perceive approximately when the text was created in real time by the different parties. The transmission and mixing are intended to be done in a general way, so that presentation can be arranged in a layout decided upon by the receiving endpoint.

Existing implementations of RFC 4103 in endpoints that do not implement the updates specified in this document cannot be expected to properly present real-time text mixed for multiparty-aware endpoints.

A negotiation mechanism is therefore needed to verify if the parties (1) are able to handle a common method for multiparty transmissions and (2) can agree on using that method.

A fallback mixing procedure is also needed for cases when the negotiation result indicates that a receiving endpoint is not capable of handling the mixed format. Multiparty-unaware endpoints would possibly otherwise present all received multiparty mixed text as if it came from the same source regardless of any accompanying source indication coded in fields in the packet. Or, they may have other undesirable ways of acting on the multiparty content. The fallback method is called the mixing procedure for multiparty-unaware endpoints. The fallback method is naturally not expected to meet all performance requirements placed on the mixing procedure for multiparty-aware endpoints.

This document updates [RFC4103] by introducing an attribute for declaring support of the RTP-mixer-based multiparty-mixing case and rules for source indications and interleaving of text from different sources.

1.1. Terminology

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

The terms "Source Description" (SDES), "Canonical Name" (CNAME), "Name" (NAME), "Synchronization Source" (SSRC), "Contributing Source" (CSRC), "CSRC list", "CSRC count" (CC), "RTP Control Protocol" (RTCP), and "RTP mixer" are defined in [RFC3550].

"real-time text" (RTT) is text transmitted instantly as it is typed or created. Recipients can immediately read the message while it is being written, without waiting.

The term "T140block" is defined in [RFC4103] to contain one or more T.140 code elements.

"TTY" stands for a textphone type used in North America.

Web Real-Time Communication (WebRTC) is specified by the World Wide Web Consortium (W3C) and the IETF. See [RFC8825].

"DTLS-SRTP" is a Datagram Transport Layer Security (DTLS) extension for use with the Secure Real-time Transport Protocol / Secure Real-time Transport Control Protocol (SRTP/SRTCP) as specified in [RFC5764].

The term "multiparty aware" describes an endpoint that (1) receives real-time text from multiple sources through a common conference mixer, (2) is able to present the text in real time, separated by source, and (3) presents the text so that a user can get an impression of the approximate relative timing of text from different parties.

The term "multiparty unaware" describes an endpoint that cannot itself separate text from different sources when the text is received through a common conference mixer.

1.2. Main Method, Fallback Method, and Considered Alternatives

A number of alternatives were considered when searching for an efficient and easily implemented multiparty method for real-time text. This section briefly explains a few of them.

Multiple RTP streams, one per participant:

One RTP stream per source would be sent in the same RTP session with the "text/red" format. From some points of view, the use of multiple RTP streams, one for each source, sent in the same RTP session would be efficient and would use exactly the same packet format as [RFC4103] and the same payload type. A couple of relevant scenarios using multiple RTP streams are specified in "RTP Topologies" [RFC7667]. One possibility of special interest is the Selective Forwarding Middlebox (SFM) topology specified in Section 3.7 of [RFC7667], which could enable end-to-end encryption. In contrast to audio and video, real-time text is only transmitted when the users actually transmit information. Thus, an SFM solution would not need to exclude any party from transmission under normal conditions. In order to allow the mixer to convey the packets with the payload preserved and encrypted, an SFM solution would need to act on some specific characteristics of the "text/red" format. The redundancy headers are part of the payload, so the receiver would need to just assume that the payload type number in the redundancy header is for "text/t140". The characters per second ("cps") parameter would need to act per stream. The relationship between the SSRC and the source would need to be conveyed in some specified way, e.g., in the CSRC. Recovery and loss detection would preferably be based on RTP sequence number gap detection. Thus, sequence number gaps in the incoming stream to the mixer would need to be reflected in the stream to the participant, with no new gaps created by the mixer. However, the RTP implementation in both mixers and endpoints needs to support multiple streams in the same RTP session in order to use this mechanism. To provide the best opportunities for deployment, it should be possible to upgrade existing endpoint solutions to be multiparty aware with a reasonable amount of effort. There is currently a lack of support for multi-stream RTP in certain implementations. This fact led to only brief mention of this solution in this document as an option for further study.

RTP-mixer-based method for multiparty-aware endpoints:
: The "text/red" format as defined in RFC 4102 and applied in RFC 4103 is sent with the RTP-mixer method indicating the source in the CSRC field. The "text/red" format with a "text/t140" payload in a single RTP stream can be sent when text is available from the call participants instead of at the regular 300 ms intervals. Transmission of packets with text from different sources can then be done smoothly while simultaneous transmission occurs as long as it is not limited by the maximum character rate "cps" value. With ten participants sending text simultaneously, the switching and transmission performance is good. With more simultaneously sending participants and with receivers at default capacity, there will be a noticeable jerkiness and delay in text presentation. The more participants who send text simultaneously, the more jerkiness will occur. Two seconds of jerkiness will be noticeable and slightly unpleasant, but it corresponds in time to what typing humans often cause by hesitating or changing position while typing. A benefit of this method is that no new packet format needs to be introduced and implemented. Since simultaneous typing by more than two parties is expected to be very rare -- as described in Section 1.3 -- this method can be used successfully with good performance. Recovery of text in the case of packet loss is based on analysis of timestamps of received redundancy versus earlier received text. Negotiation is based on a new SDP media attribute, "rtt-mixer". This method was selected to be the main method specified in this document.

Multiple sources per packet:
: A new "text" media subtype would be specified with up to 15 sources in each packet. The mechanism would make use of the RTP-mixer model specified in RTP [RFC3550]. The sources would be indicated in strict order in the CSRC list of the RTP packets. The CSRC list can have up to 15 members. Therefore, text from up to 15 sources can be included in each packet. Packets are normally sent at 300 ms intervals. The mean delay would be 150 ms. A new redundancy packet format would be specified. This method would result in good performance but would require standardization and implementation of new releases in the target technologies; these would take more time than desirable to complete. It was therefore not selected to be included in this document.

Mixing for multiparty-unaware endpoints:
: The presentation of text from multiple parties is prepared by the mixer in one single stream. It is desirable to have a method that does not require any modifications in existing user devices implementing RFC 4103 for real-time text without explicit support of multiparty sessions. This is made possible by having the mixer insert a new line and a text-formatted source label before each switch of text source in the stream. Switching the source can only be done in places in the text where it does not disturb the perception of the contents. Text from only one source at a time can be presented in real time. The delay will therefore vary. In calls where parties take turns properly by ending their entries with a new line, the limitations will have limited influence on the user experience. When only two parties send text, these two will see the text in real time with no delay. Although this method also has other limitations, it is included in this document as a fallback method.

Real-time text transport in WebRTC:
: [RFC8865] specifies how the WebRTC data channel can be used to transport real-time text. That specification contains a section briefly describing its use in multiparty sessions. The focus of this document is RTP transport. Therefore, even if the WebRTC transport provides good multiparty performance, it is only mentioned in this document in relation to providing gateways with multiparty capabilities between RTP and WebRTC technologies.

1.3. Intended Application

The method for multiparty real-time text specified in this document is primarily intended for use in transmissions between mixers and endpoints in centralized mixing configurations. It is also applicable between mixers. An often-mentioned application is for emergency service calls with real-time text and voice, where a call taker wants to make an attended handover of a call to another agent and stay on the call to observe the session. Multimedia conference sessions with support for participants to contribute with text is another example. Conferences with central support for speech-to-text conversion represent yet another example.

In all these applications, normally only one participant at a time will send long text comments. In some cases, one other participant will occasionally contribute with a longer comment simultaneously. That may also happen in some rare cases when text is translated to text in another language in a conference. Apart from these cases, other participants are only expected to contribute with very brief comments while others are sending text.

Users expect the text they send to be presented in real time in a readable way to the other participants even if they send simultaneously with other users and even when they make brief edit operations of their text by backspacing and correcting their text.

Text is supposed to be human generated, by some means of text input, such as typing on a keyboard or using speech-to-text technology. Occasional small cut-and-paste operations may appear even if that is not the initial purpose of real-time text.

The real-time characteristics of real-time text are essential for the participants to be able to contribute to a conversation. If the text is delayed too much between the typing of a character and its presentation, then, in some conference situations, the opportunity to comment will be gone and someone else will grab the turn. A delay of more than one second in such situations is an obstacle to good conversation.

2. Overview of the Two Specified Solutions and Selection of Method

This section contains a brief introduction of the two methods specified in this document.

2.1. The RTP-Mixer-Based Solution for Multiparty-Aware Endpoints

This method specifies the negotiated use of the formats described in RFC 4103, for multiparty transmissions in a single RTP stream. The main purpose of this document is to specify a method for true multiparty real-time text mixing for multiparty-aware endpoints that can be widely

deployed. The RTP-mixer-based method makes use of the current format for real-time text as provided in [RFC4103]. This method updates RFC 4103 by clarifying one way to use it in the multiparty situation. That is done by completing a negotiation for this kind of multiparty capability and by interleaving packets from different sources. The source is indicated in the CSRC element in the RTP packets. Specific considerations are made regarding the ability to recover text after packet loss.

The detailed procedures for the RTP-mixer-based multiparty-aware case are specified in Section 3.

Please refer to [RFC4103] when reading this document.

2.2. Mixing for Multiparty-Unaware Endpoints

This document also specifies a method to be used in cases when the endpoint participating in a multiparty call does not itself implement any solution or does not implement the same solution as the mixer. This method requires the mixer to insert text dividers and readable labels and only send text from one source at a time until a suitable point appears for changing the source. This solution is a fallback method with functional limitations. It operates at the presentation level.

A mixer **SHOULD** by default format and transmit text to a call participant so that the text is suitable for presentation on a multiparty-unaware endpoint that has not negotiated any method for true multiparty real-time text handling but has negotiated a "text/red" or "text/t140" format in a session. This **SHOULD** be done if nothing else is specified for the application, in order to maintain interoperability. Section 4.2 specifies how this mixing is done.

2.3. Offer/Answer Considerations

"RTP Payload for Text Conversation" [RFC4103] specifies the use of RTP [RFC3550] and a redundancy format ("text/red", as defined in [RFC4102]) for increased robustness of real-time text transmission. This document updates [RFC4103] by introducing a capability negotiation for handling multiparty real-time text, a way to indicate the source of transmitted text, and rules for efficient timing of the transmissions interleaved from different sources.

The capability negotiation for the RTP-mixer-based multiparty method is based on the use of the SDP media attribute "rtt-mixer".

The syntax is as follows:

```
a=rtt-mixer
```

If in the future any other method for RTP-based multiparty real-time text is specified by additional work, it is assumed that it will be recognized by some specific SDP feature exchange.

2.3.1. Initial Offer

A party that intends to set up a session and is willing to use the RTP-mixer-based method provided in this specification for sending, receiving, or both sending and receiving real-time text **SHALL** include the "rtt-mixer" SDP attribute in the corresponding "text" media section in the initial offer.

The party **MAY** indicate its capability regarding both the RTP-mixer-based method provided in this specification and other methods.

When the offerer has sent the offer, which includes the "rtt-mixer" attribute, it **MUST** be prepared to receive and handle real-time text formatted according to both the method for multiparty-aware parties specified in Section 3 and two-party formatted real-time text.

2.3.2. Answering the Offer

A party that receives an offer containing the "rtt-mixer" SDP attribute and is willing to use the RTP-mixer-based method provided in this specification for sending, receiving, or both sending and receiving real-time text **SHALL** include the "rtt-mixer" SDP attribute in the corresponding "text" media section in the answer.

If the offer did not contain the "rtt-mixer" attribute, the answer **MUST NOT** contain the "rtt-mixer" attribute.

Even when the "rtt-mixer" attribute is successfully negotiated, the parties **MAY** send and receive two-party coded real-time text.

An answer **MUST NOT** include acceptance of more than one method for multiparty real-time text in the same RTP session.

When the answer, which includes acceptance, is transmitted, the answerer **MUST** be prepared to act on received text in the negotiated session according to the method for multiparty-aware parties specified in Section 3. Reception of text for a two-party session **SHALL** also be supported.

2.3.3. Offerer Processing the Answer

When the answer is processed by the offerer, the offerer **MUST** follow the requirements listed in Section 2.4.

2.3.4. Modifying a Session

A session **MAY** be modified at any time by any party offering a modified SDP with or without the "rtt-mixer" SDP attribute expressing a desired change in the support of multiparty real-time text.

If the modified offer adds the indication of support for multiparty real-time text by including the "rtt-mixer" SDP attribute, the procedures specified in the previous subsections **SHALL** be applied.

If the modified offer deletes the indication of support for multiparty real-time text by excluding the "rtt-mixer" SDP attribute, the answer **MUST NOT** contain the "rtt-mixer" attribute. After processing this SDP exchange, the parties **MUST NOT** send real-time text formatted for multiparty-aware parties according to this specification.

2.4. Actions Depending on Capability Negotiation Result

A transmitting party **SHALL** send text according to the RTP-mixer-based multiparty method only when the negotiation for that method was successful and when it conveys text for another source. In all other cases, the packets **SHALL** be populated and interpreted as for a two-party session.

A party that has negotiated the "rtt-mixer" SDP media attribute and acts as an RTP mixer sending multiparty text **MUST** (1) populate the CSRC list and (2) format the packets according to Section 3.

A party that has negotiated the "rtt-mixer" SDP media attribute **MUST** interpret the contents of the CC field, the CSRC list, and the packets according to Section 3 in received RTP packets in the corresponding RTP stream.

A party that has not successfully completed the negotiation of the "rtt-mixer" SDP media attribute **MUST NOT** transmit packets interleaved from different sources in the same RTP stream, as specified in Section 3. If the party is a mixer and did declare the "rtt-mixer" SDP media attribute, it **SHOULD** perform the procedure for multiparty-unaware endpoints. If the party is not a mixer, it **SHOULD** transmit as in a two-party session according to [RFC4103].

3. Details for the RTP-Mixer-Based Mixing Method for Multiparty-Aware Endpoints

3.1. Use of Fields in the RTP Packets

The CC field **SHALL** show the number of members in the CSRC list, which **SHALL** be one (1) in transmissions from a mixer when conveying text from other sources in a multiparty session, and otherwise 0.

When text is conveyed by a mixer during a multiparty session, a CSRC list **SHALL** be included in the packet. The single member in the CSRC list **SHALL** contain the SSRC of the source of the T140blocks in the packet.

When redundancy is used, the **RECOMMENDED** level of redundancy is to use one primary and two redundant generations of T140blocks. In some cases, a primary or redundant T140block is empty but is still represented by a member in the redundancy header.

In other respects, the contents of the RTP packets will be as specified in [RFC4103].

3.2. Initial Transmission of a BOM Character

As soon as a participant is known to participate in a session with another entity and is available for text reception, a Unicode byte order mark (BOM) character **SHALL** be sent to it by the other entity according to the procedures in this section. This is useful in many configurations for opening ports and firewalls and for setting up the connection between the application and the network. If the transmitter is a mixer, then the source of this character **SHALL** be indicated to be the mixer itself.

Note that the BOM character **SHALL** be transmitted with the same redundancy procedures as any other text.

3.3. Keep-Alive

After that, the transmitter **SHALL** send keep-alive traffic to the receiver(s) at regular intervals when no other traffic has occurred during that interval, if that is decided upon for the actual connection. It is **RECOMMENDED** to use the keep-alive solution provided in [RFC6263]. The consent check [RFC7675] is a possible alternative if it is used anyway for other reasons.

3.4. Transmission Interval

A "text/red" or "text/t140" transmitter in a mixer **SHALL** send packets distributed over time as long as there is something (new or redundant T140blocks) to transmit. The maximum transmission interval between text transmissions from the same source **SHALL** then be 330 ms, when no other limitations cause a longer interval to be temporarily used. It is **RECOMMENDED** to send the next packet to a receiver as soon as new text to that receiver is available, as long as the mean character rate of new text to the receiver calculated over the last 10 one-second intervals does not exceed the "cps" value of the receiver. The intention is to keep the latency low and network load limited while keeping good protection against text loss in bursty packet loss conditions. The main purpose of the 330 ms interval is for the timing of redundant transmissions, when no new text from the same source is available.

The value of 330 ms is used, because many sources of text will transmit new text at 300 ms intervals during periods of continuous user typing, and then reception in the mixer of such new text will cause a combined transmission of the new text and the unsent redundancy from the previous transmission. Only when the user stops typing will the 330 ms interval be applied to send the redundancy.

If the characters per second ("cps") value is reached, a longer transmission interval **SHALL** be applied for text from all sources as specified in [RFC4103] and only as much of the text queued for transmission **SHALL** be sent at the end of each transmission interval as can be allowed without exceeding the "cps" value. Division of text for partial transmission **MUST** then be made at T140block borders. When the transmission rate falls below the "cps" value again, the transmission intervals **SHALL** be reset to 330 ms and transmission of new text **SHALL** again be made as soon as new text is available.

> NOTE: Extending the transmission intervals during periods of high load does not change the number of characters to be conveyed. It just evens out the load over time and reduces the number of packets per second. With human-created conversational text, the sending user will eventually take a pause, letting transmission catch up.

See also Section 8.

For a transmitter not acting as a mixer, the transmission interval principles provided in [RFC4103] apply, and the normal transmission interval **SHALL** be 300 ms.

3.5. Only One Source per Packet

New text and redundant copies of earlier text from one source **SHALL** be transmitted in the same packet if available for transmission at the same time. Text from different sources **MUST NOT** be transmitted in the same packet.

3.6. Do Not Send Received Text to the Originating Source

Text received by a mixer from a participant **SHOULD NOT** be included in transmissions from the mixer to that participant, because for text that is produced locally, the normal behavior of the endpoint is to present such text directly when it is produced.

3.7. Clean Incoming Text

A mixer **SHALL** handle reception, recovery from packet loss, deletion of superfluous redundancy, marking of possible text loss, and deletion of BOM characters from each participant before queueing received text for transmission to receiving participants as specified in [RFC4103] for single-party sources and Section 3.16 for multiparty sources (chained mixers).

3.8. Principles of Redundant Transmission

A transmitting party using redundancy **SHALL** send redundant repetitions of T140blocks already transmitted in earlier packets.

The number of redundant generations of T140blocks to include in transmitted packets **SHALL** be deduced from the SDP negotiation. It **SHALL** be set to the minimum of the number declared by the two parties negotiating a connection. It is **RECOMMENDED** to declare and transmit one original and two redundant generations of the T140blocks, because this provides good protection against text loss in the case of packet loss and also provides low overhead.

3.9. Text Placement in Packets

The mixer **SHALL** compose and transmit an RTP packet to a receiver when one or more of the following conditions have occurred:

- The transmission interval is the normal 330 ms (no matter whether the transmission interval has passed or not), and there is newly received unsent text available for transmission to that receiver.

- The current transmission interval has passed and is longer than the normal 330 ms, and there is newly received unsent text available for transmission to that receiver.
- The current transmission interval (normally 330 ms) has passed since already-transmitted text was queued for transmission as redundant text.

The principles provided in [RFC4103] apply for populating the header, the redundancy header, and the data in the packet with specific information, as detailed here and in the following sections.

At the time of transmission, the mixer **SHALL** populate the RTP packet with all T140blocks queued for transmission originating from the source selected for transmission as long as this is not in conflict with the allowed number of characters per second ("cps") or the maximum packet size. In this way, the latency of the latest received text is kept low even in moments of simultaneous transmission from many sources.

Redundant text **SHALL** also be included, and the assessment of how much new text can be included within the maximum packet size **MUST** take into account that the redundancy has priority to be transmitted in its entirety. See Section 3.4.

The SSRC of the source **SHALL** be placed as the only member in the CSRC list.

> Note: The CSRC list in an RTP packet only includes the participant whose text is included in text blocks. It is not the same as the total list of participants in a conference. With audio and video media, the CSRC list would often contain all participants who are not muted, whereas text participants that don't type are completely silent and thus are not represented in RTP packet CSRC lists.

3.10. Empty T140blocks

If no unsent T140blocks were available for a source at the time of populating a packet but already-transmitted T140blocks are available that have not yet been sent the full intended number of redundant transmissions, then the primary area in the packet is composed of an empty T140block and included (without taking up any length) in the packet for transmission. The corresponding SSRC **SHALL** be placed as usual in its place in the CSRC list.

The first packet in the session, the first after a source switch, and the first after a pause **SHALL** be populated with the available T140blocks for the source selected to be sent as the primary, and empty T140blocks for the agreed-upon number of redundancy generations.

3.11. Creation of the Redundancy

The primary T140block from a source in the latest transmitted packet is saved for populating the first redundant T140block for that source in the next transmission of text from that source. The first redundant T140block for that source from the latest transmission is saved for populating the second redundant T140block in the next transmission of text from that source.

Usually, this is the level of redundancy used. If a higher level of redundancy is negotiated, then the procedure **SHALL** be continued until all available redundant levels of T140blocks are placed in the packet. If a receiver has negotiated a lower number of "text/red" generations, then that level **SHALL** be the maximum used by the transmitter.

The T140blocks saved for transmission as redundant data are assigned a planned transmission time of 330 ms after the current time but **SHOULD** be transmitted earlier if new text for the same source gets selected for transmission before that time.

3.12. Timer Offset Fields

The timestamp offset values **SHALL** be inserted in the redundancy header, with the time offset from the RTP timestamp in the packet when the corresponding T140block was sent as the primary.

The timestamp offsets are expressed in the same clock tick units as the RTP timestamp.

The timestamp offset values for empty T140blocks have no relevance but **SHOULD** be assigned realistic values.

3.13. Other RTP Header Fields

The number of members in the CSRC list (0 or 1) **SHALL** be placed in the CC header field. Only mixers place value 1 in the CC field. A value of 0 indicates that the source is the transmitting device itself and that the source is indicated by the SSRC field. This value is used by endpoints and also by mixers sending self-sourced data.

The current time **SHALL** be inserted in the timestamp.

The SSRC header field **SHALL** contain the SSRC of the RTP session where the packet will be transmitted.

The M-bit **SHALL** be handled as specified in [RFC4103].

3.14. Pause in Transmission

When there is no new T140block to transmit and no redundant T140block that has not been retransmitted the intended number of times from any source, the transmission process **SHALL** be stopped until either new T140blocks arrive or a keep-alive method calls for transmission of keep-alive packets.

3.15. RTCP Considerations

A mixer **SHALL** send RTCP reports with SDES, CNAME, and NAME information about the sources in the multiparty call. This makes it possible for participants to compose a suitable label for text from each source.

Privacy considerations **SHALL** be taken when composing these fields. They contain name and address information that may be considered sensitive if the information is transmitted in its entirety, e.g., to unauthenticated participants.

3.16. Reception of Multiparty Contents

The "text/red" receiver included in an endpoint with presentation functions will receive RTP packets in the single stream from the mixer and **SHALL** distribute the T140blocks for presentation in presentation areas for each source. Other receiver roles, such as gateways or chained mixers, are also feasible. Whether the stream will only be forwarded or will be distributed based on the different sources must be taken into consideration.

3.16.1. Acting on the Source of the Packet Contents

If the CC field value of a received packet is 1, it indicates that the text is conveyed from a source indicated in the single member in the CSRC list, and the receiver **MUST** act on the source according to its role. If the CC value is 0, the source is indicated in the SSRC field.

3.16.2. Detection and Indication of Possible Text Loss

The receiver **SHALL** monitor the RTP sequence numbers of the received packets for gaps and for packets received out of order. If a sequence number gap appears and still exists after some defined short time for jitter and reordering resolution, the packets in the gap **SHALL** be regarded as lost.

If it is known that only one source is active in the RTP session, then it is likely that a gap equal to or larger than the agreed-upon number of redundancy generations (including the primary) causes text loss. In that case, the receiver **SHALL** create a T140block with a marker for possible text loss [T140ad1], associate it with the source, and insert it in the reception buffer for that source.

If it is known that more than one source is active in the RTP session, then it is not possible in general to evaluate if text was lost when packets were lost. With two active sources and the recommended number of redundancy generations (one original and two redundant), it can take a gap of five consecutive lost packets before any text may be lost, but text loss can also appear if three non-consecutive packets are lost when they contained consecutive data from the same source. A simple method for deciding when there is a risk of resulting text loss is to evaluate if three or more packets were lost within one second. If this simple method is used, then a T140block **SHOULD** be created with a marker for possible text loss [T140ad1] and associated with the SSRC of the RTP session as a general input from the mixer.

Implementations **MAY** apply more refined methods for more reliable detection of whether text was lost or not. Any refined method **SHOULD** prefer marking possible loss rather than not marking when it is uncertain if there was loss.

3.16.3. Extracting Text and Handling Recovery

When applying the following procedures, the effects of possible timestamp wraparound and the RTP session possibly changing the SSRC **MUST** be considered.

When a packet is received in an RTP session using the packetization for multiparty-aware endpoints, its T140blocks **SHALL** be extracted as described below.

The source **SHALL** be extracted from the CSRC list if available, and otherwise from the SSRC.

If the received packet is the first packet received from the source, then all T140blocks in the packet **SHALL** be retrieved and assigned to a receive buffer for that source, beginning with the oldest available redundant generation, continuing with the younger redundant generations in age order, and finally ending with the primary.

> Note: The normal case is that in the first packet, only the primary data has contents. The redundant data has contents in the first received packet from a source only after initial packet loss.

If the packet is not the first packet from a source, then if redundant data is available, the process **SHALL** start with the oldest generation. The timestamp of that redundant data **SHALL** be created by subtracting its timestamp offset from the RTP timestamp. If the resulting timestamp is later than the latest retrieved data from the same source, then the redundant data **SHALL** be retrieved and appended to the receive buffer. The process **SHALL** be continued in the same way for all younger generations of redundant data. After that, the timestamp of the packet **SHALL** be compared with the timestamp of the latest retrieved data from the same source and if it is later, then the primary data **SHALL** be retrieved from the packet and appended to the receive buffer for the source.

3.16.4. Delete BOM

The Unicode BOM character is used as a start indication and is sometimes used as a filler or keep-alive by transmission implementations. Any BOM characters **SHALL** be deleted after extraction from received packets.

3.17. Performance Considerations

This solution has good performance with low text delays, as long as the mean number of characters per second sent during any 10-second interval from a number of simultaneously sending participants to a receiving participant does not reach the "cps" value. At higher numbers of sent characters per second, a jerkiness is visible in the presentation of text. The solution is therefore suitable for emergency service use, relay service use, and small or well-managed larger multimedia conferences. In large unmanaged conferences with a high number of participants only, on very rare occasions, situations might arise where many participants happen to send text simultaneously. In such circumstances, the result may be unpleasantly jerky presentation of text from each sending participant. It should be noted that it is only the number of users sending text within the same moment that causes jerkiness, not the total number of users with real-time text capability.

3.18. Security for Session Control and Media

Security mechanisms to provide confidentiality, integrity protection, and peer authentication **SHOULD** be applied when possible regarding the capabilities of the participating devices by using the Session Initiation Protocol (SIP) over TLS by default according to Section 3.1.3 of [RFC5630] on the session control level and by default using DTLS-SRTP [RFC5764] at the media level. In applications where legacy endpoints without security are allowed, a negotiation **SHOULD** be performed to decide if encryption at the media level will be applied. If no other security solution is mandated for the application, then the Opportunistic Secure Real-time Transport Protocol (OSRTP) [RFC8643] is a suitable method to be applied to negotiate SRTP media security with DTLS. For simplicity, most SDP examples below are expressed without the security additions. The principles (but not all details) for applying DTLS-SRTP security [RFC5764] are shown in a couple of the following examples.

Further general security considerations are covered in Section 10.

End-to-end encryption would require further work and could be based on WebRTC as specified in Section 1.2 or on double encryption as specified in [RFC8723].

3.19. SDP Offer/Answer Examples

This section shows some examples of SDP for session negotiation of the real-time text media in SIP sessions. Audio is usually provided in the same session, and sometimes also video. The examples only show the part of importance for the real-time text media. The examples relate to the single RTP stream mixing for multiparty-aware endpoints and for multiparty-unaware endpoints.

> Note: Multiparty real-time text **MAY** also be provided through other methods, e.g., by a Selective Forwarding Middlebox (SFM). In that case, the SDP of the offer will include something specific for that method, e.g., an SDP attribute or another media format. An answer selecting the use of that method would accept it via a corresponding acknowledgement included in the SDP. The offer may also contain the "rtt-mixer" SDP media attribute for the main real-time text media when the offerer has this capability for both multiparty methods, while an answer, choosing to use SFM, will not include the "rtt-mixer" SDP media attribute.

Offer example for the "text/red" format, multiparty support, and capability for 90 characters per second:

```
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=fmtp:98 cps=90
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

Answer example from a multiparty-aware device:

```
m=text 14000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=fmtp:98 cps=90
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

Offer example for the "text/red" format, including multiparty and security:

```
a=fingerprint: (fingerprint1)
m=text 11000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

The "fingerprint" is sufficient to offer DTLS-SRTP, with the media line still indicating RTP/AVP.

> Note: For brevity, the entire value of the SDP "fingerprint" attribute is not shown in this and the following example.

Answer example from a multiparty-aware device with security:

```
a=fingerprint: (fingerprint2)
m=text 16000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
a=rtt-mixer
```

With the "fingerprint", the device acknowledges the use of DTLS-SRTP.

Answer example from a multiparty-unaware device that also does not support security:

```
m=text 12000 RTP/AVP 100 98
a=rtpmap:98 t140/1000
a=rtpmap:100 red/1000
a=fmtp:100 98/98/98
```

3.20. Packet Sequence Example from Interleaved Transmission

This example shows a symbolic flow of packets from a mixer, including loss and recovery. The sequence includes interleaved transmission of text from two real-time text sources: A and B. P indicates primary data. R1 is the first redundant generation of data, and R2 is the second redundant generation of data. A1, B1, A2, etc. are text chunks (T140blocks) received from the

respective sources and sent on to the receiver by the mixer. X indicates a dropped packet between the mixer and a receiver. The session is assumed to use the original and two redundant generations of real-time text.

```
|-----------------------|
|Seq no 101, Time=20400 |
|CC=1                   |
|CSRC list A            |
|R2: A1, Offset=600     |
|R1: A2, Offset=300     |
|P:  A3                 |
|-----------------------|
```

Assuming that earlier packets (with text A1 and A2) were received in sequence, text A3 is received from packet 101 and assigned to reception buffer A. The mixer is now assumed to have received initial text from source B 100 ms after packet 101 and will send that text. Transmission of A2 and A3 as redundancy is planned for 330 ms after packet 101 if no new text from A is ready to be sent before that.

```
 |-----------------------|
 |Seq no 102, Time=20500 |
 |CC=1                   |
 |CSRC list B            |
 |R2  Empty, Offset=600  |
 |R1: Empty, Offset=300  |
 |P:  B1                 |
 |-----------------------|
```

Packet 102 is received.

B1 is retrieved from this packet. Redundant transmission of B1 is planned 330 ms after packet 102.

```
X-----------------------|
X Seq no 103, Timer=20730|
X CC=1                   |
X CSRC list A            |
X R2: A2, Offset=630     |
X R1: A3, Offset=330     |
X P:  Empty              |
X-----------------------|
```

Packet 103 is assumed to be lost due to network problems.

It contains redundancy for A. Sending A3 as second-level redundancy is planned for 330 ms after packet 103.

```
X------------------------|
X Seq no 104, Timer=20800|
X CC=1                   |
X CSRC list B            |
X R2: Empty, Offset=600  |
X R1: B1, Offset=300     |
X P:  B2                 |
X------------------------|
```

Packet 104 contains text from B, including new B2 and redundant B1. It is assumed dropped due to network problems.

The mixer has A3 redundancy to send, but no new text appears from A, and therefore the redundancy is sent 330 ms after the previous packet with text from A.

```
|------------------------|
| Seq no 105, Timer=21060|
| CC=1                   |
| CSRC list A            |
| R2: A3, Offset=660     |
| R1: Empty, Offset=330  |
| P:  Empty              |
|------------------------|
```

Packet 105 is received.

A gap for lost packets 103 and 104 is detected. Assume that no other loss was detected during the last second. It can then be concluded that nothing was totally lost.

R2 is checked. Its original time was 21060-660=20400. A packet with text from A was received with that timestamp, so nothing needs to be recovered.

B1 and B2 still need to be transmitted as redundancy. This is planned 330 ms after packet 104. That would be at 21130.

```
|------------------------|
|Seq no 106, Timer=21130|
|CC=1                    |
|CSRC list B             |
| R2: B1, Offset=630     |
| R1: B2, Offset=330     |
| P:  Empty              |
|------------------------|
```

Packet 106 is received.

The second-level redundancy in packet 106 is B1 and has a timestamp offset of 630 ms. The timestamp of packet 106 minus 630 is 20500, which is the timestamp of packet 102 that was received. So, B1 does not need to be retrieved. The first-level redundancy in packet 106 has an offset of 330. The timestamp of packet 106 minus 330 is 20800. That is later than the latest received packet with source B. Therefore, B2 is retrieved and assigned to the input buffer for source B. No primary is available in packet 106.

After this sequence, A3, B1, and B2 have been received. In this case, no text was lost.

3.21. Maximum Character Rate "cps" Setting

The default maximum rate of reception of "text/t140" real-time text, as specified in [RFC4103], is 30 characters per second. The actual rate is calculated without regard to any redundant text transmission and is, in the multiparty case, evaluated for all sources contributing to transmission to a receiver. The value **MAY** be modified in the "cps" parameter of the "fmtp" attribute for the "text/t140" format of the "text" media section.

A mixer combining real-time text from a number of sources may occasionally have a higher combined flow of text coming from the sources. Endpoints **SHOULD** therefore include a suitable higher value for the "cps" parameter, corresponding to its real reception capability. The default "cps" value 30 can be assumed to be sufficient for small meetings and well-managed larger conferences with users only making manual text entry. A "cps" value of 90 can be assumed to be sufficient even for large unmanaged conferences and for cases when speech-to-text technologies are used for text entry. This is also a reachable performance for receivers in modern technologies, and 90 is therefore the **RECOMMENDED** "cps" value. See [RFC4103] for the format and use of the "cps" parameter. The same rules apply for the multiparty case.

4. Presentation-Level Considerations

"Protocol for multimedia application text conversation" [T140] provides the presentation-level requirements for RTP transport as described in [RFC4103]. Functions for erasure and other formatting functions are specified in [T140], which has the following general statement for the presentation:

> The display of text from the members of the conversation should 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.

Strict application of [T140] is essential for the interoperability of real-time text implementations and to fulfill the intention that the session participants have the same information conveyed in the text contents of the conversation without necessarily having the exact same layout of the conversation.

[T140] specifies a set of presentation control codes (Section 4.2.4) to include in the stream. Some of them are optional. Implementations **MUST** ignore optional control codes that they do not support.

There is no strict "message" concept in real-time text. The Unicode Line Separator character **SHALL** be used as a separator allowing a part of received text to be grouped in a presentation. The character combination "CRLF" may be used by other implementations as a replacement for the Line Separator. The "CRLF" combination **SHALL** be erased by just one erasing action, the same as the Line Separator. Presentation functions are allowed to group text for presentation in smaller groups than the Line Separators imply and present such groups with a source indication together with text groups from other sources (see the following presentation examples). Erasure has no specific limit by any delimiter in the text stream.

4.1. Presentation by Multiparty-Aware Endpoints

A multiparty-aware receiving party presenting real-time text **MUST** separate text from different sources and present them in separate presentation fields. The receiving party **MAY** separate the presentation of parts of text from a source in readable groups based on criteria other than a Line Separator and merge these groups in the presentation area when it benefits the user to most easily find and read text from the different participants. The criteria **MAY**, for example, be a received comma, a full stop, some other type of phrase delimiter, or a long pause.

When text is received from multiple original sources, the presentation **SHALL** provide a view where text is added in multiple presentation fields.

If the presentation presents text from different sources in one common area, the presenting endpoint **SHOULD** insert text from the local user, where the text ends at suitable points and is merged properly with received text to indicate the relative timing for when the text groups were completed. In this presentation mode, the receiving endpoint **SHALL** present the source of the different groups of text. This presentation style is called the "chat" style here and provides the possibility of following text arriving from multiple parties and the approximate relative time that text is received as related to text from the local user.

A view of a three-party real-time text call in chat style is shown in this example.

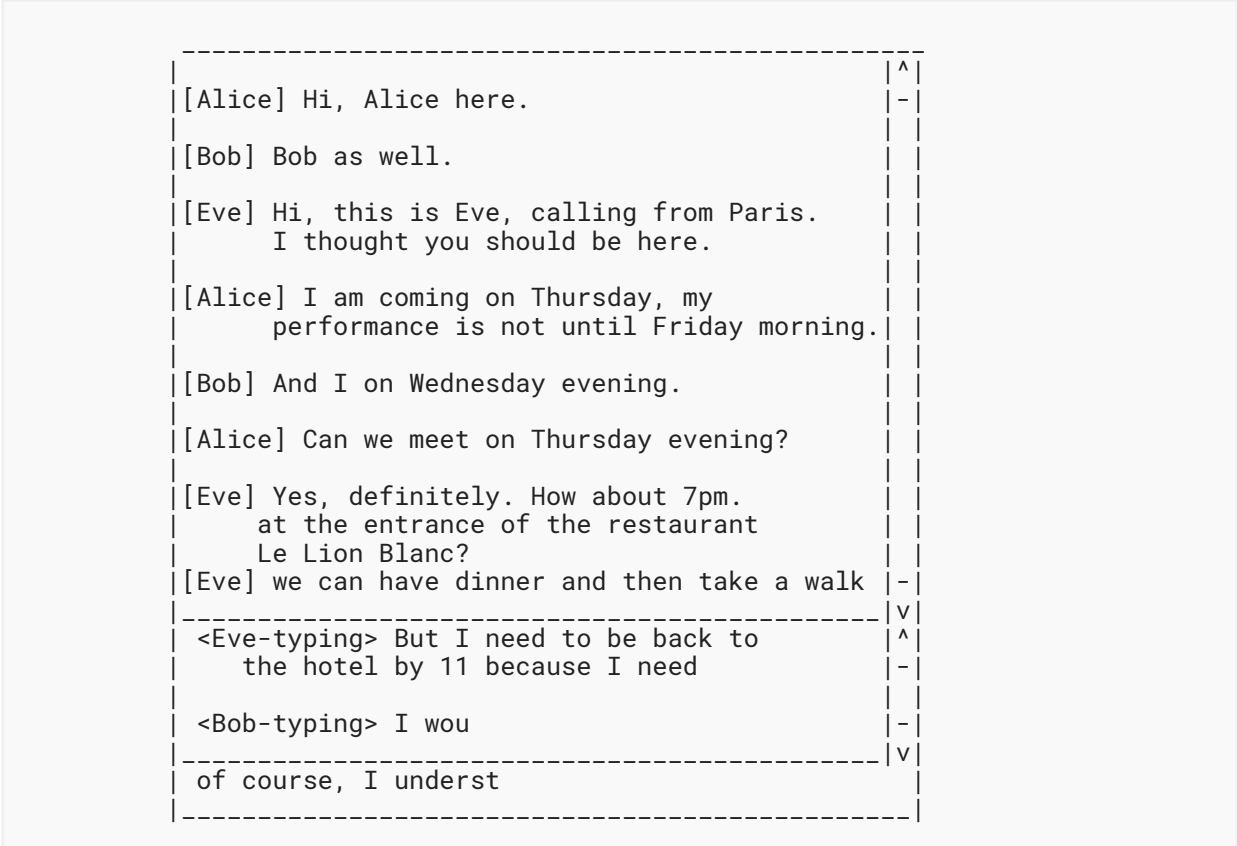

Figure 1: Example of a Three-Party Real-Time Text Call Presented in Chat Style Seen at Participant Alice's Endpoint

Presentation styles other than the chat style **MAY** be arranged.

Figure 2 shows how a coordinated column view **MAY** be presented.

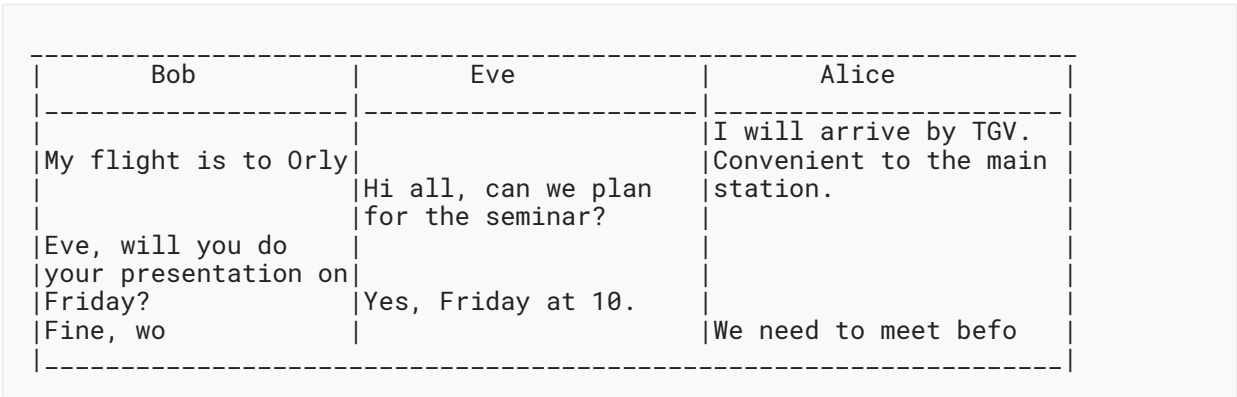

Figure 2: An Example of a Coordinated Column View of a Three-Party Session with Entries Ordered Vertically in Approximate Time Order

4.2. Multiparty Mixing for Multiparty-Unaware Endpoints

When the mixer has indicated multiparty real-time text capability in an SDP negotiation but the multiparty capability negotiation fails with an endpoint, the agreed-upon "text/red" or "text/t140" format **SHALL** be used and the mixer **SHOULD** compose a best-effort presentation of multiparty real-time text in one stream intended to be presented by an endpoint with no multiparty awareness, when that is desired in the actual implementation. The following specifies a procedure that **MAY** be applied in that situation.

This presentation format has functional limitations and **SHOULD** be used only to enable participation in multiparty calls by legacy deployed endpoints implementing only RFC 4103 without any multiparty extensions specified in this document.

The principles and procedures below do not specify any new protocol elements. They are instead composed of information provided in [T140] and an ambition to provide a best-effort presentation on an endpoint that has functions originally intended only for two-party calls.

The mixer performing the mixing for multiparty-unaware endpoints **SHALL** compose a simulated, limited multiparty real-time text view suitable for presentation in one presentation area. The mixer **SHALL** group text in suitable groups and prepare them for presentation by inserting a Line Separator between them if the transmitted text did not already end with a new line (Line Separator or CRLF). A presentable label **SHALL** be composed and sent for the source initially in the session and after each source switch. With this procedure, the time for switching from transmission of text from one source to transmission of text from another source depends on the actions of the users. In order to expedite source switching, a user can, for example, end its turn with a new line.

4.2.1. Actions by the Mixer at Reception from the Call Participants

When text is received by the mixer from the different participants, the mixer **SHALL** recover text from redundancy if any packets are lost. The marker for lost text [T140ad1] **SHALL** be inserted in the stream if unrecoverable loss appears. Any Unicode BOM characters, possibly used for keep-alives, **SHALL** be deleted. The time of creation of text (retrieved from the RTP timestamp) **SHALL** be stored together with the received text from each source in queues for transmission to the recipients in order to be able to evaluate text loss.

4.2.2. Actions by the Mixer for Transmission to the Recipients

The following procedure **SHALL** be applied for each multiparty-unaware recipient of multiparty text from the mixer.

The text for transmission **SHALL** be formatted by the mixer for each receiving user for presentation in one single presentation area. Text received from a participant **SHOULD NOT** be included in transmissions to that participant, because it is usually presented locally at transmission time. When there is text available for transmission from the mixer to a receiving party from more than one participant, the mixer **SHALL** switch between transmission of text from the different sources at suitable points in the transmitted stream.

When switching the source, the mixer **SHALL** insert a Line Separator if the already-transmitted text did not end with a new line (Line Separator or CRLF). A label **SHALL** be composed of information in the CNAME and NAME fields in RTCP reports from the participant to have its text transmitted, or from other session information for that user. The label **SHALL** be delimited by suitable characters (e.g., "[ ]") and transmitted. The CSRC **SHALL** indicate the selected source. Then, text from that selected participant **SHALL** be transmitted until a new suitable point for switching the source is reached.

Information available to the mixer for composing the label may contain sensitive personal information that **SHOULD NOT** be revealed in sessions not securely authenticated and confidentiality protected. Privacy considerations regarding how much personal information is included in the label **SHOULD** therefore be taken when composing the label.

Seeking a suitable point for switching the source **SHALL** be done when there is older text waiting for transmission from any party than the age of the last transmitted text. Suitable points for switching are:

- A completed phrase ending with a comma.
- A completed sentence.
- A new line (Line Separator or CRLF).
- A long pause (e.g., > 10 seconds) in received text from the currently transmitted source.
- If text from one participant has been transmitted with text from other sources waiting for transmission for a long time (e.g., > 1 minute) and none of the other suitable points for switching has occurred, a source switch **MAY** be forced by the mixer at the next word delimiter, and also even if a word delimiter does not occur within some period of time (e.g., 15 seconds) after the scan for a word delimiter started.

When switching the source, the source that has the oldest text in queue **SHALL** be selected to be transmitted. A character display count **SHALL** be maintained for the currently transmitted source, starting at zero after the label is transmitted for the currently transmitted source.

The status **SHALL** be maintained for the latest control code for Select Graphic Rendition (SGR) from each source. If there is an SGR code stored as the status for the current source before the source switch is done, a reset of SGR **SHALL** be sent by the sequence SGR 0 [U+009B U+0000 U +006D] after the new line and before the new label during a source switch. See Section 4.2.4 for an explanation. This transmission does not influence the display count.

If there is an SGR code stored for the new source after the source switch, that SGR code **SHALL** be transmitted to the recipient before the label. This transmission does not influence the display count.

4.2.3. Actions on Transmission of Text

Text from a source sent to the recipient **SHALL** increase the display count by one per transmitted character.

4.2.4. Actions on Transmission of Control Codes

The following control codes, as specified by T.140 [T140], require specific actions. They **SHALL** cause specific considerations in the mixer. Note that the codes presented here are expressed in UTF-16, while transmission is made in the UTF-8 encoding of these codes.

BEL (U+0007): Bell. Alert in session. Provides for alerting during an active session. The display count **SHALL NOT** be altered.

NEW LINE (U+2028): Line Separator. Check and perform a source switch if appropriate. Increase the display count by 1.

CR LF (U+000D U+000A): A supported, but not preferred, way of requesting a new line. Check and perform a source switch if appropriate. Increase the display count by 1.

INT (ESC U+0061): Interrupt (used to initiate the mode negotiation procedure). The display count **SHALL NOT** be altered.

SGR (U+009B Ps U+006D): Select Graphic Rendition. Ps represents the rendition parameters specified in [ISO6429]. (For freely available equivalent information, please see [ECMA-48].) The display count **SHALL NOT** be altered. The SGR code **SHOULD** be stored for the current source.

SOS (U+0098): Start of String. Used as a general protocol element introducer, followed by a maximum 256-byte string and the ST. The display count **SHALL NOT** be altered.

ST (U+009C): String Terminator. End of SOS string. The display count **SHALL NOT** be altered.

ESC (U+001B): Escape. Used in control strings. The display count **SHALL NOT** be altered for the complete escape code.

Byte order mark (BOM) (U+FEFF): "Zero width no-break space". Used for synchronization and keep-alive. It **SHALL** be deleted from incoming streams. It **SHALL** also be sent first after session establishment to the recipient. The display count **SHALL NOT** be altered.

Missing text mark (U+FFFD): "Replacement character". Represented as a question mark in a rhombus, or, if that is not feasible, replaced by an apostrophe ('). It marks the place in the stream of possible text loss. This mark **SHALL** be inserted by the reception procedure in the case of unrecoverable loss of packets. The display count **SHALL** be increased by one when sent as for any other character.

SGR: If a control code for SGR other than a reset of the graphic rendition (SGR 0) is sent to a recipient, that control code **SHALL** also be stored as the status for the source in the storage for SGR status. If a reset graphic rendition (SGR 0) originating from a source is sent, then the SGR status storage for that source **SHALL** be cleared. The display count **SHALL NOT** be increased.

BS (U+0008): "Back Space". Intended to erase the last entered character by a source. Erasure by backspace cannot always be performed as the erasing party intended. If an erasing action erases all text up to the end of the leading label after a source switch, then the mixer **MUST**

NOT transmit more backspaces. Instead, it is **RECOMMENDED** that a letter "X" be inserted in the text stream for each backspace as an indication of the intent to erase more. A new line is usually coded by a Line Separator, but the character combination "CRLF" **MAY** be used instead. Erasure of a new line is, in both cases, done by just one erasing action (backspace). If the display count has a positive value, it **SHALL** be decreased by one when the BS is sent. If the display count is at zero, it **SHALL NOT** be altered.

4.2.5. Packet Transmission

A mixer transmitting to a multiparty-unaware endpoint **SHALL** send primary data only from one source per packet. The SSRC **SHALL** be the SSRC of the mixer. The CSRC list **MAY** contain one member and be the SSRC of the source of the primary data.

4.2.6. Functional Limitations

When a multiparty-unaware endpoint presents a conversation in one display area in a chat style, it inserts source indications for remote text and local user text as they are merged in completed text groups. When an endpoint using this layout receives and presents text mixed for multiparty-unaware endpoints, there will be two levels of source indicators for the received text: one generated by the mixer and inserted in a label after each source switch, and another generated by the receiving endpoint and inserted after each switch between the local source and the remote source in the presentation area. This will waste display space and look inconsistent to the reader.

New text can be presented from only one source at a time. Switching the source to be presented takes place at suitable places in the text, such as the end of a phrase, the end of a sentence, or a Line Separator, or upon detecting inactivity. Therefore, the time to switch to present waiting text from other sources may grow long, and it will vary and depend on the actions of the currently presented source.

Erasure can only be done up to the latest source switch. If a user tries to erase more text, the erasing actions will be presented as a letter "X" after the label.

Text loss because of network errors may hit the label between entries from different parties, causing the risk of a misunderstanding regarding which source provided a piece of text.

Because of these facts, it is strongly **RECOMMENDED** that multiparty awareness be implemented in real-time text endpoints. The use of the mixing method for multiparty-unaware endpoints should be left for use with endpoints that are impossible to upgrade to become multiparty aware.

4.2.7. Example Views of Presentation on Multiparty-Unaware Endpoints

The following pictures are examples of the view on a participant's display for the multiparty-unaware case.

Figure 3 shows how a coordinated column view **MAY** be presented on Alice's device in a view with two columns. The mixer inserts labels to show how the sources alternate in the column with received text. The mixer alternates between the sources at suitable points in the text exchange so that text entries from each party can be conveniently read.

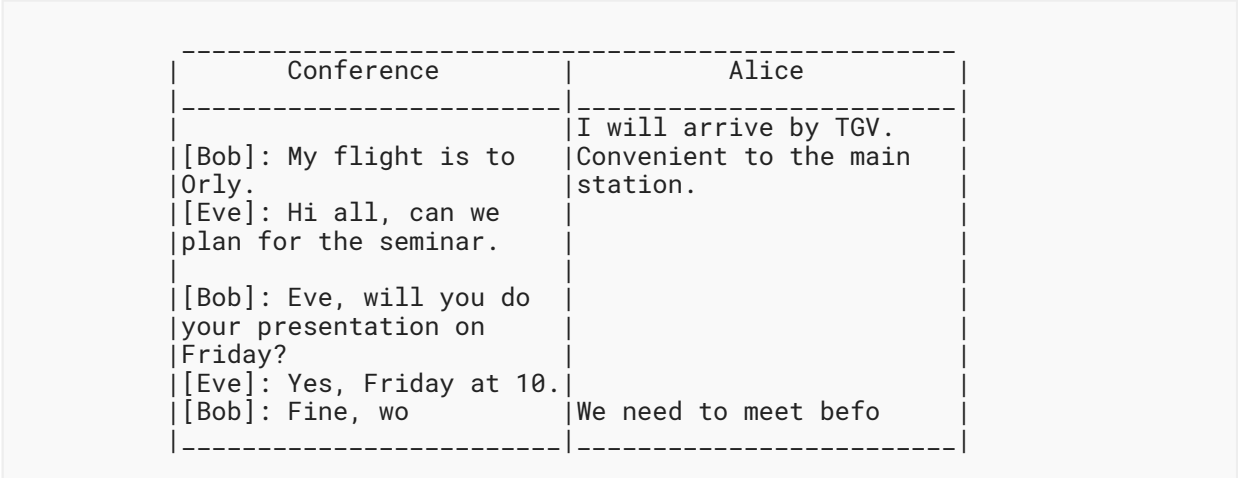

Figure 3: Alice, Who Has a Conference-Unaware Client, Is Receiving the Multiparty Real-Time Text in a Single Stream

In Figure 4, there is a tradition in receiving applications to include a label showing the source of the text, here shown with parentheses "()". The mixer also inserts source labels for the multiparty call participants, here shown with brackets "[]".

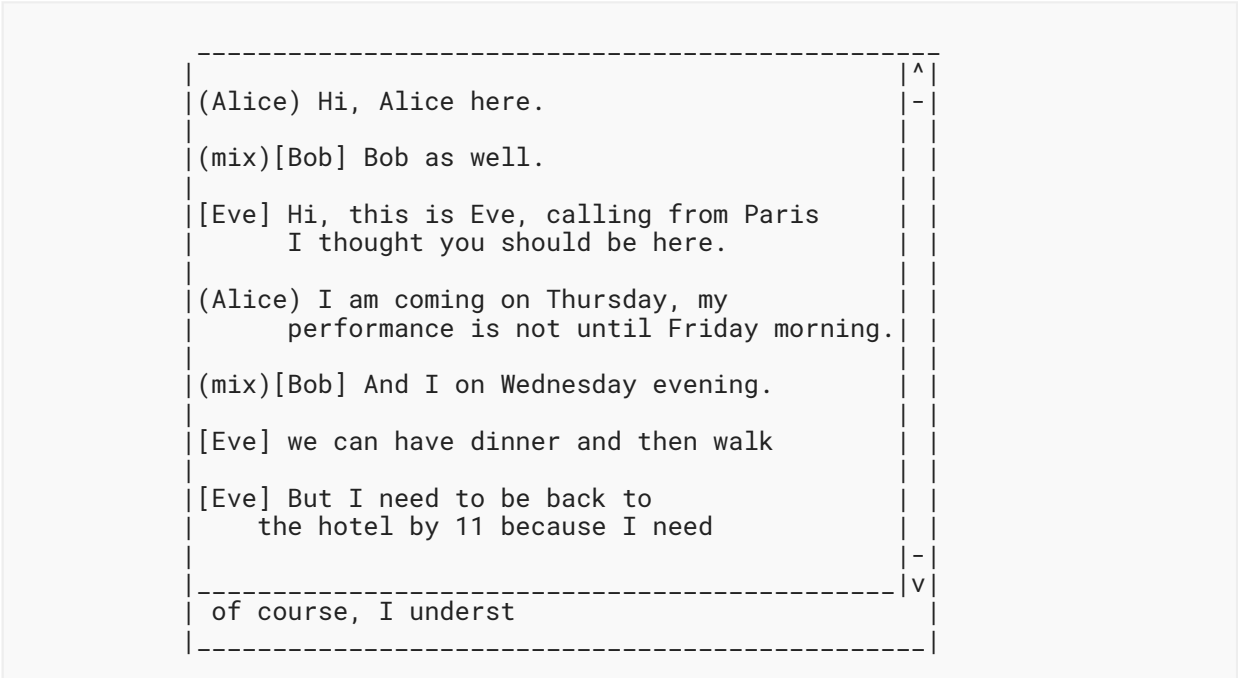

Figure 4: An Example of a View of the Multiparty-Unaware Presentation in Chat Style, Where Alice Is the Local User

5. Relationship to Conference Control

5.1. Use with SIP Centralized Conferencing Framework

The Session Initiation Protocol (SIP) conferencing framework, mainly specified in [RFC4353], [RFC4579], and [RFC4575], is suitable for coordinating sessions, including multiparty real-time text. The real-time text stream between the mixer and a participant is one and the same during the conference. Participants get announced by notifications when participants are joining or leaving, and further user information may be provided. The SSRC of the text to expect from joined users **MAY** be included in a notification. The notifications **MAY** be used for both security purposes and translation to a label for presentation to other users.

5.2. Conference Control

In managed conferences, control of the real-time text media **SHOULD** be provided in the same way as for other media, e.g., for muting and unmuting by the direction attributes in SDP [RFC8866].

Note that floor control functions may be of value for real-time text users as well as for users of other media in a conference.

6. Gateway Considerations

Multiparty real-time text sessions may involve gateways of different kinds. Gateways involved in setting up sessions **SHALL** correctly reflect the multiparty capability or unawareness of the combination of the gateway and the remote endpoint beyond the gateway.

6.1. Gateway Considerations with Textphones

One case that may occur is a gateway to the Public Switched Telephone Network (PSTN) for communication with textphones (e.g., TTYs). Textphones are limited devices with no multiparty awareness, and it **SHOULD** therefore be appropriate for the gateway to not indicate multiparty awareness for that case. Another solution is that the gateway indicates multiparty capability towards the mixer and includes the multiparty mixer function for multiparty-unaware endpoints itself. This solution makes it possible to adapt to the functional limitations of the textphone.

More information on gateways to textphones is found in [RFC5194].

6.2. Gateway Considerations with WebRTC

Gateway operation between RTP-mixer-based multiparty real-time text and WebRTC-based real-time text may also be required. Real-time text transport in WebRTC is specified in [RFC8865].

A multiparty bridge may have functionality for communicating via real-time text in both (1) RTP streams with real-time text and (2) WebRTC T.140 data channels. Other configurations may consist of a multiparty bridge with either technology for real-time text transport and a separate gateway for conversion of the text communication streams between RTP and T.140 data channels.

In WebRTC, it is assumed that for a multiparty session, one T.140 data channel is established for each source from a gateway or bridge to each participant. Each participant also has a data channel with a two-way connection with the gateway or bridge.

A T.140 data channel used for two-way communication is for text from the WebRTC user and from the bridge or gateway itself to the WebRTC user. The label parameter of this T.140 data channel is used as the NAME field in RTCP to participants on the RTP side. The other T.140 data channels are only for text from other participants to the WebRTC user.

When a new participant has entered the session with RTP transport of real-time text, a new T.140 data channel **SHOULD** be established to WebRTC users with the label parameter composed of information from the NAME field in RTCP on the RTP side.

When a new participant has entered the multiparty session with real-time text transport in a WebRTC T.140 data channel, the new participant **SHOULD** be announced by a notification to RTP users. The label parameter from the WebRTC side or other suitable information from the session or stream establishment procedure **SHOULD** be used to compose the NAME RTCP field on the RTP side.

When a participant on the RTP side is disconnected from the multiparty session, the corresponding T.140 data channel(s) **SHOULD** be closed.

When a WebRTC user of T.140 data channels disconnects from the mixer, the corresponding RTP streams or sources in an RTP-mixed stream **SHOULD** be closed.

T.140 data channels **MAY** be opened and closed by negotiation or renegotiation of the session, or by any other valid means, as specified in Section 1 of [RFC8865].

7. Updates to RFC 4103

This document updates [RFC4103] by introducing an SDP media attribute, "rtt-mixer", for negotiation of multiparty-mixing capability with the format described in [RFC4103] and by specifying the rules for packets when multiparty capability is negotiated and in use.

8. Congestion Considerations

The congestion considerations and recommended actions provided in [RFC4103] are also valid in multiparty situations.

The time values **SHALL** then be applied per source of text sent to a receiver.

In the very unlikely event that many participants in a conference send text simultaneously for a long period of time, a delay may build up for the presentation of text at the receivers if the limitation in characters per second ("cps") to be transmitted to the participants is exceeded. A delay of more than 15 seconds can cause confusion in the session. It is therefore **RECOMMENDED** that an RTP mixer discard such text causing excessive delays and insert a general indication of possible text loss [T140ad1] in the session. If the main text contributor is indicated in any way, the mixer **MAY** avoid deleting text from that participant. It should, however, be noted that human creation of text normally contains pauses, when the transmission can catch up, so that transmission-overload situations are expected to be very rare.

9. IANA Considerations

9.1. Registration of the "rtt-mixer" SDP Media Attribute

IANA has registered the new SDP attribute "rtt-mixer".

Contact name: IESG

Contact email: iesg@ietf.org

Attribute name: rtt-mixer

Attribute semantics: See RFC 9071, Section 2.3

Attribute value: none

Usage level: media

Purpose: To indicate mixer and endpoint support of multiparty mixing for real-time text transmission, using a common RTP stream for transmission of text from a number of sources mixed with one source at a time and where the source is indicated in a single CSRC-list member.

Charset Dependent: no

O/A procedures: See RFC 9071, Section 2.3

Mux Category: NORMAL

Reference: RFC 9071

10. Security Considerations

The RTP-mixer model requires the mixer to be allowed to decrypt, pack, and encrypt secured text from conference participants. Therefore, the mixer needs to be trusted to maintain confidentiality and integrity of the real-time text data. This situation is similar to the situation for handling audio and video media in centralized mixers.

The requirement to transfer information about the user in RTCP reports in SDES, CNAME, and NAME fields, and in conference notifications, may have privacy concerns, as already stated in RFC 3550 [RFC3550], and may be restricted for privacy reasons. When used for the creation of readable labels in the presentation, the receiving user will then get a more symbolic label for the source.

The services available through the real-time text mixer may be of special interest to deaf and hard-of-hearing individuals. Some users may want to refrain from revealing such characteristics broadly in conferences. Conference systems where the mixer is included **MAY** need to be designed with the confidentiality of such characteristics in mind.

Participants with malicious intentions may appear and, for example, disrupt the multiparty session by emitting a continuous flow of text. They may also send text that appears to originate from other participants. Countermeasures should include requiring secure signaling, media, and authentication, and providing higher-layer conference functions, e.g., for blocking, muting, and expelling participants.

Participants with malicious intentions may also try to disrupt the presentation by sending incomplete or malformed control codes. Handling of text from the different sources by the receivers **MUST** therefore be well separated so that the effects of such actions only affect text from the source causing the action.

Care should be taken to avoid the possibility of attacks by unauthenticated call participants, and even eavesdropping and manipulation of content by non-participants, if the use of the mixer is permitted for users both with and without security procedures.

As already stated in Section 3.18, security in media **SHOULD** be applied by using DTLS-SRTP [RFC5764] at the media level.

Further security considerations specific to this application are specified in Section 3.18.

11. References

11.1. Normative References

[ECMA-48] Ecma International, "ECMA-48: Control functions for coded character sets", 5th edition, June 1991, <https://www.ecma-international.org/publications-and-standards/standards/ecma-48/>.

[ISO6429] ISO/IEC, "Information technology - Control functions for coded character sets", ISO/IEC ISO/IEC 6429:1992, December 1992, <https://www.iso.org/obp/ui/#iso:std:iso-iec:6429:ed-3:v1:en>.

[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>.

[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>.

[RFC4102] Jones, P., "Registration of the text/red MIME Sub-Type", RFC 4102, DOI 10.17487/RFC4102, June 2005, <https://www.rfc-editor.org/info/rfc4102>.

[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>.

[RFC5630] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, DOI 10.17487/RFC5630, October 2009, <https://www.rfc-editor.org/info/rfc5630>.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <https://www.rfc-editor.org/info/rfc5764>.

[RFC6263] Marjou, X. and A. Sollaud, "Application Mechanism for Keeping Alive the NAT Mappings Associated with RTP / RTP Control Protocol (RTCP) Flows", RFC 6263, DOI 10.17487/RFC6263, June 2011, <https://www.rfc-editor.org/info/rfc6263>.

[RFC7675] Perumal, M., Wing, D., Ravindranath, R., Reddy, T., and M. Thomson, "Session Traversal Utilities for NAT (STUN) Usage for Consent Freshness", RFC 7675, DOI 10.17487/RFC7675, October 2015, <https://www.rfc-editor.org/info/rfc7675>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8865] Holmberg, C. and G. Hellström, "T.140 Real-Time Text Conversation over WebRTC Data Channels", RFC 8865, DOI 10.17487/RFC8865, January 2021, <https://www.rfc-editor.org/info/rfc8865>.

[RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <https://www.rfc-editor.org/info/rfc8866>.

[T140] ITU-T, "Protocol for multimedia application text conversation", ITU-T Recommendation T.140, February 1998, <https://www.itu.int/rec/T-REC-T.140-199802-I/en>.

[T140ad1] ITU-T, "Recommendation T.140 Addendum", February 2000, <https://www.itu.int/rec/T-REC-T.140-200002-I!Add1/en>.

11.2. Informative References

[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>.

[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>.

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

[RFC8643] Johnston, A., Aboba, B., Hutton, A., Jesske, R., and T. Stach, "An Opportunistic Approach for Secure Real-time Transport Protocol (OSRTP)", RFC 8643, DOI 10.17487/RFC8643, August 2019, <https://www.rfc-editor.org/info/rfc8643>.

[RFC8723] Jennings, C., Jones, P., Barnes, R., and A.B. Roach, "Double Encryption Procedures for the Secure Real-Time Transport Protocol (SRTP)", RFC 8723, DOI 10.17487/RFC8723, April 2020, <https://www.rfc-editor.org/info/rfc8723>.

[RFC8825] Alvestrand, H., "Overview: Real-Time Protocols for Browser-Based Applications", RFC 8825, DOI 10.17487/RFC8825, January 2021, <https://www.rfc-editor.org/info/rfc8825>.

Acknowledgements

The author wants to thank the following persons for support, reviews, and valuable comments: Bernard Aboba, Amanda Baber, Roman Danyliw, Spencer Dawkins, Martin Duke, Lars Eggert, James Hamlin, Benjamin Kaduk, Murray Kucherawy, Paul Kyzivat, Jonathan Lennox, Lorenzo Miniero, Dan Mongrain, Francesca Palombini, Colin Perkins, Brian Rosen, Rich Salz, Jürgen Schönwälder, Robert Wilton, Dale Worley, Yong Xin, and Peter Yee.

Author's Address

Gunnar Hellström
Gunnar Hellström Accessible Communication
SE-13670 Vendelsö
Sweden
Email: gunnar.hellstrom@ghaccess.se