Real-Time Text (RTT) Protocol for PEMEA

PEMEA-CONS-Spec-RTT-001

V1.0d

20 July 2021

Contributors

Name	Company
James Winterbottom	Deveryware
Javier Cabas	Deveryware
Miguel Ortega	Deveryware
Gunnar Helstrom	GH Access

Table of Contents

1. Executive Summary

Real-Time Text (RTT) communications are used extensively by people with hearing and speech disabilities around the world. These systems convey letters as they are typed from the source to the destination. The International Telecommunications Union (ITU) defines clear guidelines for what is required to support RTT. This specification defines an RTT protocol, complying with ITU guidelines, for use in the Pan-European Mobile Emergency Application (PEMEA) framework.
The specification in the present document does not preclude PEMEA from being used to support and initiate other RTT protocols or implementations.

2. Terms and Definitions

The following terms and definitions are used in this document:

AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
AESGCM	Advanced Encryption Standard key used with GCM
AP	Application Provider
CA	Certificate Authority
CPE	Customer Premises Equipment
DHE	Diffie-Hellman key Exchange
ECDHE	Elliptic-curve Diffie-Hellman key Exchange
EDS	Emergency Data Send message
ETSI	European Telecommunication Standards Institute
GCM	Galios/Counter Mode
HTTP	Hyper-Text Transfer Protocol
HTTPS	Hyper-Text Transfer Protocol secure
IETF	Internet Engineering Task Force
ITU	International Telecommunications Union
JSON	Java Script Object Notation
MAC	Message Authentication Code
PEMEA	Pan-European Mobile Emergency Application framework
PIM	PSAP Interface Module (to the PEMEA network)
PRA	PEMEA Registration Authority
PSAP	Public Safety Answering Point
RTT	Real-Time Text
RSA	Rivest Shamir Adleman public key encryption algorithm
SHA	Secure Hash Algorithm
TLS	Transport Layer Security
tPSP	Terminating PSAP Service Provider
UCS	Universal Multiple-Octect Coded Character Set
URI	Universal Resource Identifier
UTF-8	UCS Transformation Format (8 bit words)
WSS	WebSocket Secure

3. References

3.1. Normative References

[R.1] "Emergency Communication (EMTEL); Pan-European Mobile Emergency Application", TS 103 478, March 2018, ETSI
[R.2] "Emergency Communications (EMTEL); PEMEA Instant Message Extension", TS 103 756, June 2021, ETSI
[R.3] "Protocol for multimedia application text conversation", T.140, ITU-T
[R.4] IANA language subtag registry. http://www.iana.org/assignments/language-subtag-registry/language-subtag-registry
[R.5] Add the consortium repository for the RTT JSON Schema Here

3.2. Informative References

[IR.1] "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999
[IR.2] "The OAuth 2.0 Authorization Framework: Bearer Token Usage", RFC 6750, October 2012
[IR.3] "JSON Web Token (JWT)", RFC 7519, May 2015

4. Scope

The present document describes the PEMEA Real-Time Text (RTT) capability, the and the need for this functionality. The required entities and actors are identified along with the protocol, specifying message exchanges between entities. The message formats are specified and procedural descriptions of expected behaviours under different conditions are detailed.

The PEMEA RTT capability described in the current document augments the ETSI PEMEA Instant Messaging protocol specification TS 103 756 [R.2] to address transfer expediency requirements stipulated in ITU T140 [R.3]. This approach allows the leveraging of existing protocol work in such a way as to simplifies development for applications and systems that already support the PEMEA chat protocol.

5. PEMEA capability extensions

5.1. Overview of extension in PEMEA

PEMEA extension capabilities are defined in TS 103 478 [R.1] and are implemented through the use of "reach-back" URIs. The Application Provider (AP) node advertises capabilities as part of the initial forward message through the network, the emergencyDataSend(EDS) message, and the terminating PSAP Service Provider (PSP) or PSAP responds with a subset of capabilities that it supports, thus binding the emergency session between the AP and the terminating emergency node.

Specifically, the capabilities are sent as information elements in the apMoreInformation element of the EDS message. The information element and apMoreInformation structures are defined in Clauses 10.3.11 and 10.3.12 of TS 103 478 [R.1]. An information element in a PEMEA EDS message identifies a capability and each capability is made up of three distinct parts:

- typeOfInfo:- what function does the information element serve
- protocol:- the specific semantics for using the function
- value:- the URI through which the service is invoked

Table 10 in TS 103 478 [1] identifies an initial set of "typeOfInfo" values used to specify a range of capability extensions for PEMEA. However, beyond the Location_Update and SIP_Request values described in Table 11 of TS 103 478 [1], protocols are left for further study and definition in subsequent specifications such as the present document.

5.2. Service support indication and response

5.2.1. Service definition

TS 103 478 [R.1] defines the Real-Time Text, "RTT", typeOfInfo in Table 10, but does not elaborate further on protocols in Table 11. The present document provides a concrete definition of the "RTT" typeOfInfo in PEMEA through the specification of a protocol value.

Table 1: PEMEA RTT service definition

Info Type Value	Protocol Token	Description
RTT	PEMEA	Real-Time Text functionality is supported using the PEMEA message exchange protocol

5.2.2. Service support indication

An AP needing to indicate that the Application it is serving can support real-time text using the PEMEA protocol would include the following information element in the apMoreInformation element of the EDS associated with the emergency session:

```
<information typeOfInfo="RTT" protocol="PEMEA">
    https://ap.example.pemea.help/48sne8aopaop
</information>
```

5.2.3. Service support response

A terminating node that can support the "RTT" "PEMEA" capability includes this capability in the apMoreInformation element returned to the AP in the onCapSupportPost. This is described in Clause 11.1.4 of TS 103 478 [R.1] with the value for "RTT" "PEMEA" provided in the example below.

```
<apMoreInformation xmlns="urn:pemea:apps:xml:ns:pemea:base">
    <information typeOfInfo="RTT" protocol="PEMEA"/>
</apMoreInformation>
```

5.2.4. Auto response service

The original intent of emergency applications was to provide ancillary data to the PSAP that was associated with an emergency voice call that the PSAP had, or soon would, receive. As a consequence, a PIM or tPSP usually notifies the PSAP-CPE when an EDS has arrived, but doesn't respond to the AP until a PSAP call-taker has answered the call. Operating in this manner allows for smart routing solutions ensuring that only the PSAP with the call binds the PEMEA session to the AP ensuring that the data is always available to the call-taker rather than it being missing because it went to the wrong PSAP.

TS 103 478 [R.1] identifies some types of capabilities, most notably the SIP_Request capabilities, as being responded to automatically, that is, the PIM or tPSP will send an immediate onCapSupportPost message with all supported capabilities if the EDS contains a SIP_Request capability. This functionality is described in clause 8 of [R.1] and came about because there was no way for the App

to make a voice call until it has a destination SIP URI, so there was no possible way for the data to not available at the destination PSAP.

Another reason for auto-response is that no conventional carrier/mobile voice call will be placed as part of the emergency communication. That is, only PEMEA advanced services will be used for communicating between the caller and the PSAP call-taker.

The PEMEA RTT capability falls into this latter category of services, that is, it is used in place of a conventional carrier/mobile voice call. Consequently, a PSAP (PIM or tPSP) supporting this capability and with the capacity to handle the communication shall respond to the AP with an onCapSupportPost message immediately on receipt of an EDS containing. The onCapSupportPost message shall contain the RTT PEMEA capability along with any other capabilities that the PSAP supports.

If the PSAP does not have the ability or capacity to support the request then it may forward the request to a neighbouring PSAP with whom it has an agreement to do so. In this situation the original PSAP shall not send an onCapSupportPost message to the originating AP.

6. Mapping to T.140

6.1. T.140 special character support

ITU specification T.140 [R.3] defines requirements and procedures for RTT systems. For the most part these are mapped directly. With the movement to modern communications system however, some of the requirements in T.140 are no longer relevant. In other cases, functionality is not provided as it is available through other PEMEA extensions or is supported implicitly through the protocol itself rather than through special characters. The Table 2 indicates which functionality from clause 7 of T.140 [R.3] is supported and how.

Table 2: PEMEA RTT support for T.140 special characters

Name	Supported	Description
BEL	No	No alerting in in the communication is provided
BS	Yes	Backspace character is sent as 0x08, converted to UTF-8, inside a TEXT_MESSAGE
NEW LINE	Yes	New line character is sent as 0x0A, converted to UTF-8, inside a TEXT_MESSAGE
CR LF	No	No-standard and non-preferred, not supported
INT	No	No mode negotiation is required
SGR	No	Not supported
SOS	No	Not supported
ST	No	Not supported
ESC	Yes	This specification supports the sending and receiving of the ESC (0x1B) control character, however, rendering, displaying and interpretation of control sequences is not specified.
Byte order mark	No	Synchronization is not required via a Web Socket

The protocol described in the present document addresses the establishment of connections, disconnections and the transfer of data between entities, it does not attempt to address the display

requirements of T.140 [R.3]. However, the intention from T1.40 Appendix I shall be fulfilled. “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.”

All text is transferred using UTF-8 which can represent most language character sets. The language that the user intends to communicate with is provided in the JOIN message, see clause 9.3. The language shall be specified using one of the languages provided in the language sub-tags registered with IANA [R.4]. The present document does not provide guidance on whether multi-lingual session participants may switch languages during the session or not though the general recommendation is against taking this action.

Text messages consists of one or more characters. Characters are transferred from the App to the AP either in real-time, as they are typed, or in batches at 0.5 second intervals so that a character is always transferred within 0.5 seconds of having been typed. This functionality is described in clause 6.1.1 of T.140 [R.3].

6.2. ESC character sequence support

ESC character sequences in the current specification are a set of characters bounded by ESC characters (0x1B) on either side. For example 0x1B:)0x1B may display a smiley face. The present document does not define any ESC character sequences nor does it provide any guidance on rendering or interpretation beyond all characters between two ESC characters forming the escape sequence.

An entity shall ignore all escape sequence characters if an explicit escape sequence code set has not been established through some other means. The present document leaves the possibility open for a future revision of the specification to define common sets of escape sequences.

The ESC sequence, open ESC character, intermediate characters and closing ESC character shall be sent in a single message and receiver receiving a single erase character shall erase any and all characters in the ESC sequence.

Any message containing a partial ESC sequence shall be ignored.

7. Security

7.1. Transport security

The RTT service is identified to potential session participants as an HTTPS URI. The connection should be made using TLS 1.3 but may be made using 1.2 and this shall not support fallback below TLS 1.2. The connecting participant shall authenticate to the RTT service. Once the connecting entity is authenticated and authorization granted the connection is upgraded to a websocket. The websocket is expected to remain open while the entity is "online". The protocol is resilient to connections being dropped, so an entity may reconnect as long as the EDS session remains active in the PSAP.

The lists for the TLS 1.3 and TLS 1.2 acceptable cipher suites are included in ANNEX A. These lists are informative and are based on best information at the time of writing. Older cipher suites not included in either of these lists shall not be used.

7.2. Security token usage

The HTTP Authorization header field is defined in RFC 2617 [IR.1] and it specifies that the usage is a scheme followed by a value, where the value may have a structure, as is the case for the digest authentication scheme.

Security token usage in the HTTP Authorization header field was originally specified for use with OAuth and is defined in RFC 6750 [IR.2]. Here the use of OAuth "Bearer token" is specified so the scheme of the Authorization header field is Bearer, following the scheme a token is placed. The token is a base64 encoded string.

Token usage in the RTT PEMEA specification follows the Bearer scheme defined in RFC 6750 [IR.2]. Tokens issued by entities in the RTT PEMEA architecture are expected to also be the validating entities, or to have ties to the validating entities, consequently, whether the tokens are opaque or follow a convention such as JWT [IR.3] is not considered relevant to usage and so Is not specified further.

RFC 6750 [IR.2] mandates the usage of TLS for use with Bearer tokens, this usage is further defined in clause 7.1 of the present document.

8. Procedures and signalling

8.1. Service invocation

8.1.1. Service invocation procedures

Once the terminating PSP or PSAP has responded to the AP that it can support the PEMEA RTT service then the AP shall be capable of accepting a service invocation on the provided URI at any time. The AP shall only accept an RTT service invocation from the PIM or tPSP that sent the onCapSupportPost message.
The PSAP invokes the RTT service by:

A. The call-taker initiating their willingness to use RTT to the PEMEA interface module (PIM) in the PSAP or the tPSP
B. The PIM/tPSP requesting the RTT server create an RTT-session room
C. The RTT server creating an RTT-session room and returning a URI to the PIM/tPSP
D. The PIM/tPSP requesting security tokens for the PSAP and AP.
E. The PIM/tPSP returning the URI and associated security token the PSAP call-taker
F. The call-taker connecting to the RTT-session room
G. The PIM/tPSP calling the URI provided by the AP for the RTT-PEMEA service, including the URI for the RTT-session room and the AP security token in this invocation. Note that the URI is the same for the call-taker and the caller.

H. The AP indicates to the App that the PSAP wishes to communicate using RTT with the user.

I. The user indicates their willingness to communicate using RTT with the PSAP to AP

J. The AP initiates a connection to the RTT-session room.

It is important to note that it is always the AP that authenticates to the RTT-session room and consequently all messages from the App shall traverse the AP. The present document only defines the protocol between the AP and other trusted entities e.g. PSAP call-taker or First Responder, and the RTT-session room in the PSAP, it does not define the RTT Pa messaging between the App and the AP.

8.1.2. Service invocation object

The PIM/tPSP invokes the RTT PEMEA service in the AP by posting to the URI provided in the RTT PEMEA information element included in the apMoreInformation contained in the EDS. The POST message includes a body containing a JSON object. The JSON object provides the RTT-session room URL as well as a security token and corresponding expiry time.

The JSON schema for the RTT service invocation message is provided in ANNEX B for completeness but may downloaded from [R.5].

Table 3: Invocation object fields

Element Name	Presence	Description
uri	Mandatory	The uri of the RTT-session room.
token	Mandatory	A security token used to authenticate the AP to the RTT-session room. The AP shall include the token in the HTTP Authorization header. The AP shall use the token each time it needs to establish or re-establish a connection to the RTT-session room for the duration of the App emergency session. The AP shall not provide the token to the App
expiry	Mandatory	Specifies the expiry time of the security token. expiry is an integer specifying the number of second since UTC epoch, 00:00:00 1st of January 1970.

Invocation example:

```
{
    "uri": "https://rtt-server.example.com/session/534wafds21s21fdf",
    "token": "PPtzs5zzG5Pkf61KPz51",
    "expiry": "1590563357576"
}
```

8.2. RTT-session room creation and deletion

The RTT-session room is created by the RTTserver under direction of the PSAP call-taker via the PSAP interface module (PIM) or tPSP. When the RTT-session room is created, a logging function shall be created with it to scribe all messages into and out of the room. This flow is shown in Figure 1.

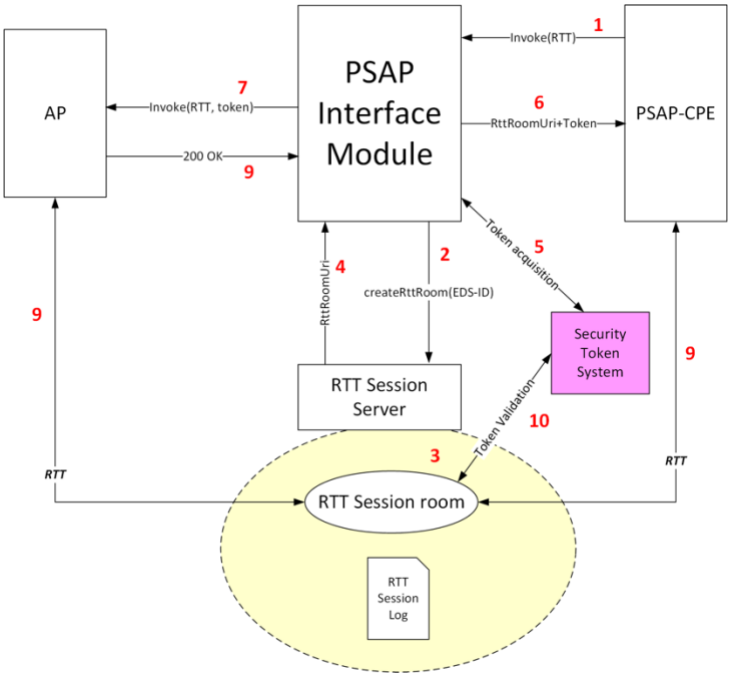

Figure 1 RTT session initiation

Once the RTT-session room is created it remains active as long as the PIM or tPSP maintains a context for the EDS. When EDS context is deleted the RTT-session room is also destroyed.

8.3. RTT-session room creation, JOINT and TEXT_MESSAGE signalling

8.3.1. Semantics

The figures in the following sub clauses show the signalling involved in establishing and subsequently joining a PEMEA RTT session. By necessity the diagrams show four distinctive types of signalling:

- Semantic signalling across the Pa interface between the App and the AP is explicitly not defined in PEMEA. So, while the message names and contents may not align with any specific implementation, the semantics of what the messages convey should be understood.
- Core PEMEA signalling are explicit messages defined in the PEMEA technical specification TS 103 478 [R.1].
- RTT semantic signalling is messaging that needs to occur between the PSAP call-taker equipment, the PIM/tPSP and the software entities and components required to establish the RTT service. These messages are intended to provide an idea of what needs to occur, not how it should be implement. Consequently, they are informative only and not normative.
- RTT normative signalling messages and semantics explicitly defined in the present document.

8.3.2. RTT service invocation

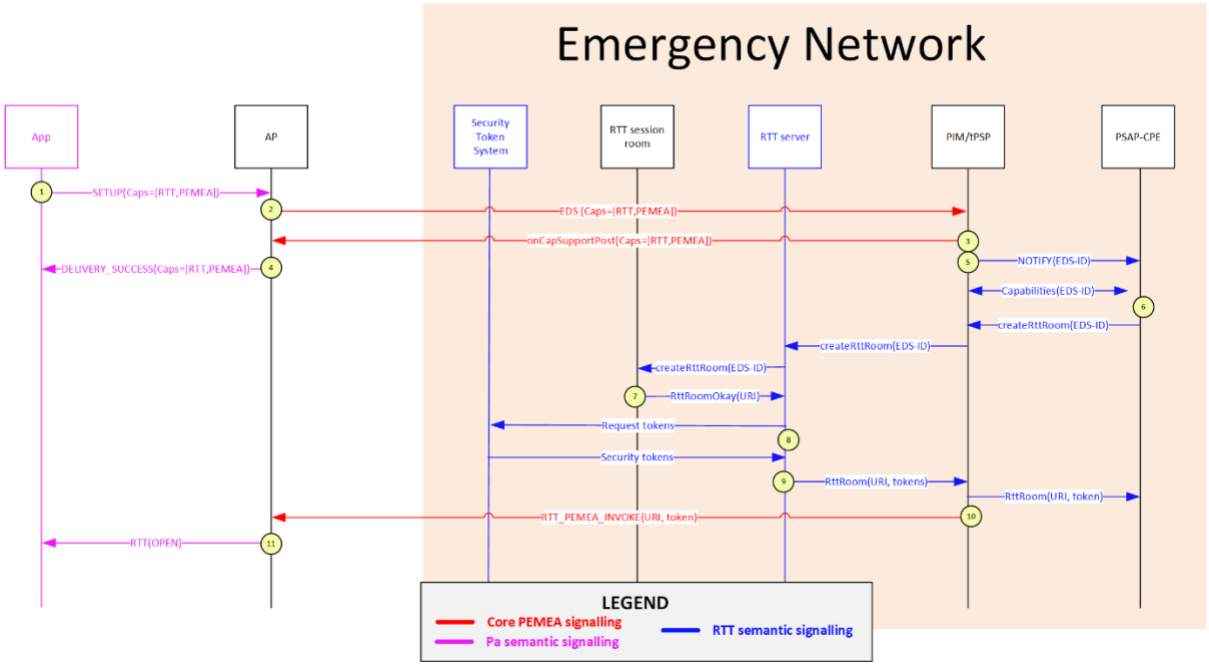

Figure 2: RTT service invocation

1. App initiates and emergency session with the AP over the Pa interface indicating that it can support the Real-Time text communications.
2. The AP creates an EDS message from the data provided by the App and includes the PEMEA RTT protocol capability. The AP then sends the EDS into the PEMEA network.
3. The EDS arrives the PIM/tPSP. The PIM supports the PEMEA RTT capability and includes this option in the onCapupportPost back to the AP.
4. The AP binds the emergency session to the PIM that sent the onCapSupportPost message and then signals to the App over the Pa interface that the PSAP can support the PEMEA RTT functionality.
5. The PIM notifies the PSAP-CPE that a new EDS has arrived
6. The PSAP call-taker via the PSAP-CPE requests the capabilities agreed with the AP, sees the PEMEA RTT capability and requests the PIM to initiate the creation of an RTT-session room. The PIM request that the RTT server create an RTT-session room
7. The RTT server creates an RTT-session room. The RTT-session room initializes its log functions and internal status and returns its contact URI to the RTT server.
8. The RTT server requests two security tokens for use with an RTT-session room from the security token system. The security token system validates the RTT-server and provides it with two RTT-session room tokens.
9. The RTT server returns the RTT-session room URI and the security tokens to the PIM. The PIM returns the RTT-session room URI and one of the security tokens to the call-taker via the PSAP-CPE.
10. The PIM invokes the PEMEA RTT capability in the AP using the URI contain provided in the capability sent in the EDS. The PIM includes the RTT-session room URI, a security token along with the token expiry time in the body of the HTTP POST used to invoke the capability in the AP.
11. The AP signals to the App over the Pa interface that the PSAP has invoked the RTT communication capability.

8.3.3. JOIN message flow

Once the RTTservice has been invoked in the AP, then the PSAP call-taker and Caller join the RTT-session room.

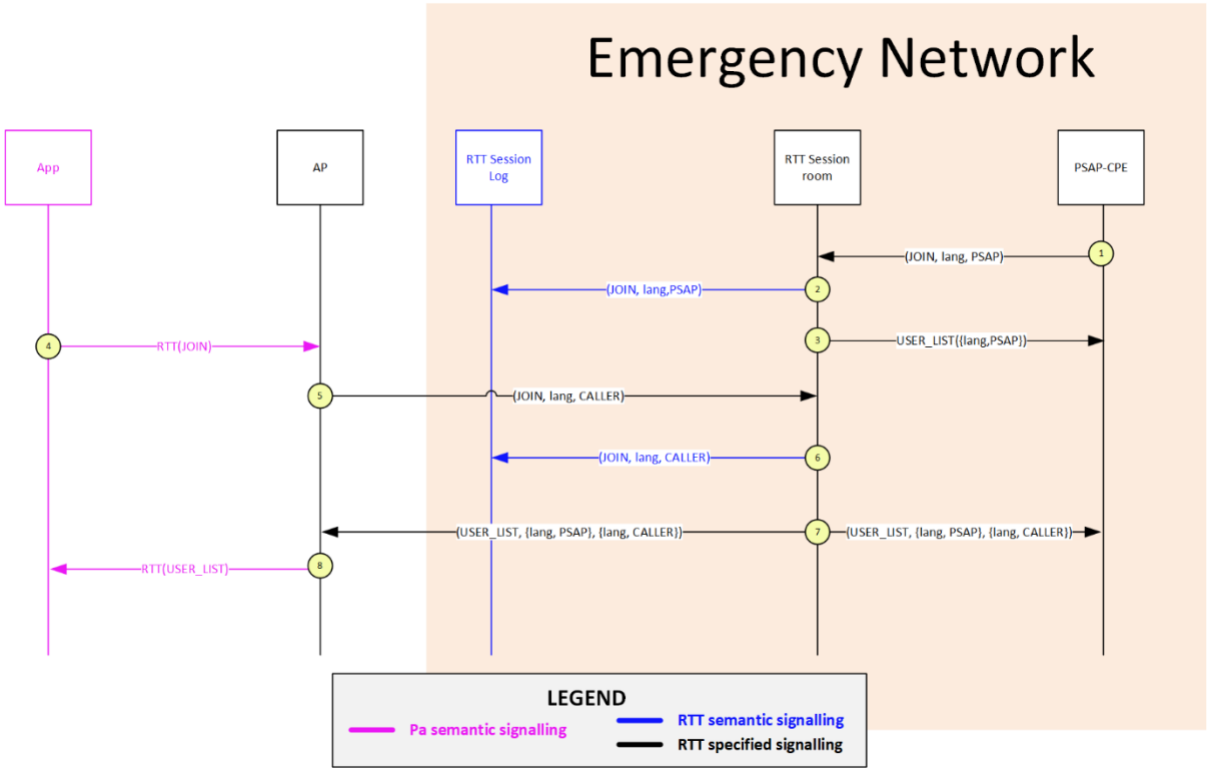

Figure 3: RTT JOIN message flow

1. The PSAP call-taker joins the RTT-session room.
2. The log is written to the session log indicating that the PSAP has joined the RTT-session.
3. The RTT-session room responds to the PSAP with the current USER_LIST, which only contains the PSAP.
4. The App signals to the AP over the Pa interface that it wishes to join the RTT session.
5. The AP connects to the RTT-session room including the authentication token in the Authorization HTTP header field. The RTT-session room authenticates the AP and the connection is promoted to a websocket.
   The AP then sends a JOIN message to the RTT-session room indicating that the connecting entity is a CALLER and includes the caller's name and uniqueId.
6. The RTT-session room accepts the JOIN message and writes it to the session log.
7. The RTT-session room then sends a USER_LIST message to both the PSAP and the AP containing the PSAP and CALLER information.
8. The AP relays the USER_LIST to the App over the Pa interface.

8.3.4. ERROR message flow

The RTT-session room sending, and user receiving, a USER_LIST message indicates a successful joining of the RTT session, a join request may also be rejected by the RTT-session room by sending an ERROR message.

A JOIN message shall be rejected by the RTT-session room if the uniqueId is already in use by an active/ONLINE user. When this occurs the WebSocket is also closed and the user shall create a new uniqueId and shall re-connect and send a new JOIN message. The security token shall remain valid.

The RTT-session room shall log the JOIN message and the ERROR message. However, the user list is not updated so no USER_LIST is subsequently sent to other session participants when an ERROR message has been sent in response to the JOIN message.

8.3.5. TEXT_MESSAGE flow

Once the AP has joined the RTT-session room it is able to send text messages to the other participants in the session.

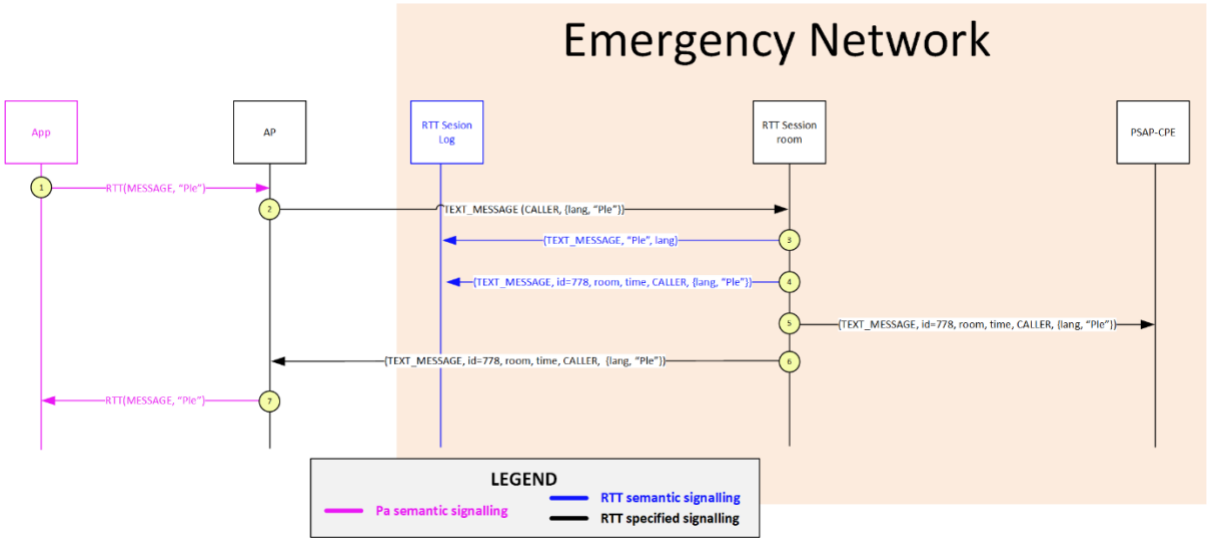

Figure 4: RTT TEXT_MESSAGE signalling flow

1. The App user sends the characters that the user types to the AP over the Pa interface. These characters may include new line or backspace characters as identified in clause 6.
2. The AP buffers the user's characters for up to 0.5 seconds then creates a TEXT_MESSAGE for transmission to the RTT-session room and sends the TEXT_MESSAGE to the session room.
3. The RTT-session room receives the message and writes it to the session log.
4. The RTT-session room adds a unique message id to the message, along with the room identifier and a time stamp and writes message this to the session log.
5. The RTT-session room sends the new TEXT_MESSAGE to the PSAP call-taker.
6. The RTT-session room sends the new TEXT_MESSAGE to the AP.
7. The AP may relay the message to the App, indicating that the characters have been successfully delivered to the other parties in the RTT-session.

8.4. Disconnects and reconnects

Despite communications networks being reliable, accidental disconnects owing to temporary issues do still occur. PEMEA does not define how the AP and the App communicate, though some high-level semantics for RTT are described in the present document. The present document describes communication between the authorized entities and the RTT-session room, most commonly the AP and the PSAP-CPE.
If the RTT-session room terminates for an unexpected reason, then the websockets used for communication between the participants and the RTT-session room will close. Should this occur,

then the participants should attempt to reconnect, with an ever-increasing exponential back-off. Participants shall reconnect using the same uniqueId (see clause 8.3.3). Failure to reconnect after a configurable period should result in the participant not attempting to continue to retry. When this occurs for the PSAP call-taker, the PSAP call-taker may request a new session be created and this will then follow the creation process described in clause 8.2.
When this occurs, the new session may have a new URI, so a new RTT-session invocation is sent to the AP including the new RTT-session room URI. The new invocation shall include a new token and expiry time. When this occurs then a participant shall connect using the same uniqueId as was used for the previous session (see clause 8.3.3), this ensures that logs can be aligned.
Since the session log provides a transcript of what information has been exchanged between room participants, it shall be persistent, so in the event of a new room creation due to a system failure of some kind, the same session log shall continue to be used in the new RTT-session room.
On receipt of the new RTT invocation, the AP shall auto join the newly provided RTT-session room URI and communication is re-established. The AP does not need to report to the App the loss of connectivity to the RTT-session room until it determines that the connectivity cannot be restored. A simplified version of this flow is provided in Figure 5.

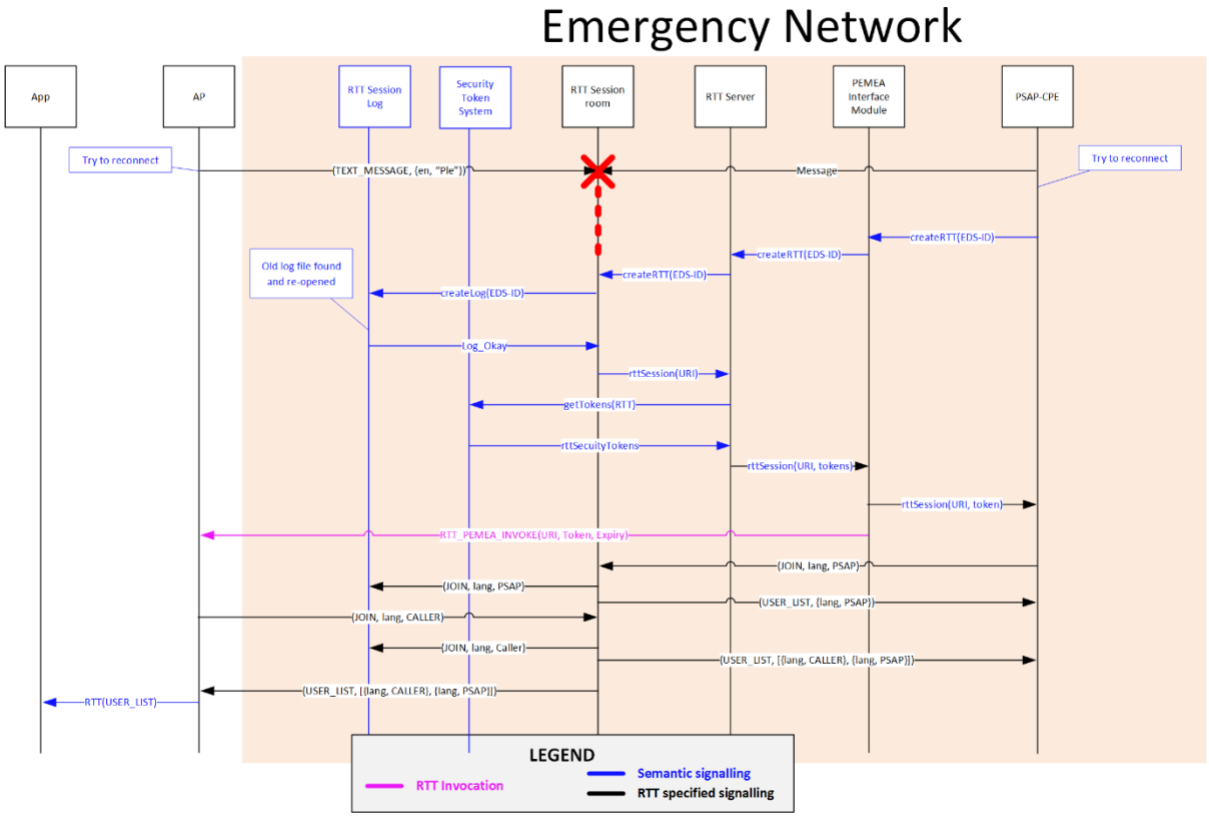

Figure 5: PEMEA RTT reconnection signalling

9. RTT PEMEA message and type definitions

9.1. Overview

The RTT PEMEA protocol messages are defined as a series of JSON documents exchanged between the AP or PEMEA terminating node and an "RTT-session room" established inside the secure emergency network. The RTT-session room is established solely for communications with a single

emergency session. Each emergency session requiring the use of the RTT PEMEA service has its own RTT-session room created. Service and message exchanges between the AP and the App are not defined in the present document and are left to application implementers.

The JSON specifications for the messages are maintained in a repository outside of this specification and are available for download from the PEMEA consortium web site. The subsequent clauses in this section of the document describe each of the RTT PEMEA messages, its function, elements and any key constraints. Messages exchanges and procedures are specified in clause 8

The JSON schema for the main protocol messages defined in the clause 9 are provided in ANNEX C for completeness but may be downloaded from the PEMEA consortium repository [R.5].

Table 4: Types of RTT PEMEA messages

Message type	Description
JOIN	Message sent from the AP to the RTT-session room when the user wants to join the RTT-session.
ERROR	Sent by the RTT-session room in the event that a JOIN request is received containing a uniqueId already in use by an active/ONLINE user.
USER_LIST	Message sent from the RTT-session room to the AP containing all users whenever a user enters or leaves the RTT-session.
TEXT_MESSAGE	Message sent either from the AP to the RTT-session room, or from the RTT-session room to the AP containing a user's characters. Message history is transferred as text messages. History is sent when a user joins the RTT-session room.

The participants leave the RTT-session by breaking their connection to the RTT-session room, so no explicit leave message is defined for this protocol.

9.2. Data types

9.2.1. language

Is the language that the user will be communicating in through the RTT session. The language may be any of the pertinent languages from the IANA language subtag registry [R.4].

9.2.2. room

Is a unique string providing a name for the RTT-session room. This is usually the URI used to specify the room attachment provided when the RTT service is invoked by the PSAP.

9.2.3. timestamp

All messages are sent with a timestamp and to avoid offsets, timezones, daylight savings changes etc, the time is always absolute. It is specified as an integer in milliseconds since the UTC epoch of 00:00:00 1st January 1970.

9.2.4. user

Defines a user in the RTT-session room. It consists of:

- name
- role
- uniqueId

The name is a string that identifies a handle to which the user relates, this may be their name "George" or their telephone number tel:+34666554433 for example.
The role defines the type of user that is associated with the name. These values are presently constrained and defined in **Table 5** but maybe extended at any time by adding the new values to a registry managed by the PEMEA consortium.

Table 5: Role values

Role	Description
CALLER	The value sent by the AP to the chat-room and used to identify the user initiating the emergency communication to all other participants in the RTT session.
PSAP	The value sent by the PSAP Call-Taker to the chat-room and used to identify the Call-Taker to all other participants in the RTT session.
POLICE	If the police are linked into the chat-room then this value is sent by them to identify that police are in the chat to all other participants in the RTT session.
FIREFIGHTER	If the fire department are linked into the chat-room then this value is sent by them to identify that firefighters are in the chat to all other participants in the RTT session.
MED	If the ambulance or medical services are linked into the chat-room then this value is sent by them to identify that they are in the chat to all other participants in the RTT session.
OTHER	A role that does not directly fall into one of the previous categories and has not been registered as a formal role.

The uniqueId shall be generated by the AP or PSAP-CPE and is used to unqiuely identify the message stream in the event that more than one user in the session uses the same name and role. The uniqueId needs to have little chance of collisions with other generated identities and so should not be based purely on static data, such the name and role, and should be large enough so that collisions are avoided. Attempts to join an RTT session with a uniqueId already use shall result in a rejection of the join request.

9.2.5. userInfo

Is used to combine information about the user:

- user:- defined in clause 9.2.4
- language:- language defined in clause 9.2.1
- status

The status field is used to describe what the user is doing:

Table 6: UserInfo status values

Status	Description
ONLINE	The user is in the RTT-session. This may be a new user joining the RTT-session, or maybe an existing user connected to the RTT-session room.
OFFLINE	The user was, but is no-longer, in the RTT session. The RTT-session room may only use this status as an indicator that a user has left the RTT session and then delete knowledge of the connection, or it may maintain a list of all users that have ever joined the RTT session.

9.2.6. message

Is a container used to convey text by a user. The language that message is written in was specified by the user when they joined the RTT-session. The language that the user is conversing in is disseminated to the other RTT-session participants in the userInfo element of the USER_LIST.

Table 7: message container elements

Element NAME	Description
message	The information to be conveyed

9.3. JOIN message

9.3.1. Message overview

The JOIN message is the message sent from the participant to the RTT-session room when the user wants to join the session. This may be the AP, the PSAP-CPE or another trusted user. The JOIN message is resent if for some reason the connection between the entity and the RTT-session room is lost but the RTT session is not concluded.
The JOIN message consists of the following required fields:

Table 8: JOIN message fields and description

Element Name	Description	
type	"JOIN" The message being sent by the end-point to the RTT-session room	
user	name	Name and role of entity joining the RTT session. The Name may be the user's name or their telephone number.
	role	The role will depend on the type of user joining the RTT session. In the case of the user initiating the emergency communication this will be "CALLER"
	uniqueId	uniqueId for this user in the RTT session
language	Is the language that the user will be communicating in over RTT. It shall be a language from the IANA language subtag registry [R.4].	
since	Send all messages after this time. The time is specified as milliseconds since epoch. A value of zero means send all messages When a participant is connecting to the RTT-session room for the first time then it will send a "since" value of zero, indicating that it wants all messages. For example, an AP may do this in case the PSAP call-taker joined the RTT-session before the caller did. This will ensure that when the history is sent from the RTT-session room to the AP that the AP receives all messages in chronological order.	

9.3.2. Examples

The JOIN message is also used to reconnect to the RTT-session room in the case that the connection terminated, AP or RTT server restarted. In this case, the AP will set the since value to be the time that the AP knew it last had a connection to the RTT-session room, often this will be last received message from the RTT-session room. On a successful connection, the RTT-session room will send all messages that have occurred "since" the specified time.

```
{
    "language":"es",
```

```
    "since":0,
    "type":"JOIN",
    "user":{
        "name":"PSAP-IXHJh219",
        "role":"PSAP",
        "uniqueId":"jgh204nq9md"
    }
}
```

9.4. ERROR message

The ERROR message is sent by the RTT-session room in response to a JOIN request containing a uniqueId already n use by an active/ONLINE user in the session. The message is not intended to be visible to the end-user since either the AP or call-taker CPE will generate the uniqueId automatically. Refer to clause 8.3.4 for procedures on message rejection.

Table 9: RTT JOIN ERROR message

Element Name	Description
type	"ERROR" The message being sent by the RTT-session room in response to the JOIN request
room	String identifying the RTT-session room
reasonCode	This is a token indicating why the JOIN was refused by the RTT-session room. The tokens are defined in the PEMEA consortium registry, PEMEA-RTT-ERROR-Tokens: The present document defined the token "idInUse" and is set when a JOIN request containing an active/online uniqueId is received.
reason	An optional text string indicating the reason for the error.
timestamp	Integer Number of milliseconds from epoch (00:00:00:00 1st January 1970). See clause 9.2.3

Example:

```
{
    "type":"ERROR",
    "room":"ttRRkzORz",
    "reasonCode":"idInUse"
    "reason":"uniqueId sh4786793384881h32jh already in use",
    "timestamp":1574092280231
}
```

9.5. USER_LIST message

The USER_LIST message is sent to all participants in the RTT session whenever a user enters and leaves the RTT session.

Table 10: USER_LIST message fields and description

Element Name	Description
type	"USER_LIST" The message being sent by the RTT-session room to the participants
room	String identifying the RTT-session room
timestamp	Integer Number of milliseconds from epoch (00:00:00:00 1st January 1970). See clause9.2.3
users	An array of userInfo containing the name, role, uniqueId and status of each user in the RTT session. See clause 9.2.4.

Each participant in the chat-room is required to keep a list of the participants so that it knows when participants join and leave the chat.

```
{
    "type":"USER_LIST",
    "room":"ttRRkzORz",
    "users":[
        {
            "language":"es",
            "user":{
                "name":"George",
                "role":"CALLER",
                "uniqueId":"jgh204nq9md"
            },
            "status":"OFFLINE"
        },
        {
            "language":"es",
            "user":{
                "name":"PSAP-IXHJh219",
                "role":"PSAP",
               "uniqueId":"ljfvgtsy26540"
            },
            "status":"ONLINE"
        }
    ],
    "timestamp":1574092280231
}
```

9.6. TEXT_Message message

The text message is used by a RTT session participant to contribute to send real-time written text to other participants in the RTT session. Every message has an identifier associated with RTT-session room so that it can be uniquely identified.
After the RTT-session room server receives a TEXT_MESSAGE message from a user, it with send it to all the participants in the RTT-session, including the sender.
Text message fields are different depending on who the sender is. When the sender of text message is a participant, then Table 11 represents the required fields:

Table 11: TEXT_MESSAGE message fields and description for participant message

Element Name	Description
type	"TEXT_MESSAGE" refer to **Table** 4
message	The details of the text being sent. Refer to clause 9.2.6

```
{
    "type":"TEXT_MESSAGE",
    "message":"hola"
}
```

The RTT-session room is responsible for relaying text messages to all the participants and the text messages it sends consists of all of the fields in Table 12.

Table 12: TEXT_MESSAGE message fields and description for a chat-server message

Element Name	Description
id	Unique identifier for this message within the RTT-session.
type	"TEXT_MESSAGE" refer to **Table** 4
room	The identifier for the RTT-session room.
timestamp	The time that the message was sent. Refer to clause 9.2.3
user	The user sending or that sent the text message. Refer to clause 9.2.4 The receiving entities should use the value uniqueId element to align character streams against user conversations.
message	The details of the text being sent and the language in which the message is composed. Refer to clause 9.2.6.

```
{
    "id":"5dd2bd8ba5568000079fa11c",
    "type":"TEXT_MESSAGE",
    "message":"text":"holajd\b\b",
    "room":"ttRRkzORz",
    "user":{
        "name":"PSAP-XqwFbQ-A",
        "role":"PSAP",
        "uniqueId":"jgh204nq9md"
    },
    "timestamp":1574092171988
}
```

10. Logging requirements

The RTT-session room is responsible for all session logging. It shall log all messages into and out of the room. This ensures that in the event of an audit there is a trail showing that the messages were sent to a specific room participant.
Message logging shall include all characters sent by a user, including backspace characters so that in the event of an incident, the entire history of the channel can be viewed.

ANNEX A Cipher Suites

Table 13: Recommended TLS 1.3 cipher suites

Cipher	TLS version	Encryption	MAC
TLS_AES_128_GCM_SHA256	1.3	AESGCM(128)	AEAD
TLS_AES_256_GCM_SHA384	1.3	AESGCM(256)	AEAD
TLS_CHACHA20_POLY1305_SHA256	1.3	CHACHA20/POLY1305(256)	AEAD

Table 14: Acceptable TLS 1.2 cipher suites

Cipher	TLS version	Encryption	MAC
ECDHE-ECDSA-AES128-GCM-SHA256	1.2	AESGCM(128)	AEAD
ECDHE-RSA-AES128-GCM-SHA256	1.2	AESGCM(128)	AEAD
ECDHE-ECDSA-AES256-GCM-SHA384	1.2	AESGCM(256)	AEAD
ECDHE-RSA-AES256-GCM-SHA384	1.2	AESGCM(256)	AEAD
ECDHE-ECDSA-CHACHA20-POLY1305	1.2	CHACHA20/POLY1305(256)	AEAD
ECDHE-RSA-CHACHA20-POLY1305	1.2	CHACHA20/POLY1305(256)	AEAD
DHE-RSA-AES128-GCM-SHA256	1.2	AESGCM(128)	AEAD
DHE-RSA-AES256-GCM-SHA384	1.2	AESGCM(256)	AEAD

ANNEX B RTT Invocation Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT invocation schema",
    "properties": {
        "token": {
            "type": "string"
        },
        "expiry": {
            "type": "number"
        },
        "uri": {
            "type": "string",
            "format": "uri"
       }
    },
    "required": ["uri", "token", "expiry"]
}
```

ANNEX C RTT Protocol Schema

C.1 JOIN SCHEMA

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT JOIN message Schema",
    "properties": {
        "type": {
            "const": "JOIN"
        },
        "language": {
            "type": "string"
        },
        "since": {
            "type": "number"
        },
        "user": {
            "$ref": "#/definitions/user"
        }
    },
    "definitions": {
        "user": {
            "type": "object",
            "properties": {
                "name": {
                    "type": "string"
                },
                "role": {
                    "$ref": "#/definitions/role"
                },
                "uniqueId": {
                    "type": "string"
                }
            },
            "required": ["name", "role", "uniqueId"]
        },
        "role": {
            "type": "string"
        }
    },
    "required": ["language","since","user", "type"]
}
```

C.2 UserList Schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT USER_LIST message Schema",
    "properties": {
        "type": {
            "const": "USER_LIST"
        },
        "room": {
          "type": "string"
        },
        "users": {
            "type": "array",
             "items": {
               "$ref": "#/definitions/userStatus"
             }
        },
        "timestamp": {
```

```
            "type": "number"
        }
    },
    "definitions": {
        "userStatus": {
          "type": "object",
          "required": ["language", "user", "status"],
          "properties": {
            "language": { "type": "string"},
            "user": {"$ref": "#/definitions/user"},
            "status": {
              "type": "string",
              "enum": ["OFFLINE", "ONLINE"]
            }
          }


        },
        "user": {
            "type": "object",
            "properties": {
                "name": {
                    "type": "string"
                },
                "role": {
                    "type": "string"
                },
                "uniqueId": {
                    "type": "string"
                }
            },
            "required": ["name", "role", "uniqueId"]
        }
    },
    "required": ["type", "room", "users", "timestamp"]
}
```

C.3 Text Message Schema for participant

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT TextMessage message Schema for participant",
    "properties": {
        "type": {
            "const": "TEXT_MESSAGE"
        },
        "message": {
            "type": "string"
        },
    },
    "required": ["message", "type"]
}
```

C.4 TextMessage Schema for rtt-server

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT TextMessage message Schema for rtt-server",
    "properties": {
        "type": {
            "const": "TEXT_MESSAGE"
        },
```

```
        "room": "{
            "type": "string"
        },
        "message": {
            "type": "string"
        },
        "user": {
            "$ref": "#/definitions/user"
        },
        "timestamp": {
            "type": "number"
        }
    },
    "definitions": {
        "user": {
            "type": "object",
            "properties": {
                "name": {
                    "type": "string"
                },
                "role": {
                    "type": "string"
                },
                "uniqueId": {
                    "type": "string"
                }
            },
            "required": ["name", "role", "uniqueId"]
        }
    },
    "required": ["message", "type", "room", "user", "timestamp"]
}
```

C.5 Error schema

```
{
    "$schema": "http://json-schema.org/draft-07/schema",
    "type": "object",
    "title": "RTT Error  Schema",
    "properties": {
        "type": {
            "const": "ERROR"
        },
        "room": {
          "type": "string"
        },
        "reasonCode": {
          "type": "string"
        },
        "reason": {
          "type": "string"
        },
        "timestamp": {
            "type": "number"
        }
    },
    "required": ["type", "room", "reasonCode", "reason", "timestamp"]
}
```

HISTORY

Document history		
V0.1	10 Jun 2021	Initial Draft
V0.2	11 June 2021	Updated after review. Significant expansion of the security sections. Removed language from text messages Changed user languages to exactly one language Added uniqueId to user in addition to the name and role to ensure a unique stream for client reconciliation. Added REJECT message in response to JOIN in the event that the uniqueId if duplicated.
V0.3	17 June 2021	Added more text around language usage. Changed REJECT message to be ERROR message and added a reasonCode token. Fixed a couple of errors Removed "text" element from "message" since message is no longer a structure. Add a reference to a yet to be defined repository for keeping the JSON schemas for download. Added appendices to put the current schemas into. Added a section about auto response for this capability since no voice call is ever expected with this function.
V0.4	18 June 2021	Interim changes with schema added
V0.5	21 June 2021	Changes to T.140 section based on Gunnar input and review
V0.6	22 June 2021	Minor changes to Schema
V1.0a	05 July 2021	Incorporated final comments from Gunnar. Stated \b sent as 0x08 converted to UTF-8. \n sent as 0x0A converted to UTF-8 Indicated that the ESC character shall be ignored if received and may be used in a revision of the spec to end emojis or pictograms.
V1.0b	09 July 2021	Addressed the issue raid by Gunnar on ESC character sequence support.
V1.0c	15July 2021	Table number for section 9.4 and more text around ESC sequences.
V1.0d	20 July 2021	final