

CLUE WG
Internet-Draft
Intended status: Standards Track
Expires: January 17, 2013

R. Even
Huawei Technologies
J. Lennox
Vidyo
July 16, 2012

Mapping RTP streams to CLUE media captures
draft-even-clue-rtp-mapping-03.txt

Abstract

This document describes mechanisms and recommended practice for mapping RTP media streams defined in SDP to CLUE media captures.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 17, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. RTP topologies for CLUE	3
4. Mapping CLUE Media Captures to RTP streams	5
4.1. Static Mapping	6
4.2. Dynamic mapping	7
4.3. Recommendations	7
5. Application to CLUE Media Requirements	7
6. Examples	9
7. Acknowledgements	9
8. IANA Considerations	9
9. Security Considerations	10
10. References	10
10.1. Normative References	10
10.2. Informative References	10
Authors' Addresses	11

1. Introduction

Telepresence systems can send and receive multiple media streams. The CLUE framework [I-D.ietf-clue-framework] defines media captures as a source of Media, such as from one or more Capture Devices. A Media Capture (MC) may be the source of one or more Media streams. A Media Capture may also be constructed from other Media streams. A middle box can express Media Captures that it constructs from Media streams it receives.

SIP offer answer [RFC3264] uses SDP [RFC4566] to describe the RTP[RFC3550] media streams. Each RTP stream has a payload type number and SSRC. The content of the RTP stream is created by the encoder in the endpoint. This may be an original content from a camera or a content created by an intermediary device like an MCU.

This document makes recommendations, for this telepresence architecture, about how RTP and RTCP streams should be encoded and transmitted, and how their relation to CLUE Media Captures should be communicated. The proposed solution supports multiple RTP topologies

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119[RFC2119] and indicate requirement levels for compliant RTP implementations.

3. RTP topologies for CLUE

The typical RTP topologies used by telepresence systems specify different behaviors for RTP and RTCP distribution. The relevant topologies include point-to-point, as well as media mixers, media-switching mixers, and source-projection mixers.

In the point-to-point topology, one peer communicates directly with a single peer over unicast. There can be one or more RTP sessions, and each RTP session can carry multiple RTP streams identified by their SSRC. All SSRCs will be recognized by the peers based on the information in the RTCP SDES report that will include the CNAME and SSRC of the sent RTP streams. There are different use point to point use cases as specified in CLUE use case [I-D.ietf-clue-telepresence-use-cases]. There may be a difference between the symmetric and asymmetric use cases. While in the symmetric use case the typical mapping will be from a Media capture device to a render device (e.g. camera to monitor) in the asymmetric

case the render device may receive different capture information (RTP stream from a different camera) if it has fewer rendering devices (monitors). In some cases, a CLUE session which, at a high-level, is point-to-point may nonetheless have RTP which is best described by one of the mixer topologies below. For example, a CLUE endpoint can produce composited or switched captures for use by a receiving system with fewer displays than the sender has cameras.

In the Media Mixer topology, the peers communicate only with the mixer. The mixer provides mixed or composited media streams, using its own SSRC for the sent streams. There are two cases here. In the first case the mixer may have separate RTP sessions with each peer (similar to the point to point topology) terminating the RTCP sessions on the mixer; this is known as Topo-RTCP-Terminating MCU in [RFC5117]. In the second case, the mixer can use a conference-wide RTP session similar to RFC 5117's Topo-mixer or Topo-Video-switching. The major difference is that for the second case, the mixer uses conference-wide RTP sessions, and distributes the RTCP reports to all the RTP session participants, enabling them to learn all the CNAMEs and SSRCs of the participants and know the contributing source or sources (CSRCs) of the original streams from the RTP header. In the first case, the Mixer terminates the RTCP and the participants cannot know all the available sources based on the RTCP information. The conference roster information including conference participants, endpoints, media and media-id (SSRC) can be available using the conference event package [RFC4575] element.

In the Media-Switching Mixer topology, the peer to mixer communication is unicast with mixer RTCP feedback. It is conceptually similar to a compositing mixer as described in the previous paragraph, except that rather than compositing or mixing multiple sources, the mixer provides one or more conceptual sources selecting one source at a time from the original sources. The Mixer creates a conference-wide RTP session by sharing remote SSRC values as CSRCs to all conference participants.

In the Source-Projection Mixer topology, the peer to mixer communication is unicast with RTCP mixer feedback. Every potential sender in the conference has a source which is "projected" by the mixer into every other session in the conference; thus, every original source is maintained with an independent RTP identity to every receiver, maintaining separate decoding state and its original RTCP SDES information. However, RTCP is terminated at the mixer, which might also perform reliability, repair, rate adaptation, or transcoding on the stream. Senders' SSRCs may be renumbered by the mixer. The sender may turn the projected sources on and off at any time, depending on which sources it thinks are most relevant for the receiver; this is the primary reason why this topology must act as an

RTP mixer rather than as a translator, as otherwise these disabled sources would appear to have enormous packet loss. Source switching is accomplished through this process of enabling and disabling projected sources, with the higher-level semantic assignment of reason for the RTP streams assigned externally.

The above topologies demonstrate two major RTP/RTCP behaviors:

1. The mixer may either use the source SSRC when forwarding RTP packets, or use its own created SSRC. Still the mixer will distribute all RTCP information to all participants creating conference-wide RTP session/s. This allows the participants to learn the available RTP sources in each RTP session. The original source information will be the SSRC or in the CSRC depending on the topology. The point to point case behaves like this.
2. The mixer terminates the RTCP from the source, creating separate RTP sessions with the peers. In this case the participants will not receive the source SSRC in the CSRC. Since this is usually a mixer topology, the source information is available from the SIP conference event package [RFC4575] Subscribing to the conference event package allows each participant to know the SSRCs of all sources in the conference.

4. Mapping CLUE Media Captures to RTP streams

The different topologies described in Section 3 support different SSRC distribution models and RTP stream multiplexing points.

Most video conferencing systems today can separate multiple RTP sources by placing them into separate RTP sessions using, the SDP description. For example, main and slides video sources are separated into separate RTP sessions based on the content attribute [RFC4796]. This solution works straightforwardly if the multiplexing point is at the UDP transport level, where each RTP stream uses a separate RTP session. This will also be true for mapping the RTP streams to Media Captures if each media capture uses a separate RTP session, and the consumer can identify it based on the receiving RTP port. In this case, SDP only needs to label the RTP session with an identifier that identifies the media capture in the CLUE description. In this case, it does not change the mapping even if the RTP session is switched using same or different SSRC. (The multiplexing is not at the SSRC level).

Even though Session multiplexing is supported by CLUE, for scaling reasons, CLUE recommends using SSRC multiplexing in a single or

multiple sessions. So we need to look at how to map RTP streams to Media Capture IDs when SSRC multiplexing is used.

When looking at SSRC multiplexing we can see that in various topologies, the SSRC behavior may be different:

1. The SSRCs are static (assigned by the MCU/Mixer), and there is an SSRC for each media capture encoding defined in the CLUE protocol. Source information may be conveyed using CSRC, or, in the case of topo-RTCP-Terminating MCU, is not conveyed.
2. The SSRCs are dynamic, representing the original source and are relayed by the Mixer/MCU to the participants.

In the above two cases the MCU/Mixer creates its own advertisement, with a virtual room capture scene.

Another case we can envision is that the MCU / Mixer relays all the capture scenes from all advertisements to all consumers. This means that the advertisement will include multiple capture scenes, each representing a separate TP room with its own coordinate system. A general tools for distributing roster information is by using an event package, for example by extending the conference event package.

4.1. Static Mapping

Static mapping is widely used in current MCU implementations. It is also common for a point to point symmetric use case when both endpoints have the same capabilities. For capture encodings with static SSRCs, it is most straightforward to indicate this mapping outside the media stream, in the CLUE or SDP signaling. An SDP source attribute [RFC5576] could be defined to associate CLUE capture IDs with SSRCs in SDP. Each SSRC will have a captureID value that will be specified also in the CLUE media capture as an attribute. The provider advertisement could, if it wished, use the same SSRC for media capture encodings that are mutually exclusive. (This would be natural, for example, if two advertised captures are implemented as different configurations of the same physical camera, zoomed in or out.)

Note: there may be more than one RTP session for a media capture like in simulcast. We still need to figure out how to describe it in SDP and CLUE.

Another method for static mapping may be to use the provider advertisement could to indicate the intended SSRC directly. The advantage of using the SDP SSRC attribute is that RFC5576 [RFC5576] the issue of SSRC collision and provide guideline how to address

them.

4.2. Dynamic mapping

Dynamic mapping using RTP header extension is described in draft-lennox-clue-rtp-usage [I-D.lennox-clue-rtp-usage] section 10.2. The draft does not specify what is the capture id value. As specified for the static case there should be a capture id attribute in the CLUE media capture information to enable this mapping.

4.3. Recommendations

The recommendation is that endpoints MUST support both the static declaration of capture encoding SSRCs, and the RTP header extension method of sharing capture IDs, with the extension in every media packet. For low bandwidth situations, this may be considered excessive overhead; in which case endpoints MAY support the combined approach from [I-D.lennox-clue-rtp-usage]. The SDP offer MAY specify the SSRC mapping to media capture. In the case of static mapping topologies there will be no need to use the header extensions in the media, since the SSRC for the RTP stream will remain the same during the call unless a collision is detected and handled according to RFC5576 [RFC5576]. If the used topology uses dynamic mapping then the RTP header extension will be used to indicate the RTP stream switch for the media capture. In this case the SDP description may be used to negotiate the initial SSRC but this will be left for the implementation. Note that if the SSRC is defined explicitly in the SDP the SSRC collision should be handled as in RFC5576.

5. Application to CLUE Media Requirements

[I-D.lennox-clue-rtp-usage] offers a number of requirements that are believed to be necessary for a CLUE RTP mapping. The solutions described in this document are believed to meet that requirement, though some of them are only possible for some of the topologies. (Since the requirements are generally of the form "it must be possible for a sender to do something", this is adequate; a sender which wishes to perform that action needs to choose a topology which allows the behavior it wants.

In this section we address only those requirements where the topologies or the association mechanisms treat the requirements differently.

Media-4: It must be possible for an original source to move among switched captures (i.e. at one time be sent for one switched capture, and at a later time be sent for another one).

This applies naturally for static sources with a Switched Mixer. For dynamic sources with a Source-Projecting Mixer, this just requires the capture tag in the header extension element to be updated appropriately.

Media-6: Whenever a given source is transmitted for a switched capture, it must be immediately possible for a receiver to determine the switched capture it corresponds to, and thus that any previous source is no longer being mapped to that switched capture.

For a Switched Mixer, this applies naturally. For a Source-Projecting mixer, this is done based on the header extension.

Media-7: It must be possible for a receiver to identify the original source that is currently being mapped to a switched capture, and correlate it with out-of-band (non-Clue) information such as rosters.

For a Switched Mixer, this is done based on the CSRC, if the mixer is providing CSRCs; if for a Source-Projecting Mixer, this is done based on the SSRC.

Media-8: It must be possible for a source to move among switched captures without requiring a refresh of decoder state (e.g., for video, a fresh I-frame), when this is unnecessary. However, it must also be possible for a receiver to indicate when a refresh of decoder state is in fact necessary.

This can be done by a Source-Projecting Mixer, but not by a Switching Mixer. The last requirement can be accomplished through an FIR message [RFC5104], though potentially a faster mechanism (not requiring a round-trip time from the receiver) would be preferable.

Media-9: If a given source is being sent on the same transport flow to satisfy more than one capture (e.g. if it corresponds to more than one switched capture at once, or to a static capture as well as a switched capture), it should be possible for a sender to send only one copy of the source.

For a Source-Projecting Mixer, this can be accomplished by sending multiple dynamic capture IDs for the same source; this can also be done for an environment with a hybrid of mixer topologies and static and dynamic captures, described below in Section 6. It is not possible for static captures from a Switched Mixer.

Media-12: If multiple sources from a single synchronization context are being sent simultaneously, it must be possible for a receiver to associate and synchronize them properly, even for sources that are mapped to switched captures.

For a Mixed or Switched Mixer topology, receivers will see only a single synchronization context (CNAME), corresponding to the mixer. For a Source-Projecting Mixer, separate projecting sources keep separate synchronization contexts based on their original CNAMEs, thus allowing independent synchronization of sources from independent rooms without needing global synchronization. In hybrid cases, however (e.g. if audio is mixed), all sources which need to be synchronized with the mixed audio must get the same CNAME (and thus a mixer-provided timebase) as the mixed audio.

6. Examples

It is possible for a CLUE device to send multiple instances of the topologies in Section 3 simultaneously. For example, an MCU which uses a traditional audio bridge with switched video would be a Mixer topology for audio, but a Switched Mixer or a Source-Projecting Mixer for video. In the latter case, the audio could be sent as a static source, whereas the video could be dynamic.

More notably, it is possible for an endpoint to send the same sources both for static and dynamic captures. Consider the example in Section 11.1 of [I-D.ietf-clue-framework], where an endpoint can provide both three cameras (VC0, VC1, and VC2) for left, center, and right views, as well as a switched view (VC3) of the loudest panel.

It is possible for a consumer to request both the (VC0 - VC2) set and VC3. It is worth noting that the content of VC3 is, at all times, exactly the content of one of VC0, VC1, or VC2. Thus, if the sender uses the Source-Selection Mixer topology for VC3, the consumer that receives these three sources would not need to send any additional media traffic over just sending (VC0 - VC2).

In this case, the advertiser could describe VC0, VC1, and VC2 in its initial advertisement or SDP with static SSRCs, whereas VC3 would need to be dynamic. The role of VC3 would move among VC0, VC1, or VC2, indicated by the RTP header extension on those streams' RTP packets.

7. Acknowledgements

place holder

8. IANA Considerations

TBD

9. Security Considerations

TBD.

10. References

10.1. Normative References

- [I-D.ietf-clue-framework]
Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino,
"Framework for Telepresence Multi-Streams",
draft-ietf-clue-framework-05 (work in progress),
February 2012.
- [I-D.lennox-clue-rtp-usage]
Lennox, J., Witty, P., and A. Romanow, "Real-Time
Transport Protocol (RTP) Usage for Telepresence Sessions",
draft-lennox-clue-rtp-usage-04 (work in progress),
June 2012.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

- [I-D.ietf-clue-telepresence-use-cases]
Romanow, A., Botzko, S., Duckworth, M., Even, R., and I.
Communications, "Use Cases for Telepresence Multi-
streams", draft-ietf-clue-telepresence-use-cases-02 (work
in progress), January 2012.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model
with Session Description Protocol (SDP)", RFC 3264,
June 2002.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
Jacobson, "RTP: A Transport Protocol for Real-Time
Applications", STD 64, RFC 3550, July 2003.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session
Description Protocol", RFC 4566, July 2006.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session
Initiation Protocol (SIP) Event Package for Conference
State", RFC 4575, August 2006.
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description

Protocol (SDP) Content Attribute", RFC 4796, February 2007.

- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.
- [RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, June 2009.
- [RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, June 2010.
- [RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, May 2011.
- [RFC6236] Johansson, I. and K. Jung, "Negotiation of Generic Image Attributes in the Session Description Protocol (SDP)", RFC 6236, May 2011.

Authors' Addresses

Roni Even
Huawei Technologies
Tel Aviv,
Israel

Email: even.roni@huawei.com

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com

CLUE WG
Internet-Draft
Intended status: Standards Track
Expires: August 5, 2013

R. Even
Huawei Technologies
J. Lennox
Vidyo
February 1, 2013

Mapping RTP streams to CLUE media captures
draft-even-clue-rtp-mapping-05.txt

Abstract

This document describes mechanisms and recommended practice for mapping RTP media streams defined in SDP to CLUE media captures.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 5, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. RTP topologies for CLUE	3
4. Mapping CLUE Media Captures to RTP streams	5
4.1. Review of current directions in MMUSIC, AVText and AVTcore	6
4.2. Requirements of a solution	7
4.3. Static Mapping	9
4.4. Dynamic mapping	9
4.4.1. RTP header extension	10
4.4.2. Restricted approach	10
4.5. Recommendations	11
5. Application to CLUE Media Requirements	11
6. Examples	13
6.1. Static mapping	13
6.2. Dynamic Mapping	16
7. Acknowledgements	16
8. IANA Considerations	16
9. Security Considerations	17
10. References	17
10.1. Normative References	17
10.2. Informative References	17
Authors' Addresses	19

1. Introduction

Telepresence systems can send and receive multiple media streams. The CLUE framework [I-D.ietf-clue-framework] defines media captures as a source of Media, such as from one or more Capture Devices. A Media Capture (MC) may be the source of one or more Media streams. A Media Capture may also be constructed from other Media streams. A middle box can express Media Captures that it constructs from Media streams it receives.

SIP offer answer [RFC3264] uses SDP [RFC4566] to describe the RTP[RFC3550] media streams. Each RTP stream has a unique SSRC within its RTP session. The content of the RTP stream is created by an encoder in the endpoint. This may be an original content from a camera or a content created by an intermediary device like an MCU.

This document makes recommendations, for this telepresence architecture, about how RTP and RTCP streams should be encoded and transmitted, and how their relation to CLUE Media Captures should be communicated. The proposed solution supports multiple RTP topologies.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119[RFC2119] and indicate requirement levels for compliant RTP implementations.

3. RTP topologies for CLUE

The typical RTP topologies used by Telepresence systems specify different behaviors for RTP and RTCP distribution. A number of RTP topologies are described in [I-D.westerlund-avtcore-rtp-topologies-update]. For telepresence, the relevant topologies include point-to-point, as well as media mixers, media- switching mixers, and source-projection mixers.

In the point-to-point topology, one peer communicates directly with a single peer over unicast. There can be one or more RTP sessions, and each RTP session can carry multiple RTP streams identified by their SSRC. All SSRCs will be recognized by the peers based on the information in the RTCP SDES report that will include the CNAME and SSRC of the sent RTP streams. There are different point to point use cases as specified in CLUE use case [I-D.ietf-clue-telepresence-use-cases]. There may be a difference

between the symmetric and asymmetric use cases. While in the symmetric use case the typical mapping will be from a Media capture device to a render device (e.g. camera to monitor) in the asymmetric case the render device may receive different capture information (RTP stream from a different camera) if it has fewer rendering devices (monitors). In some cases, a CLUE session which, at a high-level, is point-to-point may nonetheless have RTP which is best described by one of the mixer topologies below. For example, a CLUE endpoint can produce composited or switched captures for use by a receiving system with fewer displays than the sender has cameras.

In the Media Mixer topology, the peers communicate only with the mixer. The mixer provides mixed or composited media streams, using its own SSRC for the sent streams. There are two cases here. In the first case the mixer may have separate RTP sessions with each peer (similar to the point to point topology) terminating the RTCP sessions on the mixer; this is known as Topo-RTCP-Terminating MCU in [RFC5117]. In the second case, the mixer can use a conference-wide RTP session similar to RFC 5117's Topo-mixer or Topo-Video-switching. The major difference is that for the second case, the mixer uses conference-wide RTP sessions, and distributes the RTCP reports to all the RTP session participants, enabling them to learn all the CNAMEs and SSRCs of the participants and know the contributing source or sources (CSRCs) of the original streams from the RTP header. In the first case, the Mixer terminates the RTCP and the participants cannot know all the available sources based on the RTCP information. The conference roster information including conference participants, endpoints, media and media-id (SSRC) can be available using the conference event package [RFC4575] element.

In the Media-Switching Mixer topology, the peer to mixer communication is unicast with mixer RTCP feedback. It is conceptually similar to a compositing mixer as described in the previous paragraph, except that rather than compositing or mixing multiple sources, the mixer provides one or more conceptual sources selecting one source at a time from the original sources. The Mixer creates a conference-wide RTP session by sharing remote SSRC values as CSRCs to all conference participants.

In the Source-Projection Mixer topology, the peer to mixer communication is unicast with RTCP mixer feedback. Every potential sender in the conference has a source which is "projected" by the mixer into every other session in the conference; thus, every original source is maintained with an independent RTP identity to every receiver, maintaining separate decoding state and its original RTCP SDES information. However, RTCP is terminated at the mixer, which might also perform reliability, repair, rate adaptation, or transcoding on the stream. Senders' SSRCs may be renumbered by the

mixer. The sender may turn the projected sources on and off at any time, depending on which sources it thinks are most relevant for the receiver; this is the primary reason why this topology must act as an RTP mixer rather than as a translator, as otherwise these disabled sources would appear to have enormous packet loss. Source switching is accomplished through this process of enabling and disabling projected sources, with the higher-level semantic assignment of reason for the RTP streams assigned externally.

The above topologies demonstrate two major RTP/RTCP behaviors:

1. The mixer may either use the source SSRC when forwarding RTP packets, or use its own created SSRC. Still the mixer will distribute all RTCP information to all participants creating conference-wide RTP session/s. This allows the participants to learn the available RTP sources in each RTP session. The original source information will be the SSRC or in the CSRC depending on the topology. The point to point case behaves like this.
2. The mixer terminates the RTCP from the source, creating separate RTP sessions with the peers. In this case the participants will not receive the source SSRC in the CSRC. Since this is usually a mixer topology, the source information is available from the SIP conference event package [RFC4575]. Subscribing to the conference event package allows each participant to know the SSRCs of all sources in the conference.

4. Mapping CLUE Media Captures to RTP streams

The different topologies described in Section 3 support different SSRC distribution models and RTP stream multiplexing points.

Most video conferencing systems today can separate multiple RTP sources by placing them into separate RTP sessions using, the SDP description. For example, main and slides video sources are separated into separate RTP sessions based on the content attribute [RFC4796]. This solution works straightforwardly if the multiplexing point is at the UDP transport level, where each RTP stream uses a separate RTP session. This will also be true for mapping the RTP streams to Media Captures if each media capture uses a separate RTP session, and the consumer can identify it based on the receiving RTP port. In this case, SDP only needs to label the RTP session with an identifier that identifies the media capture in the CLUE description. In this case, it does not change the mapping even if the RTP session is switched using same or different SSRC. (The multiplexing is not at the SSRC level).

Even though Session multiplexing is supported by CLUE, for scaling reasons, CLUE recommends using SSRC multiplexing in a single or multiple sessions. So we need to look at how to map RTP streams to Media Captures when SSRC multiplexing is used.

When looking at SSRC multiplexing we can see that in various topologies, the SSRC behavior may be different:

1. The SSRCS are static (assigned by the MCU/Mixer), and there is an SSRC for each media capture encoding defined in the CLUE protocol. Source information may be conveyed using CSRC, or, in the case of topo-RTCP-Terminating MCU, is not conveyed.
2. The SSRCS are dynamic, representing the original source and are relayed by the Mixer/MCU to the participants.

In the above two cases the MCU/Mixer creates its own advertisement, with a virtual room capture scene.

Another case we can envision is that the MCU / Mixer relays all the capture scenes from all advertisements to all consumers. This means that the advertisement will include multiple capture scenes, each representing a separate TP room with its own coordinate system. A general tools for distributing roster information is by using an event package, for example by extending the conference event package.

4.1. Review of current directions in MMUSIC, AVText and AVTcore

Editor's note: This section provides an overview of the RFCs and drafts that can be used as a base for a mapping solution. This section is for information only, and if the WG thinks that it is the right direction, the authors will bring the required work to the relevant WGs.

The solution needs to also support the simulcast case where more than one RTP session may be advertised for a Media Capture. Support of such simulcast is out of scope for CLUE.

When looking at the available tools based on current work in MMUSIC, AVTcore and AVText for supporting SSRC multiplexing the following documents are considered to be relevant.

SDP Source attribute [RFC5576] mechanisms to describe specific attributes of RTP sources based on their SSRC.

Negotiation of generic image attributes in SDP [RFC6236] provides the means to negotiate the image size. The image attribute can be used to offer different image parameters like size but in order to offer

multiple RTP streams with different resolutions it does it using separate RTP session for each image option.

[I-D.westerlund-avtcore-max-ssrc] proposes a signaling solution for how to use multiple SSRCS within one RTP session.

[I-D.westerlund-avtext-rtcp-sdes-srcname] provides an extension that may be send in SDP, as an RTCP SDES information or as an RTP header extension that uniquely identifies a single media source. It defines an hierarchical order of the SRCNAME parameter that can be used to for example to describe multiple resolution from the same source (see section 5.1 of [I-D.westerlund-avtcore-rtp-simulcast]). Still all the examples are using RTP session multiplexing.

Other documents reviewed by the authors but are currently not used in a proposed solution include:

[I-D.lennox-mmusic-sdp-source-selection] specifies how participants in a multimedia session can request a specific source from a remote party.

[I-D.westerlund-avtext-codec-operation-point](expired) extends the codec control messages by specifying messages that let participants communicate a set of codec configuration parameters.

Using the above documents it is possible to negotiate the max number of received and sent RTP streams inside an RTP session (m-line or bundled m-line). This allows also offering allowed combinations of codec configurations using different payload type numbers

Examples: max-recv-ssrc:{96:2 & 97:3} where 96 and 96 are different payload type numbers. Or max-send-ssrc{*:4}.

In the next sections, the document will propose mechanisms to map the RTP streams to media captures addressing.

4.2. Requirements of a solution

This section lists, more briefly, the requirements a media architecture for Clue telepresence needs to achieve, summarizing the discussion of previous sections. In this section, RFC 2119 [RFC2119] language refers to requirements on a solution, not an implementation; thus, requirements keywords are not written in capital letters.

Media-1: It must not be necessary for a Clue session to use more than a single transport flow for transport of a given media type (video or audio).

Media-2: It must, however, be possible for a Clue session to use multiple transport flows for a given media type where it is considered valuable (for example, for distributed media, or differential quality-of-service).

Media-3: It must be possible for a Clue endpoint or MCU to simultaneously send sources corresponding to static, to composited, and to switched captures, in the same transport flow. (Any given device might not necessarily be able send all of these source types; but for those that can, it must be possible for them to be sent simultaneously.)

Media-4: It must be possible for an original source to move among switched captures (i.e. at one time be sent for one switched capture, and at a later time be sent for another one).

Media-5: It must be possible for a source to be placed into a switched capture even if the source is a "late joiner", i.e. was added to the conference after the receiver requested the switched source.

Media-6: Whenever a given source is assigned to a switched capture, it must be immediately possible for a receiver to determine the switched capture it corresponds to, and thus that any previous source is no longer being mapped to that switched capture.

Media-7: It must be possible for a receiver to identify the actual source that is currently being mapped to a switched capture, and correlate it with out-of-band (non-Clue) information such as rosters.

Media-8: It must be possible for a source to move among switched captures without requiring a refresh of decoder state (e.g., for video, a fresh I-frame), when this is unnecessary. However, it must also be possible for a receiver to indicate when a refresh of decoder state is in fact necessary.

Media-9: If a given source is being sent on the same transport flow for more than one reason (e.g. if it corresponds to more than one switched capture at once, or to a static capture), it should be possible for a sender to send only one copy of the source.

Media-10: On the network, media flows should, as much as possible, look and behave like currently-defined usages of existing protocols; established semantics of existing protocols must not be redefined.

Media-11: The solution should seek to minimize the processing burden for boxes that distribute media to decoding hardware.

Media-12: If multiple sources from a single synchronization context are being sent simultaneously, it must be possible for a receiver to associate and synchronize them properly, even for sources that are mapped to switched captures.

4.3. Static Mapping

Static mapping is widely used in current MCU implementations. It is also common for a point to point symmetric use case when both endpoints have the same capabilities. For capture encodings with static SSRCS, it is most straightforward to indicate this mapping outside the media stream, in the CLUE or SDP signaling. An SDP source attribute [RFC5576] can be used to associate CLUE capture IDs with SSRCS in SDP. Each SSRCS will have a captureID value that will be specified also in the CLUE media capture as an attribute. The provider advertisement could, if it wished, use the same SSRCS for media capture encodings that are mutually exclusive. (This would be natural, for example, if two advertised captures are implemented as different configurations of the same physical camera, zoomed in or out.). Section 6 provide an example of an SDP offer and CLUE advertisement.

4.4. Dynamic mapping

Dynamic mapping by tagging each media packet with the capture ID. This means that a receiver immediately knows how to interpret received media, even when an unknown SSRCS is seen. As long as the media carries a known capture ID, it can be assumed that this media stream will replace the stream currently being received with that capture ID.

This gives significant advantages to switching latency, as a switch between sources can be achieved without any form of negotiation with the receiver. [RFC5285] recommends that header extensions must be used with caution.

However, the disadvantage in using a capture ID in the stream that it introduces additional processing costs for every media packet, as capture IDs are scoped only within one hop (i.e., within a cascaded conference a capture ID that is used from the source to the first MCU is not meaningful between two MCUs, or between an MCU and a receiver), and so they may need to be added or modified at every stage.

As capture IDs are chosen by the media sender, by offering a particular capture to multiple recipients with the same ID, this requires the sender to only produce one version of the stream (assuming outgoing payload type numbers match). This reduces the

cost in the multicast case, although does not necessarily help in the switching case.

An additional issue with putting capture IDs in the RTP packets comes from cases where a non-CLUE aware endpoint is being switched by an MCU to a CLUE endpoint. In this case, we may require up to an additional 12 bytes in the RTP header, which may push a media packet over the MTU. However, as the MTU on either side of the switch may not match, it is possible that this could happen even without adding extra data into the RTP packet. The 12 additional bytes per packet could also be a significant bandwidth increase in the case of very low bandwidth audio codecs.

4.4.1. RTP header extension

The capture ID could be carried within the RTP header extension field, using [RFC5285]. This is negotiated within the SDP i.e.

```
a=extmap:1 urn:ietf:params:rtp-hdrext:clue-capture-id
```

Packets tagged by the sender with the capture ID will then contain a header extension as shown below

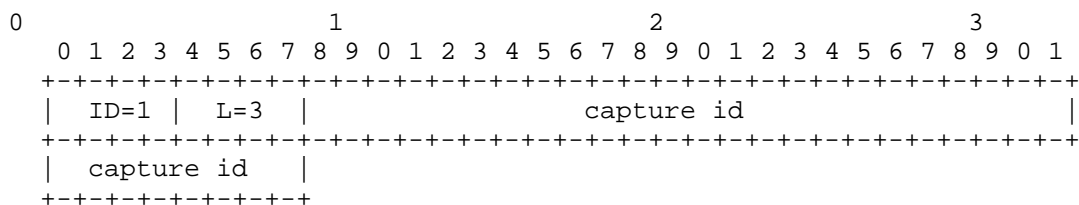


Figure : RTP header extension for encoding of the capture ID

To add or modify the capture ID can be an expensive operation, particularly if SRTP is used to authenticate the packet. Modification to the contents of the RTP header requires a reauthentication of the complete packet, and this could prove to be a limiting factor in the throughput of a multipoint device. However, it may be that reauthentication is required in any case due to the nature of SDP. SDP permits the receiver to choose payload types, meaning that a similar option to modify the payload type in the packet header will cause the need to reauthenticate.

4.4.2. Restricted approach

The flaws of the Capture ID method (high latency switching of SSRC multiplexing, high computational cost on switching nodes) can be

mitigated by sending Capture ID only on some packets of a stream. In this, the capture ID can be included in packets belonging to the first frame of media (typically an IDR/GDR) following a change in the dynamic mapping. Following this, the SSRC is used to map sources to capture IDs.

Note: in the dynamic case there is a need to verify how it will work if not all RTP streams of the same media type are multiplexed in a single RTP session.

4.5. Recommendations

The recommendation is that endpoints **MUST** support both the static declaration of capture encoding SSRCs, and the RTP header extension method of sharing capture IDs, with the extension in every media packet. For low bandwidth situations, this may be considered excessive overhead; in which case endpoints **MAY** support the approach where capture IDs are sent selectively. The SDP offer **MAY** specify the SSRC mapping to media capture. In the case of static mapping topologies there will be no need to use the header extensions in the media, since the SSRC for the RTP stream will remain the same during the call unless a collision is detected and handled according to RFC5576 [RFC5576]. If the used topology uses dynamic mapping then the RTP header extension will be used to indicate the RTP stream switch for the media capture. In this case the SDP description may be used to negotiate the initial SSRC but this will be left for the implementation. Note that if the SSRC is defined explicitly in the SDP the SSRC collision should be handled as in RFC5576.

5. Application to CLUE Media Requirements

The requirement section Section 4.2 offers a number of requirements that are believed to be necessary for a CLUE RTP mapping. The solutions described in this document are believed to meet these requirements, though some of them are only possible for some of the topologies. (Since the requirements are generally of the form "it must be possible for a sender to do something", this is adequate; a sender which wishes to perform that action needs to choose a topology which allows the behavior it wants.

In this section we address only those requirements where the topologies or the association mechanisms treat the requirements differently.

Media-4: It must be possible for an original source to move among switched captures (i.e. at one time be sent for one switched capture, and at a later time be sent for another one).

This applies naturally for static sources with a Switched Mixer. For dynamic sources with a Source-Projecting Mixer, this just requires the capture tag in the header extension element to be updated appropriately.

Media-6: Whenever a given source is transmitted for a switched capture, it must be immediately possible for a receiver to determine the switched capture it corresponds to, and thus that any previous source is no longer being mapped to that switched capture.

For a Switched Mixer, this applies naturally. For a Source-Projecting mixer, this is done based on the header extension.

Media-7: It must be possible for a receiver to identify the original source that is currently being mapped to a switched capture, and correlate it with out-of-band (non-Clue) information such as rosters.

For a Switched Mixer, this is done based on the CSRC, if the mixer is providing CSRCs; if for a Source-Projecting Mixer, this is done based on the SSRC.

Media-8: It must be possible for a source to move among switched captures without requiring a refresh of decoder state (e.g., for video, a fresh I-frame), when this is unnecessary. However, it must also be possible for a receiver to indicate when a refresh of decoder state is in fact necessary.

This can be done by a Source-Projecting Mixer, but not by a Switching Mixer. The last requirement can be accomplished through an FIR message [RFC5104], though potentially a faster mechanism (not requiring a round-trip time from the receiver) would be preferable.

Media-9: If a given source is being sent on the same transport flow to satisfy more than one capture (e.g. if it corresponds to more than one switched capture at once, or to a static capture as well as a switched capture), it should be possible for a sender to send only one copy of the source.

For a Source-Projecting Mixer, this can be accomplished by sending multiple dynamic capture IDs for the same source; this can also be done for an environment with a hybrid of mixer topologies and static and dynamic captures, described below in Section 6. It is not possible for static captures from a Switched Mixer.

Media-12: If multiple sources from a single synchronization context are being sent simultaneously, it must be possible for a receiver to associate and synchronize them properly, even for sources that are mapped to switched captures.

For a Mixed or Switched Mixer topology, receivers will see only a single synchronization context (CNAME), corresponding to the mixer. For a Source-Projecting Mixer, separate projecting sources keep separate synchronization contexts based on their original CNAMEs, thus allowing independent synchronization of sources from independent rooms without needing global synchronization. In hybrid cases, however (e.g. if audio is mixed), all sources which need to be synchronized with the mixed audio must get the same CNAME (and thus a mixer-provided timebase) as the mixed audio.

6. Examples

It is possible for a CLUE device to send multiple instances of the topologies in Section 3 simultaneously. For example, an MCU which uses a traditional audio bridge with switched video would be a Mixer topology for audio, but a Switched Mixer or a Source-Projecting Mixer for video. In the latter case, the audio could be sent as a static source, whereas the video could be dynamic.

More notably, it is possible for an endpoint to send the same sources both for static and dynamic captures. Consider the example in Section 11.1 of [I-D.ietf-clue-framework], where an endpoint can provide both three cameras (VC0, VC1, and VC2) for left, center, and right views, as well as a switched view (VC3) of the loudest panel.

It is possible for a consumer to request both the (VC0 - VC2) set and VC3. It is worth noting that the content of VC3 is, at all times, exactly the content of one of VC0, VC1, or VC2. Thus, if the sender uses the Source-Selection Mixer topology for VC3, the consumer that receives these three sources would not need to send any additional media traffic over just sending (VC0 - VC2).

In this case, the advertiser could describe VC0, VC1, and VC2 in its initial advertisement or SDP with static SSRCs, whereas VC3 would need to be dynamic. The role of VC3 would move among VC0, VC1, or VC2, indicated by the RTP header extension on those streams' RTP packets.

6.1. Static mapping

Using the video capture example from the framework for a three camera system with four monitors where one is for the presentation stream [I-D.ietf-clue-framework] document:

- o VC0- (the camera-left camera stream, purpose=main, switched:no

- o VC1- (the center camera stream, purpose=main, switched:no
- o VC2- (the camera-right camera stream), purpose=main, switched:no
- o VC3- (the loudest panel stream), purpose=main, switched:yes
- o VC4- (the loudest panel stream with PiPs), purpose=main, composed=true; switched:yes
- o VC5- (the zoomed out view of all people in the room), purpose=main, composed=no; switched:no
- o VC6- (presentation stream), purpose=presentation, switched:no

Where the physical simultaneity information is:

{VC0, VC1, VC2, VC3, VC4, VC6}

{VC0, VC2, VC5, VC6}

In this case the provider can send up to six simultaneous streams and receive four one for each monitor. This is the maximum case but it can be further limited by the capture scene entries which may propose sending only three camera streams and one presentation, still since the consumer can select any media captures that can be sent simultaneously the offer will specify 6 streams where VC5 and VC1 are using the same resource and are mutually exclusive.

In the Advertisement there may be two capture scenes:

The first capture scene may have four entries:

{VC0, VC1, VC2}

{VC3}

{VC4}

{VC5}

The second capture scene will have the following single entry.

{VC6}

We assume that an intermediary will need to look at CLUE if want to have better decision on handling specific RTP streams for example based on them being part of the same capture scene so the SDP will not group streams by capture scene.

The SIP offer may be

```
m=video 49200 RTP/AVP 99

a=extmap:1 urn:ietf:params:rtp-hdrex:clue-capture-id / for support
of dynamic mapping

a=rtpmap:99 H264/90000

a=max-send-ssrc:{*:6}

a=max-recv-ssrc:{*:4}

a=ssrc:11111 CaptureID:1

a=ssrc:22222 CaptureID:2

a=ssrc:33333 CaptureID:3

a=ssrc:44444 CaptureID:4

a=ssrc:55555 CaptureID:5

a=ssrc:66666 CaptureID:6
```

In the above example the provider can send up to five main streams and one presentation stream.

We define a new Media Capture ID attribute CaptureID which will have the mapping of the related RTP stream

Note that VC1 and VC5 have the same SSRC since they are using the same resource.

- o VC0- (the camera-left camera stream, purpose=main, switched:no, CaptureID =1
- o VC1- (the center camera stream, purpose=main, switched:no, CaptureID =2
- o VC2- (the camera-right camera stream), purpose=main, switched:no, CaptureID =3
- o VC3- (the loudest panel stream), purpose=main, switched:yes, CaptureID =4
- o VC4- (the loudest panel stream with PiPs), purpose=main, composed=true; switched:yes, CaptureID =5

- o VC5- (the zoomed out view of all people in the room),
purpose=main, composed=no; switched=no, CaptureID =2
- o VC6- (presentation stream), purpose=presentation, switched=no,
CaptureID =6

Note: We can allocate an SSRC for each MC which will not require the indirection of using a CaptureId. This will require if a switch to dynamic is done to provide information about which SSRC is being replaced by the new one.

6.2. Dynamic Mapping

For topologies that use dynamic mapping there is no need to provide the SSRCs in the offer (they may not be available if the offers from the sources will not include them when connecting to the mixer or remote endpoint) In this case the captureID (srcname) will be specified first in the advertisement.

The SIP offer may be

```
m=video 49200 RTP/AVP 99

a=extmap:1 urn:ietf:params:rtp-hdrex:clue-capture-id

a=rtpmap:99 H264/90000

a=max-send-ssrc:{*:4}

a=max-recv-ssrc:{*:4}
```

This will work for ssrc multiplex. It is not clear how it will work when RTP streams of the same media are not multiplexed in a single RTP session. How to know which encoding will be in which of the different RTP sessions.

7. Acknowledgements

The authors would like to thanks Allyn Romanow and Paul Witty for contributing text to this work.

8. IANA Considerations

TBD

9. Security Considerations

TBD.

10. References

10.1. Normative References

[I-D.ietf-clue-framework]

Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-06 (work in progress), July 2012.

[I-D.lennox-clue-rtp-usage]

Lennox, J., Witty, P., and A. Romanow, "Real-Time Transport Protocol (RTP) Usage for Telepresence Sessions", draft-lennox-clue-rtp-usage-04 (work in progress), June 2012.

[I-D.westerlund-avtcore-max-ssrc]

Westerlund, M., Burman, B., and F. Jansson, "Multiple Synchronization sources (SSRC) in RTP Session Signaling", draft-westerlund-avtcore-max-ssrc-02 (work in progress), July 2012.

[I-D.westerlund-avtext-rtcp-sdes-srcname]

Westerlund, M., Burman, B., and P. Sandgren, "RTCP SDES Item SRCNAME to Label Individual Sources", draft-westerlund-avtext-rtcp-sdes-srcname-01 (work in progress), July 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

[I-D.ietf-clue-telepresence-use-cases]

Romanow, A., Botzko, S., Duckworth, M., Even, R., and I. Communications, "Use Cases for Telepresence Multi-streams", draft-ietf-clue-telepresence-use-cases-04 (work in progress), August 2012.

[I-D.lennox-mmusic-sdp-source-selection]

Lennox, J. and H. Schulzrinne, "Mechanisms for Media Source Selection in the Session Description Protocol (SDP)", draft-lennox-mmusic-sdp-source-selection-04 (work

in progress), March 2012.

- [I-D.westerlund-avtcore-rtp-simulcast]
Westerlund, M., Burman, B., Lindqvist, M., and F. Jansson,
"Using Simulcast in RTP sessions",
draft-westerlund-avtcore-rtp-simulcast-01 (work in
progress), July 2012.
- [I-D.westerlund-avtcore-rtp-topologies-update]
Westerlund, M. and S. Wenger, "RTP Topologies",
draft-westerlund-avtcore-rtp-topologies-update-01 (work in
progress), October 2012.
- [I-D.westerlund-avtext-codec-operation-point]
Westerlund, M., Burman, B., and L. Hamm, "Codec Operation
Point RTCP Extension",
draft-westerlund-avtext-codec-operation-point-00 (work in
progress), March 2012.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model
with Session Description Protocol (SDP)", RFC 3264,
June 2002.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V.
Jacobson, "RTP: A Transport Protocol for Real-Time
Applications", STD 64, RFC 3550, July 2003.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session
Description Protocol", RFC 4566, July 2006.
- [RFC4575] Rosenberg, J., Schulzrinne, H., and O. Levin, "A Session
Initiation Protocol (SIP) Event Package for Conference
State", RFC 4575, August 2006.
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description
Protocol (SDP) Content Attribute", RFC 4796,
February 2007.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman,
"Codec Control Messages in the RTP Audio-Visual Profile
with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117,
January 2008.
- [RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP
Header Extensions", RFC 5285, July 2008.

- [RFC5576] Lennox, J., Ott, J., and T. Schierl, "Source-Specific Media Attributes in the Session Description Protocol (SDP)", RFC 5576, June 2009.
- [RFC6236] Johansson, I. and K. Jung, "Negotiation of Generic Image Attributes in the Session Description Protocol (SDP)", RFC 6236, May 2011.

Authors' Addresses

Roni Even
Huawei Technologies
Tel Aviv,
Israel

Email: roni.even@mail01.huawei.com

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com

CLUE WG
Internet-Draft
Intended status: Informational
Expires: April 24, 2013

R. Even
Huawei
October 21, 2012

Signalling of CLUE and SDP offer/answer
draft-even-clue-sdp-clue-relation-01.txt

Abstract

This document describes the relation between the different CLUE attributes as specified in the CLUE framework and the SDP attributes. The document will discuss the issues with the CLUE call signalling in order to keep the consistency between the Offer/answer state and the CLUE state.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 24, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Capture attributes	3
4. Encoding parameters	4
5. Acknowledgements	6
6. IANA Considerations	6
7. Security Considerations	6
8. References	6
8.1. Normative References	6
8.2. Informative References	6
Author's Address	7

1. Introduction

The CLUE framework[I-D.ietf-clue-framework] is used to specify the information needed for creating a Telepresence call. The model includes the Media capture information providing information about content of the streams and can provide information about the spatial information between streams based on the capture point and area of capture. A capture scene includes media captures that are part of a same scene e.g. room capture or presentation.

The other information defined in the framework is the Encoding information providing information about the abilities of the providers to send streams allowing a consumer to configure a capture to a specific encoding.

The next sections will look at the capture attributes and the encoding parameter and describe the relation to SDP [RFC4566].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119[RFC2119] and indicate requirement levels for compliant RTP implementations.

3. Capture attributes

The media capture attributes provide static information about the captures. This includes the content of information and can provide spatial information in order to allow the "being there" experience. The media capture attributes include the content describing the role of the media capture, if the content is composed or switched and spatial information providing the three dimensional position of the streams.

The media capture content attribute is based on SDP content attribute [RFC4796] The other attribute do not have similar SDP attributes.

When starting a call the initial offer may include more than one media stream of a media type with a content attribute (e.g. offer main and slide SDP content). In this case the advertisement will also include media captures for main and slides. If the initial offer did not provide content attribute there is no need to provide it later assuming that if the answerer do not support CLUE protocol it is not sure that he will support the content attribute [RFC4796]

As for the other attributes, there are no similar SDP attributes and they provide information which can be used by TelePresence systems. The media capture switched and composed attributes provide information about the creation of the content. Using RTP/RTCP SRRC and CSRC [RFC3550] can provide information about the content of the media captures.

The CLUE framework [I-D.ietf-clue-framework] recommends using one transport connection for each media type multiplexing using one RTP session. This also makes it simple to add and remove media capture by the consumer without a need for an [RFC3264] offer answer.

The exception is if the consumer prefers using a separate RTP non multiplex session. In this case when adding or removing a media capture there will be a need to have also an offer answer session to specify the UDP port for the new RTP session or to close it. (note that ICE exchange may be required too). The open question is what should be done first, CLUE configuration or SDP offer/answer.

As can be seen there is minimal duplication between SDP and these media capture attributes. There is a need to correlate between the RTP streams and media captures; this is discussed in CLUE RTP mapping [I-D.even-clue-rtp-mapping]

When multiplexing RTP streams in a single RTP session there is probably no need for offer answer exchange when the consumer send a new configuration.

4. Encoding parameters

A media capture can specify an encoding group that maps a media capture to encoding parameters. The encoding parameters are used to provide information about the ability of a CLUE endpoint to send streams. An encoding group is composed of individual encodes that may be used by a media capture to encode a stream as long as the total values of the attributes used by the individual encodes in the group do not exceed the group encode values.

The individual encode parameters include: maxBandwidth, maxH264Mbps, maxWidth, maxHeight and maxFrameRate. The encoding group parameters include MaxGroupBandWidth, MaxGroupH264Mbps, and for video MaxBandWidth. Note that the use case is to provide information about the send capabilities of the provider.

The max H264Mbps is H.264 specific and there is a similar parameters in RFC6184 [RFC6184] but in RFC6184signals the receiver capability.

The maxFrameRate is similar to SDP framerate attribute, for maxWidht and MaxHeight there are similar parameters in [RFC6236]. All these parameters carried in SDP signal the receiver capability or requested mode.

The maxBandwidth is similar to the SDP b=attribute and specifying group and individual values can be partially done using the session level and media level attributes. The major different again is that the b= attribute is used to describe receiver capability, maximum bandwidth it can receive.

The Media capture encoding information and SDP attribute are similar but they indicate different types of limitations. While the Media capture attributes are sender encoding capabilities the SDP ones specify receiver capabilities. This is because of the different usage. The SDP attributes are used by the receiver to indicate what it can receive and decode. Still in H.264 the parameter sets are used to convey some of the above parameters (like width and height) and can be conveyed in SDP using sprop-parameter-sets or sprop-level-parameter-sets. In general the "sprop" attributes are used to convey sender capabilities.

The RFC3264 [RFC3264] offer/answer is used by the receiver to limit or ask for a specific mode. Since the encoding parameters specify limits on the sending side it may look like there is no correlation problem. The next sections will look at the specific parameters and see if this assumption is correct.

The CLUE maxBandWidth is limiting the maximum bandwidth that a sender will use. The receiver can ask for higher or lower bandwidth based on H.264 level or using the SDP "b" attribute. If asking for higher than the CLUE value will limit and is asking for lower value the SDP value will be the limiting factor. A change mid-call may happen for example because of congestion. The endpoint must be aware of both values. Note that bandwidth change using RTCP TMMBR [RFC5104] can occur. If the previous bandwidth was higher than the CLUE maxBandwidth and the new band width is lower the EP must use the lowest bandwidth. Since the bandwidth in the offer and answer provide information about receiver capabilities it means that if the value is changed by an intermediary like an SBC the sender will know that the receiver now have different value and act accordingly.

The maxH264Mbps and maxFrameRate show similar behavior to maxBandwidth.

The CLUE maxWidht and maxHeight as sender capability and the image attribute in SDP are both send and receive capability. The recommendation if for using RFC6236 to negotiate these values and not

CLUE encoding. The maxH264Mbps and maxFrameRate can be sufficient to provide compute limitations on the encoder side.

Conclusion - except for the maxWidth and maxHeight there is no problem between the SDP and CLUE encoding values as long as SDP attributes are used as receiver capabilities and the relation between the two protocols is defined. Changing SDP values will not require a change in CLUE advertisement or configuration and vice versa since these parameters signal maximum values and not exact values.

5. Acknowledgements

place holder

6. IANA Considerations

TBD

7. Security Considerations

TBD.

8. References

8.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.
- [RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

8.2. Informative References

- [I-D.even-clue-rtp-mapping]
Even, R. and J. Lennox, "Mapping RTP streams to CLUE media captures", draft-even-clue-rtp-mapping-04 (work in progress), September 2012.
- [I-D.ietf-clue-framework]

Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino,
"Framework for Telepresence Multi-Streams",
draft-ietf-clue-framework-06 (work in progress),
July 2012.

- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, May 2011.
- [RFC6236] Johansson, I. and K. Jung, "Negotiation of Generic Image Attributes in the Session Description Protocol (SDP)", RFC 6236, May 2011.

Author's Address

Roni Even
Huawei
Tel Aviv,
Israel

Email: roni.even@mail01.huawei.com

CLUE WG
Internet-Draft
Intended status: Informational
Expires: March 14, 2013

C. Groves
W. Yang
R. Even
Huawei
September 10, 2012

CLUE media capture description
draft-groves-clue-capture-attr-00.txt

Abstract

This memo discusses how media captures are described and in particular the content attribute in the current CLUE framework document and proposes several alternatives.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 14, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Issues with Content attribute	4
3.1. Ambiguous definition	4
3.2. Multiple functions	5
3.3. Limited Stream Support	5
3.4. Insufficient information for individual parameters	5
3.5. Insufficient information for negotiation	5
4. Capture description attributes	6
4.1. Presentation	7
4.2. View	7
4.3. Language	8
4.4. Role	8
4.5. Priority	9
4.6. Others	9
4.6.1. Dynamic	9
4.6.2. Embedded Text	10
4.6.3. Supplementary Description	10
4.6.4. Telepresence	11
5. Summary	11
6. Acknowledgements	11
7. IANA Considerations	11
8. Security Considerations	12
9. References	12
9.1. Normative References	12
9.2. Informative References	12
Authors' Addresses	12

1. Introduction

One of the fundamental aspects of the CLUE framework is the concept of media captures. The media captures are sent from a provider to a consumer. This consumer then selects which captures it is interested in and replies back to the consumer. The question is how does the consumer choose between what may be many different media captures?

In order to be able to choose between the different media captures the consumer must have enough information regarding what the media capture represents and to distinguish between the media captures.

The CLUE framework draft currently defines several media capture attributes which provide information regarding the capture. The draft indicates that Media Capture Attributes describe static information about the captures. A provider uses the media capture attributes to describe the media captures to the consumer. The consumer will select the captures it wants to receive. Attributes are defined by a variable and its value."

One of the media capture attributes is the content attribute. As indicated in the draft it is a field with enumerated values which describes the role of the media capture and can be applied to any media type. The enumerated values are defined by [RFC4796] The values for this attribute are the same as the mediacontent values for the content attribute in [RFC4796] This attribute can have multiple values, for example content={main, speaker}."

[RFC4796] defines the values as:

- o slides: the media stream includes presentation slides. The media type can be, for example, a video stream or a number of instant messages with pictures. Typical use cases for this are online seminars and courses. This is similar to the 'presentation' role in H.239.
- o speaker: the media stream contains the image of the speaker. The media can be, for example, a video stream or a still image. Typical use cases for this are online seminars and courses.
- o sl: the media stream contains sign language. A typical use case for this is an audio stream that is translated into sign language, which is sent over a video stream.
- o main: the media stream is taken from the main source. A typical use case for this is a concert where the camera is shooting the performer.

- o alt: the media stream is taken from the alternative source. A typical use case for this is an event where the ambient sound is separated from the main sound. The alternative audio stream could be, for example, the sound of a jungle. Another example is the video of a conference room, while the main stream carries the video of the speaker. This is similar to the 'live' role in H.239.

Whilst the above values appear to be a simple way of conveying the content of a stream the Contributors believe that there are multiple issues that make the use of the existing "Content" tag insufficient for CLUE and multi-stream telepresence systems. These issues are described in section 3. Section 4 proposes new capture description attributes.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC2119[RFC2119] and indicate requirement levels for compliant RTP implementations.

3. Issues with Content attribute

3.1. Ambiguous definition

There is ambiguity in the definitions that may cause problems for interoperability. A clear example is "slides" which could be any form of presentation media. Another example is the difference between "main" and "alt". In a telepresence scenario the room would be captured by the "main cameras" and a speaker would be captured by an alternative "camera". This runs counter with the definition of "alt".

Another example is a university use case where:

The main site is a university auditorium which is equipped with three cameras. One camera is focused on the professor at the podium. A second camera is mounted on the wall behind the professor and captures the class in its entirety. The third camera is co-located with the second, and is designed to capture a close up view of a questioner in the audience. It automatically zooms in on that student using sound localization.

For the first camera, it's not clear whether to use "main" or "speaker". According to the definition and example of "speaker" in

RFC 4796, maybe it's more proper to use "speaker" here? For the third camera it could fit the definition of "main" or "alt" or "speaker".

3.2. Multiple functions

It appears that the definitions cover disparate functions. "Main" and "alt" appear to describe the source from which media is sent. "Speaker" indicates a role associated with the media stream. "Slides" and "Sign Language" indicates the actual content. Also indirectly some prioritization is applied to these parameters. For example: the IMTC document on best practices for H.239 indicates a display priority between "main" and "alt". This mixing of functions per code point can lead to ambiguous behavior and interoperability problems. It also is an issue when extending the values.

3.3. Limited Stream Support

The values above appear to be defined based on a small number of video streams that are typically supported by legacy video conferencing. E.g. a main video stream (main), a secondary one (alt) and perhaps a presentation stream (slides). It is not clear how this value scales when many media streams are present. For example if you have several main streams and several presentation streams how would an endpoint distinguish between them?

3.4. Insufficient information for individual parameters

Related to the above point is that some individual values do not provide sufficient information for an endpoint to make an educated decision on the content. For example: Sign language (sl) - If a conference provides multiple streams each one containing a sign interpretation in a different sign language how does an endpoint distinguish between the languages if "sl" is the only label? Also for accessible services other functions such a real time captioning and video description where an additional audio channel is used to describe the conference for vision impaired people should be supported.

Note: SDP provide a language attribute.

3.5. Insufficient information for negotiation

CLUE negotiation is likely to be at the start of a session initiation. At this point of time only a very simple set of SDP (i.e. limited media description) may be available (depending on call flow). In most cases the supported media captures may be agreed upon before the full SDP information for each media stream. The effect of

this is that detailed information would not be available for the initial decision about which capture to choose. The obvious solution is to provide "enough" data in the CLUE provider messages so that a consumer can choose the appropriate media captures. The current CLUE framework already partly addresses this through the "Content" attribute however based on the current "Content" values it appears that the information is not sufficient to fully describe the content of the captures.

The purpose of the CLUE work is to supply enough information for negotiating multiple streams. CLUE framework [I-D.ietf-clue-framework] addresses the spatial relation between the streams but it looks like it does not provide enough information about the semantic content of the stream to allow interoperability.

Some information is available in SDP and may be available before the CLUE exchange but there are still some information missing.

4. Capture description attributes

As indicated above it is proposed to introduce a new attribute/s that allows the definition of various pieces of information that provide metadata about a particular media capture. This information should be described in a way that it only supplies one atomic function. It should also be applicable in a multi-stream environment. It should also be extensible to allow new information elements to be introduced in the future.

As an initial list the following attributes are proposed for use as metadata associated with media captures. Further attributes may be identified in the future.

This document propose to remove the "Content" attribute. Rather than describing the "source device" in this way it may be better to describe its characteristics. i.e.

- o An attribute to indicate "Presentation" rather than the value "Slides"
- o An attribute to describe the "Role" of a capture rather than the value "Speaker".
- o An attribute to indicate the actual language used rather than a value "Sign Language". This is also applicable to multiple audio streams.

- o With respect to "main" and "alt" in a multiple stream environment it's not clear these values are needed if the characteristics of the capture are described. An assumption may be that a capture is "main" unless described otherwise.

Note: CLUE may have missed a media type "text". How about a real time captioning or a real time text conversation associated with a video meeting? It's a text based service. It's not necessarily a presentation stream. It's not audio or visual but a valid component of a conference.

The sections below contain an initial list of attributes.

4.1. Presentation

This attribute indicates that the capture originates from a presentation device, that is one that provides supplementary information to a conference through slides, video, still images, data etc. Where more information is known about the capture it may be expanded hierarchically to indicate the different types of presentation media, e.g. presentation.slides, presentation.image etc.

Note: It is expected that a number of keywords will be defined that provide more detail on the type of presentation.

4.2. View

The Area of capture attribute provides a physical indication of a region that the media capture captures. However the consumer does not know what this physical region relates to. In discussions on the IETF mailing list it is apparent that some people propose to use the "Description" attribute to describe a scene. This is a free text field and as such can be used to signal any piece of information. This leads to problems with interoperability if this field is automatically processed. For interoperability purposes it is proposed to introduce a set of keywords that could be used as a basis for the selection of captures. It is envisaged that this list would be extendable to allow for future uses not covered by the initial specification. Therefore it is proposed to introduce a number of keywords (that may be expanded) indicating what the spatial region relates to? I.e. Room, table, etc. this is an initial description of an attribute introducing these keywords.

This attribute provides a textual description of the area that a media capture captures. This provides supplementary information in addition to the spatial information (i.e. area of capture) regarding the region that is captured.

Room - Captures the entire scene.

Table - Captures the conference table with seated participants

Individual - Captures an individual participant

Lectern - Captures the region of the lectern including the presenter in classroom style conference

Audience - Captures a region showing the audience in a classroom style conference.

Others - TBD

4.3. Language

As indicated in the discussion in section 2 captures may be offered in different languages in case of multi-lingual and/or accessible conferences. It is important to allow the remote end to distinguish between them. It is noted that SDP already contains a language attribute however this may not be available at the time that an initial CLUE message is sent. Therefore a language attribute is proposed for CLUE.

This indicates which language is associated with the capture. For example: it may provide a language associated with an audio capture or a language associated with a video capture when sign interpretation or text is used. The possible values for a language tag are the values of the 'Subtag' column for the "Type: language" entries in the "Language Subtag Registry" defined in [RFC5646]

4.4. Role

The original definition of "Content" allows the indication that a particular media stream is related to the speaker. CLUE should also allow this identification for captures. In addition with the advent of XCON there may be other formal roles that may be associated with media/captures. For instance: a remote end may like to always view the floor controller. It is envisaged that a remote end may also chose captures depending on the role of the person/s captured. For example: the people at the remote end may wish to always view the chairmen. This indicates that the capture is associated with an entity that has a particular role in the conference. The values are:

Speaker - indicates that the capture relates to the current speaker

Floor - indicates that the capture relates to the current floor controller of the conference

Chairman- indicates who the chairman of the meeting is.

Others - ?

4.5. Priority

As has been highlighted in discussions on the CLUE mailing list there appears to be some desire to provide some relative priority between captures when multiple alternatives are supplied. This priority can be used to determine which captures contain the most important information (according to the provider). This may be important in case where the consumer has limited resources and can only render a subset of captures. Priority may also be advantageous in congestion scenarios where media from one capture may be favoured over other captures in any control algorithms. This could be supplied via "ordering" in a CLUE data structure however this may be problematic if people assume some spatial meaning behind ordering, i.e. given three captures VC1, VC2, VC3: it would be natural to send VC1,VC2,VC3 if the images are composed this way. However if your boss sits in the middle view the priority may be VC2,VC1,VC3. Explicit signalling is better.

Additionally currently there are no hints to relative priority among captures from different capture scenes. In order to prevent any misunderstanding with implicit ordering a numeric number that may be assigned to each capture.

The "priority" attribute indicates a relative priority between captures. For example it is possible to assign a priority between two presentation captures that would allow a remote endpoint to determine which presentation is more important. Priority is assigned at the individual capture level. It represents the provider's view of the relative priority between captures with a priority. The same priority number may be used across multiple captures. It indicates they are equally as important. If no priority is assigned no assumptions regarding relative importance of the capture can be assumed.

4.6. Others

4.6.1. Dynamic

In the framework it has been assumed that the capture point is a fixed point within a telepresence session. However depending on the conference scenario this may not be the case. In tele-medical or tele-education cases a conference may include cameras that move during the conference. For example: a camera may be placed at different positions in order to provide the best angle to capture a

work task, or may include a camera worn by a participant. This would have an effect of changing the capture point, capture axis and area of capture. In order that the remote endpoint can chose to layout/render the capture appropriately an indication of if the camera is dynamic should be indicated in the initial capture description.

This indicates that the spatial information related to the capture may be dynamic and change through the conference. Thus captures may be characterised as static, dynamic or highly dynamic. The capture point of a static capture does not move for the life of the conference. The capture point of dynamic captures is categorised by a change in position followed by a reasonable period of stability. High dynamic captures are categorised by a capture point that is constantly moving. This may assist an endpoint in determining the correct display layout. If the "area of capture", "capture point" and "line of capture" attributes are included with dynamic or highly dynamic captures they indicate spatial information at the time a CLUE message is sent. No information regarding future spatial information should be assumed.

4.6.2. Embedded Text

In accessible conferences textual information may be added to a capture before it is transmitted to the remote end. In the case where multiple video captures are presented the remote end may benefit from the ability to choose a video stream containing text over one that does not.

This attribute indicates that a capture provides embedded textual information. For example the video capture may contain speech to text information composed with the video image. This attribute is only applicable to video captures and presentation streams with visual information.

4.6.3. Supplementary Description

Some conferences utilise translators or facilitators that provide an additional audio stream (i.e. a translation or description of the conference). These persons may not be pictured in a video capture. Where multiple audio captures are presented it may be advantageous for an endpoint to select a supplementary stream instead of or additional to an audio feed associated with the participants from a main video capture. Therefore an attribute is proposed for this. Depending on the results of the discussion of the source device this parameter may be another value for the source.

This indicates that a capture provides additional description of the conference. For example an additional audio stream that provides a

commentary of a conference that provides supplementary information (e.g. a translation) or extra information to participants in accessible conferences.

4.6.4. Telepresence

In certain use cases scenarios it is important to maintain a feeling of "Telepresence" associated with captures when they are played at the remote end. For example: in medical use cases it is important to maintain the colour of images. It is important to note that CLUE is used to describe multi-stream conferences. These may or may not be "telepresence" conferences. Alternatively it could be assumed that all captures possess this attribute and the only captures not subject to processing to create "telepresence" this are those marked with "presentation". We did discuss the aspect of how an endpoint determines if a capture relates to a computer generated image or a real environment. An endpoint may apply different images processing depending on a source, i.e. it may or not apply image processing to adjust lighting levels for a telepresence experience.

This parameter indicates that "telepresence" should be associated with the capture. E.g. real world environmental conditions are associated with this capture. Lighting, spatial and timing information are important aspects of the telepresence session. The remote should apply the appropriate capture processing to maintain integrity of this information. For example: the colour related information associated with the original capture is important and should be replicated when displayed/played.

5. Summary

The main proposal is a to remove the Content Attribute in favour of describing the characteristics of captures in a more functional(atomic) way using the above attributes as the attributes to describe metadata regarding a capture.

6. Acknowledgements

place holder

7. IANA Considerations

TBD

8. Security Considerations

TBD.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[I-D.ietf-clue-framework]
Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino,
"Framework for Telepresence Multi-Streams",
draft-ietf-clue-framework-06 (work in progress),
July 2012.

[RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

Authors' Addresses

Christian Groves
Huawei
Australia

Email: Christian.Groves@nteczone.com

Weiwei Yang
Huawei
P.R. China

Email: tommy@huawei.com

Roni Even
Huawei
Tel Aviv,
Israel

Email: roni.even@mail01.huawei.com

CLUE
Internet-Draft
Intended status: Informational
Expires: August 22, 2013

C. Groves, Ed.
W. Yang
R. Even
Huawei
February 18, 2013

CLUE media capture description
draft-groves-clue-capture-attr-01

Abstract

This memo discusses how media captures are described and in particular the content attribute in the current CLUE framework document and proposes several alternatives.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 22, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Introduction

One of the fundamental aspects of the CLUE framework is the concept of media captures. The media captures are sent from a provider to a consumer. This consumer then selects which captures it is interested in and replies back to the consumer. The question is how does the consumer choose between what may be many different media captures?

In order to be able to choose between the different media captures the consumer must have enough information regarding what the media capture represents and to distinguish between the media captures.

The CLUE framework draft currently defines several media capture attributes which provide information regarding the capture. The draft indicates that Media Capture Attributes describe static information about the captures. A provider uses the media capture attributes to describe the media captures to the consumer. The consumer will select the captures it wants to receive. Attributes are defined by a variable and its value.

One of the media capture attributes is the content attribute. As indicated in the draft it is a field with enumerated values which describes the role of the media capture and can be applied to any media type. The enumerated values are defined by RFC 4796 [RFC4796]. The values for this attribute are the same as the media content values for the content attribute in RFC 4796 [RFC4796]. This attribute can have multiple values, for example content={main, speaker}.

RFC 4796 [RFC4796] defines the values as:

slides: the media stream includes presentation slides. The media type can be, for example, a video stream or a number of instant messages with pictures. Typical use cases for this are online seminars and courses. This is similar to the 'presentation' role in H.239.

speaker: the media stream contains the image of the speaker. The media can be, for example, a video stream or a still image. Typical use cases for this are online seminars and courses.

sl: the media stream contains sign language. A typical use case for this is an audio stream that is translated into sign language, which is sent over a video stream.

Whilst the above values appear to be a simple way of conveying the content of a stream the Contributors believe that there are multiple issues that make the use of the existing "Content" tag insufficient for CLUE and multi-stream telepresence systems. These issues are

described in section 3. Section 4 proposes new capture description attributes.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This document draws liberally from the terminology defined in the CLUE Framework [I-D.ietf-clue-framework]

3. Issues with Content attribute

3.1. Ambiguous definition

*There is ambiguity in the definitions that may cause problems for interoperability. A clear example is "slides" which could be any form of presentation media. Another example is the difference between "main" and "alt". In a telepresence scenario the room would be captured by the "main cameras" and a speaker would be captured by an alternative "camera". This runs counter with the definition of "alt".

Another example is a university use case where:

The main site is a university auditorium which is equipped with three cameras. One camera is focused on the professor at the podium. A second camera is mounted on the wall behind the professor and captures the class in its entirety. The third camera is co-located with the second, and is designed to capture a close up view of a questioner in the audience. It automatically zooms in on that student using sound localization.

For the first camera, it's not clear whether to use "main" or "speaker". According to the definition and example of "speaker" in RFC 4796 [RFC4796], maybe it's more proper to use "speaker" here? For the third camera it could fit the definition of "main" or "alt" or "speaker".

3.2. Multiple functions

It appears that the definitions cover disparate functions. "Main" and "alt" appear to describe the source from which media is sent. "Speaker" indicates a role associated with the media stream.

"Slides" and "Sign Language" indicates the actual content. Also indirectly some prioritization is applied to these parameters. For example: the IMTC document on best practices for H.239 indicates a display priority between "main" and "alt". This mixing of functions per code point can lead to ambiguous behavior and interoperability problems. It also is an issue when extending the values.

3.3. Limited Stream Support

The values above appear to be defined based on a small number of video streams that are typically supported by legacy video conferencing. E.g. a main video stream (main), a secondary one (alt) and perhaps a presentation stream (slides). It is not clear how this value scales when many media streams are present. For example if you have several main streams and several presentation streams how would an endpoint distinguish between them?

3.4. Insufficient information for individual parameters

Related to the above point is that some individual values do not provide sufficient information for an endpoint to make an educated decision on the content. For example: Sign language (sl) - If a conference provides multiple streams each one containing a sign interpretation in a different sign language how does an endpoint distinguish between the languages if "sl" is the only label? Also for accessible services other functions such as a real time captioning and video description where an additional audio channel is used to describe the conference for vision impaired people should be supported.

Note: SDP provide a language attribute.

3.5. Insufficient information for negotiation

CLUE negotiation is likely to be at the start of a session initiation. At this point of time only a very simple set of SDP (i.e. limited media description) may be available (depending on call flow). In most cases the supported media captures may be agreed upon before the full SDP information for each media stream. The effect of this is that detailed information would not be available for the initial decision about which capture to choose. The obvious solution is to provide "enough" data in the CLUE provider messages so that a consumer can choose the appropriate media captures. The current CLUE framework already partly addresses this through the "Content" attribute however based on the current "Content" values it appears that the information is not sufficient to fully describe the content of the captures.

The purpose of the CLUE work is to supply enough information for negotiating multiple streams. CLUE framework [I-D.ietf-clue-framework] addresses the spatial relation between the streams but it looks like it does not provide enough information about the semantic content of the stream to allow interoperability.

Some information is available in SDP and may be available before the CLUE exchange but there are still some information missing.

4. Capture description attributes

As indicated above it is proposed to introduce a new attribute/s that allows the definition of various pieces of information that provide metadata about a particular media capture. This information should be described in a way that it only supplies one atomic function. It should also be applicable in a multi-stream environment. It should also be extensible to allow new information elements to be introduced in the future.

As an initial list the following attributes are proposed for use as metadata associated with media captures. Further attributes may be identified in the future.

This document propose to remove the "Content" attribute. Rather than describing the "source device" in this way it may be better to describe its characteristics. i.e.

An attribute to indicate "Presentation" rather than the value "Slides".

An attribute to describe the "Role" of a capture rather than the value "Speaker".

An attribute to indicate the actual language used rather than a value "Sign Language". This is also applicable to multiple audio streams.

With respect to "main" and "alt" in a multiple stream environment it's not clear these values are needed if the characteristics of the capture are described. An assumption may be that a capture is "main" unless described otherwise.

Note: CLUE may have missed a media type "text". How about a real time captioning or a real time text conversation associated with a video meeting? It's a text based service. It's not necessarily a presentation stream. It's not audio or visual but a valid component of a conference.

The sections below contain an initial list of attributes.

4.1. Presentation

This attribute indicates that the capture originates from a presentation device, that is one that provides supplementary information to a conference through slides, video, still images, data etc. Where more information is known about the capture it may be expanded hierarchically to indicate the different types of presentation media, e.g. presentation.slides, presentation.image etc.

Note: It is expected that a number of keywords will be defined that provide more detail on the type of presentation.

4.2. View

The Area of capture attribute provides a physical indication of a region that the media capture captures. However the consumer does not know what this physical region relates to. In discussions on the IETF mailing list it is apparent that some people propose to use the "Description" attribute to describe a scene. This is a free text field and as such can be used to signal any piece of information. This leads to problems with interoperability if this field is automatically processed. For interoperability purposes it is proposed to introduce a set of keywords that could be used as a basis for the selection of captures. It is envisaged that this list would be extendable to allow for future uses not covered by the initial specification. Therefore it is proposed to introduce a number of keywords (that may be expanded) indicating what the spatial region relates to? I.e. Room, table, etc. this is an initial description of an attribute introducing these keywords.

This attribute provides a textual description of the area that a media capture captures. This provides supplementary information in addition to the spatial information (i.e. area of capture) regarding the region that is captured.

Room - Captures the entire scene.

Table - Captures the conference table with seated participants

Individual - Captures an individual participant

Lectern - Captures the region of the lectern including the presenter in classroom style conference

Audience - Captures a region showing the audience in a classroom style conference.

Others - TBD

4.3. Language

Captures may be offered in different languages in case of multi-lingual and/or accessible conferences. It is important to allow the remote end to distinguish between them. It is noted that SDP already contains a language attribute however this may not be available at the time that an initial CLUE message is sent. Therefore a language attribute is needed in CLUE to indicate the language used by the capture.

This indicates which language is associated with the capture. For example: it may provide a language associated with an audio capture or a language associated with a video capture when sign interpretation or text is used.

An example where multiple languages may be used is where a capture includes multiple conference participants who use different languages.

The possible values for the language tag are the values of the 'Subtag' column for the "Type: language" entries in the "Language Subtag Registry" defined in RFC 5646 [RFC5646].

4.4. Role

The original definition of "Content" allows the indication that a particular media stream is related to the speaker. CLUE should also allow this identification for captures. In addition with the advent of XCON there may be other formal roles that may be associated with media/captures. For instance: a remote end may like to always view the floor controller. It is envisaged that a remote end may also chose captures depending on the role of the person/s captured. For example: the people at the remote end may wish to always view the chairman. This indicates that the capture is associated with an entity that has a particular role in the conference. It is possible for the attribute to have multiple values where the capture has multiple roles.

The values are grouped into two types: Person roles and Conference Roles

4.4.1. Person Roles

The roles are related to the titles of the person/s associated with the capture.

Manager - Indicates that the capture is assigned to a person with a senior position.

Chairman- indicates who the chairman of the meeting is.

Secretary - indicates that the capture is associated with the conference secretary.

Lecturer - indicates that the capture is associated with the conference lecturer.

Audience - indicates that the capture is associated with the conference audience.

Others

4.4.2. Conference Roles

These roles are related to the establishment and maintenance of the multimedia conference and is related to the conference system.

Speaker - indicates that the capture relates to the current speaker.

Controller - indicates that the capture relates to the current floor controller of the conference.

Others

An example is:

```
AC1 [Role=Speaker]
VC1 [Role=Lecturer,Speaker]
```

4.5. Priority

As has been highlighted in discussions on the CLUE mailing list there appears to be some desire to provide some relative priority between captures when multiple alternatives are supplied. This priority can be used to determine which captures contain the most important information (according to the provider). This may be important in case where the consumer has limited resources and can only render a subset of captures. Priority may also be advantageous in congestion scenarios where media from one capture may be favoured over other captures in any control algorithms. This could be supplied via "ordering" in a CLUE data structure however this may be problematic if people assume some spatial meaning behind ordering, i.e. given three captures VC1, VC2, VC3: it would be natural to send VC1,VC2,VC3

if the images are composed this way. However if your boss sits in the middle view the priority may be VC2,VC1,VC3. Explicit signalling is better.

Additionally currently there are no hints to relative priority among captures from different capture scenes. In order to prevent any misunderstanding with implicit ordering a numeric number that may be assigned to each capture.

The "priority" attribute indicates a relative priority between captures. For example it is possible to assign a priority between two presentation captures that would allow a remote endpoint to determine which presentation is more important. Priority is assigned at the individual capture level. It represents the provider's view of the relative priority between captures with a priority. The same priority number may be used across multiple captures. It indicates they are equally as important. If no priority is assigned no assumptions regarding relative important of the capture can be assumed.

4.6. Others

4.6.1. Dynamic

In the framework it has been assumed that the capture point is a fixed point within a telepresence session. However depending on the conference scenario this may not be the case. In tele-medical or tele-education cases a conference may include cameras that move during the conference. For example: a camera may be placed at different positions in order to provide the best angle to capture a work task, or may include a camera worn by a participant. This would have an effect of changing the capture point, capture axis and area of capture. In order that the remote endpoint can chose to layout/render the capture appropriately an indication of if the camera is dynamic should be indicated in the initial capture description.

This indicates that the spatial information related to the capture may be dynamic and change through the conference. Thus captures may be characterised as static, dynamic or highly dynamic. The capture point of a static capture does not move for the life of the conference. The capture point of dynamic captures is categorised by a change in position followed by a reasonable period of stability. High dynamic captures are categorised by a capture point that is constantly moving. This may assist an endpoint in determining the correct display layout. If the "area of capture", "capture point" and "line of capture" attributes are included with dynamic or highly dynamic captures they indicate spatial information at the time a CLUE message is sent. No information regarding future spatial information

should be assumed.

4.6.2. Embedded Text

In accessible conferences textual information may be added to a capture before it is transmitted to the remote end. In the case where multiple video captures are presented the remote end may benefit from the ability to choose a video stream containing text over one that does not.

This attribute indicates that a capture provides embedded textual information. For example the video capture may contain speech to text information composed with the video image. This attribute is only applicable to video captures and presentation streams with visual information.

The EmbeddedText attribute contains a language value according to RFC 5646 [RFC5646] and may use a script sub-tag. For example:

```
EmbeddedText=zh-Hans
```

Which indicates embedded text in Chinese written using the simplified Chinese script.

4.6.3. Complementary Feed

Some conferences utilise translators or facilitators that provide an additional audio stream (i.e. a translation or description of the conference). These persons may not be pictured in a video capture. Where multiple audio captures are presented it may be advantageous for an endpoint to select a complementary stream instead of or additional to an audio feed associated with the participants from a main video capture.

This indicates that a capture provides additional description of the conference. For example an additional audio stream that provides a commentary of a conference that provides complementary information (e.g. a translation) or extra information to participants in accessible conferences.

An example is where an additional capture provides a translation of another capture:

```
AC1 [Language = English]  
AC2 [ComplementaryFeed = AC1, Language=Chinese]
```

The complementary feed attribute indicates the capture to which it is providing additional information.

5. Summary

The main proposal is a to remove the Content Attribute in favour of describing the characteristics of captures in a more functional(atomic) way using the above attributes as the attributes to describe metadata regarding a capture.

6. Acknowledgements

This template was derived from an initial version written by Pekka Savola and contributed by him to the xml2rfc project.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

TBD

9. Changes and Status Since Last Version

Changes from 00 to 01:

1. Changed source to XML.
2. 4.1 Presentation : No comments or concerns. No changes.
3. 4.2 View : No comments or concerns. No changes.
4. 4.3 Language: There were comments that multiple languages need to be supported e.g. audio in one, embedded text in another. The text need to be clear whether it is supported or preferred language however it was clarified it is neither. Its the language of the content/capture. It was also noted that different speakers using different languages could talk on the main speakers capture therefore language should be a list. Seemed to be support for this. Text was adapted accordingly.
5. 4.4 Role: There were a couple of responses for support for this attribute. The actual values still need some work. It was noted that there were two possible sets of roles: One group related to the titles of the person: i.e. Boss,

Chairman, Secretary, Lecturer, Audience. Another group related to conference functions: i.e. Conference initiator, controller, speaker. Text was adapted accordingly.

6. 4.5 Priority: No direct comment on the proposal. There appeared to be some interest in a prioritisation scheme during discussions on the framework. No changes.
7. 4.6.1 Dynamic : No comments or concerns. No changes.
8. 4.6.2 Embedded text: There was a comment that "text" media capture was needed. It was also indicated that it should be possible to associate a language with embedded text. It should be possible to also specify language and script. e.g. Embedded text could have its own language. Text adapted accordingly.
9. 4.6.3 Supplementary Description: There were comments that it could be interpreted as a free text field. The intention is that its more of a flag. A better name could be "Complementary feed"? There was also a comment that perhaps a specific "translator flag" is needed. It was noted the usage was like: AC1 Language=English or AC2 Supplementary Description = TRUE, Language=Chinese. Text updated accordingly.
10. 4.6.4 Telepresence There were a couple of comments questioning the need for this parameter. Attribute removed.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

10.2. Informative References

- [I-D.ietf-clue-framework]
Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams",
draft-ietf-clue-framework-08 (work in progress).
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796,
February 2007.

[RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.

Authors' Addresses

Christian Groves (editor)
Huawei
Melbourne,
Australia

Email: Christian.Groves@nteczone.com

Weiwei Yang
Huawei
P.R.China

Email: tommy@huawei.com

Roni Even
Huawei
Tel Aviv,
Israel

Email: roni.even@mail01.huawei.com

CLUE WG
Internet-Draft
Intended status: Informational
Expires: April 25, 2013

A. Romanow
Cisco Systems
M. Duckworth, Ed.
Polycom
A. Pepperell
Silverflare
B. Baldino
Cisco Systems
October 22, 2012

Framework for Telepresence Multi-Streams
draft-ietf-clue-framework-07.txt

Abstract

This memo offers a framework for a protocol that enables devices in a telepresence conference to interoperate by specifying the relationships between multiple media streams.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Definitions	3
4. Overview of the Framework/Model	6
5. Spatial Relationships	7
6. Media Captures and Capture Scenes	8
6.1. Media Captures	9
6.1.1. Media Capture Attributes	9
6.2. Capture Scene	12
6.2.1. Capture scene attributes	13
6.2.2. Capture scene entry attributes	14
6.3. Simultaneous Transmission Set Constraints	15
7. Encodings	16
7.1. Individual Encodings	16
7.2. Encoding Group	17
8. Associating Media Captures with Encoding Groups	19
9. Consumer's Choice of Streams to Receive from the Provider	19
9.1. Local preference	20
9.2. Physical simultaneity restrictions	20
9.3. Encoding and encoding group limits	21
9.4. Message Flow	21
10. Extensibility	22
11. Examples - Using the Framework	22
11.1. Three screen endpoint media provider	23
11.2. Encoding Group Example	29
11.3. The MCU Case	30
11.4. Media Consumer Behavior	30
11.4.1. One screen consumer	31
11.4.2. Two screen consumer configuring the example	31
11.4.3. Three screen consumer configuring the example	32
12. Acknowledgements	32
13. IANA Considerations	32
14. Security Considerations	32
15. Changes Since Last Version	32
16. Informative References	34
Authors' Addresses	35

1. Introduction

Current telepresence systems, though based on open standards such as RTP [RFC3550] and SIP [RFC3261], cannot easily interoperate with each other. A major factor limiting the interoperability of telepresence systems is the lack of a standardized way to describe and negotiate the use of the multiple streams of audio and video comprising the media flows. This draft provides a framework for a protocol to enable interoperability by handling multiple streams in a standardized way. It is intended to support the use cases described in draft-ietf-clue-telepresence-use-cases-02 and to meet the requirements in draft-ietf-clue-telepresence-requirements-01.

The solution described here is strongly focused on what is being done today, rather than on a vision of future conferencing. At the same time, the highest priority has been given to creating an extensible framework to make it easy to accommodate future conferencing functionality as it evolves.

The purpose of this effort is to make it possible to handle multiple streams of media in such a way that a satisfactory user experience is possible even when participants are using different vendor equipment, and also when they are using devices with different types of communication capabilities. Information about the relationship of media streams at the provider's end must be communicated so that streams can be chosen and audio/video rendering can be done in the best possible manner.

There is no attempt here to dictate to the renderer what it should do. What the renderer does is up to the renderer.

After the following Definitions, a short section introduces key concepts. The body of the text comprises several sections about the key elements of the framework, how a consumer chooses streams to receive, and some examples. The appendix describe topics that are under discussion for adding to the document.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Definitions

The definitions marked with an "*" are new; all the others are from

draft-wenger-clue-definitions-00-01.txt.

*Audio Capture: Media Capture for audio. Denoted as ACn.

Camera-Left and Right: For media captures, camera-left and camera-right are from the point of view of a person observing the rendered media. They are the opposite of stage-left and stage-right.

Capture Device: A device that converts audio and video input into an electrical signal, in most cases to be fed into a media encoder. Cameras and microphones are examples for capture devices.

*Capture Encoding: A specific encoding of a media capture, to be sent by a media provider to a media consumer via RTP.

*Capture Scene: a structure representing the scene that is captured by a collection of capture devices. A capture scene includes attributes and one or more capture scene entries, with each entry including one or more media captures.

*Capture Scene Entry: a list of media captures of the same media type that together form one way to represent the capture scene.

Conference: used as defined in [RFC4353], A Framework for Conferencing within the Session Initiation Protocol (SIP).

*Individual Encoding: A variable with a set of attributes that describes the maximum values of a single audio or video capture encoding. The attributes include: maximum bandwidth- and for video maximum macroblocks (for H.264), maximum width, maximum height, maximum frame rate.

*Encoding Group: A set of encoding parameters representing a media provider's encoding capabilities. Media stream providers formed of multiple physical units, in each of which resides some encoding capability, would typically advertise themselves to the remote media stream consumer using multiple encoding groups. Within each encoding group, multiple potential encodings are possible, with the sum of the chosen encodings' characteristics constrained to being less than or equal to the group-wide constraints.

Endpoint: The logical point of final termination through receiving, decoding and rendering, and/or initiation through capturing, encoding, and sending of media streams. An endpoint consists of one or more physical devices which source and sink media streams, and exactly one [RFC4353] Participant (which, in turn, includes exactly one SIP User Agent). In contrast to an endpoint, an MCU may also send and receive media streams, but it is not the initiator nor the

final terminator in the sense that Media is Captured or Rendered. Endpoints can be anything from multiscreen/multicamera rooms to handheld devices.

Front: the portion of the room closest to the cameras. In going towards back you move away from the cameras.

MCU: Multipoint Control Unit (MCU) - a device that connects two or more endpoints together into one single multimedia conference [RFC5117]. An MCU includes an [RFC4353] Mixer. [Edt. RFC4353 is tardy in requiring that media from the mixer be sent to EACH participant. I think we have practical use cases where this is not the case. But the bug (if it is one) is in 4353 and not herein.]

Media: Any data that, after suitable encoding, can be conveyed over RTP, including audio, video or timed text.

*Media Capture: a source of Media, such as from one or more Capture Devices. A Media Capture (MC) may be the source of one or more capture encodings. A Media Capture may also be constructed from other Media streams. A middle box can express Media Captures that it constructs from Media streams it receives.

*Media Consumer: an Endpoint or middle box that receives media streams

*Media Provider: an Endpoint or middle box that sends Media streams

Model: a set of assumptions a telepresence system of a given vendor adheres to and expects the remote telepresence system(s) also to adhere to.

*Plane of Interest: The spatial plane containing the most relevant subject matter.

Render: the process of generating a representation from a media, such as displayed motion video or sound emitted from loudspeakers.

*Simultaneous Transmission Set: a set of media captures that can be transmitted simultaneously from a Media Provider.

Spatial Relation: The arrangement in space of two objects, in contrast to relation in time or other relationships. See also Camera-Left and Right.

Stage-Left and Right: For media captures, stage-left and stage-right are the opposite of camera-left and camera-right. For the case of a person facing (and captured by) a camera, stage-left and stage-right

are from the point of view of that person.

*Stream: a capture encoding sent from a media provider to a media consumer via RTP [RFC3550].

Stream Characteristics: the media stream attributes commonly used in non-CLUE SIP/SDP environments (such as: media codec, bit rate, resolution, profile/level etc.) as well as CLUE specific attributes, such as the ID of a capture or a spatial location.

Telepresence: an environment that gives non co-located users or user groups a feeling of (co-located) presence - the feeling that a Local user is in the same room with other Local users and the Remote parties. The inclusion of Remote parties is achieved through multimedia communication including at least audio and video signals of high fidelity.

*Video Capture: Media Capture for video. Denoted as VCn.

Video composite: A single image that is formed from combining visual elements from separate sources.

4. Overview of the Framework/Model

The CLUE framework specifies how multiple media streams are to be handled in a telepresence conference.

The main goals include:

- o Interoperability
- o Extensibility
- o Flexibility

Interoperability is achieved by the media provider describing the relationships between media streams in constructs that are understood by the consumer, who can then render the media. Extensibility is achieved through abstractions and the generality of the model, making it easy to add new parameters. Flexibility is achieved largely by having the consumer choose what content and format it wants to receive from what the provider is capable of sending.

A transmitting endpoint or MCU describes specific aspects of the content of the media and the formatting of the media streams it can send (advertisement); and the receiving end responds to the provider by specifying which content and media streams it wants to receive

(configuration). The provider then transmits the asked for content in the specified streams.

This advertisement and configuration occurs at call initiation but may also happen at any time throughout the conference, whenever there is a change in what the consumer wants or the provider can send.

An endpoint or MCU typically acts as both provider and consumer at the same time, sending advertisements and sending configurations in response to receiving advertisements. (It is possible to be just one or the other.)

The data model is based around two main concepts: a capture and an encoding. A media capture (MC), such as audio or video, describes the content a provider can send. Media captures are described in terms of CLUE-defined attributes, such as spatial relationships and purpose of the capture. Providers tell consumers which media captures they can provide, described in terms of the media capture attributes.

A provider organizes its media captures that represent the same scene into capture scenes. A consumer chooses which media captures it wants to receive according to the capture scenes sent by the provider.

In addition, the provider sends the consumer a description of the individual encodings it can send in terms of the media attributes of the encodings, in particular, well-known audio and video parameters such as bandwidth, frame rate, macroblocks per second.

The provider also specifies constraints on its ability to provide media, and the consumer must take these into account in choosing the content and capture encodings it wants. Some constraints are due to the physical limitations of devices - for example, a camera may not be able to provide zoom and non-zoom views simultaneously. Other constraints are system based constraints, such as maximum bandwidth and maximum macroblocks/second.

The following sections discuss these constructs and processes in detail, followed by use cases showing how the framework specification can be used.

5. Spatial Relationships

In order for a consumer to perform a proper rendering, it is often necessary to provide spatial information about the streams it is receiving. CLUE defines a coordinate system that allows media

providers to describe the spatial relationships of their media captures to enable proper scaling and spatial rendering of their streams. The coordinate system is based on a few principles:

- o Simple systems which do not have multiple Media Captures to associate spatially need not use the coordinate model.
- o Coordinates can either be in real, physical units (millimeters), have an unknown scale or have no physical scale. Systems which know their physical dimensions should always provide those real-world measurements. Systems which don't know specific physical dimensions but still know relative distances should use 'unknown scale'. 'No scale' is intended to be used where Media Captures from different devices (with potentially different scales) will be forwarded alongside one another (e.g. in the case of a middle box).
 - * "millimeters" means the scale is in millimeters
 - * "Unknown" means the scale is not necessarily millimeters, but the scale is the same for every capture in the capture scene.
 - * "No Scale" means the scale could be different for each capture - an MCU provider that advertises two adjacent captures and picks sources (which can change quickly) from different endpoints might use this value; the scale could be different and changing for each capture. But the areas of capture still represent a spatial relation between captures.
- o The coordinate system is Cartesian X, Y, Z with the origin at a spot of the provider's choosing. The provider must use the same coordinate system with same scale and origin for all coordinates within the same capture scene.

The direction of increasing coordinate values is:

X increases from camera left to camera right

Y increases from front to back

Z increases from low to high

6. Media Captures and Capture Scenes

This section describes how media providers can describe the content of media to consumers.

6.1. Media Captures

Media captures are the fundamental representations of streams that a device can transmit. What a Media Capture actually represents is flexible:

- o It can represent the immediate output of a physical source (e.g. camera, microphone) or 'synthetic' source (e.g. laptop computer, DVD player).
- o It can represent the output of an audio mixer or video composer
- o It can represent a concept such as 'the loudest speaker'
- o It can represent a conceptual position such as 'the leftmost stream'

To distinguish between multiple instances, video and audio captures are numbered such as: VC1, VC2 and AC1, AC2. VC1 and VC2 refer to two different video captures and AC1 and AC2 refer to two different audio captures.

Each Media Capture can be associated with attributes to describe what it represents.

6.1.1. Media Capture Attributes

Media Capture Attributes describe static information about the captures. A provider uses the media capture attributes to describe the media captures to the consumer. The consumer will select the captures it wants to receive. Attributes are defined by a variable and its value. The currently defined attributes and their values are:

Content: {slides, speaker, sl, main, alt}

A field with enumerated values which describes the role of the media capture and can be applied to any media type. The enumerated values are defined by [RFC4796]. The values for this attribute are the same as the mediacontent values for the content attribute in [RFC4796]. This attribute can have multiple values, for example content={main, speaker}.

Composed: {true, false}

A field with a Boolean value which indicates whether or not the Media Capture is a mix (audio) or composition (video) of streams.

This attribute is useful for a media consumer to avoid nesting a composed video capture into another composed capture or rendering. This attribute is not intended to describe the layout a media provider uses when composing video streams.

Audio Channel Format: {mono, stereo} A field with enumerated values which describes the method of encoding used for audio.

A value of 'mono' means the Audio Capture has one channel.

A value of 'stereo' means the Audio Capture has two audio channels, left and right.

This attribute applies only to Audio Captures. A single stereo capture is different from two mono captures that have a left-right spatial relationship. A stereo capture maps to a single RTP stream, while each mono audio capture maps to a separate RTP stream.

Switched: {true, false}

A field with a Boolean value which indicates whether or not the Media Capture represents the (dynamic) most appropriate subset of a 'whole'. What is 'most appropriate' is up to the provider and could be the active speaker, a lecturer or a VIP.

Point of Capture: {(X, Y, Z)}

A field with a single Cartesian (X, Y, Z) point value which describes the spatial location, virtual or physical, of the capturing device (such as camera).

When the Point of Capture attribute is specified, it must include X, Y and Z coordinates. If the point of capture is not specified, it means the consumer should not assume anything about the spatial location of the capturing device. Even if the provider specifies an area of capture attribute, it does not need to specify the point of capture.

Point on Line of Capture: {(X,Y,Z)}

A field with a single Cartesian (X, Y, Z) point value (virtual or physical) which describes a position in space of a second point on the axis of the capturing device; the first point being the Point of Capture (see above). This point MUST lie between the Point of Capture and the Area of Capture.

The Point on Line of Capture MUST be ignored if the Point of Capture is not present for this capture device. When the Point on Line of

Capture attribute is specified, it must include X, Y and Z coordinates. These coordinates MUST NOT be identical to the Point of Capture coordinates. If the Point on Line of Capture is not specified, no assumptions are made about the axis of the capturing device.

Area of Capture:

```
{bottom left(X1, Y1, Z1), bottom right(X2, Y2, Z2), top left(X3, Y3, Z3), top right(X4, Y4, Z4)}
```

A field with a set of four (X, Y, Z) points as a value which describe the spatial location of what is being "captured". By comparing the Area of Capture for different Media Captures within the same capture scene a consumer can determine the spatial relationships between them and render them correctly.

The four points should be co-planar. The four points form a quadrilateral, not necessarily a rectangle.

The quadrilateral described by the four (X, Y, Z) points defines the plane of interest for the particular media capture.

If the area of capture attribute is specified, it must include X, Y and Z coordinates for all four points. If the area of capture is not specified, it means the media capture is not spatially related to any other media capture (but this can change in a subsequent provider advertisement).

For a switched capture that switches between different sections within a larger area, the area of capture should use coordinates for the larger potential area.

EncodingGroup: {<encodeGroupID value>}

A field with a value equal to the encodeGroupID of the encoding group associated with the media capture.

Max Capture Encodings: {unsigned integer}

An optional attribute indicating the maximum number of capture encodings that can be simultaneously active for the media capture. If absent, this parameter defaults to 1. The minimum value for this attribute is 1. The number of simultaneous capture encodings is also limited by the restrictions of the encoding group for the media capture.

6.2. Capture Scene

In order for a provider's individual media captures to be used effectively by a consumer, the provider organizes the media captures into capture scenes, with the structure and contents of these capture scenes being sent from the provider to the consumer.

A capture scene is a structure representing the scene that is captured by a collection of capture devices. A capture scene includes one or more capture scene entries, with each entry including one or more media captures. A capture scene represents, for example, the video image of a group of people seated next to each other, along with the sound of their voices, which could be represented by some number of VCs and ACs in the capture scene entries. A middle box may also express capture scenes that it constructs from media streams it receives.

A provider may advertise multiple capture scenes or just a single capture scene. A media provider might typically use one capture scene for main participant media and another capture scene for a computer generated presentation. A capture scene may include more than one type of media. For example, a capture scene can include several capture scene entries for video captures, and several capture scene entries for audio captures.

A provider can express spatial relationships between media captures that are included in the same capture scene. But there is no spatial relationship between media captures that are in different capture scenes.

A media provider arranges media captures in a capture scene to help the media consumer choose which captures it wants. The capture scene entries in a capture scene are different alternatives the provider is suggesting for representing the capture scene. The media consumer can choose to receive all media captures from one capture scene entry for each media type (e.g. audio and video), or it can pick and choose media captures regardless of how the provider arranges them in capture scene entries. Different capture scene entries of the same media type are not necessarily mutually exclusive alternatives.

Media captures within the same capture scene entry must be of the same media type - it is not possible to mix audio and video captures in the same capture scene entry, for instance. The provider must be capable of encoding and sending all media captures in a single entry simultaneously. A consumer may decide to receive all the media captures in a single capture scene entry, but a consumer could also decide to receive just a subset of those captures. A consumer can also decide to receive media captures from different capture scene

entries.

When a provider advertises a capture scene with multiple entries, it is essentially signaling that there are multiple representations of the same scene available. In some cases, these multiple representations would typically be used simultaneously (for instance a "video entry" and an "audio entry"). In some cases the entries would conceptually be alternatives (for instance an entry consisting of 3 video captures versus an entry consisting of just a single video capture). In this latter example, the provider would in the simple case end up providing to the consumer the entry containing the number of video captures that most closely matched the media consumer's number of display devices.

The following is an example of 4 potential capture scene entries for an endpoint-style media provider:

1. (VC0, VC1, VC2) - left, center and right camera video captures
2. (VC3) - video capture associated with loudest room segment
3. (VC4) - video capture zoomed out view of all people in the room
4. (AC0) - main audio

The first entry in this capture scene example is a list of video captures with a spatial relationship to each other. Determination of the order of these captures (VC0, VC1 and VC2) for rendering purposes is accomplished through use of their Area of Capture attributes. The second entry (VC3) and the third entry (VC4) are additional alternatives of how to capture the same room in different ways. The inclusion of the audio capture in the same capture scene indicates that AC0 is associated with those video captures, meaning it comes from the same scene. The audio should be rendered in conjunction with any rendered video captures from the same capture scene.

6.2.1. Capture scene attributes

Attributes can be applied to capture scenes as well as to individual media captures. Attributes specified at this level apply to all constituent media captures.

Description attribute - list of {<description text>, <language tag>}

The optional description attribute is a list of human readable text strings which describe the capture scene. If there is more than one string in the list, then each string in the list should contain the same description, but in a different language. A provider that

advertises multiple capture scenes can provide descriptions for each of them. This attribute can contain text in any number of languages.

The language tag identifies the language of the corresponding description text. The possible values for a language tag are the values of the 'Subtag' column for the "Type: language" entries in the "Language Subtag Registry" at [IANA-Lan] originally defined in [RFC5646]. A particular language tag value MUST NOT be used more than once in the description attribute list.

Area of Scene attribute

The area of scene attribute for a capture scene has the same format as the area of capture attribute for a media capture. The area of scene is for the entire scene, which is captured by the one or more media captures in the capture scene entries. If the provider does not specify the area of scene, but does specify areas of capture, then the consumer may assume the area of scene is greater than or equal to the outer extents of the individual areas of capture.

Scale attribute

An optional attribute indicating if the numbers used for area of scene, area of capture and point of capture are in terms of millimeters, unknown scale factor, or not any scale, as described in Section 5. If any media captures have an area of capture attribute or point of capture attribute, then this scale attribute must also be defined. The possible values for this attribute are:

- "millimeters"
- "unknown"
- "no scale"

6.2.2. Capture scene entry attributes

Attributes can be applied to capture scene entries. Attributes specified at this level apply to the capture scene entry as a whole.

Scene-switch-policy: {site-switch, segment-switch}

A media provider uses this scene-switch-policy attribute to indicate its support for different switching policies. In the provider's advertisement, this attribute can have multiple values, which means the provider supports each of the indicated policies. The consumer, when it requests media captures from this capture scene entry, should also include this attribute but with only the single value (from among the values indicated by the provider) indicating the consumer's choice for which policy it wants the provider to use. If the

provider does not support any of these policies, it should omit this attribute.

The "site-switch" policy means all captures are switched at the same time to keep captures from the same endpoint site together. Let's say the speaker is at site A and everyone else is at a "remote" site. When the room at site A is shown, all the camera images from site A are forwarded to the remote sites. Therefore at each receiving remote site, all the screens display camera images from site A. This can be used to preserve full size image display, and also provide full visual context of the displayed far end, site A. In site switching, there is a fixed relation between the cameras in each room and the displays in remote rooms. The room or participants being shown is switched from time to time based on who is speaking or by manual control.

The "segment-switch" policy means different captures can switch at different times, and can be coming from different endpoints. Still using site A as where the speaker is, and "remote" to refer to all the other sites, in segment switching, rather than sending all the images from site A, only the image containing the speaker at site A is shown. The camera images of the current speaker and previous speakers (if any) are forwarded to the other sites in the conference. Therefore the screens in each site are usually displaying images from different remote sites - the current speaker at site A and the previous ones. This strategy can be used to preserve full size image display, and also capture the non-verbal communication between the speakers. In segment switching, the display depends on the activity in the remote rooms - generally, but not necessarily based on audio / speech detection.

6.3. Simultaneous Transmission Set Constraints

The provider may have constraints or limitations on its ability to send media captures. One type is caused by the physical limitations of capture mechanisms; these constraints are represented by a simultaneous transmission set. The second type of limitation reflects the encoding resources available - bandwidth and macroblocks/second. This type of constraint is captured by encoding groups, discussed below.

An endpoint or MCU can send multiple captures simultaneously, however sometimes there are constraints that limit which captures can be sent simultaneously with other captures. A device may not be able to be used in different ways at the same time. Provider advertisements are made so that the consumer will choose one of several possible mutually exclusive usages of the device. This type of constraint is expressed in a Simultaneous Transmission Set, which lists all the

media captures that can be sent at the same time. This is easier to show in an example.

Consider the example of a room system where there are 3 cameras each of which can send a separate capture covering 2 persons each- VC0, VC1, VC2. The middle camera can also zoom out and show all 6 persons, VC3. But the middle camera cannot be used in both modes at the same time - it has to either show the space where 2 participants sit or the whole 6 seats, but not both at the same time.

Simultaneous transmission sets are expressed as sets of the MCs that could physically be transmitted at the same time, (though it may not make sense to do so). In this example the two simultaneous sets are shown in Table 1. The consumer must make sure that it chooses one and not more of the mutually exclusive sets. A consumer may choose any subset of the media captures in a simultaneous set, it does not have to choose all the captures in a simultaneous set if it does not want to receive all of them.

Simultaneous Sets	
{VC0, VC1, VC2}	
{VC0, VC3, VC2}	

Table 1: Two Simultaneous Transmission Sets

A media provider includes the simultaneous sets in its provider advertisement. These simultaneous set constraints apply across all the captures scenes in the advertisement. The simultaneous transmission sets MUST allow all the media captures in a particular capture scene entry to be used simultaneously.

7. Encodings

We have considered how providers can describe the content of media to consumers. We will now consider how the providers communicate information about their abilities to send streams. We introduce two constructs - individual encodings and encoding groups. Consumers will then map the media captures they want onto the encodings with encoding parameters they want. This process is then described.

7.1. Individual Encodings

An individual encoding represents a way to encode a media capture to become a capture encoding, to be sent as an encoded media stream from

the media provider to the media consumer. An individual encoding has a set of parameters characterizing how the media is encoded. Different media types have different parameters, and different encoding algorithms may have different parameters. An individual encoding can be assigned to only one capture encoding at a time.

The parameters of an individual encoding represent the maximum values for certain aspects of the encoding. A particular instantiation into a capture encoding might use lower values than these maximums.

The following tables show the variables for audio and video encoding.

Name	Description
encodeID	A unique identifier for the individual encoding
maxBandwidth	Maximum number of bits per second
maxH264Mbps	Maximum number of macroblocks per second: $((\text{width} + 15) / 16) * ((\text{height} + 15) / 16) * \text{framesPerSecond}$
maxWidth	Video resolution's maximum supported width, expressed in pixels
maxHeight	Video resolution's maximum supported height, expressed in pixels
maxFrameRate	Maximum supported frame rate

Table 2: Individual Video Encoding Parameters

Name	Description
maxBandwidth	Maximum number of bits per second

Table 3: Individual Audio Encoding Parameters

7.2. Encoding Group

An encoding group includes a set of one or more individual encodings, plus some parameters that apply to the group as a whole. By grouping multiple individual encodings together, an encoding group describes additional constraints on bandwidth and other parameters for the group. Table 4 shows the parameters and individual encoding sets that are part of an encoding group.

Name	Description
encodeGroupID	A unique identifier for the encoding group
maxGroupBandwidth	Maximum number of bits per second relating to all encodings combined
maxGroupH264Mbps	Maximum number of macroblocks per second relating to all video encodings combined
videoEncodings[]	Set of potential encodings (list of encodeIDs)
audioEncodings[]	Set of potential encodings (list of encodeIDs)

Table 4: Encoding Group

When the individual encodings in a group are instantiated into capture encodings, each capture encoding has a bandwidth that must be less than or equal to the maxBandwidth for the particular individual encoding. The maxGroupBandwidth parameter gives the additional restriction that the sum of all the individual capture encoding bandwidths must be less than or equal to the maxGroupBandwidth value.

Likewise, the sum of the macroblocks per second of each instantiated encoding in the group must not exceed the maxGroupH264Mbps value.

The following diagram illustrates the structure of a media provider's Encoding Groups and their contents.

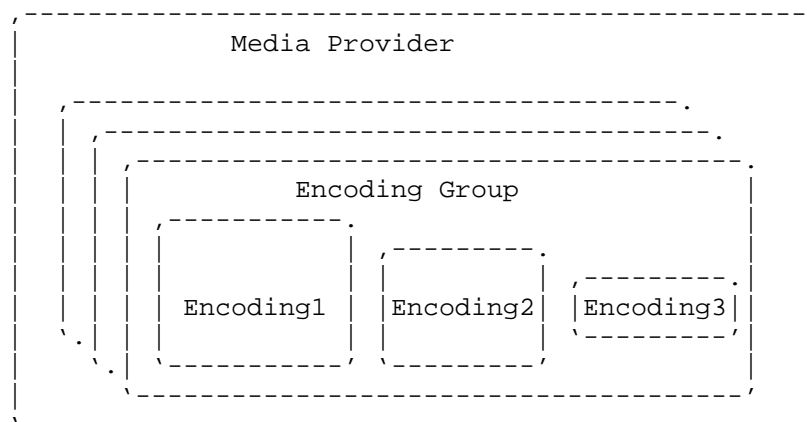


Figure 1: Encoding Group Structure

A media provider advertises one or more encoding groups. Each

encoding group includes one or more individual encodings. Each individual encoding can represent a different way of encoding media. For example one individual encoding may be 1080p60 video, another could be 720p30, with a third being CIF.

While a typical 3 codec/display system might have one encoding group per "codec box", there are many possibilities for the number of encoding groups a provider may be able to offer and for the encoding values in each encoding group.

There is no requirement for all encodings within an encoding group to be instantiated at once.

8. Associating Media Captures with Encoding Groups

Every media capture is associated with an encoding group, which is used to instantiate that media capture into one or more capture encodings. Each media capture has an encoding group attribute. The value of this attribute is the encodeGroupID for the encoding group with which it is associated. More than one media capture may use the same encoding group.

The maximum number of streams that can result from a particular encoding group constraint is equal to the number of individual encodings in the group. The actual number of capture encodings used at any time may be less than this maximum. Any of the media captures that use a particular encoding group can be encoded according to any of the individual encodings in the group. If there are multiple individual encodings in the group, then the media consumer can configure the media provider to encode a single media capture into multiple different capture encodings at the same time, subject to the Max Capture Encodings constraint, with each capture encoding following the constraints of a different individual encoding.

The Encoding Groups MUST allow all the media captures in a particular capture scene entry to be used simultaneously.

9. Consumer's Choice of Streams to Receive from the Provider

After receiving the provider's advertised media captures and associated constraints, the consumer must choose which media captures it wishes to receive, and which individual encodings from the provider it wants to use to encode the captures. Each media capture has an encoding group ID attribute which specifies which individual encodings are available to be used for that media capture.

For each media capture the consumer wants to receive, it configures one or more of the encodings in that capture's encoding group. The consumer does this by telling the provider the resolution, frame rate, bandwidth, etc. when asking for capture encodings for its chosen captures. Upon receipt of this configuration command from the consumer, the provider generates a stream for each such configured capture encoding and sends those streams to the consumer.

The consumer must have received at least one capture advertisement from the provider to be able to configure the provider's generation of media streams.

The consumer is able to change its configuration of the provider's encodings any number of times during the call, either in response to a new capture advertisement from the provider or autonomously. The consumer need not send a new configure message to the provider when it receives a new capture advertisement from the provider unless the contents of the new capture advertisement cause the consumer's current configure message to become invalid.

When choosing which streams to receive from the provider, and the encoding characteristics of those streams, the consumer needs to take several things into account: its local preference, simultaneity restrictions, and encoding limits.

9.1. Local preference

A variety of local factors will influence the consumer's choice of streams to be received from the provider:

- o if the consumer is an endpoint, it is likely that it would choose, where possible, to receive video and audio captures that match the number of display devices and audio system it has
- o if the consumer is a middle box such as an MCU, it may choose to receive loudest speaker streams (in order to perform its own media composition) and avoid pre-composed video captures
- o user choice (for instance, selection of a new layout) may result in a different set of media captures, or different encoding characteristics, being required by the consumer

9.2. Physical simultaneity restrictions

There may be physical simultaneity constraints imposed by the provider that affect the provider's ability to simultaneously send all of the captures the consumer would wish to receive. For instance, a middle box such as an MCU, when connected to a multi-

camera room system, might prefer to receive both individual camera streams of the people present in the room and an overall view of the room from a single camera. Some endpoint systems might be able to provide both of these sets of streams simultaneously, whereas others may not (if the overall room view were produced by changing the zoom level on the center camera, for instance).

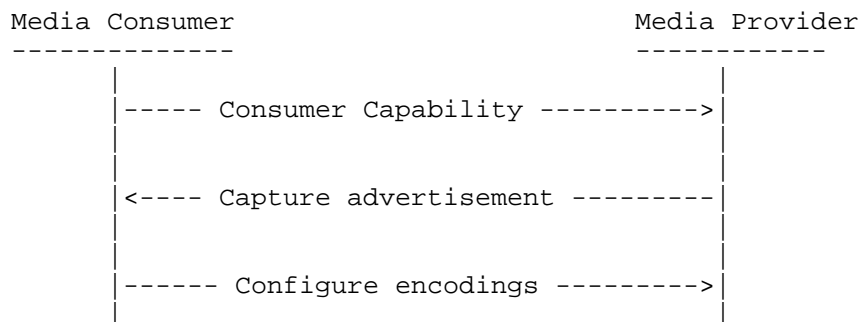
9.3. Encoding and encoding group limits

Each of the provider's encoding groups has limits on bandwidth and macroblocks per second, and the constituent potential encodings have limits on the bandwidth, macroblocks per second, video frame rate, and resolution that can be provided. When choosing the media captures to be received from a provider, a consumer device must ensure that the encoding characteristics requested for each individual media capture fits within the capability of the encoding it is being configured to use, as well as ensuring that the combined encoding characteristics for media captures fit within the capabilities of their associated encoding groups. In some cases, this could cause an otherwise "preferred" choice of capture encodings to be passed over in favour of different capture encodings - for instance, if a set of 3 media captures could only be provided at a low resolution then a 3 screen device could switch to favoring a single, higher quality, capture encoding.

9.4. Message Flow

The following diagram shows the basic flow of messages between a media provider and a media consumer. The usage of the "capture advertisement" and "configure encodings" message is described above. The consumer also sends its own capability message to the provider which may contain information about its own capabilities or restrictions.

Diagram for Message Flow



In order for a maximally-capable provider to be able to advertise a manageable number of video captures to a consumer, there is a potential use for the consumer, at the start of CLUE, to be able to inform the provider of its capabilities. One example here would be the video capture attribute set - a consumer could tell the provider the complete set of video capture attributes it is able to understand and so the provider would be able to reduce the capture scene it advertises to be tailored to the consumer.

TBD - the content of the consumer capability message needs to be better defined. The authors believe there is a need for this message, but have not worked out the details yet.

10. Extensibility

One of the most important characteristics of the Framework is its extensibility. Telepresence is a relatively new industry and while we can foresee certain directions, we also do not know everything about how it will develop. The standard for interoperability and handling multiple streams must be future-proof.

The framework itself is inherently extensible through expanding the data model types. For example:

- o Adding more types of media, such as telemetry, can be done by defining additional types of captures in addition to audio and video.
- o Adding new functionalities , such as 3-D, say, will require additional attributes describing the captures.
- o Adding a new codecs, such as H.265, can be accomplished by defining new encoding variables.

The infrastructure is designed to be extended rather than requiring new infrastructure elements. Extension comes through adding to defined types.

Assuming the implementation is in something like XML, adding data elements and attributes makes extensibility easy.

11. Examples - Using the Framework

This section shows some examples in more detail how to use the framework to represent a typical case for telepresence rooms. First an endpoint is illustrated, then an MCU case is shown.

11.1. Three screen endpoint media provider

Consider an endpoint with the following description:

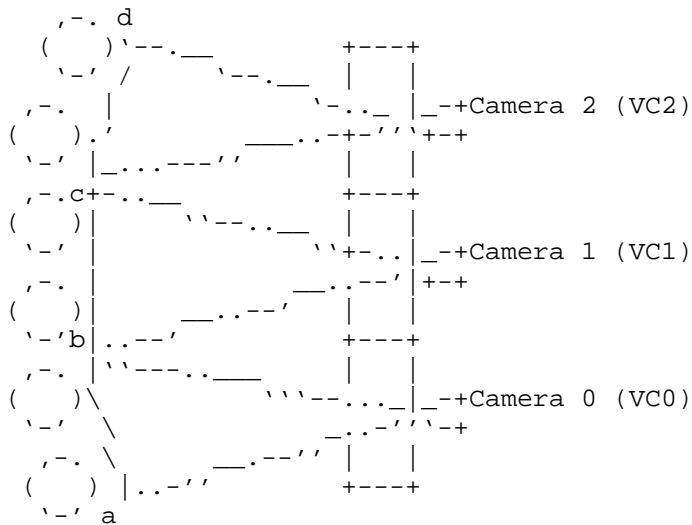
- o 3 cameras, 3 displays, a 6 person table
- o Each video device can provide one capture for each 1/3 section of the table
- o A single capture representing the active speaker can be provided
- o A single capture representing the active speaker with the other 2 captures shown picture in picture within the stream can be provided
- o A capture showing a zoomed out view of all 6 seats in the room can be provided

The audio and video captures for this endpoint can be described as follows.

Video Captures:

- o VC0- (the camera-left camera stream), encoding group=EG0, content=main, switched=false
- o VC1- (the center camera stream), encoding group=EG1, content=main, switched=false
- o VC2- (the camera-right camera stream), encoding group=EG2, content=main, switched=false
- o VC3- (the loudest panel stream), encoding group=EG1, content=main, switched=true
- o VC4- (the loudest panel stream with PiPs), encoding group=EG1, content=main, composed=true, switched=true
- o VC5- (the zoomed out view of all people in the room), encoding group=EG1, content=main, composed=false, switched=false
- o VC6- (presentation stream), encoding group=EG1, content=slides, switched=false

The following diagram is a top view of the room with 3 cameras, 3 displays, and 6 seats. Each camera is capturing 2 people. The six seats are not all in a straight line.



The two points labeled b and c are intended to be at the midpoint between the seating positions, and where the fields of view of the cameras intersect.

The plane of interest for VC0 is a vertical plane that intersects points 'a' and 'b'.

The plane of interest for VC1 intersects points 'b' and 'c'.

The plane of interest for VC2 intersects points 'c' and 'd'.

This example uses an area scale of millimeters.

Areas of capture:

	bottom left	bottom right	top left	top right
VC0	(-2011,2850,0)	(-673,3000,0)	(-2011,2850,757)	(-673,3000,757)
VC1	(-673,3000,0)	(673,3000,0)	(-673,3000,757)	(673,3000,757)
VC2	(673,3000,0)	(2011,2850,0)	(673,3000,757)	(2011,3000,757)
VC3	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
VC4	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
VC5	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
VC6	none			

Points of capture:

VC0 (-1678,0,800)
 VC1 (0,0,800)
 VC2 (1678,0,800)
 VC3 none
 VC4 none
 VC5 (0,0,800)
 VC6 none

In this example, the right edge of the VC0 area lines up with the

left edge of the VC1 area. It doesn't have to be this way. There could be a gap or an overlap. One additional thing to note for this example is the distance from a to b is equal to the distance from b to c and the distance from c to d. All these distances are 1346 mm. This is the planar width of each area of capture for VC0, VC1, and VC2.

Note the text in parentheses (e.g. "the camera-left camera stream") is not explicitly part of the model, it is just explanatory text for this example, and is not included in the model with the media captures and attributes. Also, the "composed" boolean attribute doesn't say anything about how a capture is composed, so the media consumer can't tell based on this attribute that VC4 is composed of a "loudest panel with PiPs".

Audio Captures:

- o AC0 (camera-left), encoding group=EG3, content=main, channel format=mono
- o AC1 (camera-right), encoding group=EG3, content=main, channel format=mono
- o AC2 (center) encoding group=EG3, content=main, channel format=mono
- o AC3 being a simple pre-mixed audio stream from the room (mono), encoding group=EG3, content=main, channel format=mono
- o AC4 audio stream associated with the presentation video (mono) encoding group=EG3, content=slides, channel format=mono

Areas of capture:

	bottom left	bottom right	top left	top right
AC0	(-2011,2850,0)	(-673,3000,0)	(-2011,2850,757)	(-673,3000,757)
AC1	(673,3000,0)	(2011,2850,0)	(673,3000,757)	(2011,3000,757)
AC2	(-673,3000,0)	(673,3000,0)	(-673,3000,757)	(673,3000,757)
AC3	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
AC4	none			

The physical simultaneity information is:

Simultaneous transmission set #1 {VC0, VC1, VC2, VC3, VC4, VC6}

Simultaneous transmission set #2 {VC0, VC2, VC5, VC6}

This constraint indicates it is not possible to use all the VCs at the same time. VC5 can not be used at the same time as VC1 or VC3 or

VC4. Also, using every member in the set simultaneously may not make sense - for example VC3(loudest) and VC4 (loudest with PIP). (In addition, there are encoding constraints that make choosing all of the VCs in a set impossible. VC1, VC3, VC4, VC5, VC6 all use EG1 and EG1 has only 3 ENCs. This constraint shows up in the encoding groups, not in the simultaneous transmission sets.)

In this example there are no restrictions on which audio captures can be sent simultaneously.

Encoding Groups:

This example has three encoding groups associated with the video captures. Each group can have 3 encodings, but with each potential encoding having a progressively lower specification. In this example, 1080p60 transmission is possible (as ENC0 has a maxMbps value compatible with that) as long as it is the only active encoding in the group(as maxMbps for the entire encoding group is also 489600). Significantly, as up to 3 encodings are available per group, it is possible to transmit some video captures simultaneously that are not in the same entry in the capture scene. For example VC1 and VC3 at the same time.

It is also possible to transmit multiple capture encodings of a single video capture. For example VC0 can be encoded using ENC0 and ENC1 at the same time, as long as the encoding parameters satisfy the constraints of ENC0, ENC1, and EG0, such as one at 1080p30 and one at 720p30.

```

encodeGroupID=EG0, maxGroupH264Mbps=489600, maxGroupBandwidth=6000000
  encodeID=ENC0, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=489600, maxBandwidth=4000000
  encodeID=ENC1, maxWidth=1280, maxHeight=720, maxFrameRate=30,
    maxH264Mbps=108000, maxBandwidth=4000000
  encodeID=ENC2, maxWidth=960, maxHeight=544, maxFrameRate=30,
    maxH264Mbps=61200, maxBandwidth=4000000

encodeGroupID=EG1 maxGroupH264Mbps=489600 maxGroupBandwidth=6000000
  encodeID=ENC3, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=489600, maxBandwidth=4000000
  encodeID=ENC4, maxWidth=1280, maxHeight=720, maxFrameRate=30,
    maxH264Mbps=108000, maxBandwidth=4000000
  encodeID=ENC5, maxWidth=960, maxHeight=544, maxFrameRate=30,
    maxH264Mbps=61200, maxBandwidth=4000000

encodeGroupID=EG2 maxGroupH264Mbps=489600 maxGroupBandwidth=6000000
  encodeID=ENC6, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=489600, maxBandwidth=4000000
  encodeID=ENC7, maxWidth=1280, maxHeight=720, maxFrameRate=30,
    maxH264Mbps=108000, maxBandwidth=4000000
  encodeID=ENC8, maxWidth=960, maxHeight=544, maxFrameRate=30,
    maxH264Mbps=61200, maxBandwidth=4000000

```

Figure 2: Example Encoding Groups for Video

For audio, there are five potential encodings available, so all five audio captures can be encoded at the same time.

```

encodeGroupID=EG3, maxGroupH264Mbps=0, maxGroupBandwidth=320000
  encodeID=ENC9, maxBandwidth=64000
  encodeID=ENC10, maxBandwidth=64000
  encodeID=ENC11, maxBandwidth=64000
  encodeID=ENC12, maxBandwidth=64000
  encodeID=ENC13, maxBandwidth=64000

```

Figure 3: Example Encoding Group for Audio

Capture Scenes:

The following table represents the capture scenes for this provider. Recall that a capture scene is composed of alternative capture scene entries covering the same scene. Capture Scene #1 is for the main people captures, and Capture Scene #2 is for presentation.

Each row in the table is a separate entry in the capture scene

+	-----	+
	Capture Scene #1	
+	-----	+
	VC0, VC1, VC2	
	VC3	
	VC4	
	VC5	
	AC0, AC1, AC2	
	AC3	
+	-----	+
+	-----	+
	Capture Scene #2	
+	-----	+
	VC6	
	AC4	
+	-----	+

Different capture scenes are unique to each other, non-overlapping. A consumer can choose an entry from each capture scene. In this case the three captures VC0, VC1, and VC2 are one way of representing the video from the endpoint. These three captures should appear adjacent next to each other. Alternatively, another way of representing the Capture Scene is with the capture VC3, which automatically shows the person who is talking. Similarly for the VC4 and VC5 alternatives.

As in the video case, the different entries of audio in Capture Scene #1 represent the "same thing", in that one way to receive the audio is with the 3 audio captures (AC0, AC1, AC2), and another way is with the mixed AC3. The Media Consumer can choose an audio capture entry it is capable of receiving.

The spatial ordering is understood by the media capture attributes area and point of capture.

A Media Consumer would likely want to choose a capture scene entry to receive based in part on how many streams it can simultaneously receive. A consumer that can receive three people streams would probably prefer to receive the first entry of Capture Scene #1 (VC0, VC1, VC2) and not receive the other entries. A consumer that can receive only one people stream would probably choose one of the other entries.

If the consumer can receive a presentation stream too, it would also choose to receive the only entry from Capture Scene #2 (VC6).

11.2. Encoding Group Example

This is an example of an encoding group to illustrate how it can express dependencies between encodings.

```
encodeGroupID=EG0, maxGroupH264Mbps=489600, maxGroupBandwidth=6000000
  encodeID=VIDENC0, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=244800, maxBandwidth=4000000
  encodeID=VIDENC1, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=244800, maxBandwidth=4000000
  encodeID=AUDENC0, maxBandwidth=96000
  encodeID=AUDENC1, maxBandwidth=96000
  encodeID=AUDENC2, maxBandwidth=96000
```

Here, the encoding group is EG0. It can transmit up to two 1080p30 capture encodings (Mbps for 1080p = 244800), but it is capable of transmitting a maxFrameRate of 60 frames per second (fps). To achieve the maximum resolution (1920 x 1088) the frame rate is limited to 30 fps. However 60 fps can be achieved at a lower resolution if required by the consumer. Although the encoding group is capable of transmitting up to 6Mbit/s, no individual video encoding can exceed 4Mbit/s.

This encoding group also allows up to 3 audio encodings, AUDENC<0-2>. It is not required that audio and video encodings reside within the same encoding group, but if so then the group's overall maxBandwidth value is a limit on the sum of all audio and video encodings configured by the consumer. A system that does not wish or need to combine bandwidth limitations in this way should instead use separate encoding groups for audio and video in order for the bandwidth limitations on audio and video to not interact.

Audio and video can be expressed in separate encoding groups, as in this illustration.

```
encodeGroupID=EG0, maxGroupH264Mbps=489600, maxGroupBandwidth=6000000
  encodeID=VIDENC0, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=244800, maxBandwidth=4000000
  encodeID=VIDENC1, maxWidth=1920, maxHeight=1088, maxFrameRate=60,
    maxH264Mbps=244800, maxBandwidth=4000000

encodeGroupID=EG1, maxGroupH264Mbps=0, maxGroupBandwidth=500000
  encodeID=AUDENC0, maxBandwidth=96000
  encodeID=AUDENC1, maxBandwidth=96000
  encodeID=AUDENC2, maxBandwidth=96000
```

11.3. The MCU Case

This section shows how an MCU might express its Capture Scenes, intending to offer different choices for consumers that can handle different numbers of streams. A single audio capture stream is provided for all single and multi-screen configurations that can be associated (e.g. lip-synced) with any combination of video captures at the consumer.

Capture Scene #1	note
VC0	video capture for single screen consumer
VC1, VC2	video capture for 2 screen consumer
VC3, VC4, VC5	video capture for 3 screen consumer
VC6, VC7, VC8, VC9	video capture for 4 screen consumer
AC0	audio capture representing all participants

If / when a presentation stream becomes active within the conference, the MCU might re-advertise the available media as:

Capture Scene #2	note
VC10	video capture for presentation
AC1	presentation audio to accompany VC10

11.4. Media Consumer Behavior

This section gives an example of how a media consumer might behave when deciding how to request streams from the three screen endpoint described in the previous section.

The receive side of a call needs to balance its requirements, based on number of screens and speakers, its decoding capabilities and available bandwidth, and the provider's capabilities in order to optimally configure the provider's streams. Typically it would want to receive and decode media from each capture scene advertised by the provider.

A sane, basic, algorithm might be for the consumer to go through each capture scene in turn and find the collection of video captures that best matches the number of screens it has (this might include consideration of screens dedicated to presentation video display rather than "people" video) and then decide between alternative entries in the video capture scenes based either on hard-coded

preferences or user choice. Once this choice has been made, the consumer would then decide how to configure the provider's encoding groups in order to make best use of the available network bandwidth and its own decoding capabilities.

11.4.1. One screen consumer

VC3, VC4 and VC5 are all different entries by themselves, not grouped together in a single entry, so the receiving device should choose between one of those. The choice would come down to whether to see the greatest number of participants simultaneously at roughly equal precedence (VC5), a switched view of just the loudest region (VC3) or a switched view with PiPs (VC4). An endpoint device with a small amount of knowledge of these differences could offer a dynamic choice of these options, in-call, to the user.

11.4.2. Two screen consumer configuring the example

Mixing systems with an even number of screens, "2n", and those with "2n+1" cameras (and vice versa) is always likely to be the problematic case. In this instance, the behavior is likely to be determined by whether a "2 screen" system is really a "2 decoder" system, i.e., whether only one received stream can be displayed per screen or whether more than 2 streams can be received and spread across the available screen area. To enumerate 3 possible behaviors here for the 2 screen system when it learns that the far end is "ideally" expressed via 3 capture streams:

1. Fall back to receiving just a single stream (VC3, VC4 or VC5 as per the 1 screen consumer case above) and either leave one screen blank or use it for presentation if / when a presentation becomes active
2. Receive 3 streams (VC0, VC1 and VC2) and display across 2 screens (either with each capture being scaled to 2/3 of a screen and the centre capture being split across 2 screens) or, as would be necessary if there were large bezels on the screens, with each stream being scaled to 1/2 the screen width and height and there being a 4th "blank" panel. This 4th panel could potentially be used for any presentation that became active during the call.
3. Receive 3 streams, decode all 3, and use control information indicating which was the most active to switch between showing the left and centre streams (one per screen) and the centre and right streams.

For an endpoint capable of all 3 methods of working described above, again it might be appropriate to offer the user the choice of display

mode.

11.4.3. Three screen consumer configuring the example

This is the most straightforward case - the consumer would look to identify a set of streams to receive that best matched its available screens and so the VC0 plus VC1 plus VC2 should match optimally. The spatial ordering would give sufficient information for the correct video capture to be shown on the correct screen, and the consumer would either need to divide a single encoding group's capability by 3 to determine what resolution and frame rate to configure the provider with or to configure the individual video captures' encoding groups with what makes most sense (taking into account the receive side decode capabilities, overall call bandwidth, the resolution of the screens plus any user preferences such as motion vs sharpness).

12. Acknowledgements

Mark Gorzyinski contributed much to the approach. We want to thank Stephen Botzko for helpful discussions on audio.

13. IANA Considerations

TBD

14. Security Considerations

TBD

15. Changes Since Last Version

NOTE TO THE RFC-Editor: Please remove this section prior to publication as an RFC.

Changes from 06 to 07:

1. Ticket #9. Rename Axis of Capture Point attribute to Point on Line of Capture. Clarify the description of this attribute.
2. Ticket #17. Add "capture encoding" definition. Use this new term throughout document as appropriate, replacing some usage of the terms "stream" and "encoding".

3. Ticket #18. Add Max Capture Encodings media capture attribute.
4. Add clarification that different capture scene entries are not necessarily mutually exclusive.

Changes from 05 to 06:

1. Capture scene description attribute is a list of text strings, each in a different language, rather than just a single string.
2. Add new Axis of Capture Point attribute.
3. Remove appendices A.1 through A.6.
4. Clarify that the provider must use the same coordinate system with same scale and origin for all coordinates within the same capture scene.

Changes from 04 to 05:

1. Clarify limitations of "composed" attribute.
2. Add new section "capture scene entry attributes" and add the attribute "scene-switch-policy".
3. Add capture scene description attribute and description language attribute.
4. Editorial changes to examples section for consistency with the rest of the document.

Changes from 03 to 04:

1. Remove sentence from overview - "This constitutes a significant change ..."
2. Clarify a consumer can choose a subset of captures from a capture scene entry or a simultaneous set (in section "capture scene" and "consumer's choice...").
3. Reword first paragraph of Media Capture Attributes section.
4. Clarify a stereo audio capture is different from two mono audio captures (description of audio channel format attribute).
5. Clarify what it means when coordinate information is not specified for area of capture, point of capture, area of scene.

6. Change the term "producer" to "provider" to be consistent (it was just in two places).
7. Change name of "purpose" attribute to "content" and refer to RFC4796 for values.
8. Clarify simultaneous sets are part of a provider advertisement, and apply across all capture scenes in the advertisement.
9. Remove sentence about lip-sync between all media captures in a capture scene.
10. Combine the concepts of "capture scene" and "capture set" into a single concept, using the term "capture scene" to replace the previous term "capture set", and eliminating the original separate capture scene concept.

16. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, February 2006.
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.
- [RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.
- [RFC5646] Phillips, A. and M. Davis, "Tags for Identifying Languages", BCP 47, RFC 5646, September 2009.
- [IANA-Lan] IANA, "Language Subtag Registry",

[http://www.iana.org/assignments/
language-subtag-registry](http://www.iana.org/assignments/language-subtag-registry)>.

Authors' Addresses

Allyn Romanow
Cisco Systems
San Jose, CA 95134
USA

Email: allyn@cisco.com

Mark Duckworth (editor)
Polycom
Andover, MA 01810
USA

Email: mark.duckworth@polycom.com

Andrew Pepperell
Silverflare
Uxbridge, England
UK

Email: apeppere@gmail.com

Brian Baldino
Cisco Systems
San Jose, CA 95134
USA

Email: bbaldino@cisco.com

CLUE WG
Internet Draft
Intended status: Standards Track
Expires: July 8, 2016

M. Duckworth, Ed.
Polycom
A. Pepperell
Acano
S. Wenger
Vidyo
January 8, 2016

Framework for Telepresence Multi-Streams
draft-ietf-clue-framework-25.txt

Abstract

This document defines a framework for a protocol to enable devices in a telepresence conference to interoperate. The protocol enables communication of information about multiple media streams so a sending system and receiving system can make reasonable decisions about transmitting, selecting and rendering the media streams. This protocol is used in addition to SIP signaling and SDP negotiation for setting up a telepresence session.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on July 8, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction.....	3
2. Terminology.....	4
3. Definitions.....	4
4. Overview and Motivation.....	7
5. Description of the Framework/Model.....	10
6. Spatial Relationships.....	15
7. Media Captures and Capture Scenes.....	17
7.1. Media Captures.....	17
7.1.1. Media Capture Attributes.....	18
7.2. Multiple Content Capture.....	24
7.2.1. MCC Attributes.....	25
7.3. Capture Scene.....	30
7.3.1. Capture Scene attributes.....	33
7.3.2. Capture Scene View attributes.....	33
7.4. Global View List.....	34
8. Simultaneous Transmission Set Constraints.....	35
9. Encodings.....	37
9.1. Individual Encodings.....	37
9.2. Encoding Group.....	38
9.3. Associating Captures with Encoding Groups.....	39
10. Consumer's Choice of Streams to Receive from the Provider....	40
10.1. Local preference.....	43
10.2. Physical simultaneity restrictions.....	43
10.3. Encoding and encoding group limits.....	43
11. Extensibility.....	44
12. Examples - Using the Framework (Informative).....	44
12.1. Provider Behavior.....	44
12.1.1. Three screen Endpoint Provider.....	44
12.1.2. Encoding Group Example.....	51
12.1.3. The MCU Case.....	52

12.2. Media Consumer Behavior.....	53
12.2.1. One screen Media Consumer.....	53
12.2.2. Two screen Media Consumer configuring the example..	54
12.2.3. Three screen Media Consumer configuring the example	55
12.3. Multipoint Conference utilizing Multiple Content Captures	55
12.3.1. Single Media Captures and MCC in the same Advertisement.....	55
12.3.2. Several MCCs in the same Advertisement.....	59
12.3.3. Heterogeneous conference with switching and composition.....	60
12.3.4. Heterogeneous conference with voice activated switching.....	67
13. Acknowledgements.....	70
14. IANA Considerations.....	70
15. Security Considerations.....	70
16. Changes Since Last Version.....	73
17. Normative References.....	81
18. Informative References.....	82
19. Authors' Addresses.....	83

1. Introduction

Current telepresence systems, though based on open standards such as RTP [RFC3550] and SIP [RFC3261], cannot easily interoperate with each other. A major factor limiting the interoperability of telepresence systems is the lack of a standardized way to describe and negotiate the use of multiple audio and video streams comprising the media flows. This document provides a framework for protocols to enable interoperability by handling multiple streams in a standardized way. The framework is intended to support the use cases described in Use Cases for Telepresence Multistreams [RFC7205] and to meet the requirements in Requirements for Telepresence Multistreams [RFC7262]. This includes cases using multiple media streams that are not necessarily telepresence.

This document occasionally refers to the term "CLUE", in capital letters. CLUE is an acronym for "ControLLing mUltiple streams for tElepresence", which is the name of the IETF working group in which this document and certain companion documents have been developed. Often, CLUE-something refers to something that has been designed by the CLUE working group; for example, this document may be called the CLUE-framework.

The basic session setup for the use cases is based on SIP [RFC3261] and SDP offer/answer [RFC3264]. In addition to basic SIP & SDP offer/answer, CLUE specific signaling is required to exchange the information describing the multiple media streams. The motivation for this framework, an overview of the signaling, and information required to be exchanged is described in subsequent sections of this document. Companion documents describe the signaling details [I-D.ietf-clue-signaling] and the data model [I-D.ietf-clue-data-model-schema] and protocol [I-D.ietf-clue-protocol].

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Definitions

The terms defined below are used throughout this document and companion documents. In order to easily identify the use of a defined term, those terms are capitalized.

Advertisement: a CLUE message a Media Provider sends to a Media Consumer describing specific aspects of the content of the media, and any restrictions it has in terms of being able to provide certain Streams simultaneously.

Audio Capture: Media Capture for audio. Denoted as ACn in the examples in this document.

Capture: Same as Media Capture.

Capture Device: A device that converts physical input, such as audio, video or text, into an electrical signal, in most cases to be fed into a media encoder.

Capture Encoding: A specific encoding of a Media Capture, to be sent by a Media Provider to a Media Consumer via RTP.

Capture Scene: a structure representing a spatial region captured by one or more Capture Devices, each capturing media representing a portion of the region. The spatial region represented by a Capture Scene may correspond to a real region in physical space, such as a room. A Capture Scene includes attributes and one or more Capture Scene Views, with each view including one or more Media Captures.

Capture Scene View (CSV): a list of Media Captures of the same media type that together form one way to represent the entire Capture Scene.

CLUE-capable device: A device that supports the CLUE data channel [I-D.ietf-clue-datachannel], the CLUE protocol [I-D.ietf-clue-protocol] and the principles of CLUE negotiation, and seeks CLUE-enabled calls.

CLUE-enabled call: A call in which two CLUE-capable devices have successfully negotiated support for a CLUE data channel in SDP [RFC4566]. A CLUE-enabled call is not necessarily immediately able to send CLUE-controlled media; negotiation of the data channel and of the CLUE protocol must complete first. Calls between two CLUE-capable devices which have not yet successfully completed negotiation of support for the CLUE data channel in SDP are not considered CLUE-enabled.

Conference: used as defined in [RFC4353], A Framework for Conferencing within the Session Initiation Protocol (SIP).

Configure Message: A CLUE message a Media Consumer sends to a Media Provider specifying which content and Media Streams it wants to receive, based on the information in a corresponding Advertisement message.

Consumer: short for Media Consumer.

Encoding: short for Individual Encoding.

Encoding Group: A set of encoding parameters representing a total media encoding capability to be sub-divided across potentially multiple Individual Encodings.

Endpoint: A CLUE-capable device which is the logical point of final termination through receiving, decoding and rendering, and/or initiation through capturing, encoding, and sending of media streams. An endpoint consists of one or more physical devices

which source and sink media streams, and exactly one [RFC4353] Participant (which, in turn, includes exactly one SIP User Agent). Endpoints can be anything from multiscreen/multicamera rooms to handheld devices.

Global View: A set of references to one or more Capture Scene Views of the same media type that are defined within Scenes of the same advertisement. A Global View is a suggestion from the Provider to the Consumer for one set of CSVs that provide a useful representation of all the scenes in the advertisement.

Global View List: A list of Global Views included in an Advertisement. A Global View List may include Global Views of different media types.

Individual Encoding: a set of parameters representing a way to encode a Media Capture to become a Capture Encoding.

Multipoint Control Unit (MCU): a CLUE-capable device that connects two or more endpoints together into one single multimedia conference [RFC5117]. An MCU includes an [RFC4353]-like Mixer, without the [RFC4353] requirement to send media to each participant.

Media: Any data that, after suitable encoding, can be conveyed over RTP, including audio, video or timed text.

Media Capture: a source of Media, such as from one or more Capture Devices or constructed from other Media streams.

Media Consumer: a CLUE-capable device that intends to receive Capture Encodings.

Media Provider: a CLUE-capable device that intends to send Capture Encodings.

Multiple Content Capture (MCC): A Capture that mixes and/or switches other Captures of a single type. (E.g. all audio or all video.) Particular Media Captures may or may not be present in the resultant Capture Encoding depending on time or space. Denoted as MCCn in the example cases in this document.

Plane of Interest: The spatial plane within a scene containing the most relevant subject matter.

Provider: Same as Media Provider.

Render: the process of generating a representation from media, such as displayed motion video or sound emitted from loudspeakers.

Scene: Same as Capture Scene

Simultaneous Transmission Set: a set of Media Captures that can be transmitted simultaneously from a Media Provider.

Single Media Capture: A capture which contains media from a single source capture device, e.g. an audio capture from a single microphone, a video capture from a single camera.

Spatial Relation: The arrangement in space of two objects, in contrast to relation in time or other relationships.

Stream: a Capture Encoding sent from a Media Provider to a Media Consumer via RTP [RFC3550].

Stream Characteristics: the media stream attributes commonly used in non-CLUE SIP/SDP environments (such as: media codec, bit rate, resolution, profile/level etc.) as well as CLUE specific attributes, such as the Capture ID or a spatial location.

Video Capture: Media Capture for video. Denoted as VCn in the example cases in this document.

Video Composite: A single image that is formed, normally by an RTP mixer inside an MCU, by combining visual elements from separate sources.

4. Overview and Motivation

This section provides an overview of the functional elements defined in this document to represent a telepresence or multistream system. The motivations for the framework described in this document are also provided.

Two key concepts introduced in this document are the terms "Media Provider" and "Media Consumer". A Media Provider represents the entity that sends the media and a Media Consumer represents the entity that receives the media. A Media Provider provides Media in the form of RTP packets, a Media Consumer consumes those RTP packets. Media Providers and Media Consumers can reside in

Endpoints or in Multipoint Control Units (MCUs). A Media Provider in an Endpoint is usually associated with the generation of media for Media Captures; these Media Captures are typically sourced from cameras, microphones, and the like. Similarly, the Media Consumer in an Endpoint is usually associated with renderers, such as screens and loudspeakers. In MCUs, Media Providers and Consumers can have the form of outputs and inputs, respectively, of RTP mixers, RTP translators, and similar devices. Typically, telepresence devices such as Endpoints and MCUs would perform as both Media Providers and Media Consumers, the former being concerned with those devices' transmitted media and the latter with those devices' received media. In a few circumstances, a CLUE-capable device includes only Consumer or Provider functionality, such as recorder-type Consumers or webcam-type Providers.

The motivations for the framework outlined in this document include the following:

(1) Endpoints in telepresence systems typically have multiple Media Capture and Media Render devices, e.g., multiple cameras and screens. While previous system designs were able to set up calls that would capture media using all cameras and display media on all screens, for example, there was no mechanism that could associate these Media Captures with each other in space and time, in a cross-vendor interoperable way.

(2) The mere fact that there are multiple capturing and rendering devices, each of which may be configurable in aspects such as zoom, leads to the difficulty that a variable number of such devices can be used to capture different aspects of a region. The Capture Scene concept allows for the description of multiple setups for those multiple capture devices that could represent sensible operation points of the physical capture devices in a room, chosen by the operator. A Consumer can pick and choose from those configurations based on its rendering abilities and inform the Provider about its choices. Details are provided in section 7.

(3) In some cases, physical limitations or other reasons disallow the concurrent use of a device in more than one setup. For example, the center camera in a typical three-camera conference room can set its zoom objective either to capture only the middle few seats, or all seats of a room, but not both concurrently. The Simultaneous Transmission Set concept allows a Provider to signal

such limitations. Simultaneous Transmission Sets are part of the Capture Scene description, and are discussed in section 8.

(4) Often, the devices in a room do not have the computational complexity or connectivity to deal with multiple encoding options simultaneously, even if each of these options is sensible in certain scenarios, and even if the simultaneous transmission is also sensible (i.e. in case of multicast media distribution to multiple endpoints). Such constraints can be expressed by the Provider using the Encoding Group concept, described in section 9.

(5) Due to the potentially large number of RTP streams required for a Multimedia Conference involving potentially many Endpoints, each of which can have many Media Captures and media renderers, it has become common to multiplex multiple RTP streams onto the same transport address, so to avoid using the port number as a multiplexing point and the associated shortcomings such as NAT/firewall traversal. The large number of possible permutations of sensible options a Media Provider can make available to a Media Consumer makes a mechanism desirable that allows it to narrow down the number of possible options that a SIP offer/answer exchange has to consider. Such information is made available using protocol mechanisms specified in this document and companion documents. The Media Provider and Media Consumer may use information in CLUE messages to reduce the complexity of SIP offer/answer messages. Also, there are aspects of the control of both Endpoints and MCUs that dynamically change during the progress of a call, such as audio-level based screen switching, layout changes, and so on, which need to be conveyed. Note that these control aspects are complementary to those specified in traditional SIP based conference management such as BFCP. An exemplary call flow can be found in section 5.

Finally, all this information needs to be conveyed, and the notion of support for it needs to be established. This is done by the negotiation of a "CLUE channel", a data channel negotiated early during the initiation of a call. An Endpoint or MCU that rejects the establishment of this data channel, by definition, does not support CLUE based mechanisms, whereas an Endpoint or MCU that accepts it is indicating support for CLUE as specified in this document and its companion documents.

5. Description of the Framework/Model

The CLUE framework specifies how multiple media streams are to be handled in a telepresence conference.

A Media Provider (transmitting Endpoint or MCU) describes specific aspects of the content of the media and the media stream encodings it can send in an Advertisement; and the Media Consumer responds to the Media Provider by specifying which content and media streams it wants to receive in a Configure message. The Provider then transmits the asked-for content in the specified streams.

This Advertisement and Configure typically occur during call initiation, after CLUE has been enabled in a call, but MAY also happen at any time throughout the call, whenever there is a change in what the Consumer wants to receive or (perhaps less common) the Provider can send.

An Endpoint or MCU typically act as both Provider and Consumer at the same time, sending Advertisements and sending Configurations in response to receiving Advertisements. (It is possible to be just one or the other.)

The data model [I-D.ietf-clue-data-model-schema] is based around two main concepts: a Capture and an Encoding. A Media Capture (MC), such as of type audio or video, has attributes to describe the content a Provider can send. Media Captures are described in terms of CLUE-defined attributes, such as spatial relationships and purpose of the capture. Providers tell Consumers which Media Captures they can provide, described in terms of the Media Capture attributes.

A Provider organizes its Media Captures into one or more Capture Scenes, each representing a spatial region, such as a room. A Consumer chooses which Media Captures it wants to receive from the Capture Scenes.

In addition, the Provider can send the Consumer a description of the Individual Encodings it can send in terms of identifiers which relate to items in SDP [RFC4566].

The Provider can also specify constraints on its ability to provide Media, and a sensible design choice for a Consumer is to take these into account when choosing the content and Capture Encodings it requests in the later offer/answer exchange. Some constraints are

due to the physical limitations of devices--for example, a camera may not be able to provide zoom and non-zoom views simultaneously. Other constraints are system based, such as maximum bandwidth.

The following diagram illustrates the information contained in an Advertisement.

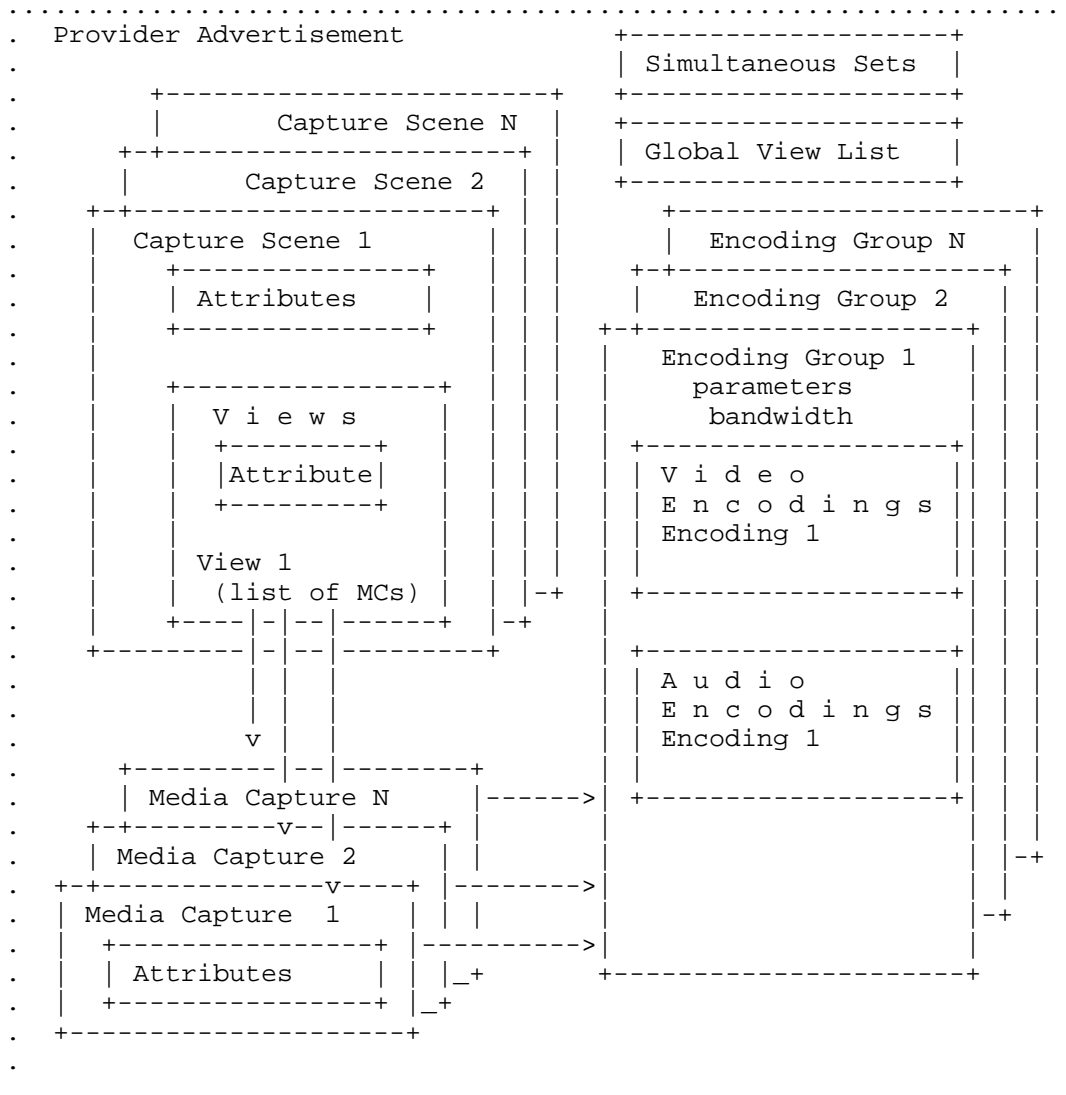


Figure 1: Advertisement Structure

A very brief outline of the call flow used by a simple system (two Endpoints) in compliance with this document can be described as follows, and as shown in the following figure.

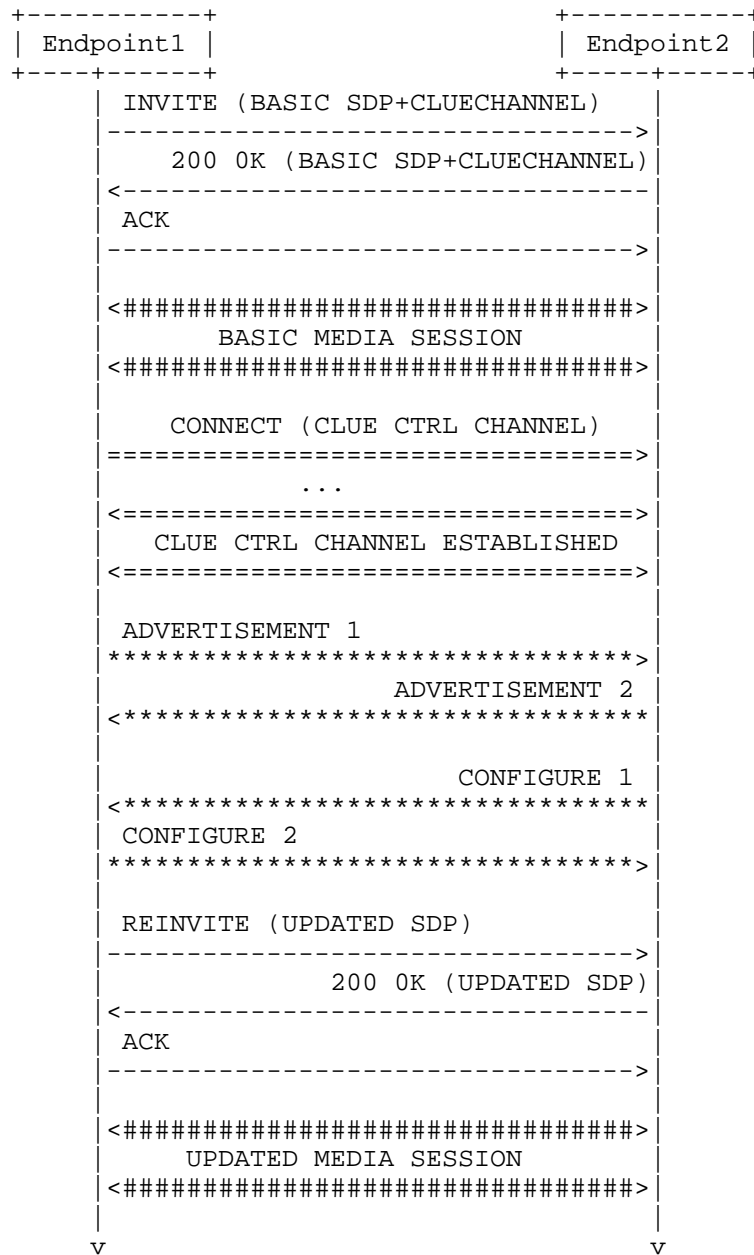


Figure 2: Basic Information Flow

An initial offer/answer exchange establishes a basic media session, for example audio-only, and a CLUE channel between two Endpoints. With the establishment of that channel, the endpoints have consented to use the CLUE protocol mechanisms and, therefore, **MUST** adhere to the CLUE protocol suite as outlined herein.

Over this CLUE channel, the Provider in each Endpoint conveys its characteristics and capabilities by sending an Advertisement as specified herein. The Advertisement is typically not sufficient to set up all media. The Consumer in the Endpoint receives the information provided by the Provider, and can use it for several purposes. It uses it, along with information from an offer/answer exchange, to construct a CLUE Configure message to tell the Provider what the Consumer wishes to receive. Also, the Consumer may use the information provided to tailor the SDP it is going to send during any following SIP offer/answer exchange, and its reaction to SDP it receives in that step. It is often a sensible implementation choice to do so. Spatial relationships associated with the Media can be included in the Advertisement, and it is often sensible for the Media Consumer to take those spatial relationships into account when tailoring the SDP. The Consumer can also limit the number of encodings it must set up resources to receive, and not waste resources on unwanted encodings, because it has the Provider's Advertisement information ahead of time to determine what it really wants to receive. The Consumer can also use the Advertisement information for local rendering decisions.

This initial CLUE exchange is followed by an SDP offer/answer exchange that not only establishes those aspects of the media that have not been "negotiated" over CLUE, but has also the effect of setting up the media transmission itself, involving potentially security exchanges, ICE, and whatnot. This step is plain vanilla SIP.

During the lifetime of a call, further exchanges **MAY** occur over the CLUE channel. In some cases, those further exchanges lead to a modified system behavior of Provider or Consumer (or both) without any other protocol activity such as further offer/answer exchanges. For example, a Configure Message requesting the Provider to place a different Capture source into a Capture Encoding, signaled over the CLUE channel, ought not to lead to heavy-handed mechanisms like SIP re-invites. However, in other cases, after the CLUE negotiation an additional offer/answer exchange becomes necessary. For example,

if both sides decide to upgrade the call from a single screen to a multi-screen call and more bandwidth is required for the additional video channels compared to what was previously negotiated using offer/answer, a new O/A exchange is required.

One aspect of the protocol outlined herein and specified in more detail in companion documents is that it makes available, to the Consumer, information regarding the Provider's capabilities to deliver Media, and attributes related to that Media such as their spatial relationship. The operation of the renderer inside the Consumer is unspecified in that it can choose to ignore some information provided by the Provider, and/or not render media streams available from the Provider (although the Consumer follows the CLUE protocol and, therefore, gracefully receives and responds to the Provider's information using a Configure operation).

A CLUE-capable device interoperates with a device that does not support CLUE. The CLUE-capable device can determine, by the result of the initial offer/answer exchange, if the other device supports and wishes to use CLUE. The specific mechanism for this is described in [I-D.ietf-clue-signaling]. If the other device does not use CLUE, then the CLUE-capable device falls back to behavior that does not require CLUE.

As for the media, Provider and Consumer have an end-to-end communication relationship with respect to (RTP transported) media; and the mechanisms described herein and in companion documents do not change the aspects of setting up those RTP flows and sessions. In other words, the RTP media sessions conform to the negotiated SDP whether or not CLUE is used.

6. Spatial Relationships

In order for a Consumer to perform a proper rendering, it is often necessary or at least helpful for the Consumer to have received spatial information about the streams it is receiving. CLUE defines a coordinate system that allows Media Providers to describe the spatial relationships of their Media Captures to enable proper scaling and spatially sensible rendering of their streams. The coordinate system is based on a few principles:

- o Each Capture Scene has a distinct coordinate system, unrelated to the coordinate systems of other scenes.

- o Simple systems which do not have multiple Media Captures to associate spatially need not use the coordinate model, although it can still be useful to provide an Area of Capture.
- o Coordinates can be either in real, physical units (millimeters), have an unknown scale or have no physical scale. Systems which know their physical dimensions (for example professionally installed Telepresence room systems) MUST provide those real-world measurements to enable the best user experience for advanced receiving systems that can utilize this information. Systems which don't know specific physical dimensions but still know relative distances MUST use 'unknown scale'. 'No scale' is intended to be used only where Media Captures from different devices (with potentially different scales) will be forwarded alongside one another (e.g. in the case of an MCU).
 - * "Millimeters" means the scale is in millimeters.
 - * "Unknown" means the scale is not necessarily millimeters, but the scale is the same for every Capture in the Capture Scene.
 - * "No Scale" means the scale could be different for each capture- an MCU Provider that advertises two adjacent captures and picks sources (which can change quickly) from different endpoints might use this value; the scale could be different and changing for each capture. But the areas of capture still represent a spatial relation between captures.
- o The coordinate system is right-handed Cartesian X, Y, Z with the origin at a spatial location of the Provider's choosing. The Provider MUST use the same coordinate system with the same scale and origin for all coordinates within the same Capture Scene.

The direction of increasing coordinate values is:

X increases from left to right, from the point of view of an observer at the front of the room looking toward the back
Y increases from the front of the room to the back of the room
Z increases from low to high (i.e. floor to ceiling)

Cameras in a scene typically point in the direction of increasing Y, from front to back. But there could be multiple cameras pointing in different directions. If the physical space does not have a well-defined front and back, the provider chooses any direction for X and Y and Z consistent with right-handed coordinates.

7. Media Captures and Capture Scenes

This section describes how Providers can describe the content of media to Consumers.

7.1. Media Captures

Media Captures are the fundamental representations of streams that a device can transmit. What a Media Capture actually represents is flexible:

- o It can represent the immediate output of a physical source (e.g. camera, microphone) or 'synthetic' source (e.g. laptop computer, DVD player)
- o It can represent the output of an audio mixer or video composer
- o It can represent a concept such as 'the loudest speaker'
- o It can represent a conceptual position such as 'the leftmost stream'

To identify and distinguish between multiple Capture instances Captures have a unique identity. For instance: VC1, VC2 and AC1, AC2, where VC1 and VC2 refer to two different video captures and AC1 and AC2 refer to two different audio captures.

Some key points about Media Captures:

- . A Media Capture is of a single media type (e.g. audio or video)
- . A Media Capture is defined in a Capture Scene and is given an Advertisement unique identity. The identity may be referenced outside the Capture Scene that defines it through a Multiple Content Capture (MCC)
- . A Media Capture may be associated with one or more Capture Scene Views
- . A Media Capture has exactly one set of spatial information
- . A Media Capture can be the source of at most one Capture Encoding

Each Media Capture can be associated with attributes to describe what it represents.

7.1.1.1. Media Capture Attributes

Media Capture Attributes describe information about the Captures. A Provider can use the Media Capture Attributes to describe the Captures for the benefit of the Consumer of the Advertisement message. All these attributes are optional. Media Capture Attributes include:

- . Spatial information, such as point of capture, point on line of capture, and area of capture, all of which, in combination define the capture field of, for example, a camera
- . Other descriptive information to help the Consumer choose between captures (e.g. description, presentation, view, priority, language, person information and type)

The sub-sections below define the Capture attributes.

7.1.1.1.1. Point of Capture

The Point of Capture attribute is a field with a single Cartesian (X, Y, Z) point value which describes the spatial location of the capturing device (such as camera). For an Audio Capture with multiple microphones, the Point of Capture defines the nominal mid-point of the microphones.

7.1.1.1.2. Point on Line of Capture

The Point on Line of Capture attribute is a field with a single Cartesian (X, Y, Z) point value which describes a position in space of a second point on the axis of the capturing device, toward the direction it is pointing; the first point being the Point of Capture (see above).

Together, the Point of Capture and Point on Line of Capture define the direction and axis of the capturing device, for example the optical axis of a camera or the axis of a microphone. The Media Consumer can use this information to adjust how it renders the received media if it so chooses.

For an Audio Capture, the Media Consumer can use this information along with the Audio Capture Sensitivity Pattern to define a 3-dimensional volume of capture where sounds can be expected to be picked up by the microphone providing this specific audio capture. If the Consumer wants to associate an Audio Capture with a Video Capture, it can compare this volume with the area of capture for

video media to provide a check on whether the audio capture is indeed spatially associated with the video capture. For example, a video area of capture that fails to intersect at all with the audio volume of capture, or is at such a long radial distance from the microphone point of capture that the audio level would be very low, would be inappropriate.

7.1.1.3. Area of Capture

The Area of Capture is a field with a set of four (X, Y, Z) points as a value which describes the spatial location of what is being "captured". This attribute applies only to video captures, not other types of media. By comparing the Area of Capture for different Video Captures within the same Capture Scene a Consumer can determine the spatial relationships between them and render them correctly.

The four points MUST be co-planar, forming a quadrilateral, which defines the Plane of Interest for the particular Media Capture.

If the Area of Capture is not specified, it means the Video Capture might be spatially related to other Captures in the same Scene, but there is no detailed information on the relationship. For a switched Capture that switches between different sections within a larger area, the area of capture MUST use coordinates for the larger potential area.

7.1.1.4. Mobility of Capture

The Mobility of Capture attribute indicates whether or not the point of capture, line on point of capture, and area of capture values stay the same over time, or are expected to change (potentially frequently). Possible values are static, dynamic, and highly dynamic.

An example for "dynamic" is a camera mounted on a stand which is occasionally hand-carried and placed at different positions in order to provide the best angle to capture a work task. A camera worn by a person who moves around the room is an example for "highly dynamic". In either case, the effect is that the capture point, capture axis and area of capture change with time.

The capture point of a static Capture MUST NOT move for the life of the CLUE session. The capture point of dynamic Captures is categorized by a change in position followed by a reasonable period

of stability--in the order of magnitude of minutes. Highly dynamic captures are categorized by a capture point that is constantly moving. If the "area of capture", "capture point" and "line of capture" attributes are included with dynamic or highly dynamic Captures they indicate spatial information at the time of the Advertisement.

7.1.1.5. Audio Capture Sensitivity Pattern

The Audio Capture Sensitivity Pattern attribute applies only to audio captures. This attribute gives information about the nominal sensitivity pattern of the microphone which is the source of the Capture. Possible values include patterns such as omni, shotgun, cardioid, hyper-cardioid.

7.1.1.6. Description

The Description attribute is a human-readable description (which could be in multiple languages) of the Capture.

7.1.1.7. Presentation

The Presentation attribute indicates that the capture originates from a presentation device, that is one that provides supplementary information to a conference through slides, video, still images, data etc. Where more information is known about the capture it MAY be expanded hierarchically to indicate the different types of presentation media, e.g. presentation.slides, presentation.image etc.

Note: It is expected that a number of keywords will be defined that provide more detail on the type of presentation. Refer to [I-D.ietf-clue-data-model-schema] for how to extend the model.

7.1.1.8. View

The View attribute is a field with enumerated values, indicating what type of view the Capture relates to. The Consumer can use this information to help choose which Media Captures it wishes to receive. Possible values are:

Room - Captures the entire scene

Table - Captures the conference table with seated people

Individual - Captures an individual person

Lectern - Captures the region of the lectern including the presenter, for example in a classroom style conference room

Audience - Captures a region showing the audience in a classroom style conference room

7.1.1.9. Language

The Language attribute indicates one or more languages used in the content of the Media Capture. Captures MAY be offered in different languages in case of multilingual and/or accessible conferences. A Consumer can use this attribute to differentiate between them and pick the appropriate one.

Note that the Language attribute is defined and meaningful both for audio and video captures. In case of audio captures, the meaning is obvious. For a video capture, "Language" could, for example, be sign interpretation or text.

The Language attribute is coded per [RFC5646].

7.1.1.10. Person Information

The Person Information attribute allows a Provider to provide specific information regarding the people in a Capture (regardless of whether or not the capture has a Presentation attribute). The Provider may gather the information automatically or manually from a variety of sources however the xCard [RFC6351] format is used to convey the information. This allows various information such as Identification information (section 6.2/[RFC6350]), Communication Information (section 6.4/[RFC6350]) and Organizational information (section 6.6/[RFC6350]) to be communicated. A Consumer may then automatically (i.e. via a policy) or manually select Captures based on information about who is in a Capture. It also allows a Consumer to render information regarding the people participating in the conference or to use it for further processing.

The Provider may supply a minimal set of information or a larger set of information. However it MUST be compliant to [RFC6350] and supply a "VERSION" and "FN" property. A Provider may supply multiple xCards per Capture of any KIND (section 6.1.4/[RFC6350]).

In order to keep CLUE messages compact the Provider SHOULD use a URI to point to any LOGO, PHOTO or SOUND contained in the xCARD rather than transmitting the LOGO, PHOTO or SOUND data in a CLUE message.

7.1.1.11. Person Type

The Person Type attribute indicates the type of people contained in the capture with respect to the meeting agenda (regardless of whether or not the capture has a Presentation attribute). As a capture may include multiple people the attribute may contain multiple values. However values MUST NOT be repeated within the attribute.

An Advertiser associates the person type with an individual capture when it knows that a particular type is in the capture. If an Advertiser cannot link a particular type with some certainty to a capture then it is not included. A Consumer on reception of a capture with a person type attribute knows with some certainty that the capture contains that person type. The capture may contain other person types but the Advertiser has not been able to determine that this is the case.

The types of Captured people include:

- . Chair - the person responsible for running the meeting according to the agenda.
- . Vice-Chair - the person responsible for assisting the chair in running the meeting.
- . Minute Taker - the person responsible for recording the minutes of the meeting.
- . Attendee - the person has no particular responsibilities with respect to running the meeting.
- . Observer - an Attendee without the right to influence the discussion.
- . Presenter - the person is scheduled on the agenda to make a presentation in the meeting. Note: This is not related to any "active speaker" functionality.
- . Translator - the person is providing some form of translation or commentary in the meeting.
- . Timekeeper - the person is responsible for maintaining the meeting schedule.

Furthermore the person type attribute may contain one or more strings allowing the Provider to indicate custom meeting specific types.

7.1.1.12. Priority

The Priority attribute indicates a relative priority between different Media Captures. The Provider sets this priority, and the Consumer MAY use the priority to help decide which Captures it wishes to receive.

The "priority" attribute is an integer which indicates a relative priority between Captures. For example it is possible to assign a priority between two presentation Captures that would allow a remote Endpoint to determine which presentation is more important. Priority is assigned at the individual Capture level. It represents the Provider's view of the relative priority between Captures with a priority. The same priority number MAY be used across multiple Captures. It indicates they are equally important. If no priority is assigned no assumptions regarding relative importance of the Capture can be assumed.

7.1.1.13. Embedded Text

The Embedded Text attribute indicates that a Capture provides embedded textual information. For example the video Capture may contain speech to text information composed with the video image.

7.1.1.14. Related To

The Related To attribute indicates the Capture contains additional complementary information related to another Capture. The value indicates the identity of the other Capture to which this Capture is providing additional information.

For example, a conference can utilize translators or facilitators that provide an additional audio stream (i.e. a translation or description or commentary of the conference). Where multiple captures are available, it may be advantageous for a Consumer to select a complementary Capture instead of or in addition to a Capture it relates to.

7.2. Multiple Content Capture

The MCC indicates that one or more Single Media Captures are multiplexed (temporally and/or spatially) or mixed in one Media Capture. Only one Capture type (i.e. audio, video, etc.) is allowed in each MCC instance. The MCC may contain a reference to the Single Media Captures (which may have their own attributes) as well as attributes associated with the MCC itself. A MCC may also contain other MCCs. The MCC MAY reference Captures from within the Capture Scene that defines it or from other Capture Scenes. No ordering is implied by the order that Captures appear within a MCC. A MCC MAY contain no references to other Captures to indicate that the MCC contains content from multiple sources but no information regarding those sources is given. MCCs either contain the referenced Captures and no others, or have no referenced captures and therefore may contain any Capture.

One or more MCCs may also be specified in a CSV. This allows an Advertiser to indicate that several MCC captures are used to represent a capture scene. Table 14 provides an example of this case.

As outlined in section 7.1. each instance of the MCC has its own Capture identity i.e. MCC1. It allows all the individual captures contained in the MCC to be referenced by a single MCC identity.

The example below shows the use of a Multiple Content Capture:

Capture Scene #1	
VC1	{MC attributes}
VC2	{MC attributes}
VC3	{MC attributes}
MCC1(VC1,VC2,VC3)	{MC and MCC attributes}
CSV(MCC1)	

Table 1: Multiple Content Capture concept

This indicates that MCC1 is a single capture that contains the Captures VC1, VC2 and VC3 according to any MCC1 attributes.

7.2.1.1. MCC Attributes

Media Capture Attributes may be associated with the MCC instance and the Single Media Captures that the MCC references. A Provider should avoid providing conflicting attribute values between the MCC and Single Media Captures. Where there is conflict the attributes of the MCC override any that may be present in the individual Captures.

A Provider MAY include as much or as little of the original source Capture information as it requires.

There are MCC specific attributes that MUST only be used with Multiple Content Captures. These are described in the sections below. The attributes described in section 7.1.1. MAY also be used with MCCs.

The spatial related attributes of an MCC indicate its area of capture and point of capture within the scene, just like any other media capture. The spatial information does not imply anything about how other captures are composed within an MCC.

For example: A virtual scene could be constructed for the MCC capture with two Video Captures with a "MaxCaptures" attribute set to 2 and an "Area of Capture" attribute provided with an overall area. Each of the individual Captures could then also include an "Area of Capture" attribute with a sub-set of the overall area. The Consumer would then know how each capture is related to others within the scene, but not the relative position of the individual captures within the composed capture.

Capture Scene #1	
VC1	AreaofCapture=(0,0,0)(9,0,0) (0,0,9)(9,0,9)
VC2	AreaofCapture=(10,0,0)(19,0,0) (10,0,9)(19,0,9)
MCC1(VC1,VC2)	MaxCaptures=2 AreaofCapture=(0,0,0)(19,0,0) (0,0,9)(19,0,9)
CSV(MCC1)	

Table 2: Example of MCC and Single Media Capture attributes

The sub-sections below describe the MCC only attributes.

7.2.1.1. Maximum Number of Captures within a MCC

The Maximum Number of Captures MCC attribute indicates the maximum number of individual Captures that may appear in a Capture Encoding at a time. The actual number at any given time can be less than or equal to this maximum. It may be used to derive how the Single Media Captures within the MCC are composed / switched with regards to space and time.

A Provider can indicate that the number of Captures in a MCC Capture Encoding is equal "=" to the MaxCaptures value or that there may be any number of Captures up to and including "<=" the MaxCaptures value. This allows a Provider to distinguish between a MCC that purely represents a composition of sources versus a MCC that represents switched or switched and composed sources.

MaxCaptures may be set to one so that only content related to one of the sources are shown in the MCC Capture Encoding at a time or it may be set to any value up to the total number of Source Media Captures in the MCC.

The bullets below describe how the setting of MaxCapture versus the number of Captures in the MCC affects how sources appear in a Capture Encoding:

- . When MaxCaptures is set to <= 1 and the number of Captures in the MCC is greater than 1 (or not specified) in the MCC this is a switched case. Zero or 1 Captures may be switched into the Capture Encoding. Note: zero is allowed because of the "<=".
- . When MaxCaptures is set to = 1 and the number of Captures in the MCC is greater than 1 (or not specified) in the MCC this is a switched case. Only one Capture source is contained in a Capture Encoding at a time.
- . When MaxCaptures is set to <= N (with N > 1) and the number of Captures in the MCC is greater than N (or not specified) this is a switched and composed case. The Capture Encoding may contain purely switched sources (i.e. <=2 allows for 1 source on its own), or may contain composed and switched sources (i.e. a composition of 2 sources switched between the sources).
- . When MaxCaptures is set to = N (with N > 1) and the number of Captures in the MCC is greater than N (or not specified) this

is a switched and composed case. The Capture Encoding contains composed and switched sources (i.e. a composition of N sources switched between the sources). It is not possible to have a single source.

- . When MaxCaptures is set to \leq to the number of Captures in the MCC this is a switched and composed case. The Capture Encoding may contain media switched between any number (up to the MaxCaptures) of composed sources.
- . When MaxCaptures is set to $=$ to the number of Captures in the MCC this is a composed case. All the sources are composed into a single Capture Encoding.

If this attribute is not set then as default it is assumed that all source media capture content can appear concurrently in the Capture Encoding associated with the MCC.

For example: The use of MaxCaptures equal to 1 on a MCC with three Video Captures VC1, VC2 and VC3 would indicate that the Advertiser in the Capture Encoding would switch between VC1, VC2 or VC3 as there may be only a maximum of one Capture at a time.

7.2.1.2. Policy

The Policy MCC Attribute indicates the criteria that the Provider uses to determine when and/or where media content appears in the Capture Encoding related to the MCC.

The attribute is in the form of a token that indicates the policy and an index representing an instance of the policy. The same index value can be used for multiple MCCs.

The tokens are:

SoundLevel - This indicates that the content of the MCC is determined by a sound level detection algorithm. The loudest (active) speaker (or a previous speaker, depending on the index value) is contained in the MCC.

RoundRobin - This indicates that the content of the MCC is determined by a time based algorithm. For example: the Provider provides content from a particular source for a period of time and then provides content from another source and so on.

An index is used to represent an instance in the policy setting. An index of 0 represents the most current instance of the policy, i.e.

the active speaker, 1 represents the previous instance, i.e. the previous active speaker and so on.

The following example shows a case where the Provider provides two media streams, one showing the active speaker and a second stream showing the previous speaker.

Capture Scene #1	
VC1	
VC2	
MCC1(VC1,VC2)	Policy=SoundLevel:0 MaxCaptures=1
MCC2(VC1,VC2)	Policy=SoundLevel:1 MaxCaptures=1
CSV(MCC1,MCC2)	

Table 3: Example Policy MCC attribute usage

7.2.1.3. Synchronisation Identity

The Synchronisation Identity MCC attribute indicates how the individual Captures in multiple MCC Captures are synchronised. To indicate that the Capture Encodings associated with MCCs contain Captures from the same source at the same time a Provider should set the same Synchronisation Identity on each of the concerned MCCs. It is the Provider that determines what the source for the Captures is, so a Provider can choose how to group together Single Media Captures into a combined "source" for the purpose of switching them together to keep them synchronized according to the SynchronisationID attribute. For example when the Provider is in an MCU it may determine that each separate CLUE Endpoint is a remote source of media. The Synchronisation Identity may be used across media types, i.e. to synchronize audio and video related MCCs.

Without this attribute it is assumed that multiple MCCs may provide content from different sources at any particular point in time.

For example:

Capture Scene #1	
VC1	Description=Left
VC2	Description=Centre
VC3	Description=Right
AC1	Description=Room
CSV(VC1,VC2,VC3)	
CSV(AC1)	
Capture Scene #2	
VC4	Description=Left
VC5	Description=Centre
VC6	Description=Right
AC2	Description=Room
CSV(VC4,VC5,VC6)	
CSV(AC2)	
Capture Scene #3	
VC7	
AC3	
Capture Scene #4	
VC8	
AC4	
Capture Scene #5	
MCC1(VC1,VC4,VC7)	SynchronisationID=1
	MaxCaptures=1
MCC2(VC2,VC5,VC8)	SynchronisationID=1
	MaxCaptures=1
MCC3(VC3,VC6)	MaxCaptures=1
MCC4(AC1,AC2,AC3,AC4)	SynchronisationID=1
	MaxCaptures=1
CSV(MCC1,MCC2,MCC3)	
CSV(MCC4)	

Table 4: Example Synchronisation Identity MCC attribute usage

The above Advertisement would indicate that MCC1, MCC2, MCC3 and MCC4 make up a Capture Scene. There would be four Capture Encodings (one for each MCC). Because MCC1 and MCC2 have the same SynchronisationID, each Encoding from MCC1 and MCC2 respectively would together have content from only Capture Scene 1 or only Capture Scene 2 or the combination of VC7 and VC8 at a particular point in time. In this case the Provider has decided the sources to be synchronized are Scene #1, Scene #2, and Scene #3 and #4 together. The Encoding from MCC3 would not be synchronised with MCC1 or MCC2. As MCC4 also has the same Synchronisation Identity as MCC1 and MCC2 the content of the audio Encoding will be synchronised with the video content.

7.2.1.4. Allow Subset Choice

The Allow Subset Choice MCC attribute is a boolean value, indicating whether or not the Provider allows the Consumer to choose a specific subset of the Captures referenced by the MCC. If this attribute is true, and the MCC references other Captures, then the Consumer MAY select (in a Configuremessage) a specific subset of those Captures to be included in the MCC, and the Provider MUST then include only that subset. If this attribute is false, or the MCC does not reference other Captures, then the Consumer MUST NOT select a subset.

7.3. Capture Scene

In order for a Provider's individual Captures to be used effectively by a Consumer, the Provider organizes the Captures into one or more Capture Scenes, with the structure and contents of these Capture Scenes being sent from the Provider to the Consumer in the Advertisement.

A Capture Scene is a structure representing a spatial region containing one or more Capture Devices, each capturing media representing a portion of the region. A Capture Scene includes one or more Capture Scene Views (CSV), with each CSV including one or more Media Captures of the same media type. There can also be Media Captures that are not included in a Capture Scene View. A Capture Scene represents, for example, the video image of a group of people seated next to each other, along with the sound of their voices, which could be represented by some number of VCs and ACs in the Capture Scene Views. An MCU can also describe in Capture Scenes what it constructs from media Streams it receives.

A Provider MAY advertise one or more Capture Scenes. What constitutes an entire Capture Scene is up to the Provider. A simple Provider might typically use one Capture Scene for participant media (live video from the room cameras) and another Capture Scene for a computer generated presentation. In more complex systems, the use of additional Capture Scenes is also sensible. For example, a classroom may advertise two Capture Scenes involving live video, one including only the camera capturing the instructor (and associated audio), the other including camera(s) capturing students (and associated audio).

A Capture Scene MAY (and typically will) include more than one type of media. For example, a Capture Scene can include several Capture Scene Views for Video Captures, and several Capture Scene Views for Audio Captures. A particular Capture MAY be included in more than one Capture Scene View.

A Provider MAY express spatial relationships between Captures that are included in the same Capture Scene. However, there is no spatial relationship between Media Captures from different Capture Scenes. In other words, Capture Scenes each use their own spatial measurement system as outlined above in section 6.

A Provider arranges Captures in a Capture Scene to help the Consumer choose which captures it wants to render. The Capture Scene Views in a Capture Scene are different alternatives the Provider is suggesting for representing the Capture Scene. Each Capture Scene View is given an advertisement unique identity. The order of Capture Scene Views within a Capture Scene has no significance. The Media Consumer can choose to receive all Media Captures from one Capture Scene View for each media type (e.g. audio and video), or it can pick and choose Media Captures regardless of how the Provider arranges them in Capture Scene Views. Different Capture Scene Views of the same media type are not necessarily mutually exclusive alternatives. Also note that the presence of multiple Capture Scene Views (with potentially multiple encoding options in each view) in a given Capture Scene does not necessarily imply that a Provider is able to serve all the associated media simultaneously (although the construction of such an over-rich Capture Scene is probably not sensible in many cases). What a Provider can send simultaneously is determined through the Simultaneous Transmission Set mechanism, described in section 8.

Captures within the same Capture Scene View MUST be of the same media type - it is not possible to mix audio and video captures in

the same Capture Scene View, for instance. The Provider MUST be capable of encoding and sending all Captures (that have an encoding group) in a single Capture Scene View simultaneously. The order of Captures within a Capture Scene View has no significance. A Consumer can decide to receive all the Captures in a single Capture Scene View, but a Consumer could also decide to receive just a subset of those captures. A Consumer can also decide to receive Captures from different Capture Scene Views, all subject to the constraints set by Simultaneous Transmission Sets, as discussed in section 8.

When a Provider advertises a Capture Scene with multiple CSVs, it is essentially signaling that there are multiple representations of the same Capture Scene available. In some cases, these multiple views would be used simultaneously (for instance a "video view" and an "audio view"). In some cases the views would conceptually be alternatives (for instance a view consisting of three Video Captures covering the whole room versus a view consisting of just a single Video Capture covering only the center of a room). In this latter example, one sensible choice for a Consumer would be to indicate (through its Configure and possibly through an additional offer/answer exchange) the Captures of that Capture Scene View that most closely matched the Consumer's number of display devices or screen layout.

The following is an example of 4 potential Capture Scene Views for an endpoint-style Provider:

1. (VC0, VC1, VC2) - left, center and right camera Video Captures
2. (MCC3) - Video Capture associated with loudest room segment
3. (VC4) - Video Capture zoomed out view of all people in the room
4. (AC0) - main audio

The first view in this Capture Scene example is a list of Video Captures which have a spatial relationship to each other. Determination of the order of these captures (VC0, VC1 and VC2) for rendering purposes is accomplished through use of their Area of Capture attributes. The second view (MCC3) and the third view (VC4) are alternative representations of the same room's video, which might be better suited to some Consumers' rendering capabilities. The inclusion of the Audio Capture in the same Capture Scene indicates that AC0 is associated with all of those

Video Captures, meaning it comes from the same spatial region. Therefore, if audio were to be rendered at all, this audio would be the correct choice irrespective of which Video Captures were chosen.

7.3.1. Capture Scene attributes

Capture Scene Attributes can be applied to Capture Scenes as well as to individual media captures. Attributes specified at this level apply to all constituent Captures. Capture Scene attributes include

- . Human-readable description of the Capture Scene, which could be in multiple languages;
- . xCard scene information
- . Scale information (millimeters, unknown, no scale), as described in Section 6.

7.3.1.1. Scene Information

The Scene information attribute provides information regarding the Capture Scene rather than individual participants. The Provider may gather the information automatically or manually from a variety of sources. The scene information attribute allows a Provider to indicate information such as: organizational or geographic information allowing a Consumer to determine which Capture Scenes are of interest in order to then perform Capture selection. It also allows a Consumer to render information regarding the Scene or to use it for further processing.

As per 7.1.1.10. the xCard format is used to convey this information and the Provider may supply a minimal set of information or a larger set of information.

In order to keep CLUE messages compact the Provider SHOULD use a URI to point to any LOGO, PHOTO or SOUND contained in the xCARD rather than transmitting the LOGO, PHOTO or SOUND data in a CLUE message.

7.3.2. Capture Scene View attributes

A Capture Scene can include one or more Capture Scene Views in addition to the Capture Scene wide attributes described above. Capture Scene View attributes apply to the Capture Scene View as a

whole, i.e. to all Captures that are part of the Capture Scene View.

Capture Scene View attributes include:

- . Human-readable description (which could be in multiple languages) of the Capture Scene View

7.4. Global View List

An Advertisement can include an optional Global View list. Each item in this list is a Global View. The Provider can include multiple Global Views, to allow a Consumer to choose sets of captures appropriate to its capabilities or application. The choice of how to make these suggestions in the Global View list for what represents all the scenes for which the Provider can send media is up to the Provider. This is very similar to how each CSV represents a particular scene.

As an example, suppose an advertisement has three scenes, and each scene has three CSVs, ranging from one to three video captures in each CSV. The Provider is advertising a total of nine video Captures across three scenes. The Provider can use the Global View list to suggest alternatives for Consumers that can't receive all nine video Captures as separate media streams. For accommodating a Consumer that wants to receive three video Captures, a Provider might suggest a Global View containing just a single CSV with three Captures and nothing from the other two scenes. Or a Provider might suggest a Global View containing three different CSVs, one from each scene, with a single video Capture in each.

Some additional rules:

- . The ordering of Global Views in the Global View list is insignificant.
- . The ordering of CSVs within each Global View is insignificant.
- . A particular CSV may be used in multiple Global Views.
- . The Provider must be capable of encoding and sending all Captures within the CSVs of a given Global View simultaneously.

The following figure shows an example of the structure of Global Views in a Global View List.

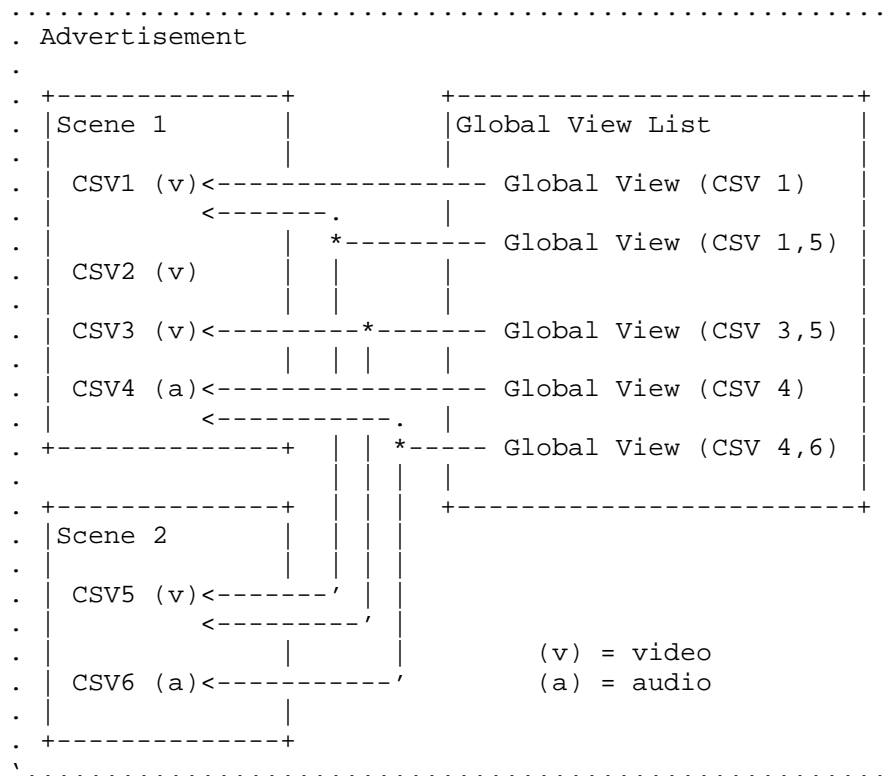


Figure 3: Global View List Structure

8. Simultaneous Transmission Set Constraints

In many practical cases, a Provider has constraints or limitations on its ability to send Captures simultaneously. One type of limitation is caused by the physical limitations of capture mechanisms; these constraints are represented by a Simultaneous Transmission Set. The second type of limitation reflects the encoding resources available, such as bandwidth or video encoding throughput (macroblocks/second). This type of constraint is captured by Individual Encodings and Encoding Groups, discussed below.

Some Endpoints or MCUs can send multiple Captures simultaneously; however sometimes there are constraints that limit which Captures can be sent simultaneously with other Captures. A device may not

be able to be used in different ways at the same time. Provider Advertisements are made so that the Consumer can choose one of several possible mutually exclusive usages of the device. This type of constraint is expressed in a Simultaneous Transmission Set, which lists all the Captures of a particular media type (e.g. audio, video, text) that can be sent at the same time. There are different Simultaneous Transmission Sets for each media type in the Advertisement. This is easier to show in an example.

Consider the example of a room system where there are three cameras each of which can send a separate Capture covering two persons each- VC0, VC1, VC2. The middle camera can also zoom out (using an optical zoom lens) and show all six persons, VC3. But the middle camera cannot be used in both modes at the same time - it has to either show the space where two participants sit or the whole six seats, but not both at the same time. As a result, VC1 and VC3 cannot be sent simultaneously.

Simultaneous Transmission Sets are expressed as sets of the Media Captures that the Provider could transmit at the same time (though, in some cases, it is not intuitive to do so). If a Multiple Content Capture is included in a Simultaneous Transmission Set it indicates that the Capture Encoding associated with it could be transmitted as the same time as the other Captures within the Simultaneous Transmission Set. It does not imply that the Single Media Captures contained in the Multiple Content Capture could all be transmitted at the same time.

In this example the two Simultaneous Transmission Sets are shown in Table 5. If a Provider advertises one or more mutually exclusive Simultaneous Transmission Sets, then for each media type the Consumer MUST ensure that it chooses Media Captures that lie wholly within one of those Simultaneous Transmission Sets.

+-----+	
	Simultaneous Sets
+-----+	
	{VC0, VC1, VC2}
	{VC0, VC3, VC2}
+-----+	

Table 5: Two Simultaneous Transmission Sets

A Provider OPTIONALLY can include the Simultaneous Transmission Sets in its Advertisement. These constraints apply across all the

Capture Scenes in the Advertisement. It is a syntax conformance requirement that the Simultaneous Transmission Sets MUST allow all the media Captures in any particular Capture Scene View to be used simultaneously. Similarly, the Simultaneous Transmission Sets MUST reflect the simultaneity expressed by any Global View.

For shorthand convenience, a Provider MAY describe a Simultaneous Transmission Set in terms of Capture Scene Views and Capture Scenes. If a Capture Scene View is included in a Simultaneous Transmission Set, then all Media Captures in the Capture Scene View are included in the Simultaneous Transmission Set. If a Capture Scene is included in a Simultaneous Transmission Set, then all its Capture Scene Views (of the corresponding media type) are included in the Simultaneous Transmission Set. The end result reduces to a set of Media Captures, of a particular media type, in either case.

If an Advertisement does not include Simultaneous Transmission Sets, then the Provider MUST be able to simultaneously provide all the Captures from any one CSV of each media type from each Capture Scene. Likewise, if there are no Simultaneous Transmission Sets and there is a Global View list, then the Provider MUST be able to simultaneously provide all the Captures from any particular Global View (of each media type) from the Global View list.

If an Advertisement includes multiple Capture Scene Views in a Capture Scene then the Consumer MAY choose one Capture Scene View for each media type, or MAY choose individual Captures based on the Simultaneous Transmission Sets.

9. Encodings

Individual encodings and encoding groups are CLUE's mechanisms allowing a Provider to signal its limitations for sending Captures, or combinations of Captures, to a Consumer. Consumers can map the Captures they want to receive onto the Encodings, with the encoding parameters they want. As for the relationship between the CLUE-specified mechanisms based on Encodings and the SIP offer/answer exchange, please refer to section 5.

9.1. Individual Encodings

An Individual Encoding represents a way to encode a Media Capture as a Capture Encoding, to be sent as an encoded media stream from the Provider to the Consumer. An Individual Encoding has a set of parameters characterizing how the media is encoded.

Different media types have different parameters, and different encoding algorithms may have different parameters. An Individual Encoding can be assigned to at most one Capture Encoding at any given time.

Individual Encoding parameters are represented in SDP [RFC4566], not in CLUE messages. For example, for a video encoding using H.26x compression technologies, this can include parameters such as:

- . Maximum bandwidth;
- . Maximum picture size in pixels;
- . Maximum number of pixels to be processed per second;

The bandwidth parameter is the only one that specifically relates to a CLUE Advertisement, as it can be further constrained by the maximum group bandwidth in an Encoding Group.

9.2. Encoding Group

An Encoding Group includes a set of one or more Individual Encodings, and parameters that apply to the group as a whole. By grouping multiple individual Encodings together, an Encoding Group describes additional constraints on bandwidth for the group. A single Encoding Group MAY refer to Encodings for different media types.

The Encoding Group data structure contains:

- . Maximum bitrate for all encodings in the group combined;
- . A list of identifiers for the Individual Encodings belonging to the group.

When the Individual Encodings in a group are instantiated into Capture Encodings, each Capture Encoding has a bitrate that MUST be less than or equal to the max bitrate for the particular Individual Encoding. The "maximum bitrate for all encodings in the group" parameter gives the additional restriction that the sum of all the individual Capture Encoding bitrates MUST be less than or equal to this group value.

The following diagram illustrates one example of the structure of a media Provider's Encoding Groups and their contents.

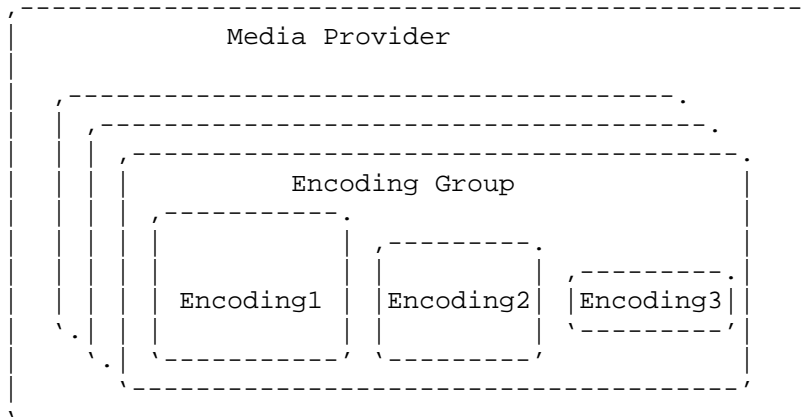


Figure 4: Encoding Group Structure

A Provider advertises one or more Encoding Groups. Each Encoding Group includes one or more Individual Encodings. Each Individual Encoding can represent a different way of encoding media. For example one Individual Encoding may be 1080p60 video, another could be 720p30, with a third being CIF, all in, for example, H.264 format.

While a typical three codec/display system might have one Encoding Group per "codec box" (physical codec, connected to one camera and one screen), there are many possibilities for the number of Encoding Groups a Provider may be able to offer and for the encoding values in each Encoding Group.

There is no requirement for all Encodings within an Encoding Group to be instantiated at the same time.

9.3. Associating Captures with Encoding Groups

Each Media Capture, including MCCs, MAY be associated with one Encoding Group. To be eligible for configuration, a Media Capture MUST be associated with one Encoding Group, which is used to instantiate that Capture into a Capture Encoding. When an MCC is configured all the Media Captures referenced by the MCC will appear in the Capture Encoding according to the attributes of the chosen encoding of the MCC. This allows an Advertiser to specify encoding attributes associated with the Media Captures without the need to provide an individual Capture Encoding for each of the inputs.

If an Encoding Group is assigned to a Media Capture referenced by the MCC it indicates that this Capture may also have an individual Capture Encoding.

For example:

Capture Scene #1	
VC1	EncodeGroupID=1
VC2	
MCC1(VC1,VC2)	EncodeGroupID=2
CSV(VC1)	
CSV(MCC1)	

Table 6: Example usage of Encoding with MCC and source Captures

This would indicate that VC1 may be sent as its own Capture Encoding from EncodeGroupID=1 or that it may be sent as part of a Capture Encoding from EncodeGroupID=2 along with VC2.

More than one Capture MAY use the same Encoding Group.

The maximum number of Capture Encodings that can result from a particular Encoding Group constraint is equal to the number of individual Encodings in the group. The actual number of Capture Encodings used at any time MAY be less than this maximum. Any of the Captures that use a particular Encoding Group can be encoded according to any of the Individual Encodings in the group.

It is a protocol conformance requirement that the Encoding Groups MUST allow all the Captures in a particular Capture Scene View to be used simultaneously.

10. Consumer's Choice of Streams to Receive from the Provider

After receiving the Provider's Advertisement message (that includes media captures and associated constraints), the Consumer composes its reply to the Provider in the form of a Configure message. The Consumer is free to use the information in the Advertisement as it chooses, but there are a few obviously sensible design choices, which are outlined below.

If multiple Providers connect to the same Consumer (i.e. in an MCU-less multiparty call), it is the responsibility of the Consumer to compose Configures for each Provider that both fulfill each Provider's constraints as expressed in the Advertisement, as well as its own capabilities.

In an MCU-based multiparty call, the MCU can logically terminate the Advertisement/Configure negotiation in that it can hide the characteristics of the receiving endpoint and rely on its own capabilities (transcoding/transrating/...) to create Media Streams that can be decoded at the Endpoint Consumers. The timing of an MCU's sending of Advertisements (for its outgoing ports) and Configures (for its incoming ports, in response to Advertisements received there) is up to the MCU and implementation dependent.

As a general outline, a Consumer can choose, based on the Advertisement it has received, which Captures it wishes to receive, and which Individual Encodings it wants the Provider to use to encode the Captures.

On receipt of an Advertisement with an MCC the Consumer treats the MCC as per other non-MCC Captures with the following differences:

- The Consumer would understand that the MCC is a Capture that includes the referenced individual Captures (or any Captures, if none are referenced) and that these individual Captures are delivered as part of the MCC's Capture Encoding.
- The Consumer may utilise any of the attributes associated with the referenced individual Captures and any Capture Scene attributes from where the individual Captures were defined to choose Captures and for rendering decisions.
- If the MCC attribute Allow Subset Choice is true, then the Consumer may or may not choose to receive all the indicated Captures. It can choose to receive a sub-set of Captures indicated by the MCC.

For example if the Consumer receives:

```
MCC1(VC1,VC2,VC3){attributes}
```

A Consumer could choose all the Captures within a MCC however if the Consumer determines that it doesn't want VC3 it can return MCC1(VC1,VC2). If it wants all the individual Captures then it

returns only the MCC identity (i.e. MCC1). If the MCC in the advertisement does not reference any individual captures, or the Allow Subset Choice attribute is false, then the Consumer cannot choose what is included in the MCC, it is up to the Provider to decide.

A Configure Message includes a list of Capture Encodings. These are the Capture Encodings the Consumer wishes to receive from the Provider. Each Capture Encoding refers to one Media Capture and one Individual Encoding.

For each Capture the Consumer wants to receive, it configures one of the Encodings in that Capture's Encoding Group. The Consumer does this by telling the Provider, in its Configure Message, which Encoding to use for each chosen Capture. Upon receipt of this Configure from the Consumer, common knowledge is established between Provider and Consumer regarding sensible choices for the media streams. The setup of the actual media channels, at least in the simplest case, is left to a following offer/answer exchange. Optimized implementations may speed up the reaction to the offer/answer exchange by reserving the resources at the time of finalization of the CLUE handshake.

CLUE advertisements and configure messages don't necessarily require a new SDP offer/answer for every CLUE message exchange. But the resulting encodings sent via RTP must conform to the most recent SDP offer/answer result.

In order to meaningfully create and send an initial Configure, the Consumer needs to have received at least one Advertisement, and an SDP offer defining the Individual Encodings, from the Provider.

In addition, the Consumer can send a Configure at any time during the call. The Configure MUST be valid according to the most recently received Advertisement. The Consumer can send a Configure either in response to a new Advertisement from the Provider or on its own, for example because of a local change in conditions (people leaving the room, connectivity changes, multipoint related considerations).

When choosing which Media Streams to receive from the Provider, and the encoding characteristics of those Media Streams, the Consumer advantageously takes several things into account: its local preference, simultaneity restrictions, and encoding limits.

10.1. Local preference

A variety of local factors influence the Consumer's choice of Media Streams to be received from the Provider:

- o if the Consumer is an Endpoint, it is likely that it would choose, where possible, to receive video and audio Captures that match the number of display devices and audio system it has
- o if the Consumer is an MCU, it may choose to receive loudest speaker streams (in order to perform its own media composition) and avoid pre-composed video Captures
- o user choice (for instance, selection of a new layout) may result in a different set of Captures, or different encoding characteristics, being required by the Consumer

10.2. Physical simultaneity restrictions

Often there are physical simultaneity constraints of the Provider that affect the Provider's ability to simultaneously send all of the captures the Consumer would wish to receive. For instance, an MCU, when connected to a multi-camera room system, might prefer to receive both individual video streams of the people present in the room and an overall view of the room from a single camera. Some Endpoint systems might be able to provide both of these sets of streams simultaneously, whereas others might not (if the overall room view were produced by changing the optical zoom level on the center camera, for instance).

10.3. Encoding and encoding group limits

Each of the Provider's encoding groups has limits on bandwidth, and the constituent potential encodings have limits on the bandwidth, computational complexity, video frame rate, and resolution that can be provided. When choosing the Captures to be received from a Provider, a Consumer device MUST ensure that the encoding characteristics requested for each individual Capture fits within the capability of the encoding it is being configured to use, as well as ensuring that the combined encoding characteristics for Captures fit within the capabilities of their associated encoding groups. In some cases, this could cause an otherwise "preferred" choice of capture encodings to be passed over in favor of different Capture Encodings--for instance, if a set of three Captures could only be provided at a low resolution

then a three screen device could switch to favoring a single, higher quality, Capture Encoding.

11. Extensibility

One important characteristics of the Framework is its extensibility. The standard for interoperability and handling multiple streams must be future-proof. The framework itself is inherently extensible through expanding the data model types. For example:

- o Adding more types of media, such as telemetry, can done by defining additional types of Captures in addition to audio and video.
- o Adding new functionalities, such as 3-D video Captures, say, may require additional attributes describing the Captures.

The infrastructure is designed to be extended rather than requiring new infrastructure elements. Extension comes through adding to defined types.

12. Examples - Using the Framework (Informative)

This section gives some examples, first from the point of view of the Provider, then the Consumer, then some multipoint scenarios

12.1. Provider Behavior

This section shows some examples in more detail of how a Provider can use the framework to represent a typical case for telepresence rooms. First an endpoint is illustrated, then an MCU case is shown.

12.1.1. Three screen Endpoint Provider

Consider an Endpoint with the following description:

3 cameras, 3 displays, a 6 person table

- o Each camera can provide one Capture for each 1/3 section of the table

- o A single Capture representing the active speaker can be provided (voice activity based camera selection to a given encoder input port implemented locally in the Endpoint)
- o A single Capture representing the active speaker with the other 2 Captures shown picture in picture (PiP) within the stream can be provided (again, implemented inside the endpoint)
- o A Capture showing a zoomed out view of all 6 seats in the room can be provided

The video and audio Captures for this Endpoint can be described as follows.

Video Captures:

- o VC0- (the left camera stream), encoding group=EG0, view=table
- o VC1- (the center camera stream), encoding group=EG1, view=table
- o VC2- (the right camera stream), encoding group=EG2, view=table
- o MCC3- (the loudest panel stream), encoding group=EG1, view=table, MaxCaptures=1, policy=SoundLevel
- o MCC4- (the loudest panel stream with PiPs), encoding group=EG1, view=room, MaxCaptures=3, policy=SoundLevel
- o VC5- (the zoomed out view of all people in the room), encoding group=EG1, view=room
- o VC6- (presentation stream), encoding group=EG1, presentation

The following diagram is a top view of the room with 3 cameras, 3 displays, and 6 seats. Each camera captures 2 people. The six seats are not all in a straight line.

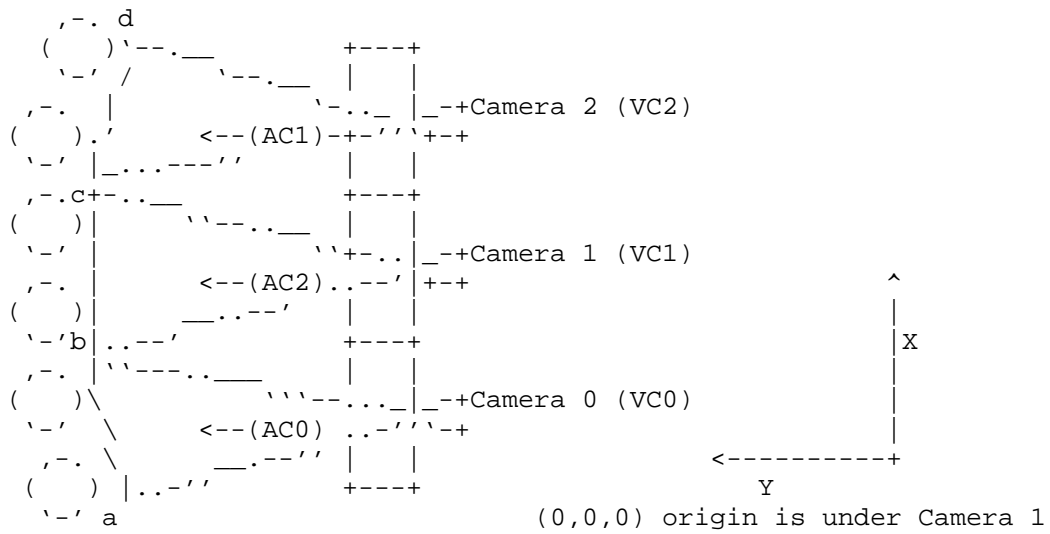


Figure 5: Room Layout Top View

The two points labeled b and c are intended to be at the midpoint between the seating positions, and where the fields of view of the cameras intersect.

The plane of interest for VC0 is a vertical plane that intersects points 'a' and 'b'.

The plane of interest for VC1 intersects points 'b' and 'c'. The plane of interest for VC2 intersects points 'c' and 'd'.

This example uses an area scale of millimeters.

Areas of capture:

	bottom left	bottom right	top left	top right
VC0	(-2011,2850,0)	(-673,3000,0)	(-2011,2850,757)	(-673,3000,757)
VC1	(-673,3000,0)	(673,3000,0)	(-673,3000,757)	(673,3000,757)
VC2	(673,3000,0)	(2011,2850,0)	(673,3000,757)	(2011,3000,757)
MCC3	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
MCC4	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
VC5	(-2011,2850,0)	(2011,2850,0)	(-2011,2850,757)	(2011,3000,757)
VC6	none			

Points of capture:

VC0 (-1678,0,800)
VC1 (0,0,800)
VC2 (1678,0,800)
MCC3 none
MCC4 none
VC5 (0,0,800)
VC6 none

In this example, the right edge of the VC0 area lines up with the left edge of the VC1 area. It doesn't have to be this way. There could be a gap or an overlap. One additional thing to note for this example is the distance from a to b is equal to the distance from b to c and the distance from c to d. All these distances are 1346 mm. This is the planar width of each area of capture for VC0, VC1, and VC2.

Note the text in parentheses (e.g. "the left camera stream") is not explicitly part of the model, it is just explanatory text for this example, and is not included in the model with the media

captures and attributes. Also, MCC4 doesn't say anything about how a capture is composed, so the media consumer can't tell based on this capture that MCC4 is composed of a "loudest panel with PiPs".

Audio Captures:

Three ceiling microphones are located between the cameras and the table, at the same height as the cameras. The microphones point down at an angle toward the seating positions.

- o AC0 (left), encoding group=EG3
- o AC1 (right), encoding group=EG3
- o AC2 (center) encoding group=EG3
- o AC3 being a simple pre-mixed audio stream from the room (mono), encoding group=EG3
- o AC4 audio stream associated with the presentation video (mono) encoding group=EG3, presentation

Point of capture:	Point on Line of Capture:
AC0 (-1342,2000,800)	(-1342,2925,379)
AC1 (1342,2000,800)	(1342,2925,379)
AC2 (0,2000,800)	(0,3000,379)
AC3 (0,2000,800)	(0,3000,379)
AC4 none	

The physical simultaneity information is:

Simultaneous transmission set #1 {VC0, VC1, VC2, MCC3, MCC4, VC6}

Simultaneous transmission set #2 {VC0, VC2, VC5, VC6}

This constraint indicates it is not possible to use all the VCs at the same time. VC5 cannot be used at the same time as VC1 or MCC3 or MCC4. Also, using every member in the set simultaneously may not make sense - for example MCC3(loudest) and MCC4 (loudest with PiP). In addition, there are encoding constraints that make choosing all of the VCs in a set impossible. VC1, MCC3, MCC4, VC5, VC6 all use EG1 and EG1 has only 3 ENCs. This constraint

shows up in the encoding groups, not in the simultaneous transmission sets.

In this example there are no restrictions on which Audio Captures can be sent simultaneously.

Encoding Groups:

This example has three encoding groups associated with the video captures. Each group can have 3 encodings, but with each potential encoding having a progressively lower specification. In this example, 1080p60 transmission is possible (as ENC0 has a maxPps value compatible with that). Significantly, as up to 3 encodings are available per group, it is possible to transmit some video Captures simultaneously that are not in the same view in the Capture Scene. For example VC1 and MCC3 at the same time. The information below about Encodings is a summary of what would be conveyed in SDP, not directly in the CLUE Advertisement.

```
encodeGroupID=EG0, maxGroupBandwidth=6000000
  encodeID=ENC0, maxWidht=1920, maxHeight=1088, maxFrameRate=60,
    maxPps=124416000, maxBandwidth=4000000
  encodeID=ENC1, maxWidht=1280, maxHeight=720, maxFrameRate=30,
    maxPps=27648000, maxBandwidth=4000000
  encodeID=ENC2, maxWidht=960, maxHeight=544, maxFrameRate=30,
    maxPps=15552000, maxBandwidth=4000000
encodeGroupID=EG1, maxGroupBandwidth=6000000
  encodeID=ENC3, maxWidht=1920, maxHeight=1088, maxFrameRate=60,
    maxPps=124416000, maxBandwidth=4000000
  encodeID=ENC4, maxWidht=1280, maxHeight=720, maxFrameRate=30,
    maxPps=27648000, maxBandwidth=4000000
  encodeID=ENC5, maxWidht=960, maxHeight=544, maxFrameRate=30,
    maxPps=15552000, maxBandwidth=4000000
encodeGroupID=EG2, maxGroupBandwidth=6000000
  encodeID=ENC6, maxWidht=1920, maxHeight=1088, maxFrameRate=60,
    maxPps=124416000, maxBandwidth=4000000
  encodeID=ENC7, maxWidht=1280, maxHeight=720, maxFrameRate=30,
    maxPps=27648000, maxBandwidth=4000000
  encodeID=ENC8, maxWidht=960, maxHeight=544, maxFrameRate=30,
    maxPps=15552000, maxBandwidth=4000000
```

Figure 6: Example Encoding Groups for Video

For audio, there are five potential encodings available, so all five Audio Captures can be encoded at the same time.

```

encodeGroupID=EG3, maxGroupBandwidth=320000
  encodeID=ENC9, maxBandwidth=64000
  encodeID=ENC10, maxBandwidth=64000
  encodeID=ENC11, maxBandwidth=64000
  encodeID=ENC12, maxBandwidth=64000
  encodeID=ENC13, maxBandwidth=64000

```

Figure 7: Example Encoding Group for Audio

Capture Scenes:

The following table represents the Capture Scenes for this Provider. Recall that a Capture Scene is composed of alternative Capture Scene Views covering the same spatial region. Capture Scene #1 is for the main people captures, and Capture Scene #2 is for presentation.

Each row in the table is a separate Capture Scene View

+	-----	+
	Capture Scene #1	
+	-----	+
	VC0, VC1, VC2	
	MCC3	
	MCC4	
	VC5	
	AC0, AC1, AC2	
	AC3	
+	-----	+
+	-----	+
	Capture Scene #2	
+	-----	+
	VC6	
	AC4	
+	-----	+

Table 7: Example Capture Scene Views

Different Capture Scenes are distinct from each other, and are non-overlapping. A Consumer can choose a view from each Capture Scene. In this case the three Captures VC0, VC1, and VC2 are one way of representing the video from the Endpoint. These three Captures should appear adjacent next to each other. Alternatively, another way of representing the Capture Scene is

with the capture MCC3, which automatically shows the person who is talking. Similarly for the MCC4 and VC5 alternatives.

As in the video case, the different views of audio in Capture Scene #1 represent the "same thing", in that one way to receive the audio is with the 3 Audio Captures (AC0, AC1, AC2), and another way is with the mixed AC3. The Media Consumer can choose an audio CSV it is capable of receiving.

The spatial ordering is understood by the Media Capture attributes Area of Capture, Point of Capture and Point on Line of Capture.

A Media Consumer would likely want to choose a Capture Scene View to receive based in part on how many streams it can simultaneously receive. A consumer that can receive three video streams would probably prefer to receive the first view of Capture Scene #1 (VC0, VC1, VC2) and not receive the other views. A consumer that can receive only one video stream would probably choose one of the other views.

If the consumer can receive a presentation stream too, it would also choose to receive the only view from Capture Scene #2 (VC6).

12.1.1.2. Encoding Group Example

This is an example of an Encoding Group to illustrate how it can express dependencies between Encodings. The information below about Encodings is a summary of what would be conveyed in SDP, not directly in the CLUE Advertisement.

```
encodeGroupID=EG0 maxGroupBandwidth=6000000
  encodeID=VIDENC0, maxWidth=1920, maxHeight=1088,
    maxFrameRate=60, maxPps=62208000, maxBandwidth=4000000
  encodeID=VIDENC1, maxWidth=1920, maxHeight=1088,
    maxFrameRate=60, maxPps=62208000, maxBandwidth=4000000
  encodeID=AUDENC0, maxBandwidth=96000
  encodeID=AUDENC1, maxBandwidth=96000
  encodeID=AUDENC2, maxBandwidth=96000
```

Here, the Encoding Group is EG0. Although the Encoding Group is capable of transmitting up to 6Mbit/s, no individual video Encoding can exceed 4Mbit/s.

This encoding group also allows up to 3 audio encodings, AUDENC<0-2>. It is not required that audio and video encodings reside

within the same encoding group, but if so then the group's overall maxBandwidth value is a limit on the sum of all audio and video encodings configured by the consumer. A system that does not wish or need to combine bandwidth limitations in this way should instead use separate encoding groups for audio and video in order for the bandwidth limitations on audio and video to not interact.

Audio and video can be expressed in separate encoding groups, as in this illustration.

```

encodeGroupID=EG0 maxGroupBandwidth=6000000
  encodeID=VIDENC0, maxWidth=1920, maxHeight=1088,
    maxFrameRate=60, maxPps=62208000, maxBandwidth=4000000
  encodeID=VIDENC1, maxWidth=1920, maxHeight=1088,
    maxFrameRate=60, maxPps=62208000, maxBandwidth=4000000
encodeGroupID=EG1 maxGroupBandwidth=500000
  encodeID=AUDENC0, maxBandwidth=96000
  encodeID=AUDENC1, maxBandwidth=96000
  encodeID=AUDENC2, maxBandwidth=96000

```

12.1.1.3. The MCU Case

This section shows how an MCU might express its Capture Scenes, intending to offer different choices for consumers that can handle different numbers of streams. Each MCC is for video. A single Audio Capture is provided for all single and multi-screen configurations that can be associated (e.g. lip-synced) with any combination of Video Captures (the MCCs) at the consumer.

Capture Scene #1	
MCC	for a single screen consumer
MCC1, MCC2	for a two screen consumer
MCC3, MCC4, MCC5	for a three screen consumer
MCC6, MCC7, MCC8, MCC9	for a four screen consumer
AC0	AC representing all participants
CSV(MCC0)	
CSV(MCC1,MCC2)	
CSV(MCC3,MCC4,MCC5)	
CSV(MCC6,MCC7, MCC8,MCC9)	
CSV(AC0)	

Table 8: MCU main Capture Scenes

If / when a presentation stream becomes active within the conference the MCU might re-advertise the available media as:

Capture Scene #2	note
VC10	video capture for presentation
AC1	presentation audio to accompany VC10
CSV(VC10)	
CSV(AC1)	

Table 9: MCU presentation Capture Scene

12.2. Media Consumer Behavior

This section gives an example of how a Media Consumer might behave when deciding how to request streams from the three screen endpoint described in the previous section.

The receive side of a call needs to balance its requirements, based on number of screens and speakers, its decoding capabilities and available bandwidth, and the provider's capabilities in order to optimally configure the provider's streams. Typically it would want to receive and decode media from each Capture Scene advertised by the Provider.

A sane, basic, algorithm might be for the consumer to go through each Capture Scene View in turn and find the collection of Video Captures that best matches the number of screens it has (this might include consideration of screens dedicated to presentation video display rather than "people" video) and then decide between alternative views in the video Capture Scenes based either on hard-coded preferences or user choice. Once this choice has been made, the consumer would then decide how to configure the provider's encoding groups in order to make best use of the available network bandwidth and its own decoding capabilities.

12.2.1. One screen Media Consumer

MCC3, MCC4 and VC5 are all different views by themselves, not grouped together in a single view, so the receiving device should choose between one of those. The choice would come down to

whether to see the greatest number of participants simultaneously at roughly equal precedence (VC5), a switched view of just the loudest region (MCC3) or a switched view with PiPs (MCC4). An endpoint device with a small amount of knowledge of these differences could offer a dynamic choice of these options, in-call, to the user.

12.2.2. Two screen Media Consumer configuring the example

Mixing systems with an even number of screens, "2n", and those with "2n+1" cameras (and vice versa) is always likely to be the problematic case. In this instance, the behavior is likely to be determined by whether a "2 screen" system is really a "2 decoder" system, i.e., whether only one received stream can be displayed per screen or whether more than 2 streams can be received and spread across the available screen area. To enumerate 3 possible behaviors here for the 2 screen system when it learns that the far end is "ideally" expressed via 3 capture streams:

1. Fall back to receiving just a single stream (MCC3, MCC4 or VC5 as per the 1 screen consumer case above) and either leave one screen blank or use it for presentation if / when a presentation becomes active.
2. Receive 3 streams (VC0, VC1 and VC2) and display across 2 screens (either with each capture being scaled to 2/3 of a screen and the center capture being split across 2 screens) or, as would be necessary if there were large bezels on the screens, with each stream being scaled to 1/2 the screen width and height and there being a 4th "blank" panel. This 4th panel could potentially be used for any presentation that became active during the call.
3. Receive 3 streams, decode all 3, and use control information indicating which was the most active to switch between showing the left and center streams (one per screen) and the center and right streams.

For an endpoint capable of all 3 methods of working described above, again it might be appropriate to offer the user the choice of display mode.

12.2.3. Three screen Media Consumer configuring the example

This is the most straightforward case - the Media Consumer would look to identify a set of streams to receive that best matched its available screens and so the VC0 plus VC1 plus VC2 should match optimally. The spatial ordering would give sufficient information for the correct Video Capture to be shown on the correct screen, and the consumer would either need to divide a single encoding group's capability by 3 to determine what resolution and frame rate to configure the provider with or to configure the individual Video Captures' Encoding Groups with what makes most sense (taking into account the receive side decode capabilities, overall call bandwidth, the resolution of the screens plus any user preferences such as motion vs. sharpness).

12.3. Multipoint Conference utilizing Multiple Content Captures

The use of MCCs allows the MCU to construct outgoing Advertisements describing complex media switching and composition scenarios. The following sections provide several examples.

Note: In the examples the identities of the CLUE elements (e.g. Captures, Capture Scene) in the incoming Advertisements overlap. This is because there is no co-ordination between the endpoints. The MCU is responsible for making these unique in the outgoing advertisement.

12.3.1. Single Media Captures and MCC in the same Advertisement

Four endpoints are involved in a Conference where CLUE is used. An MCU acts as a middlebox between the endpoints with a CLUE channel between each endpoint and the MCU. The MCU receives the following Advertisements.

Capture Scene #1	Description=AustralianConfRoom
VC1	Description=Audience
CSV(VC1)	EncodeGroupID=1

Table 10: Advertisement received from Endpoint A

Capture Scene #1	Description=ChinaConfRoom
VC1	Description=Speaker EncodeGroupID=1
VC2	Description=Audience EncodeGroupID=1
CSV(VC1, VC2)	

Table 11: Advertisement received from Endpoint B

Capture Scene #1	Description=USAConfRoom
VC1	Description=Audience EncodeGroupID=1
CSV(VC1)	

Table 12: Advertisement received from Endpoint C

Note: Endpoint B above indicates that it sends two streams.

If the MCU wanted to provide a Multiple Content Capture containing a round robin switched view of the audience from the 3 endpoints and the speaker it could construct the following advertisement:

Advertisement sent to Endpoint F

Capture Scene #1	Description=AustralianConfRoom
VC1 CSV(VC1)	Description=Audience
Capture Scene #2	Description=ChinaConfRoom
VC2 VC3 CSV(VC2, VC3)	Description=Speaker Description=Audience
Capture Scene #3	Description=USAConfRoom
VC4 CSV(VC4)	Description=Audience
Capture Scene #4	
MCC1(VC1,VC2,VC3,VC4) CSV(MCC1)	Policy=RoundRobin:1 MaxCaptures=1 EncodingGroup=1

Table 13: Advertisement sent to Endpoint F - One Encoding

Alternatively if the MCU wanted to provide the speaker as one media stream and the audiences as another it could assign an encoding group to VC2 in Capture Scene 2 and provide a CSV in Capture Scene #4 as per the example below.

Advertisement sent to Endpoint F

Capture Scene #1	Description=AustralianConfRoom
VC1 CSV(VC1)	Description=Audience
Capture Scene #2	Description=ChinaConfRoom
VC2	Description=Speaker
VC3	EncodingGroup=1
CSV(VC2, VC3)	Description=Audience
Capture Scene #3	Description=USAConfRoom
VC4	Description=Audience
CSV(VC4)	
Capture Scene #4	
MCC1(VC1,VC3,VC4)	Policy=RoundRobin:1
	MaxCaptures=1
	EncodingGroup=1
MCC2(VC2)	AllowSubset=True
	MaxCaptures=1
CSV2(MCC1,MCC2)	EncodingGroup=1

Table 14: Advertisement sent to Endpoint F - Two Encodings

Therefore a Consumer could choose whether or not to have a separate speaker related stream and could choose which endpoints to see. If it wanted the second stream but not the Australian conference room it could indicate the following captures in the Configure message:

MCC1(VC3,VC4)	Encoding
VC2	Encoding

Table 15: MCU case: Consumer Response

12.3.2. Several MCCs in the same Advertisement

Multiple MCCs can be used where multiple streams are used to carry media from multiple endpoints. For example:

A conference has three endpoints D, E and F. Each end point has three video captures covering the left, middle and right regions of each conference room. The MCU receives the following advertisements from D and E.

Capture Scene #1	Description=AustralianConfRoom
VC1	CaptureArea=Left EncodingGroup=1
VC2	CaptureArea=Centre EncodingGroup=1
VC3	CaptureArea=Right EncodingGroup=1
CSV(VC1,VC2,VC3)	

Table 16: Advertisement received from Endpoint D

Capture Scene #1	Description=ChinaConfRoom
VC1	CaptureArea=Left EncodingGroup=1
VC2	CaptureArea=Centre EncodingGroup=1
VC3	CaptureArea=Right EncodingGroup=1
CSV(VC1,VC2,VC3)	

Table 17: Advertisement received from Endpoint E

The MCU wants to offer Endpoint F three Capture Encodings. Each Capture Encoding would contain all the Captures from either Endpoint D or Endpoint E depending based on the active speaker. The MCU sends the following Advertisement:

Capture Scene #1	Description=AustralianConfRoom
VC1 VC2 VC3 CSV(VC1,VC2,VC3)	
Capture Scene #2	Description=ChinaConfRoom
VC4 VC5 VC6 CSV(VC4,VC5,VC6)	
Capture Scene #3	
MCC1(VC1,VC4)	CaptureArea=Left MaxCaptures=1 SynchronisationID=1 EncodingGroup=1
MCC2(VC2,VC5)	CaptureArea=Centre MaxCaptures=1 SynchronisationID=1 EncodingGroup=1
MCC3(VC3,VC6)	CaptureArea=Right MaxCaptures=1 SynchronisationID=1 EncodingGroup=1
CSV(MCC1,MCC2,MCC3)	

Table 18: Advertisement sent to Endpoint F

12.3.3. Heterogeneous conference with switching and composition

Consider a conference between endpoints with the following characteristics:

Endpoint A - 4 screens, 3 cameras

Endpoint B - 3 screens, 3 cameras

Endpoint C - 3 screens, 3 cameras

Endpoint D - 3 screens, 3 cameras

Endpoint E - 1 screen, 1 camera

Endpoint F - 2 screens, 1 camera

Endpoint G - 1 screen, 1 camera

This example focuses on what the user in one of the 3-camera multi-screen endpoints sees. Call this person User A, at Endpoint A. There are 4 large display screens at Endpoint A. Whenever somebody at another site is speaking, all the video captures from that endpoint are shown on the large screens. If the talker is at a 3-camera site, then the video from those 3 cameras fills 3 of the screens. If the talker is at a single-camera site, then video from that camera fills one of the screens, while the other screens show video from other single-camera endpoints.

User A hears audio from the 4 loudest talkers.

User A can also see video from other endpoints, in addition to the current talker, although much smaller in size. Endpoint A has 4 screens, so one of those screens shows up to 9 other Media Captures in a tiled fashion. When video from a 3 camera endpoint appears in the tiled area, video from all 3 cameras appears together across the screen with correct spatial relationship among those 3 images.

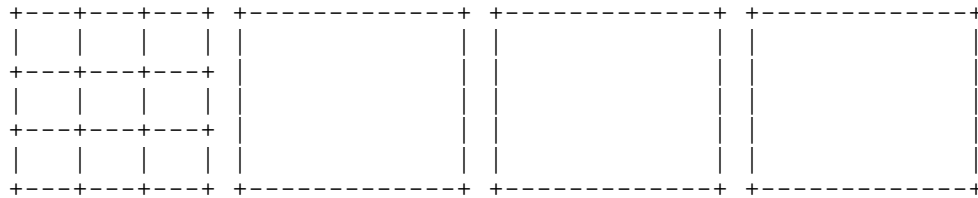


Figure 8: Endpoint A - 4 Screen Display

User B at Endpoint B sees a similar arrangement, except there are only 3 screens, so the 9 other Media Captures are spread out across the bottom of the 3 displays, in a picture-in-picture (PiP) format. When video from a 3 camera endpoint appears in the PiP area, video from all 3 cameras appears together across a single screen with correct spatial relationship.

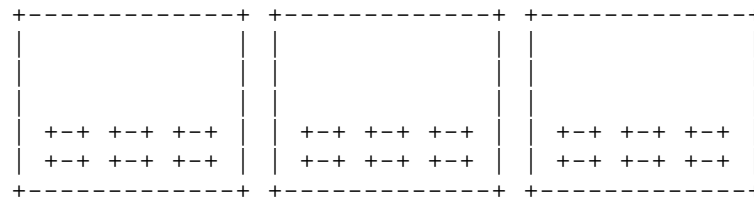


Figure 9: Endpoint B - 3 Screen Display with PiPs

When somebody at a different endpoint becomes the current talker, then User A and User B both see the video from the new talker appear on their large screen area, while the previous talker takes one of the smaller tiled or PiP areas. The person who is the current talker doesn't see themselves; they see the previous talker in their large screen area.

One of the points of this example is that endpoints A and B each want to receive 3 capture encodings for their large display areas, and 9 encodings for their smaller areas. A and B are able to each send the same Configure message to the MCU, and each receive the same conceptual Media Captures from the MCU. The differences are in how they are rendered and are purely a local matter at A and B.

The Advertisements for such a scenario are described below.

Capture Scene #1	Description=Endpoint x
VC1	EncodingGroup=1
VC2	EncodingGroup=1
VC3	EncodingGroup=1
AC1	EncodingGroup=2
CSV1(VC1, VC2, VC3)	
CSV2(AC1)	

Table 19: Advertisement received at the MCU from Endpoints A to D

Capture Scene #1	Description=Endpoint y
VC1	EncodingGroup=1
AC1	EncodingGroup=2
CSV1(VC1)	
CSV2(AC1)	

Table 20: Advertisement received at the MCU from Endpoints E to G

Rather than considering what is displayed CLUE concentrates more on what the MCU sends. The MCU doesn't know anything about the number of screens an endpoint has.

As Endpoints A to D each advertise that three Captures make up a Capture Scene, the MCU offers these in a "site" switching mode. That is that there are three Multiple Content Captures (and Capture Encodings) each switching between Endpoints. The MCU switches in the applicable media into the stream based on voice activity. Endpoint A will not see a capture from itself.

Using the MCC concept the MCU would send the following Advertisement to endpoint A:

Capture Scene #1	Description=Endpoint B
VC4	CaptureArea=Left
VC5	CaptureArea=Center
VC6	CaptureArea=Right
AC1	
CSV(VC4,VC5,VC6)	
CSV(AC1)	
Capture Scene #2	Description=Endpoint C
VC7	CaptureArea=Left
VC8	CaptureArea=Center
VC9	CaptureArea=Right
AC2	
CSV(VC7,VC8,VC9)	
CSV(AC2)	
Capture Scene #3	Description=Endpoint D

VC10 VC11 VC12 AC3 CSV(VC10,VC11,VC12) CSV(AC3)	CaptureArea=Left CaptureArea=Center CaptureArea=Right
Capture Scene #4	Description=Endpoint E
VC13 AC4 CSV(VC13) CSV(AC4)	
Capture Scene #5	Description=Endpoint F
VC14 AC5 CSV(VC14) CSV(AC5)	
Capture Scene #6	Description=Endpoint G
VC15 AC6 CSV(VC15) CSV(AC6)	

Table 21: Advertisement sent to endpoint A - Source Part

The above part of the Advertisement presents information about the sources to the MCC. The information is effectively the same as the received Advertisements except that there are no Capture Encodings associated with them and the identities have been re-numbered.

In addition to the source Capture information the MCU advertises "site" switching of Endpoints B to G in three streams.

Capture Scene #7	Description=Output3streammix
MCC1(VC4,VC7,VC10, VC13)	CaptureArea=Left MaxCaptures=1

	SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC2(VC5,VC8,VC11, VC14)	CaptureArea=Center MaxCaptures=1 SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC3(VC6,VC9,VC12, VC15)	CaptureArea=Right MaxCaptures=1 SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC4() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:0 EncodingGroup=2
MCC5() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:1 EncodingGroup=2
MCC6() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:2 EncodingGroup=2
MCC7() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:3 EncodingGroup=2
CSV(MCC1,MCC2,MCC3) CSV(MCC4,MCC5,MCC6, MCC7)	

Table 22: Advertisement send to endpoint A - switching part

The above part describes the switched 3 main streams that relate to site switching. MaxCaptures=1 indicates that only one Capture from

the MCC is sent at a particular time. SynchronisationID=1 indicates that the source sending is synchronised. The provider can choose to group together VC13, VC14, and VC15 for the purpose of switching according to the SynchronisationID. Therefore when the provider switches one of them into an MCC, it can also switch the others even though they are not part of the same Capture Scene.

All the audio for the conference is included in this Scene #7. There isn't necessarily a one to one relation between any audio capture and video capture in this scene. Typically a change in loudest talker will cause the MCU to switch the audio streams more quickly than switching video streams.

The MCU can also supply nine media streams showing the active and previous eight speakers. It includes the following in the Advertisement:

Capture Scene #8	Description=Output9stream
MCC8(VC4,VC5,VC6,VC7,VC8,VC9,VC10,VC11,VC12,VC13,VC14,VC15)	MaxCaptures=1 Policy=SoundLevel:0 EncodingGroup=1
MCC9(VC4,VC5,VC6,VC7,VC8,VC9,VC10,VC11,VC12,VC13,VC14,VC15)	MaxCaptures=1 Policy=SoundLevel:1 EncodingGroup=1
to	to
MCC16(VC4,VC5,VC6,VC7,VC8,VC9,VC10,VC11,VC12,VC13,VC14,VC15)	MaxCaptures=1 Policy=SoundLevel:8 EncodingGroup=1
CSV(MCC8,MCC9,MCC10,MCC11,MCC12,MCC13,MCC14,MCC15,MCC16)	

Table 23: Advertisement sent to endpoint A - 9 switched part

The above part indicates that there are 9 capture encodings. Each of the Capture Encodings may contain any captures from any source site with a maximum of one Capture at a time. Which Capture is

present is determined by the policy. The MCCs in this scene do not have any spatial attributes.

Note: The Provider alternatively could provide each of the MCCs above in its own Capture Scene.

If the MCU wanted to provide a composed Capture Encoding containing all of the 9 captures it could advertise in addition:

Capture Scene #9	Description=NineTiles
MCC13(MCC8,MCC9,MCC10, MCC11,MCC12,MCC13, MCC14,MCC15,MCC16)	MaxCaptures=9 EncodingGroup=1
CSV(MCC13)	

Table 24: Advertisement sent to endpoint A - 9 composed part

As MaxCaptures is 9 it indicates that the capture encoding contains information from 9 sources at a time.

The Advertisement to Endpoint B is identical to the above other than the captures from Endpoint A would be added and the captures from Endpoint B would be removed. Whether the Captures are rendered on a four screen display or a three screen display is up to the Consumer to determine. The Consumer wants to place video captures from the same original source endpoint together, in the correct spatial order, but the MCCs do not have spatial attributes. So the Consumer needs to associate incoming media packets with the original individual captures in the advertisement (such as VC4, VC5, and VC6) in order to know the spatial information it needs for correct placement on the screens. The Provider can use the RTCP CaptureId SDES item and associated RTP header extension, as described in [I-D.ietf-clue-rtp-mapping], to convey this information to the Consumer.

12.3.4. Heterogeneous conference with voice activated switching

This example illustrates how multipoint "voice activated switching" behavior can be realized, with an endpoint making its own decision about which of its outgoing video streams is considered the "active

talker" from that endpoint. Then an MCU can decide which is the active talker among the whole conference.

Consider a conference between endpoints with the following characteristics:

Endpoint A - 3 screens, 3 cameras

Endpoint B - 3 screens, 3 cameras

Endpoint C - 1 screen, 1 camera

This example focuses on what the user at endpoint C sees. The user would like to see the video capture of the current talker, without composing it with any other video capture. In this example endpoint C is capable of receiving only a single video stream. The following tables describe advertisements from A and B to the MCU, and from the MCU to C, that can be used to accomplish this.

Capture Scene #1	Description=Endpoint x
VC1	CaptureArea=Left EncodingGroup=1
VC2	CaptureArea=Center EncodingGroup=1
VC3	CaptureArea=Right EncodingGroup=1
MCC1(VC1,VC2,VC3)	MaxCaptures=1 CaptureArea=whole scene Policy=SoundLevel:0 EncodingGroup=1
AC1	CaptureArea=whole scene EncodingGroup=2
CSV1(VC1, VC2, VC3) CSV2(MCC1) CSV3(AC1)	

Table 25: Advertisement received at the MCU from Endpoints A and B

Endpoints A and B are advertising each individual video capture, and also a switched capture MCC1 which switches between the other three based on who is the active talker. These endpoints do not

advertise distinct audio captures associated with each individual video capture, so it would be impossible for the MCU (as a media consumer) to make its own determination of which video capture is the active talker based just on information in the audio streams.

Capture Scene #1	Description=conference
MCC1()	CaptureArea=Left MaxCaptures=1 SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC2()	CaptureArea=Center MaxCaptures=1 SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC3()	CaptureArea=Right MaxCaptures=1 SynchronisationID=1 Policy=SoundLevel:0 EncodingGroup=1
MCC4()	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:0 EncodingGroup=1
MCC5() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:0 EncodingGroup=2
MCC6() (for audio)	CaptureArea=whole scene MaxCaptures=1 Policy=SoundLevel:1 EncodingGroup=2
CSV1(MCC1,MCC2,MCC3 CSV2(MCC4) CSV3(MCC5,MCC6)	

Table 26: Advertisement sent from the MCU to C

The MCU advertises one scene, with four video MCCs. Three of them in CSV1 give a left, center, right view of the conference, with "site switching". MCC4 provides a single video capture representing a view of the whole conference. The MCU intends for MCC4 to be switched between all the other original source captures. In this example advertisement the MCU is not giving all the information about all the other endpoints' scenes and which of those captures is included in the MCCs. The MCU could include all that information if it wants to give the consumers more information, but it is not necessary for this example scenario.

The Provider advertises MCC5 and MCC6 for audio. Both are switched captures, with different SoundLevel policies indicating they are the top two dominant talkers. The Provider advertises CSV3 with both MCCs, suggesting the Consumer should use both if it can.

Endpoint C, in its configure message to the MCU, requests to receive MCC4 for video, and MCC5 and MCC6 for audio. In order for the MCU to get the information it needs to construct MCC4, it has to send configure messages to A and B asking to receive MCC1 from each of them, along with their AC1 audio. Now the MCU can use audio energy information from the two incoming audio streams from A and B to determine which of those alternatives is the current talker. Based on that, the MCU uses either MCC1 from A or MCC1 from B as the source of MCC4 to send to C.

13. Acknowledgements

Allyn Romanow and Brian Baldino were authors of early versions. Mark Gorzynski also contributed much to the initial approach. Many others also contributed, including Christian Groves, Jonathan Lennox, Paul Kyzivat, Rob Hansen, Roni Even, Christer Holmberg, Stephen Botzko, Mary Barnes, John Leslie, Paul Coverdale.

14. IANA Considerations

None.

15. Security Considerations

There are several potential attacks related to telepresence, and specifically the protocols used by CLUE, in the case of

conferencing sessions, due to the natural involvement of multiple endpoints and the many, often user-invoked, capabilities provided by the systems.

An MCU involved in a CLUE session can experience many of the same attacks as that of a conferencing system such as that enabled by the XCON framework [RFC5239]. Examples of attacks include the following: an endpoint attempting to listen to sessions in which it is not authorized to participate, an endpoint attempting to disconnect or mute other users, and theft of service by an endpoint in attempting to create telepresence sessions it is not allowed to create. Thus, it is RECOMMENDED that an MCU implementing the protocols necessary to support CLUE, follow the security recommendations specified in the conference control protocol documents. In the case of CLUE, SIP is the conferencing protocol, thus the security considerations in [RFC4579] MUST be followed. Other security issues related to MCUs are discussed in the XCON framework [RFC5239]. The use of xCard with potentially sensitive information provides another reason to implement recommendations of section 11/[RFC5239].

One primary security concern, surrounding the CLUE framework introduced in this document, involves securing the actual protocols and the associated authorization mechanisms. These concerns apply to endpoint to endpoint sessions, as well as sessions involving multiple endpoints and MCUs. Figure 2 in section 5 provides a basic flow of information exchange for CLUE and the protocols involved.

As described in section 5, CLUE uses SIP/SDP to establish the session prior to exchanging any CLUE specific information. Thus the security mechanisms recommended for SIP [RFC3261], including user authentication and authorization, MUST be supported. In addition, the media MUST be secured. DTLS/SRTP MUST be supported and SHOULD be used unless the media, which is based on RTP, is secured by other means (see [RFC7201] [RFC7202]). Media security is also discussed in [I-D.ietf-clue-signaling] and [I-D.ietf-clue-rtp-mapping]. Note that SIP call setup is done before any CLUE specific information is available so the authentication and authorization are based on the SIP mechanisms. The entity that will be authenticated may use the Endpoint identity or the endpoint user identity; this is an application issue and not a CLUE specific issue.

A separate data channel is established to transport the CLUE protocol messages. The contents of the CLUE protocol messages are based on information introduced in this document. The CLUE data model [I-D.ietf-clue-data-model-schema] defines through an XML schema the syntax to be used. Some of the information which could possibly introduce privacy concerns is the xCard information as described in section 7.1.1.10. The decision about which xCard information to send in the CLUE channel is an application policy for point to point and multipoint calls based on the authenticated identity that can be the endpoint identity or the user of the endpoint. For example the telepresence multipoint application can authenticate a user before starting a CLUE exchange with the telepresence system and have a policy per user.

In addition, the (text) description field in the Media Capture attribute (section 7.1.1.6) could possibly reveal sensitive information or specific identities. The same would be true for the descriptions in the Capture Scene (section 7.3.1) and Capture Scene View (7.3.2) attributes. An implementation SHOULD give users control over what sensitive information is sent in an Advertisement. One other important consideration for the information in the xCard as well as the description field in the Media Capture and Capture Scene View attributes is that while the endpoints involved in the session have been authenticated, there is no assurance that the information in the xCard or description fields is authentic. Thus, this information MUST NOT be used to make any authorization decisions.

While other information in the CLUE protocol messages does not reveal specific identities, it can reveal characteristics and capabilities of the endpoints. That information could possibly uniquely identify specific endpoints. It might also be possible for an attacker to manipulate the information and disrupt the CLUE sessions. It would also be possible to mount a DoS attack on the CLUE endpoints if a malicious agent has access to the data channel. Thus, it MUST be possible for the endpoints to establish a channel which is secure against both message recovery and message modification. Further details on this are provided in the CLUE data channel solution document [I-D.ietf-clue-datachannel].

There are also security issues associated with the authorization to perform actions at the CLUE endpoints to invoke specific capabilities (e.g., re-arranging screens, sharing content, etc.). However, the policies and security associated with these actions

are outside the scope of this document and the overall CLUE solution.

16. Changes Since Last Version

NOTE TO THE RFC-Editor: Please remove this section prior to publication as an RFC.

Changes from 24 to 25:

Updates from IESG review.

1. A few clarifications in various places.
2. Change references to RFC5239 and RFC5646 from informative to normative.

Changes from 23 to 24:

1. Updates to Security Considerations section.
2. Update version number of references to other CLUE documents in progress.

Changes from 22 to 23:

1. Updates to Security Considerations section.
2. Update version number of references to other CLUE documents in progress.
3. Change some "MAY" to "may".
4. Fix a few grammatical errors.

Changes from 21 to 22:

1. Add missing references.
2. Update version number of referenced working group drafts.
3. Minor updates for idnits issues.

Changes from 20 to 21:

1. Clarify CLUE can be useful for multi-stream non-telepresence cases.
2. Remove unnecessary ambiguous sentence about optional use of CLUE protocol.

3. Clarify meaning if Area of Capture is not specified.
4. Remove use of "conference" where it didn't fit according to the definition. Use "CLUE session" or "meeting" instead.
5. Embedded Text Attribute: Remove restriction it is for video only.
6. Minor cleanup in section 12 examples.
7. Minor editorial corrections suggested by Christian Groves.

Changes from 19 to 20:

1. Define term "CLUE" in introduction.
2. Add MCC attribute Allow Subset Choice.
3. Remove phrase about reducing SDP size, replace with potentially saving consumer resources.
4. Change example of a CLUE exchange that does not require SDP exchange.
5. Language attribute uses RFC5646.
6. Change Member person type to Attendee. Add Observer type.
7. Clarify DTLS/SRTP MUST be supported.
8. Change SHOULD NOT to MUST NOT regarding using xCard or description information for authorization decisions.
9. Clarify definition of Global View.
10. Refer to signaling doc regarding interoperating with a device that does not support CLUE.
11. Various minor editorial changes from working group last call feedback.
12. Capitalize defined terms.

Changes from 18 to 19:

1. Remove the Max Capture Encodings media capture attribute.
2. Refer to RTP mapping document in the MCC example section.
3. Update references to current versions of drafts in progress.

Changes from 17 to 18:

1. Add separate definition of Global View List.
2. Add diagram for Global View List structure.
3. Tweak definitions of Media Consumer and Provider.

Changes from 16 to 17:

1. Ticket #59 - rename Capture Scene Entry (CSE) to Capture Scene View (CSV)
2. Ticket #60 - rename Global CSE List to Global View List
3. Ticket #61 - Proposal for describing the coordinate system. Describe it better, without conflicts if cameras point in different directions.
4. Minor clarifications and improved wording for Synchronisation Identity, MCC, Simultaneous Transmission Set.
5. Add definitions for CLUE-capable device and CLUE-enabled call, taken from the signaling draft.
6. Update definitions of Capture Device, Media Consumer, Media Provider, Endpoint, MCU, MCC.
7. Replace "middle box" with "MCU".
8. Explicitly state there can also be Media Captures that are not included in a Capture Scene View.
9. Explicitly state "A single Encoding Group MAY refer to encodings for different media types."
10. In example 12.1.1 add axes and audio captures to the diagram, and describe placement of microphones.
11. Add references to data model and signaling drafts.
12. Split references into Normative and Informative sections. Add heading number for references section.

Changes from 15 to 16:

1. Remove Audio Channel Format attribute

2. Add Audio Capture Sensitivity Pattern attribute
3. Clarify audio spatial information regarding point of capture and point on line of capture. Area of capture does not apply to audio.
4. Update section 12 example for new treatment of audio spatial information.
5. Clean up wording of some definitions, and various places in sections 5 and 10.
6. Remove individual encoding parameter paragraph from section 9.
7. Update Advertisement diagram.
8. Update Acknowledgements.
9. References to use cases and requirements now refer to RFCs.
10. Minor editorial changes.

Changes from 14 to 15:

1. Add "=" and "<=" qualifiers to MaxCaptures attribute, and clarify the meaning regarding switched and composed MCC.
2. Add section 7.3.3 Global Capture Scene Entry List, and a few other sentences elsewhere that refer to global CSE sets.
3. Clarify: The Provider MUST be capable of encoding and sending all Captures (*that have an encoding group*) in a single Capture Scene Entry simultaneously.
4. Add voice activated switching example in section 12.
5. Change name of attributes Participant Info/Type to Person Info/Type.
6. Clarify the Person Info/Type attributes have the same meaning regardless of whether or not the capture has a Presentation attribute.

7. Update example section 12.1 to be consistent with the rest of the document, regarding MCC and capture attributes.
8. State explicitly each CSE has a unique ID.

Changes from 13 to 14:

1. Fill in section for Security Considerations.
2. Replace Role placeholder with Participant Information, Participant Type, and Scene Information attributes.
3. Spatial information implies nothing about how constituent media captures are combined into a composed MCC.
4. Clean up MCC example in Section 12.3.3. Clarify behavior of tiled and PIP display windows. Add audio. Add new open issue about associating incoming packets to original source capture.
5. Remove editor's note and associated statement about RTP multiplexing at end of section 5.
6. Remove editor's note and associated paragraph about overloading media channel with both CLUE and non-CLUE usage, in section 5.
7. In section 10, clarify intent of media encodings conforming to SDP, even with multiple CLUE message exchanges. Remove associated editor's note.

Changes from 12 to 13:

1. Added the MCC concept including updates to existing sections to incorporate the MCC concept. New MCC attributes: MaxCaptures, SynchronisationID and Policy.
2. Removed the "composed" and "switched" Capture attributes due to overlap with the MCC concept.
3. Removed the "Scene-switch-policy" CSE attribute, replaced by MCC and SynchronisationID.
4. Editorial enhancements including numbering of the Capture attribute sections, tables, figures etc.

Changes from 11 to 12:

1. Ticket #44. Remove note questioning about requiring a Consumer to send a Configure after receiving Advertisement.
2. Ticket #43. Remove ability for consumer to choose value of attribute for scene-switch-policy.
3. Ticket #36. Remove computational complexity parameter, MaxGroupPps, from Encoding Groups.
4. Reword the Abstract and parts of sections 1 and 4 (now 5) based on Mary's suggestions as discussed on the list. Move part of the Introduction into a new section Overview & Motivation.
5. Add diagram of an Advertisement, in the Overview of the Framework/Model section.
6. Change Intended Status to Standards Track.
7. Clean up RFC2119 keyword language.

Changes from 10 to 11:

1. Add description attribute to Media Capture and Capture Scene Entry.
2. Remove contradiction and change the note about open issue regarding always responding to Advertisement with a Configure message.
3. Update example section, to cleanup formatting and make the media capture attributes and encoding parameters consistent with the rest of the document.

Changes from 09 to 10:

1. Several minor clarifications such as about SDP usage, Media Captures, Configure message.
2. Simultaneous Set can be expressed in terms of Capture Scene and Capture Scene Entry.
3. Removed Area of Scene attribute.

4. Add attributes from draft-groves-clue-capture-attr-01.
5. Move some of the Media Capture attribute descriptions back into this document, but try to leave detailed syntax to the data model. Remove the OUTSOURCE sections, which are already incorporated into the data model document.

Changes from 08 to 09:

1. Use "document" instead of "memo".
2. Add basic call flow sequence diagram to introduction.
3. Add definitions for Advertisement and Configure messages.
4. Add definitions for Capture and Provider.
5. Update definition of Capture Scene.
6. Update definition of Individual Encoding.
7. Shorten definition of Media Capture and add key points in the Media Captures section.
8. Reword a bit about capture scenes in overview.
9. Reword about labeling Media Captures.
10. Remove the Consumer Capability message.
11. New example section heading for media provider behavior
12. Clarifications in the Capture Scene section.
13. Clarifications in the Simultaneous Transmission Set section.
14. Capitalize defined terms.
15. Move call flow example from introduction to overview section
16. General editorial cleanup
17. Add some editors' notes requesting input on issues

18. Summarize some sections, and propose details be outsourced to other documents.

Changes from 06 to 07:

1. Ticket #9. Rename Axis of Capture Point attribute to Point on Line of Capture. Clarify the description of this attribute.
2. Ticket #17. Add "capture encoding" definition. Use this new term throughout document as appropriate, replacing some usage of the terms "stream" and "encoding".
3. Ticket #18. Add Max Capture Encodings media capture attribute.
4. Add clarification that different capture scene entries are not necessarily mutually exclusive.

Changes from 05 to 06:

1. Capture scene description attribute is a list of text strings, each in a different language, rather than just a single string.
2. Add new Axis of Capture Point attribute.
3. Remove appendices A.1 through A.6.
4. Clarify that the provider must use the same coordinate system with same scale and origin for all coordinates within the same capture scene.

Changes from 04 to 05:

1. Clarify limitations of "composed" attribute.
2. Add new section "capture scene entry attributes" and add the attribute "scene-switch-policy".
3. Add capture scene description attribute and description language attribute.
4. Editorial changes to examples section for consistency with the rest of the document.

Changes from 03 to 04:

1. Remove sentence from overview - "This constitutes a significant change ..."
2. Clarify a consumer can choose a subset of captures from a capture scene entry or a simultaneous set (in section "capture scene" and "consumer's choice...").
3. Reword first paragraph of Media Capture Attributes section.
4. Clarify a stereo audio capture is different from two mono audio captures (description of audio channel format attribute).
5. Clarify what it means when coordinate information is not specified for area of capture, point of capture, area of scene.
6. Change the term "producer" to "provider" to be consistent (it was just in two places).
7. Change name of "purpose" attribute to "content" and refer to RFC4796 for values.
8. Clarify simultaneous sets are part of a provider advertisement, and apply across all capture scenes in the advertisement.
9. Remove sentence about lip-sync between all media captures in a capture scene.
10. Combine the concepts of "capture scene" and "capture set" into a single concept, using the term "capture scene" to replace the previous term "capture set", and eliminating the original separate capture scene concept.

17. Normative References

[I-D.ietf-clue-datachannel]

Holmberg, C., "CLUE Protocol Data Channel", draft-ietf-clue-datachannel-11 (work in progress), November 2015.

[I-D.ietf-clue-data-model-schema]

Presta, R., Romano, S P., "An XML Schema for the CLUE data model", draft-ietf-clue-data-model-schema-11 (work in progress), October 2015.

- [I-D.ietf-clue-protocol]
Presta, R. and S. Romano, "CLUE protocol", draft-ietf-clue-protocol-06 (work in progress), October 2015.
- [I-D.ietf-clue-signaling]
Kyzivat, P., Xiao, L., Groves, C., Hansen, R., "CLUE Signaling", draft-ietf-clue-signaling-06 (work in progress), August 2015.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [RFC3264] Rosenberg, J., Schulzrinne, H., "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002.
- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.
- [RFC4566] Handley, M., Jacobsen, V., Perkins, C., "SDP: Session Description Protocol", RFC 4566, July 2006.
- [RFC4579] Johnston, A., Levin, O., "SIP Call Control - Conferencing for User Agents", RFC 4579, August 2006
- [RFC5239] Barnes, M., Boulton, C., Levin, O., "A Framework for Centralized Conferencing", RFC 5239, June 2008.
- [RFC5646] Phillips, A., Davis, M., "Tags for Identifying Languages", RFC 5646, September 2009.
- [RFC6350] Perreault, S., "vCard Format Specification", RFC 6350, August 2011.
- [RFC6351] Perreault, S., "xCard: vCard XML Representation", RFC 6351, August 2011.

18. Informative References

- [I-D.ietf-clue-rtp-mapping]
Even, R., Lennox, J., "Mapping RP streams to CLUE media captures", draft-ietf-clue-rtp-mapping-05 (work in progress), October 2015.
- [RFC4353] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol (SIP)", RFC 4353, February 2006.
- [RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.
- [RFC7201] Westerlund, M., Perkins, C., "Options for Securing RTP Sessions", RFC 7201, April 2014.
- [RFC7202] Perkins, C., Westerlund, M., "Why RTP Does Not Mandate a Single Media Security Solution ", RFC 7202, April 2014.
- [RFC7205] Romanow, A., Botzko, S., Duckworth, M., Even, R., "Use Cases for Telepresence Multistreams", RFC 7205, April 2014.
- [RFC7262] Romanow, A., Botzko, S., Barnes, M., "Requirements for Telepresence Multistreams", RFC 7262, June 2014.

19. Authors' Addresses

Mark Duckworth (editor)
Polycom
Andover, MA 01810
USA

Email: mark.duckworth@polycom.com

Andrew Pepperell
Acano
Uxbridge, England
UK

Email: apeppere@gmail.com

Stephan Wenger
Vidyo, Inc.
433 Hackensack Ave.
Hackensack, N.J. 07601
USA

Email: stewe@stewe.org

CLUE
Internet-Draft
Intended status: Informational
Expires: March 24, 2013

P. Kyzivat
September 20, 2012

CLUE Telemedical Use Case Callflow
draft-kyzivat-clue-telemedical-callflow-01

Abstract

This is the beginning of an example call flow for an instantiation of the use case described in the telemedical use case [I-D.xiao-clue-telemedical-use-case]. More detail will be added later.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 24, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Scenario being illustrated	3
3. Proposed Relationship of O/A to CLUE messages	3
4. Ladder Diagram	4
4.1. Comments	8
5. Message Bodies	8
6. TO-DO list	8
7. Change Log	9
8. Security Considerations	9
9. IANA Considerations	9
10. References	9
10.1. Normative References	9
10.2. Informative References	9
Author's Address	9

1. Introduction

This is a first cut at the call flow. So far it only includes the ladder diagram showing the exchange of SIP and CLUE messages. The content of those messages will be added later. Before doing that it will be useful to agree on at least one acceptable sequence of messages to realize this use case.

2. Scenario being illustrated

The case considered here consists of one surgery room, one remote expert, and one remote classroom, connected by an MCU. The classroom connects first and waits until the surgery begins. The surgery room connects second. At that point the classroom and surgery are joined by the MCU. The expert joins last.

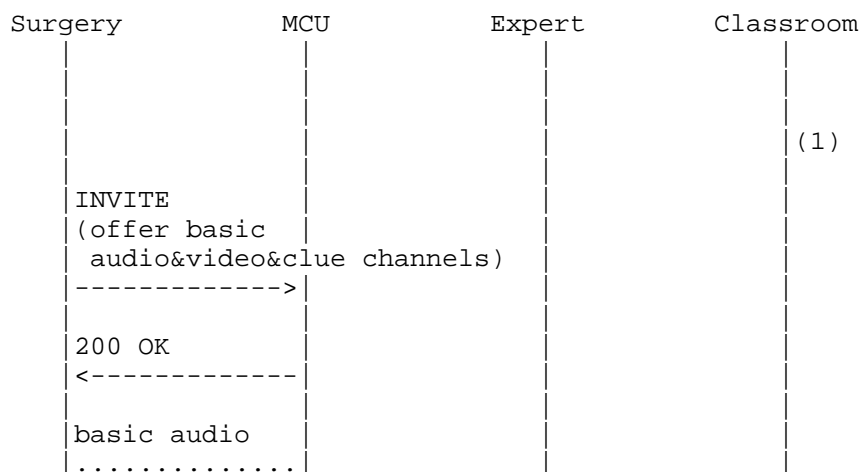
3. Proposed Relationship of O/A to CLUE messages

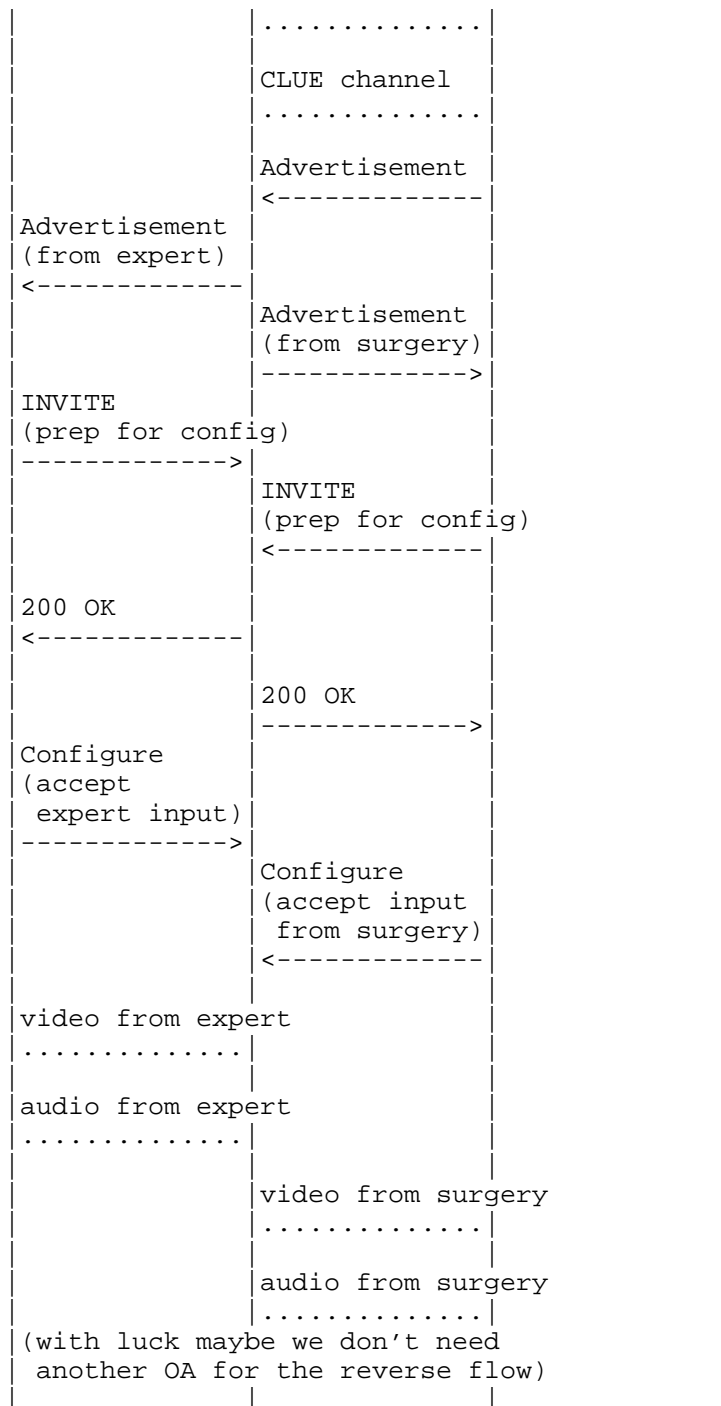
The call flow is constructed in accord with this proposed approach for ordering messages.

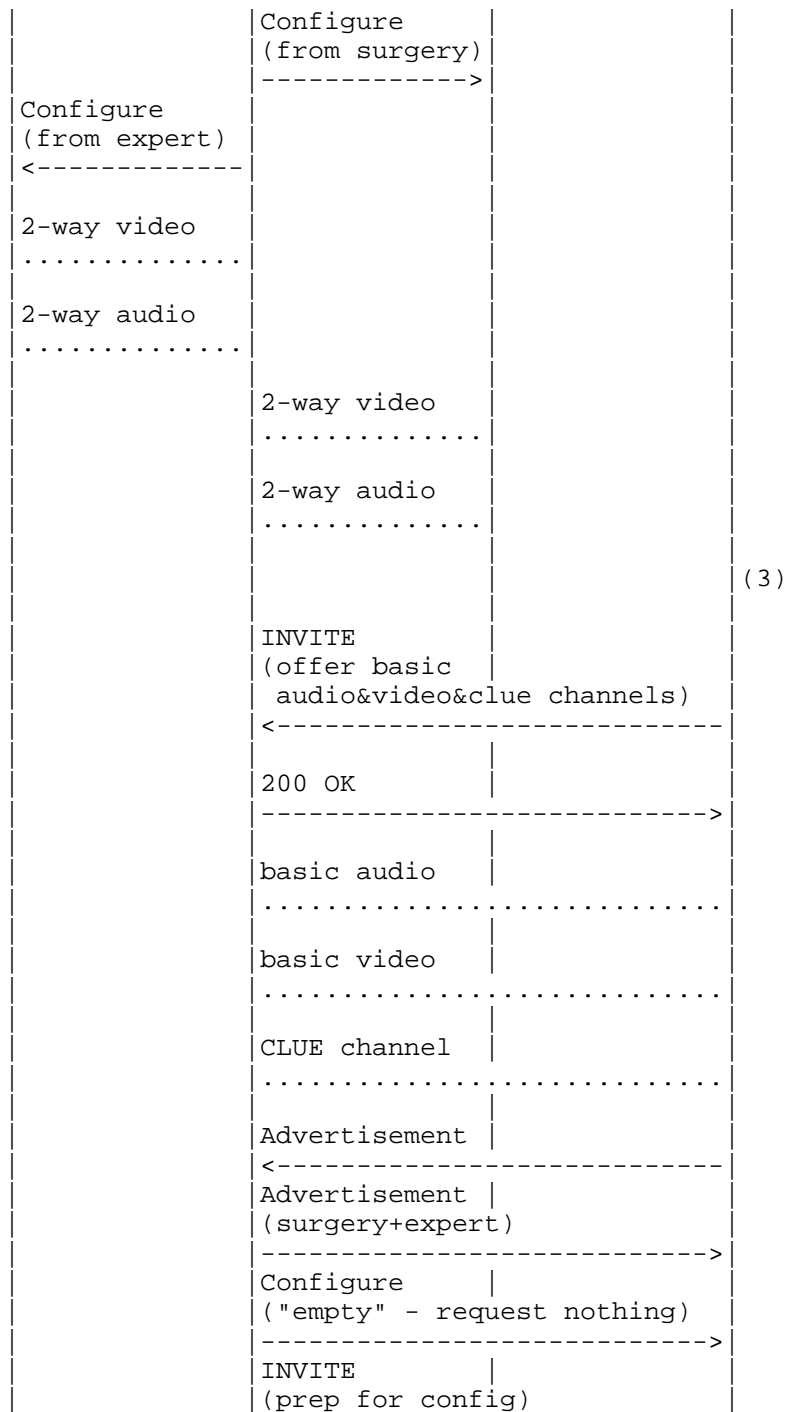
- o What media is sent at any time is determined by the most recently received valid config message, constrained by most recent O/A.
 - * When constraint is bandwidth, sender decides what to drop.
 - * A new O/A can make it possible to send more (or less) of the current config.
- o An O/A should be consistent with the most recent advertisement in each direction.
 - * This gives context to understand why to accept what is offered. (The answerer has no motivation to accept things that aren't.)
- o A config message always explicitly refers to an advertisement.
 - * It is invalid, and will be rejected, if it doesn't refer to the most recently sent advertisement. (This implies there is a message to NACK a config msg.)
- o Typically the endpoint that sends the config message makes an offer to enable it.
 - * This end is the first to know what is needed to support the config.
 - * It is also the end that cares.
 - * Can be before or after the config, or both.
 - * Must accommodate the most recent config received from the other side.
 - * But either side may send an O/A at any time for whatever reason, or may send an offerless INVITE to solicit an offer. (This is basic SIP.)

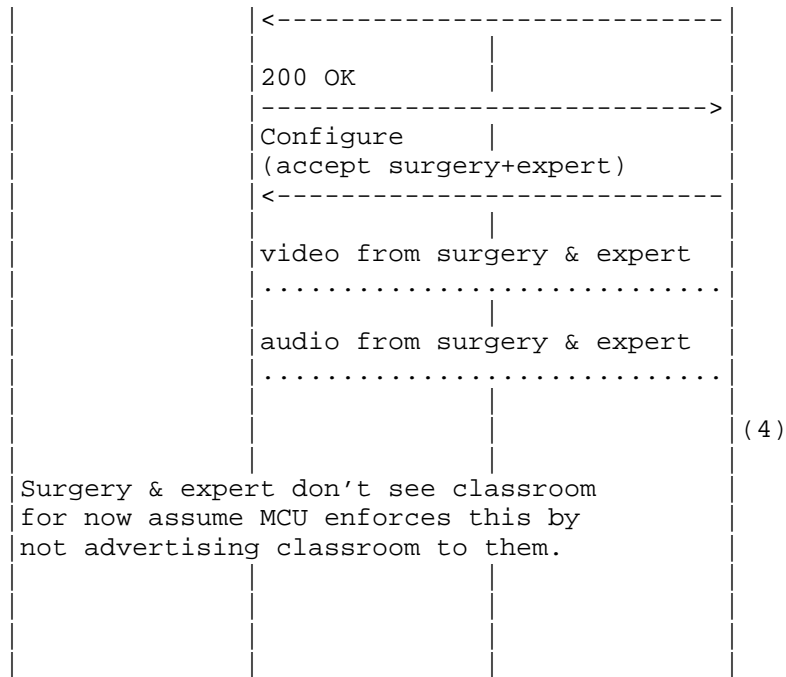
- o Require a config message in response to each advertisement.
 - * Ensures no need for continued support of old advertisement.
 - * Until a new config is received, sender of the advertisement may cease sending media that aren't consistent with the new advertisement.
- o During call major changes requiring O/A will typically happen independently at each endpoint.
 - * But at start of call there will be an exchange of advertisements and configs.
 - * We want to avoid unnecessary O/As in this case.
- o If receive an advertisement after sending one, before receiving a config:
 - * Should send config before initiating O/A.
 - * If receive offer before sending one, it will typically be sufficient to accommodate your config.
 - * If so, won't need to initiate another O/A.
 - * There may still be glare at SIP level. Use standard SIP remedies for that.
 - * If so, the O/A that glared may not need to be retried.
- o First O/A of session occurs before any advertisements or configurations.
 - * Must include the CLUE channel.
 - * Everything else is speculative until advertisements are exchanged.
 - * May be best guess at what is needed, or placeholder intended to maximize interop with arbitrary devices.
 - * Before the first config is received, there should be at most a single capture, chosen by sender, per RTP session.

4. Ladder Diagram









4.1. Comments

There are some issues with the ladder diagram above due to the tool I've used to generate it. The CLUE messages are shown with "--->" rather than "...>" because the tool can't do the latter.

5. Message Bodies

6. TO-DO list

- o Reference [I-D.westerlund-avtcore-max-ssrc] in the initial offer proposing how many RTP streams can be sent and received simultaneously in the offered RTP session.

For example the offerer can send up to 6 RTP streams and receive up to 4. This will help with providing a better advertisements from both sides:

```

m=video 49200 RTP/AVP 99
a=rtpmap:99 H264/90000
a=max-send-ssrc:{*:6}
a=max-recv-ssrc:{*:4}
  
```

7. Change Log

1. Captured suggestion from Roni in a TODO list for later. And made a number of adjustments/cleanups to the diagram. Incorporated a comment from Espen for the MCU to send "empty" configuration initially, and about a missing config message.

Reworked to incorporate and follow the approach I proposed on the first day of the interim.

8. Security Considerations

TBS

9. IANA Considerations

This specification has no IANA considerations

10. References

10.1. Normative References

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[I-D.xiao-clue-telemedical-use-case]
Xiao, L. and R. Even, "Use Case for Telemedical with Multi-streams", draft-xiao-clue-telemedical-use-case-00 (work in progress), July 2012.

[I-D.westerlund-avtcore-max-ssrc]
Westerlund, M., Burman, B., and F. Jansson, "Multiple Synchronization sources (SSRC) in RTP Session Signaling", draft-westerlund-avtcore-max-ssrc-02 (work in progress), July 2012.

10.2. Informative References

Author's Address

Paul H. Kyzivat

Email: pkyzivat@alum.mit.edu

CLUE
Internet-Draft
Intended status: Informational
Expires: September 19, 2013

P. Kyzivat
Huawei
March 18, 2013

CLUE Telemedical Use Case Callflow
draft-kyzivat-clue-telemedical-callflow-02

Abstract

This is the beginning of an example call flow for an instantiation of the use case described in the telemedical use case [I-D.xiao-clue-telemedical-use-case]. More detail will be added later.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 19, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Scenario being illustrated	2
3. Proposed Relationship of O/A to CLUE messages	2
4. Ladder Diagram	4
4.1. Comments	8
5. Message Bodies	8
6. TO-DO list	8
7. Change Log	8
8. Security Considerations	9
9. IANA Considerations	9
10. Normative References	9
Author's Address	9

1. Introduction

This is a first cut at the call flow. So far it only includes the ladder diagram showing the exchange of SIP and CLUE messages. The content of those messages will be added later. Before doing that it will be useful to agree on at least one acceptable sequence of messages to realize this use case.

2. Scenario being illustrated

The case considered here consists of one surgery room, one remote expert, and one remote classroom, connected by an MCU. The classroom connects first and waits until the surgery begins. The surgery room connects second. At that point the classroom and surgery are joined by the MCU. The expert joins last.

3. Proposed Relationship of O/A to CLUE messages

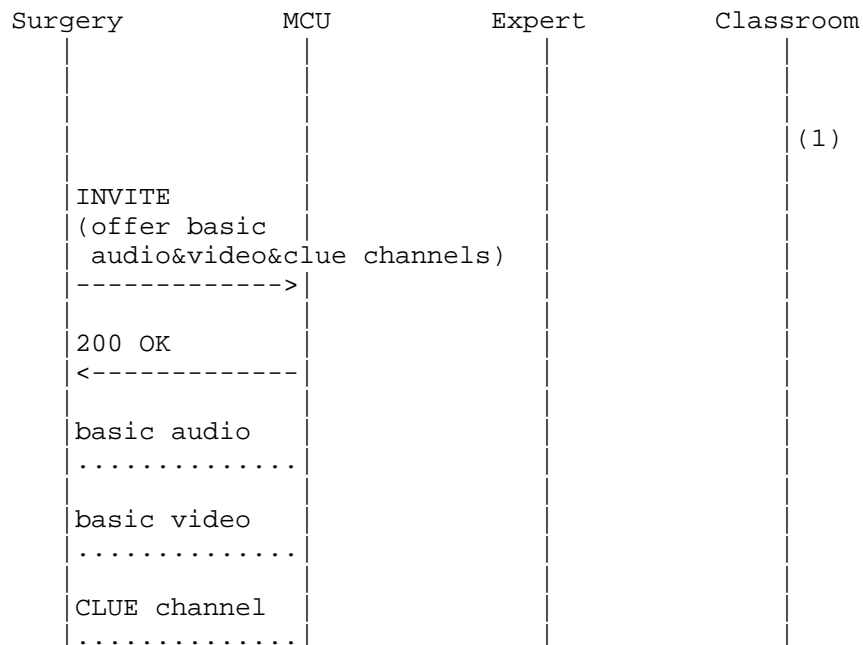
The call flow is constructed in accord with this proposed approach for ordering messages.

- o What media is sent at any time is determined by the most recently received valid config message, constrained by most recent O/A.
 - * When constraint is bandwidth, sender decides what to drop.
 - * A new O/A can make it possible to send more (or less) of the current config.
- o An O/A should be consistent with the most recent advertisement in each direction.

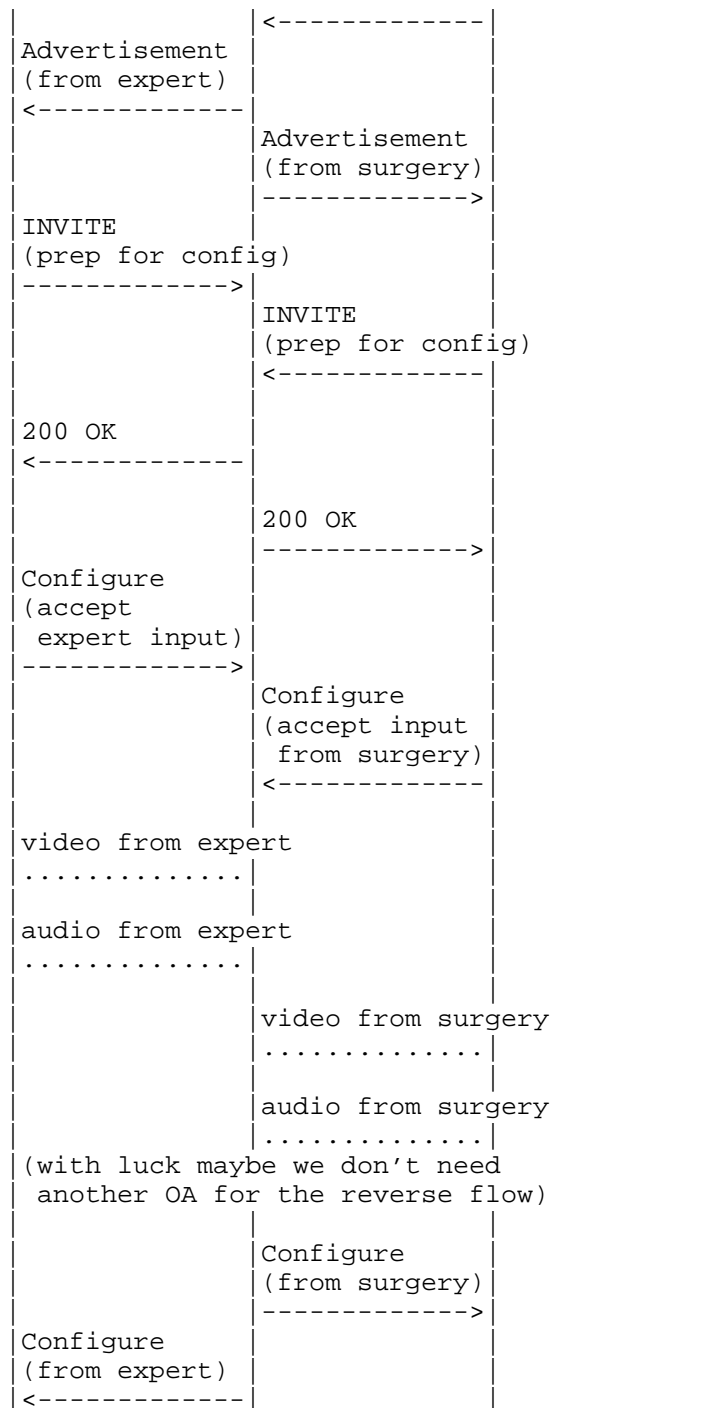
- * This gives context to understand why to accept what is offered.
(The answerer has no motivation to accept things that aren't.)
- o A config message always explicitly refers to an advertisement.
 - * It is invalid, and will be rejected, if it doesn't refer to the most recently sent advertisement. (This implies there is a message to NACK a config msg.)
- o Typically the endpoint that sends the config message makes an offer to enable it.
 - * This end is the first to know what is needed to support the config.
 - * It is also the end that cares.
 - * Can be before or after the config, or both.
 - * Must accommodate the most recent config received from the other side.
 - * But either side may send an O/A at any time for whatever reason, or may send an offerless INVITE to solicit an offer.
(This is basic SIP.)
- o Require a config message in response to each advertisement.
 - * Ensures no need for continued support of old advertisement.
 - * Until a new config is received, sender of the advertisement may cease sending media that aren't consistent with the new advertisement.
- o During call major changes requiring O/A will typically happen independently at each endpoint.
 - * But at start of call there will be an exchange of advertisements and configs.
 - * We want to avoid unnecessary O/As in this case.
- o If receive an advertisement after sending one, before receiving a config:
 - * Should send config before initiating O/A.

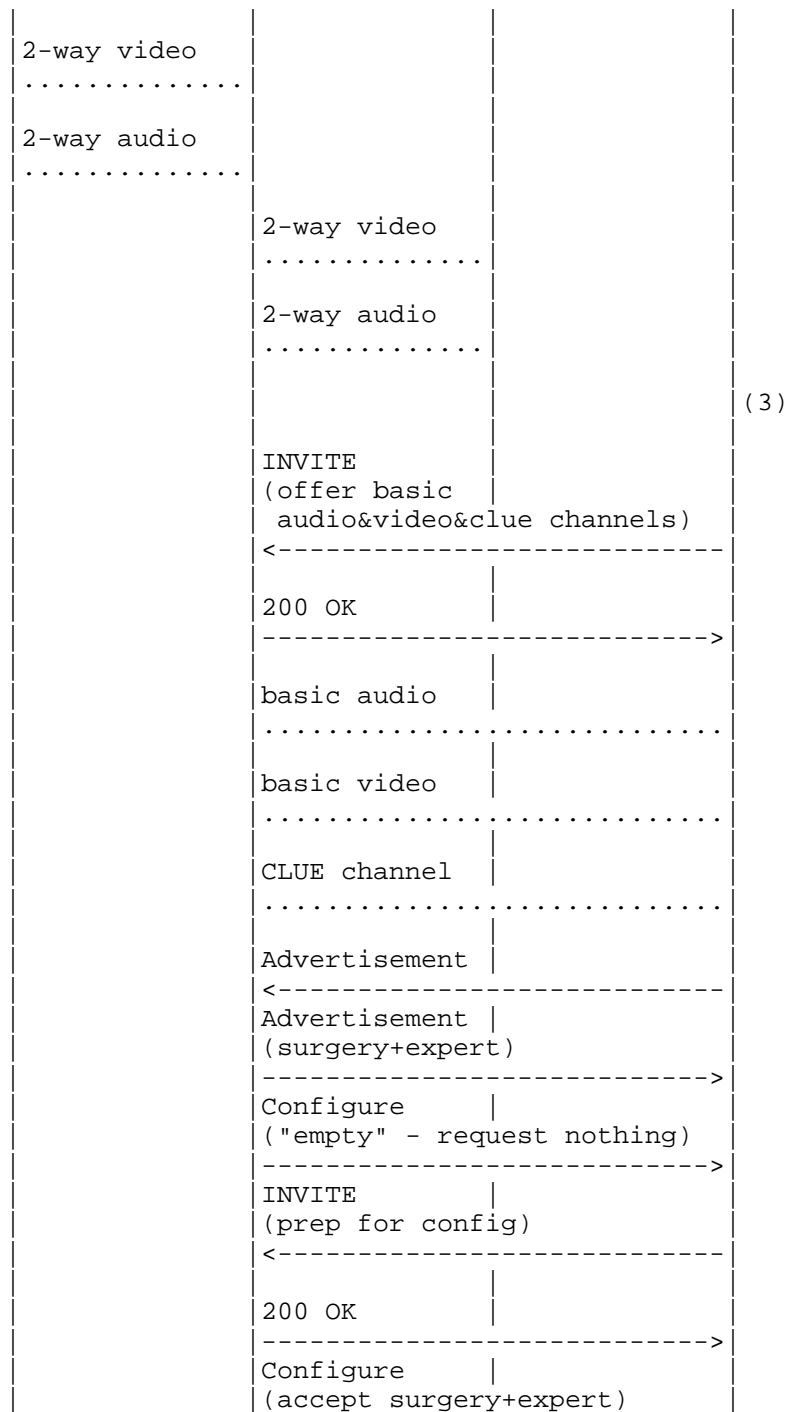
- * If receive offer before sending one, it will typically be sufficient to accommodate your config.
- * If so, won't need to initiate another O/A.
- * There may still be glare at SIP level. Use standard SIP remedies for that.
- * If so, the O/A that glared may not need to be retried.
- o First O/A of session occurs before any advertisements or configurations.
 - * Must include the CLUE channel.
 - * Everything else is speculative until advertisements are exchanged.
 - * May be best guess at what is needed, or placeholder intended to maximize interop with arbitrary devices.
 - * Before the first config is received, there should be at most a single capture, chosen by sender, per RTP session.

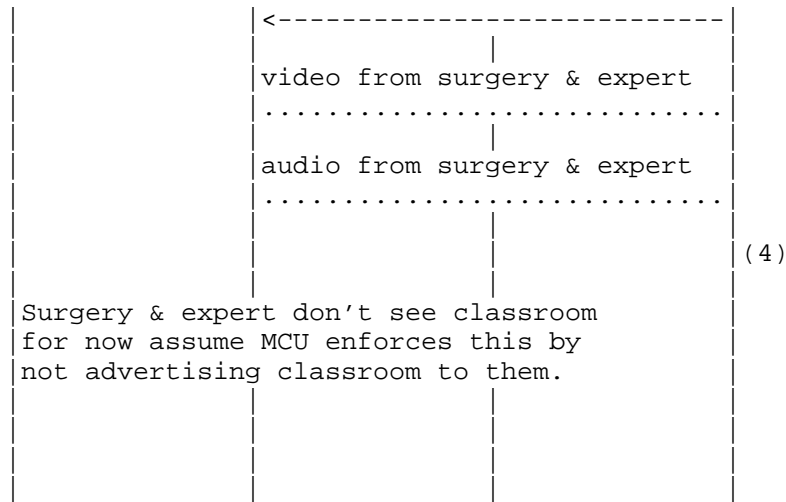
4. Ladder Diagram











4.1. Comments

There are some issues with the ladder diagram above due to the tool I've used to generate it. The CLUE messages are shown with "--->" rather than "...>" because the tool can't do the latter.

5. Message Bodies

TBS

6. TO-DO list

- o Reference [I-D.westerlund-avtcore-max-ssrc] in the initial offer proposing how many RTP streams can be sent and received simultaneously in the offered RTP session.

For example the offerer can send up to 6 RTP streams and receive up to 4. This will help with providing a better advertisements from both sides:

```

m=video 49200 RTP/AVP 99
a=rtpmap:99 H264/90000
a=max-send-ssrc:{*:6}
a=max-recv-ssrc:{*:4}
  
```

7. Change Log

1. Captured suggestion from Roni in a TODO list for later. And made a number of adjustments/cleanups to the diagram. Incorporated a comment from Espen for the MCU to send "empty" configuration initially, and about a missing config message.

Reworked to incorporate and follow the approach I proposed on the first day of the interim.

8. Security Considerations

TBS

9. IANA Considerations

This specification has no IANA considerations

10. Normative References

- [RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.
- [I-D.xiao-clue-telemedical-use-case]
Xiao, L. and R. Even, "Use Case for Telemedical with Multi-streams", draft-xiao-clue-telemedical-use-case-00 (work in progress), July 2012.
- [I-D.westerlund-avtcore-max-ssrc]
Westerlund, M., Burman, B., and F. Jansson, "Multiple Synchronization sources (SSRC) in RTP Session Signaling", draft-westerlund-avtcore-max-ssrc-02 (work in progress), July 2012.

Author's Address

Paul H. Kyzivat
Huawei

Email: pkyzivat@alum.mit.edu

CLUE
Internet-Draft
Intended status: Standards Track
Expires: December 3, 2012

J. Lennox
Vidyo
P. Witty

A. Romanow
Cisco Systems
June 1, 2012

Real-Time Transport Protocol (RTP) Usage for Telepresence Sessions
draft-lennox-clue-rtp-usage-04

Abstract

This document describes mechanisms and recommended practice for transmitting the media streams of telepresence sessions using the Real-Time Transport Protocol (RTP).

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 3, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as

described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. RTP requirements for CLUE	3
4. RTCP requirements for CLUE	5
5. Multiplexing multiple streams or multiple sessions?	6
6. Use of multiple transport flows	6
7. Use Cases	7
8. Other implementation constraints	9
9. Requirements of a solution	9
10. Mapping streams to requested captures	11
10.1. Sending SSRC to capture ID mapping outside the media stream	11
10.2. Sending capture IDs in the media stream	12
10.2.1. Multiplex ID shim	13
10.2.2. RTP header extension	13
10.2.3. Combined approach	14
10.3. Recommendations	16
11. Security Considerations	16
12. IANA Considerations	16
13. References	16
13.1. Normative References	16
13.2. Informative References	17
Authors' Addresses	18

1. Introduction

Telepresence systems, of the architecture described by [I-D.ietf-clue-telepresence-use-cases] and [I-D.ietf-clue-telepresence-requirements], will send and receive multiple media streams, where the number of streams in use is potentially large and asymmetric between endpoints, and streams can come and go dynamically. These characteristics lead to a number of architectural design choices which, while still in the scope of potential architectures envisioned by the Real-Time Transport Protocol [RFC3550], must be fairly different than those typically implemented by the current generation of voice or video conferencing systems.

Furthermore, captures, as defined by the CLUE Framework [I-D.ietf-clue-framework], are a somewhat different concept than RTP's concept of media streams, so there is a need to communicate the associations between them.

This document makes recommendations, for this telepresence architecture, about how streams should be encoded and transmitted in RTP, and how their relation to captures should be communicated.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

3. RTP requirements for CLUE

CLUE will permit a SIP call to include multiple media streams: easily dozens at a time (given, e.g., a continuous presence screen in a multi-point conference), potentially out of a possible pool of hundreds. Furthermore, endpoints will have an asymmetric number of media streams.

Two main backwards compatibility issues exist: firstly, on an initial SIP offer we can not be sure that the far end will support CLUE, and therefore a CLUE endpoint must not offer a selection of RTP sessions which would confuse a CLUEless endpoint. Secondly, there exist many SIP devices in the network through which calls may be routed; even if we know that the far end supports CLUE, re-offering with a larger selection of RTP sessions may fall foul of one of these middle boxes.

We also desire to simplify NAT and firewall traversal by allowing endpoints to deal with only a single static address/port mapping per media type rather than multiple mappings which change dynamically over the duration of the call.

A SIP call in common usage today will typically offer one or two video RTP sessions (one for presentation, one for main video), and one audio session. Each of these RTP sessions will be used to send either zero or one media streams in either direction, with the presence of these streams negotiated in the SDP (offering a particular session as send only, receive only, or send and receive), and through BFCP (for presentation video).

In a CLUE environment this model -- sending zero or one source (in each direction) per RTP session -- doesn't scale as discussed above, and mapping asymmetric numbers of sources to sessions is needlessly complex.

Therefore, telepresence systems SHOULD use a single RTP session per media type, as shown in Figure 1, except where there's a need to give sessions different transport treatment. All sources of the same media type, although from distinct captures, are sent over this single RTP session.

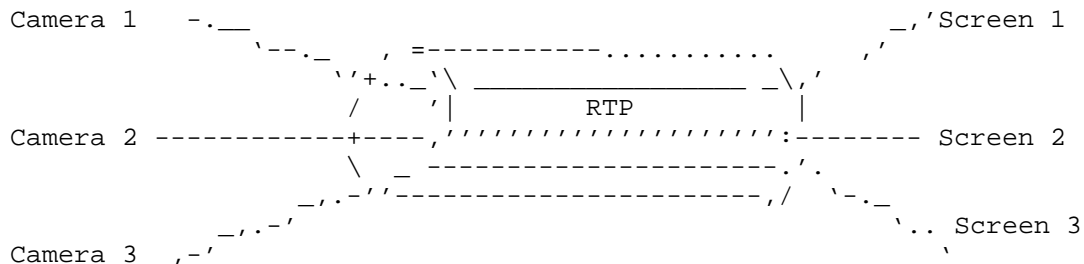


Figure 1: Multiplexing multiple media streams into one RTP session

During call setup, a single RTP session is negotiated for each media type. In SDP, only one media line is negotiated per media and multiple media streams are sent over the same UDP channel negotiated using the SDP media line.

A number of protocol issues involved in multiplexing RTP streams into a single session are discussed in [I-D.westerlund-avtcore-multiplex-architecture] and [I-D.lennox-rtcweb-rtp-media-type-mux]. In the rest of this document we concentrate on examining the mapping of RTP streams to requested

CLUE captures in the specific context of telepresence systems.

The CLUE architecture requires more than simply source multiplexing, as defined by [RFC3550]. The key issue is how a receiver interprets the multiplexed streams it receives, and correlates them with the captures it has requested. In some cases, the CLUE Framework [I-D.ietf-clue-framework]'s concept of the "capture" maps cleanly to the RTP concept of an SSRC, but in many cases it does not.

First we will consider the cases that need to be considered. We will then examine the two most obvious approaches to mapping streams for captures, showing their pros and cons. We then describe a third possible alternative.

4. RTCP requirements for CLUE

When sending media streams, we are also required to send corresponding RTCP information. However, while a unidirectional RTP stream (as identified by a single SSRC) will contain a single stream of media, the associated RTCP stream will include sender information about the stream, but will also include feedback for streams sent in the opposite direction. On a simple point-to-point case, it may be possible to naively forward on RTCP in a similar manner to RTP, but in more complicated use cases where multipoint devices are switching streams to multiple receivers, this simple approach is insufficient.

As an example, receiver report messages are sent with the source SSRC of a single media stream sent in the same direction as the RTCP, but contain within the message zero or more receiver report blocks for streams sent in the other direction. Forwarding on the receiver report packets to the same endpoints which are receiving the media stream tagged with that SSRC will provide no useful information to endpoints receiving the messages, and does not guarantee that the reports will ever reach the origin of the media streams on which they are reporting.

CLUE therefore requires devices to more intelligently deal with received RTCP messages, which will require full packet inspection, including SRTCP decryption. The low rate of RTCP transmission/reception makes this feasible to do.

RTCP also carries information to establish clock synchronization between multiple RTP streams. For CLUE, this information will be crucial, not only for traditional lip-sync between video and audio, but also for synchronized playout of multiple video streams from the same room. This information needs to be provided even in the case of switched captures, to provide clock synchronization for sources that

are temporarily being shown for a switched capture.

5. Multiplexing multiple streams or multiple sessions?

It may not be immediately obvious whether this problem is best described as multiplexing multiple RTP sessions onto a single transport layer, or as multiplexing multiple media streams onto a single RTP session. Certainly, the different captures represent independent purposes for the media that is sent; however, as any stream may be switched into any of the multiplexed captures, we maintain the requirement that all media streams within a CLUE call must have a unique SSRC -- this is also a requirement for the above use of RTCP.

Because of this, CLUE's use of RTP can best be described as multiplexing multiple streams onto one RTP session, but with additional data about the streams to identify their intended destinations. A solution to perform this multiplexing may also be sufficient to multiplex multiple RTP sessions onto one transport session, but this is not a requirement.

6. Use of multiple transport flows

Most existing videoconferencing systems use separate RTP sessions for main and presentation video sources, distinguished by the SDP content attribute [RFC4796]. The use of the CLUE telepresence framework [I-D.ietf-clue-framework] to describe multiplexed streams can remove the need to establish separate RTP sessions (and transport flows) for these sessions, as the relevant information can be provided by CLUE messaging instead.

However, it can still be useful in many cases to establish multiple RTP sessions (and transport flows) for a single CLUE session. Two clear cases would be for disaggregated media (where media is being sent to devices with different transport addresses), or scenarios where different sources should get different quality-of-service treatment. To support such scenarios, the use of multiple RTP sessions, with SDP m lines with different transport addresses, would be necessary.

To support this case, CLUE messaging needs to be able to indicate the RTP session in which a requested capture is intended to be received.

7. Use Cases

There are three distinct use cases relevant for telepresence systems: static stream choice, dynamically changing streams chosen from a finite set, and dynamic changing streams chosen from an unbounded set.

Static stream choice:

In this case, the streams sent over the multiplex are constant over the complete session. An example is a triple-camera system to MCU in which left, center and right streams are sent for the duration of the session.

This describes an endpoint to endpoint, endpoint to multipoint device, and equivalently a transcoding multipoint device to endpoint.

This is illustrated in Figure 2.

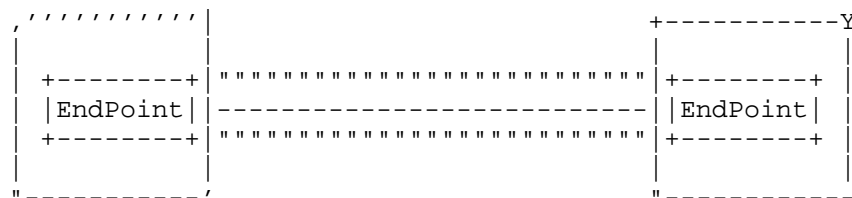


Figure 2: Point to Point Static Streams

Dynamic streams from a finite set:

In this case, the receiver has requested a smaller number of streams than the number of media sources that are available, and expects the sender to switch the sources being sent based on criteria chosen by the sender. (This is called auto-switched in the CLUE Framework [I-D.ietf-clue-framework].)

An example is a triple-camera system to two-screen system, in which the sender needs to switch either LC \rightarrow LR, or CR \rightarrow LR. (Note in particular, in this example, that the center camera stream could be sent as either the left or the right auto-switched capture.)

This describes an endpoint to endpoint, endpoint to multipoint device, and a transcoding device to endpoint.

This is illustrated in Figure 3.

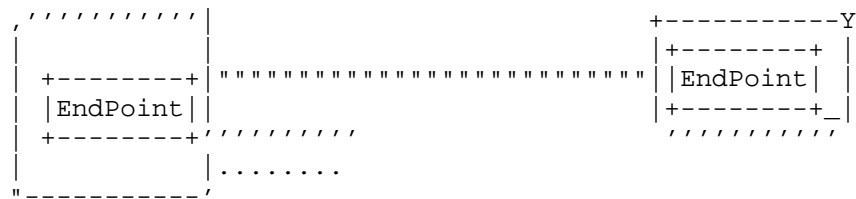


Figure 3: Point to Point Finite Source Streams

Dynamic streams from an unbounded set:

This case describes a switched multipoint device to endpoint, in which the multipoint device can choose to send any streams received from any other endpoints within the conference to the endpoint.

For example, in an MCU to triple-screen system, the MCU could send e.g. LCR of a triple-camera system -> LCR, or CCC of three single-camera endpoints -> LCR.

This is illustrated in Figure 4.

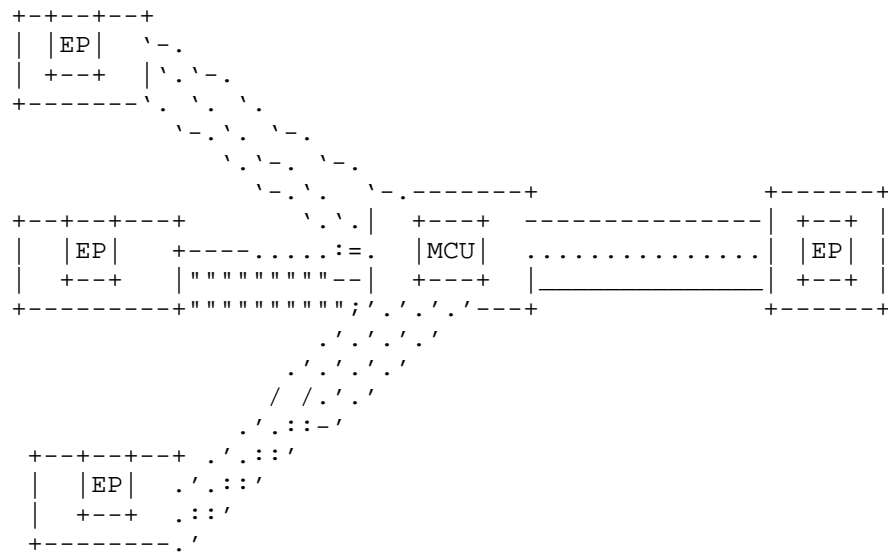


Figure 4: Multipoint Unbounded Streams

Within any of these cases, every stream within the multiplexed

session MUST have a unique SSRC. The SSRC is chosen at random [RFC3550] to ensure uniqueness (within the conference), and contains no meaningful information.

Any source may choose to restart a stream at any time, resulting in a new SSRC. For example, a transcoding MCU might, for reasons of load balancing, transfer an encoder onto a different DSP, and throw away all context of the encoding at this state, sending an RTCP BYE message for the old SSRC, and picking a new SSRC for the stream when started on the new DSP.

Because of this possibility of changing the SSRC at any time, all our use cases can be considered as simplifications of the third and most difficult case, that of dynamic streams from an unbounded set. Thus, this is the primary case we will consider.

8. Other implementation constraints

To cope with receivers with limited decoding resources, for example a hardware based telepresence endpoint with a fixed number of decoding modules, each capable of handling only a single stream, it is particularly important to ensure that the number of streams which the transmitter is expecting the receiver to decode never exceeds the maximum number the receiver has requested. In this case the receiver will be forced to drop some of the received streams, causing a poor user experience, and potentially higher bandwidth usage, should it be required to retransmit I-frames.

On a change of stream, such a receiver can be expected to have a one-out, one-in policy, so that the decoder of the stream currently being received on a given capture is stopped before starting the decoder for the stream replacing it. The sender MUST therefore indicate to the receiver which stream will be replaced upon a stream change.

9. Requirements of a solution

This section lists, more briefly, the requirements a media architecture for Clue telepresence needs to achieve, summarizing the discussion of previous sections. In this section, RFC 2119 language refers to requirements on a solution, not an implementation; thus, requirements keywords are not written in capital letters.

- Media-1: It must not be necessary for a Clue session to use more than a single transport flow for transport of a given media type (video or audio).
- Media-2: It must, however, be possible for a Clue session to use multiple transport flows for a given media type where it is considered valuable (for example, for distributed media, or differential quality-of-service).
- Media-3: It must be possible for a Clue endpoint or MCU to simultaneously send sources corresponding to static, to composited, and to switched captures, in the same transport flow. (Any given device might not necessarily be able send all of these source types; but for those that can, it must be possible for them to be sent simultaneously.)
- Media-4: It must be possible for an original source to move among switched captures (i.e. at one time be sent for one switched capture, and at a later time be sent for another one).
- Media-5: It must be possible for a source to be placed into a switched capture even if the source is a "late joiner", i.e. was added to the conference after the receiver requested the switched source.
- Media-6: Whenever a given source is assigned to a switched capture, it must be immediately possible for a receiver to determine the switched capture it corresponds to, and thus that any previous source is no longer being mapped to that switched capture.
- Media-7: It must be possible for a receiver to identify the actual source that is currently being mapped to a switched capture, and correlate it with out-of-band (non-Clue) information such as rosters.
- Media-8: It must be possible for a source to move among switched captures without requiring a refresh of decoder state (e.g., for video, a fresh I-frame), when this is unnecessary. However, it must also be possible for a receiver to indicate when a refresh of decoder state is in fact necessary.
- Media-9: If a given source is being sent on the same transport flow for more than one reason (e.g. if it corresponds to more than one switched capture at once, or to a static capture), it should be possible for a sender to send only one copy of the source.
- Media-10: On the network, media flows should, as much as possible, look and behave like currently-defined usages of existing protocols; established semantics of existing protocols must not be redefined.
- Media-11: The solution should seek to minimize the processing burden for boxes that distribute media to decoding hardware.
- Media-12: If multiple sources from a single synchronization context are being sent simultaneously, it must be possible for a receiver to associate and synchronize them properly, even for sources that are mapped to switched captures.

10. Mapping streams to requested captures

The goal of any scheme is to allow the receiver to match the received streams to the requested captures. As discussed in Section 7, during the lifetime of the transmission of one capture, we may see one or multiple media streams which belong to this capture, and during the lifetime of one media stream, it may be assigned to one or more captures.

Topologically, the requirements in Section 9 are best addressed by implementing static and a switched captures with an RTP Media Translator, i.e. the topology that RTP Topologies [RFC5117] defines as Topo-Media-Translator. (A composited capture would be the topology described by Topo-Mixer; an MCU can easily produce either or both as appropriate, simultaneously.). The MCU selectively forwards certain sources, corresponding to those sources which it currently assigns to the requested switched captures.

Demultiplexing of streams is done by SSRC; each stream is known to have a unique SSRC. However, this SSRC contains no information about capture IDs. There are two obvious choices for providing the mapping from SSRC to captures: sending the mapping outside of the media stream, or tagging media packets with the capture ID. (There may be other choices, e.g., payload type number, which might be appropriate for multiplexing one audio with one video stream on the same RTP session, but this not relevant for the cases discussed here.)

(An alternative architecture would be to map all captures directly to SSRCs, and then to use a Topo-Mixer topology to represent switched captures as a "mixed" source with a single contributing CSRC. However, such an architecture would not be able to satisfy the requirements Media-8, Media-9, or Media-12 described in Section 9, without substantial changes to the semantics of RTP.)

10.1. Sending SSRC to capture ID mapping outside the media stream

Every RTP packet includes an SSRC, which can be used to demultiplex the streams. However, although the SSRC uniquely identifies a stream, it does not indicate which of the requested captures that stream is tied to. If more than one capture is requested, a mapping from SSRC to capture ID is therefore required so that the media receiver can treat each received stream correctly.

As described above, the receiver may need to know in advance of receiving the media stream how to allocate its decoding resources. Although implementations MAY cache incoming media received before knowing which multiplexed stream it applies to, this is optional, and other implementations may choose to discard media, potentially

requiring an expensive state refresh, such as an Full Intra Request (FIR) [RFC5104].

In addition, a receiver will have to store lookup tables of SSRCs to stream IDs/decoders etc. Because of the large SSRC space (32 bits), this will have to be in the form of something like a hash map, and a lookup will have to be performed for every incoming packet, which may prove costly for e.g. MCUs processing large numbers of incoming streams.

Consider the choices for where to put the mapping from SSRC to capture ID. This mapping could be sent in the CLUE messaging. The use of a reliable transport means that it can be sure that the mapping will not be lost, but if this reliability is achieved through retransmission, the time taken for the mapping to reach all receivers (particularly in a very large scale conference, e.g., with thousands of users) could result in very poor switching times, providing a bad user experience.

A second option for sending the mapping is in RTCP, for instance as a new SDES item. This is likely to follow the same path as media, and therefore if the mapping data is sent slightly in advance of the media, it can be expected to be received in advance of the media. However, because RTCP is lossy and, due to its timing rules, cannot always be sent immediately, the mapping may not be received for some time, resulting in the receiver of the media not knowing how to route the received media. A system of acks and retransmissions could mitigate this, but this results in the same high switching latency behaviour as discussed for using CLUE as a transport for the mapping.

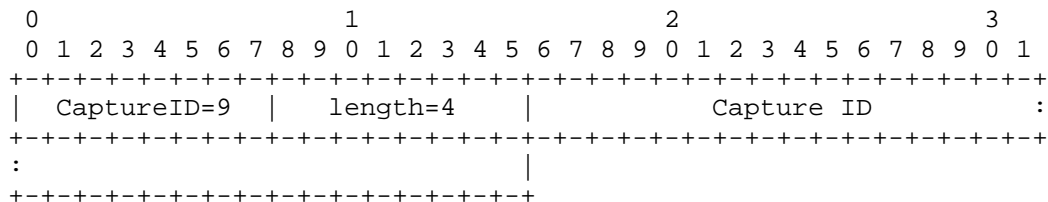


Figure 5: SDES item for encoding of the Capture ID

10.2. Sending capture IDs in the media stream

The second option is to tag each media packet with the capture ID. This means that a receiver immediately knows how to interpret received media, even when an unknown SSRC is seen. As long as the

media carries a known capture ID, it can be assumed that this media stream will replace the stream currently being received with that capture ID.

This gives significant advantages to switching latency, as a switch between sources can be achieved without any form of negotiation with the receiver. There is no chance of receiving media without knowing to which switched capture it belongs.

However, the disadvantage in using a capture ID in the stream that it introduces additional processing costs for every media packet, as capture IDs are scoped only within one hop (i.e., within a cascaded conference a capture ID that is used from the source to the first MCU is not meaningful between two MCUs, or between an MCU and a receiver), and so they may need to be added or modified at every stage.

As capture IDs are chosen by the media sender, by offering a particular capture to multiple recipients with the same ID, this requires the sender to only produce one version of the stream (assuming outgoing payload type numbers match). This reduces the cost in the multicast case, although does not necessarily help in the switching case.

An additional issue with putting capture IDs in the RTP packets comes from cases where a non-CLUE aware endpoint is being switched by an MCU to a CLUE endpoint. In this case, we may require up to an additional 12 bytes in the RTP header, which may push a media packet over the MTU. However, as the MTU on either side of the switch may not match, it is possible that this could happen even without adding extra data into the RTP packet. The 12 additional bytes per packet could also be a significant bandwidth increase in the case of very low bandwidth audio codecs.

10.2.1. Multiplex ID shim

As in draft-westerlund-avtcore-transport-multiplexing

10.2.2. RTP header extension

The capture ID could be carried within the RTP header extension field, using [RFC5285]. This is negotiated within the SDP i.e.

```
a=extmap:1 urn:ietf:params:rtp-hdrex:clue-capture-id
```

Packets tagged by the sender with the capture ID will then contain a header extension as shown below

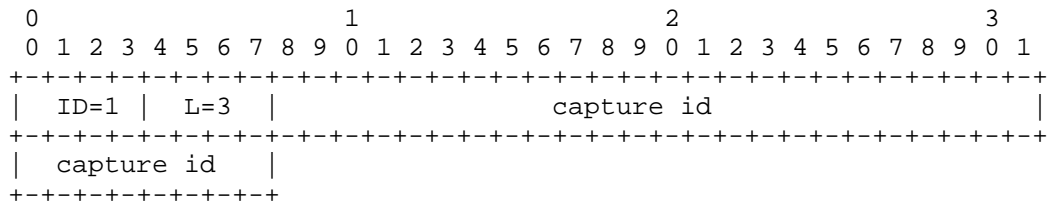


Figure 6: RTP header extension for encoding of the capture ID

To add or modify the capture ID can be an expensive operation, particularly if SRTP is used to authenticate the packet. Modification to the contents of the RTP header requires a reauthentication of the complete packet, and this could prove to be a limiting factor in the throughput of a multipoint device. However, it may be that reauthentication is required in any case due to the nature of SDP. SDP permits the receiver to choose payload types, meaning that a similar option to modify the payload type in the packet header will cause the need to reauthenticate.

10.2.3. Combined approach

The two major flaws of the above methods (high latency switching of SSRC multiplexing, high computational cost on switching nodes) can be mitigated with a combined method. In this, the multiplex ID can be included in packets belonging to the first frame of media (typically an IDR/GDR), but following this only the SSRC is used to demultiplex.

10.2.3.1. Behaviour of receivers

A receiver of a stream should demultiplex on SSRC if it knows the capture ID for the given SSRC, otherwise it should look within the packet for the presence of the stream ID. This has an issue where a stream switches from one capture to a second - for example, in the second use case described in Section 7, where the transmitter chooses to switch the center stream from the receiver's right capture to the left capture, and so the receiver will already know an incorrect mapping from that stream's SSRC to a capture ID.

In this case the receiver should, at the RTP level, detect the presence of the capture ID and update its SSRC to capture ID map. This could potentially have issues where the demultiplexer has now sent the packet to the wrong physical device - this could be solved by checking for the presence of a capture ID in every packet, but this will have speed implications. If a packet is received where the receiver does not already know the mapping between SSRC and capture ID, and the packet does not contain a capture ID, the receiver may

discard it, and MUST request a transmission of the capture ID (see below).

10.2.3.2. Choosing when to send capture IDs

The updated capture ID needs to be known as soon as possible on a switch of SSRCs, as the receiver may be unable to allocate resources to decode the incoming stream, and may throw away the received packets. It can be assumed that the incoming stream is undecodable until the capture ID is received.

In common video codecs (e.g. H.264), decoder refresh frames (either IDR or GDR) also have this property, in that it is impossible to decode any video without first receiving the refresh point. It therefore seems natural to include the capture ID within every packet of an IDR or GDR.

For most audio codecs, where every packet can be decoded independently, there is not such an obvious place to put this information. Placing the capture ID within the first *n* packets of a stream on a switch is the most simple solution, where *n* needs to be sufficiently large that it can be expected that at least one packet will have reached the receiver. For example, *n*=50 on 20ms audio packets will give 1 second of capture IDs, which should give reasonable confidence of arrival.

In the case where a stream is switched between captures, for reasons of coding efficiency, it may be desirable to avoid sending a new IDR frame for this stream, if the receiver's architecture allows the same decoding state to be used for its various captures. In this case, the capture ID could be sent for a small number of frames after the source switches capture, similarly to audio.

10.2.3.3. Requesting Capture ID retransmits

There will, unfortunately, always be cases where a receiver misses the beginning of a stream, and therefore does not have the mapping. One proposal could be to send the capture ID in SDP with every SDP packet; this should ensure that within ~5 seconds of receiving a stream, the capture ID will be received. However, a faster method for requesting the transmission of a capture ID would be preferred.

Again, we look towards the present solution to this problem with video. RFC5104 provides an Full Intra Refresh feedback message, which requests that the encoder provide the stream such that receivers need only the stream after that point. A video receiver without the start of the stream will naturally need to make this request, so by always including the capture ID in refresh frames, we

can be sure that the receiver will have all the information it needs to decode the stream (both a refresh point, and a capture ID).

For audio, we can reuse this message. If a receiver receives an audio stream for which it has no SSRC to capture mapping, it should send a FIR message for the received SSRC. Upon receiving this, an audio encoder must then tag outgoing media packets with the capture ID for a short period of time.

Alternately, a new RTCP feedback message could be defined which would explicitly request a refresh of the capture ID mapping.

10.3. Recommendations

We recommend that endpoints **MUST** support the RTP header extension method of sharing capture IDs, with the extension in every media packet. For low bandwidth situations, this may be considered excessive overhead; in which case endpoints **MAY** support the combined approach.

This will be advertised in the SDP (in a way yet to be determined); if a receiver advertises support for the combined approach, transmitters which support sending the combined approach **SHOULD** use it in preference.

11. Security Considerations

The security considerations for multiplexed RTP do not seem to be different than for non-multiplexed RTP.

Capture IDs need to be integrity-protected in secure environments; however, they do not appear to need confidentiality.

12. IANA Considerations

Depending on the decisions, the new RTP header extension element, the new RTCP SDES item, and/or the new AVPF feedback message will need to be registered.

13. References

13.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

13.2. Informative References

- [I-D.ietf-clue-framework]
Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-05 (work in progress), May 2012.
- [I-D.ietf-clue-telepresence-requirements]
Romanow, A. and S. Botzko, "Requirements for Telepresence Multi-Streams", draft-ietf-clue-telepresence-requirements-01 (work in progress), October 2011.
- [I-D.ietf-clue-telepresence-use-cases]
Romanow, A., Botzko, S., Duckworth, M., Even, R., and I. Communications, "Use Cases for Telepresence Multi-streams", draft-ietf-clue-telepresence-use-cases-02 (work in progress), January 2012.
- [I-D.lennox-rtcweb-rtp-media-type-mux]
Lennox, J. and J. Rosenberg, "Multiplexing Multiple Media Types In a Single Real-Time Transport Protocol (RTP) Session", draft-lennox-rtcweb-rtp-media-type-mux-00 (work in progress), October 2011.
- [I-D.westerlund-avtcore-multiplex-architecture]
Westerlund, M., Burman, B., and C. Perkins, "RTP Multiplexing Architecture", draft-westerlund-avtcore-multiplex-architecture-01 (work in progress), March 2012.
- [RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.
- [RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.
- [RFC5117] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.
- [RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

Authors' Addresses

Jonathan Lennox
Vidyo, Inc.
433 Hackensack Avenue
Seventh Floor
Hackensack, NJ 07601
US

Email: jonathan@vidyo.com

Paul Witty
England
UK

Email: paul.witty@balliol.oxon.org

Allyn Romanow
Cisco Systems
San Jose, CA 95134
USA

Email: allyn@cisco.com

CLUE Working Group
Internet-Draft
Intended status: Informational
Expires: April 4, 2013

R. Presta
S P. Romano
University of Napoli
October 1, 2012

An XML Schema for the CLUE data model
draft-presta-clue-data-model-schema-01

Abstract

This document provides an XML schema file for the definition of CLUE data model types.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 4, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. XML Schema	3
4. Sample XML file	10
5. Diff with -00 version	15
6. Informative References	15

1. Introduction

This document provides an XML schema file for the definition of CLUE data model types. The schema is based on information contained in [I-D.ietf-clue-framework] and also relates to the data model sketched in [I-D.romanow-clue-data-model]. The document actually represents a strawman proposal aiming at the definition of a coherent structure for all the information associated with the description of a telepresence scenario.

2. Terminology

[TBD] Copy text from framework document.

3. XML Schema

This section contains the proposed CLUE data model schema definition. The overall structure of the CLUE data has been derived starting from a data type called "clueInfoType" which comprises the following subelements:

mediaCaptures: the list of media captures available

captureScenes: the list of capture scenes

encodings: the list of individual encodings

encodingGroups: the list of encodings which have been grouped together

simultaneousSets: the list of simultaneous capture sets

All of the above elements refer to concepts that have been introduced and explained in [I-D.ietf-clue-framework] and [I-D.romanow-clue-data-model]. Indeed, the mapping is not exactly one-to-one because: (i) some details of the current data model were not clear enough to be properly included in the schema herein presented; (ii) we slightly modified naming conventions in order to let schema elements better adhere to current XML best naming practices; (iii) we sometimes changed the structure of some of the data elements presented in [I-D.romanow-clue-data-model] in order to enhance the semantic expressiveness of CLUE XML documents.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema
  targetNamespace="urn:ietf:params:xml:ns:clue-info"
```

```
xmlns:tns="urn:ietf:params:xml:ns:clue-info"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="urn:ietf:params:xml:ns:clue-info"
elementFormDefault="qualified"
attributeFormDefault="unqualified">

  <!--This imports the xml:language definition-->
  <xs:import
    namespace="http://www.w3.org/XML/1998/namespace"
    schemaLocation="http://www.w3.org/2001/03/xml.xsd"/>

  <!-- CLUE INFO ELEMENT -->
  <xs:element name="clueInfo" type="clueInfoType"/>

  <!-- CLUE INFO TYPE -->
  <xs:complexType name="clueInfoType">
    <xs:sequence>
      <xs:element name="mediaCaptures" type="mediaCapturesType"/>
      <xs:element name="captureScenes" type="captureScenesType"/>
      <xs:element name="encodings" type="encodingsType"/>
      <xs:element name="encodingGroups" type="encodingGroupsType"/>
      <xs:element name="simultaneousSets" type="simultaneousSetsType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs
="unbounded"/>
    </xs:sequence>
    <!-- TODO: check if any attribute can be needed -->
    <xs:attribute name="clueInfoID" type="xs:ID" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- ENCODINGS TYPE -->
  <xs:complexType name="encodingsType">
    <xs:sequence>
      <xs:element name="encoding" type="encodingType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- ENCODING TYPE -->
  <xs:complexType name="encodingType" abstract="true">
    <xs:sequence>
      <xs:element name="encodingName" type="xs:string"/>
      <xs:element name="maxBandwidth" type="xs:integer"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs
="unbounded"/>
    </xs:sequence>
    <xs:attribute name="encodingID" type="xs:ID" use="required"/>
    <xs:anyAttribute namespace="##any" processContents="lax"/>
  </xs:complexType>
```



```

</xs:complexType>

<!-- AUDIO ENCODING TYPE -->
<xs:complexType name="audioEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="audio" minO
ccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- VIDEO ENCODING TYPE -->
<xs:complexType name="videoEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="video" minO
ccurs="0"/>
        <xs:element name="maxWidth" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxHeight" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxFrameRate" type="xs:integer" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- H264 ENCODING TYPE -->
<xs:complexType name="h264EncodingType">
  <xs:complexContent>
    <xs:extension base="tns:videoEncodingType">
      <xs:sequence>
        <xs:element name="maxH264MacroBlocksRate" type="xs:integer" minOccur
s="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- ENCODING GROUPS TYPE -->
<xs:complexType name="encodingGroupsType">
  <xs:sequence>
    <xs:element name="encodingGroup" type="tns:encodingGroupType" maxOccurs="u
nbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- ENCODING GROUP TYPE -->
<xs:complexType name="encodingGroupType">
  <xs:sequence>

```

```

        <xs:element name="maxGroupBandwidth" type="xs:integer"/>
        <xs:element name="maxGroupH264MacroBlockRate" type="xs:integer" minOccurs="0"/>
        <xs:element name="encodingIDList" type="encodingIDListType"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="encodingGroupID" type="xs:ID" use="required"/>
    <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>

<!-- ENCODING ID LIST TYPE -->
<xs:complexType name="encodingIDListType">
    <xs:sequence>
        <xs:element name="encIDREF" type="xs:IDREF" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<!-- MEDIA CAPTURES TYPE -->
<xs:complexType name="mediaCapturesType">
    <xs:sequence>
        <xs:element name="mediaCapture" type="mediaCaptureType" maxOccurs="unbounded"/>
    </xs:sequence>
</xs:complexType>

<!-- MEDIA CAPTURE TYPE -->
<xs:complexType name="mediaCaptureType" abstract="true">
    <xs:sequence>
        <xs:element name="description" type="xs:string" minOccurs="0"/>
        <xs:element name="encGroupIDREF" type="xs:IDREF"/>
        <xs:element name="capturedMedia" type="xs:string"/>
        <xs:element name="spatialDescription" type="spatialDescriptionType"/>
        <xs:element name="content" type="xs:string" minOccurs="0"/>
        <xs:element name="derived" type="xs:boolean" minOccurs="0"/>
        <xs:element name="switched" type="xs:boolean" minOccurs="0"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="captureID" type="xs:ID" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- AUDIO CAPTURE TYPE -->
<xs:complexType name="audioCaptureType">
    <xs:complexContent>
        <xs:extension base="tns:mediaCaptureType">
            <xs:sequence>
                <xs:element name="audioChannelFormat" type="audioChannelFormatType" minOccurs="0"/>
            </xs:sequence>
        </xs:extension>
    </xs:complexContent>

```

```
</xs:complexType>

<!-- AUDIO CHANNEL FORMAT TYPE -->
<xs:simpleType name="audioChannelFormatType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="mono"/>
    <xs:enumeration value="stereo"/>
  </xs:restriction>
</xs:simpleType>

<!-- VIDEO CAPTURE TYPE -->
<xs:complexType name="videoCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
      <xs:sequence>
        <xs:element name="nativeAspectRatio" type="xs:string" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- SPATIAL DESCRIPTION TYPE -->
<xs:complexType name="spatialDescriptionType">
  <xs:sequence>
    <xs:element name="capturePoint" type="capturePointType" minOccurs="0"/>
    <xs:element name="captureArea" type="captureAreaType" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="scale" type="scaleType" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- SCALE TYPE -->
<xs:simpleType name="scaleType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="millimeters"/>
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="noscale"/>
  </xs:restriction>
</xs:simpleType>

<!-- POINT TYPE -->
<xs:complexType name="pointType">
  <xs:sequence>
    <xs:element name="x" type="xs:decimal"/>
    <xs:element name="y" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>
```

```

        <xs:element name="z" type="xs:decimal"/>
      </xs:sequence>
    </xs:complexType>

    <!-- CAPTURE POINT TYPE -->
    <xs:complexType name="capturePointType">
      <xs:complexContent>
        <xs:extension base="pointType">
          <xs:sequence>
            <xs:element name="captureAxisPoint" type="tns:pointType" minOccurs="0"
/>
          </xs:sequence>
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>

    <!-- CAPTURE AREA TYPE -->
    <xs:complexType name="captureAreaType">
      <xs:sequence>
        <xs:element name="bottomLeft" type="pointType"/>
        <xs:element name="bottomRight" type="pointType"/>
        <xs:element name="topLeft" type="pointType"/>
        <xs:element name="topRight" type="pointType"/>
      </xs:sequence>
    </xs:complexType>

    <!-- CAPTURE SCENES TYPE -->
    <xs:complexType name="captureScenesType">
      <xs:sequence>
        <xs:element name="captureScene" type="captureSceneType" maxOccurs="unb
ounded"/>
      </xs:sequence>
    </xs:complexType>

    <!-- CAPTURE SCENE TYPE -->
    <xs:complexType name="captureSceneType">
      <xs:sequence>
        <xs:element name="sceneDescription" type="xs:string" minOccurs="
0"/>
        <xs:element name="sceneLanguge" type="xs:string" minOccurs="0"/>
        <xs:element name="sceneArea" type="sceneAreaType" minOccurs="0"/
>
        <xs:element name="sceneEntries" type="sceneEntriesType"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="sceneID" type="xs:ID" use="required"/>
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!-- SCENE AREA TYPE -->
    <xs:complexType name="sceneAreaType">

```

```

    <xs:complexContent>
      <xs:extension base="captureAreaType">
        <xs:attribute name="scale" type="scaleType" use="required"/>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <!-- SCENE ENTRIES TYPE -->
  <xs:complexType name="sceneEntriesType">
    <xs:sequence>
      <xs:element name="sceneEntry" type="sceneEntryType" maxOccurs="unbounded"/
>
    </xs:sequence>
  </xs:complexType>

  <!-- SCENE ENTRY TYPE -->
  <xs:complexType name="sceneEntryType">
    <xs:sequence>
      <xs:element name="sceneEntryDescription" type="xs:string"/>
      <xs:element name="captureIDList" type="captureIDListType"/>
      <xs:element name="switchingPolicies" type="switchingPoliciesType" minOccur
s="0"/>
    </xs:sequence>
    <xs:attribute name="entryID" type="xs:ID" use="required"/>
    <xs:attribute name="mediaType" type="xs:string" use="required"/>
  </xs:complexType>

  <!-- CAPTURE ID LIST TYPE -->
  <xs:complexType name="captureIDListType">
    <xs:sequence>
      <xs:element name="captureIDREF" type="xs:IDREF" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="ID" type="xs:ID" use="required"/>
  </xs:complexType>

  <!-- SWITCHING POLICIES TYPE -->
  <xs:complexType name="switchingPoliciesType">
    <xs:sequence>
      <xs:element name="siteSwitching" type="xs:boolean" minOccurs="0"/>
      <xs:element name="segmentSwitching" type="xs:boolean" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <!-- SIMULTANEOUS SETS TYPE -->
  <xs:complexType name="simultaneousSetsType">
    <xs:sequence>
      <xs:element name="simultaneousSet" type="captureIDListType" maxOccurs="unb
ounded"/>
    </xs:sequence>
  </xs:complexType>

```

```
</xs:schema>
```

4. Sample XML file

The following XML document represents a schema compliant example of a CLUE telepresence scenario.

The example describes a scenario similar to the one in section 11.1 of [I-D.ietf-clue-framework]. There are 6 video captures:

VC0: the video from the left camera

VC1: the video from the central camera

VC2: the video from the right camera

VC3: the video of the loudest segment of the room taken from the central camera

VC4: the overall view of the telepresence room taken from the central camera

VC5: the video associated with the slide stream

There are 2 audio captures:

AC0: the overall room audio

AC1: the audio associated with the slide stream presentation

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<clueInfo xmlns="urn:ietf:params:xml:ns:clue-info" clueInfoID="01">
  <mediaCaptures>
    <mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC0">
      <description>left camera video</description>
      <encGroupIDREF>EG1</encGroupIDREF>
      <capturedMedia>video</capturedMedia>
      <content>main</content>
      <derived>>false</derived>
      <switched>>false</switched>
    </mediaCapture>
    <mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC1">
      <description>central camera video</description>
```

```

        <encGroupIDREF>EG1</encGroupIDREF>
        <capturedMedia>video</capturedMedia>
        <content>main</content>
        <derived>>false</derived>
        <switched>>false</switched>
    </mediaCapture>
    <mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC2">
        <description>right camera video</description>
        <encGroupIDREF>EG1</encGroupIDREF>
        <capturedMedia>video</capturedMedia>
        <content>main</content>
        <derived>>false</derived>
        <switched>>false</switched>
    </mediaCapture>
    <mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC3">
        <description>video capture of the loudest room segment</description>
        <encGroupIDREF>EG1</encGroupIDREF>
        <capturedMedia>video</capturedMedia>
        <content>main</content>
        <derived>>false</derived>
        <switched>>true</switched>
    </mediaCapture>
    <mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC4">
        <description>zoomed out view of the room</description>
        <encGroupIDREF>EG1</encGroupIDREF>
        <capturedMedia>video</capturedMedia>
        <spatialDescription scale="unknown">
            <captureArea>
                <bottomLeft>
                    <x>-2011</x>
                    <y>2850</y>
                    <z>0</z>
                </bottomLeft>
                <bottomRight>
                    <x>-673</x>
                    <y>3000</y>
                    <z>0</z>
                </bottomRight>
                <topLeft>
                    <x>-2011</x>
                    <y>2850</y>
                    <z>757</z>
                </topLeft>
                <topRight>
                    <x>-673</x>
                    <y>3000</y>
                    <z>757</z>
                </topRight>
            </captureArea>
        </spatialDescription>
    </mediaCapture>

```

```

        </captureArea>
    </spatialDescription>
    <content>main</content>
    <derived>>false</derived>
    <switched>>false</switched>
</mediaCapture>
<mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="videoCaptureType" captureID="VC5">
    <description>presentation slides</description>
    <encGroupIDREF>EGL</encGroupIDREF>
    <capturedMedia>video</capturedMedia>
    <content>slides</content>
    <derived>>false</derived>
    <switched>>false</switched>
</mediaCapture>
<mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="audioCaptureType" captureID="AC0">
    <description>room audio</description>
    <encGroupIDREF>EGL</encGroupIDREF>
    <capturedMedia>audio</capturedMedia>
    <content>main</content>
    <derived>>false</derived>
    <spatialDescription scale="unknown">
        <capturePoint xsi:type="capturePointType">
            <x>0</x>
            <y>0</y>
            <z>0</z>
            <captureAxisPoint>
                <x>0</x>
                <y>1</y>
                <z>1</z>
            </captureAxisPoint>
        </capturePoint>
    </spatialDescription>
    <audioChannelFormat>mono</audioChannelFormat>
</mediaCapture>
<mediaCapture xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:t
ype="audioCaptureType" captureID="AC1">
    <description>presentation audio</description>
    <encGroupIDREF>EGL</encGroupIDREF>
    <capturedMedia>audio</capturedMedia>
    <content>slides</content>
    <derived>>false</derived>
    <audioChannelFormat>mono</audioChannelFormat>
</mediaCapture>
</mediaCaptures>
<captureScenes>
    <captureScene sceneID="CS1">
        <sceneDescription>main room scene</sceneDescription>
        <sceneEntries>
            <sceneEntry entryID="SE1" mediaType="video">

```



```

        <sceneEntryDescription>zoomed in participant videos</sceneEnt
ryDescription>
        <captureIDList>
            <captureIDREF>VC0</captureIDREF>
            <captureIDREF>VC1</captureIDREF>
            <captureIDREF>VC2</captureIDREF>
        </captureIDList>
    </sceneEntry>
    <sceneEntry entryID="SE2" mediaType="audio">
        <sceneEntryDescription>room audio</sceneEntryDescription>
        <captureIDList>
            <captureIDREF>AC0</captureIDREF>
        </captureIDList>
    </sceneEntry>
    <sceneEntry entryID="SE3" mediaType="video">
        <sceneEntryDescription>loudest room segment</sceneEntryDescri
ption>
        <captureIDList>
            <captureIDREF>VC3</captureIDREF>
        </captureIDList>
    </sceneEntry>
    <sceneEntry entryID="SE4" mediaType="video">
        <sceneEntryDescription>room view</sceneEntryDescription>
        <captureIDList>
            <captureIDREF>VC4</captureIDREF>
        </captureIDList>
    </sceneEntry>
</sceneEntries>
</captureScene>
<captureScene sceneID="CS2">
    <sceneDescription>presentation view</sceneDescription>
    <sceneEntries>
        <sceneEntry entryID="SE5" mediaType="video">
            <sceneEntryDescription>presentation video</sceneEntryDescript
ion>
            <captureIDList>
                <captureIDREF>VC5</captureIDREF>
            </captureIDList>
        </sceneEntry>
        <sceneEntry entryID="SE6" mediaType="audio">
            <sceneEntryDescription>presentation audio</sceneEntryDescript
ion>
            <captureIDList>
                <captureIDREF>AC1</captureIDREF>
            </captureIDList>
        </sceneEntry>
    </sceneEntries>
</captureScene>
</captureScenes>
<encodings>
    <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
"h264EncodingType" encodingID="V_ENC1">
        <encodingName>H264</encodingName>

```

```
<maxBandwidth>14</maxBandwidth>
<encodedMedia>video</encodedMedia>
<maxWidth>1080</maxWidth>
<maxHeight>60</maxHeight>
<maxFrameRate>11</maxFrameRate>
<maxH264MacroBlocksRate>12</maxH264MacroBlocksRate>
</encoding>
<encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type=
"audioEncodingType" encodingID="A_ENC1">
  <encodingName>G711</encodingName>
  <maxBandwidth>64</maxBandwidth>
  <encodedMedia>audio</encodedMedia>
</encoding>
</encodings>
<encodingGroups>
  <encodingGroup encodingGroupID="EG1">
    <maxGroupBandwidth>17</maxGroupBandwidth>
    <maxGroupH264MacroBlockRate>19</maxGroupH264MacroBlockRate>
    <encodingIDList>
      <encIDREF>V_ENC1</encIDREF>
      <encIDREF>A_ENC1</encIDREF>
    </encodingIDList>
  </encodingGroup>
</encodingGroups>
<simultaneousSets>
  <simultaneousSet ID="S1">
    <captureIDREF>VC0</captureIDREF>
    <captureIDREF>VC1</captureIDREF>
    <captureIDREF>VC2</captureIDREF>
    <captureIDREF>VC5</captureIDREF>
    <captureIDREF>AC0</captureIDREF>
    <captureIDREF>AC1</captureIDREF>
  </simultaneousSet>
  <simultaneousSet ID="S2">
    <captureIDREF>VC0</captureIDREF>
    <captureIDREF>VC3</captureIDREF>
    <captureIDREF>VC2</captureIDREF>
    <captureIDREF>VC5</captureIDREF>
    <captureIDREF>AC0</captureIDREF>
    <captureIDREF>AC1</captureIDREF>
  </simultaneousSet>
  <simultaneousSet ID="S3">
    <captureIDREF>VC0</captureIDREF>
    <captureIDREF>VC4</captureIDREF>
    <captureIDREF>VC2</captureIDREF>
    <captureIDREF>VC5</captureIDREF>
    <captureIDREF>AC0</captureIDREF>
    <captureIDREF>AC1</captureIDREF>
  </simultaneousSet>
</simultaneousSets>
```

```
</simultaneousSets>
</clueInfo>
```

5. Diff with -00 version

captureAreaType: bottomUp & co. typos corrected (see Roni's email)

spatialDescription in mediaCaptureType now is mandatory

captureAxisPoint now embedded in capturePointType as an optional element

nativeAspectRatio as described in [I-D.romanow-clue-data-model]
added as an optional videoCaptureType element

clueInfoID added as required attribute for the clueInfoType

spatialDescription added for capture VC4 and AC0 in the XML
example, which is now coherent with the new XML Schema

6. Informative References

- | | |
|-------------------------------|--|
| [I-D.ietf-clue-framework] | Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-06 (work in progress), July 2012. |
| [I-D.romanow-clue-data-model] | Romanow, A. and A. Pepperell, "Data model for the CLUE Framework", draft-romanow-clue-data-model-01 (work in progress), June 2012. |

Authors' Addresses

Roberta Presta
University of Napoli
Via Claudio 21
Napoli 80125
Italy

EMail: roberta.presta@unina.it

Simon Pietro Romano
University of Napoli
Via Claudio 21
Napoli 80125
Italy

EMail: spromano@unina.it

CLUE Working Group
Internet-Draft
Intended status: Informational
Expires: September 9, 2013

R. Presta
S P. Romano
University of Napoli
March 8, 2013

An XML Schema for the CLUE data model
draft-presta-clue-data-model-schema-03

Abstract

This document provides an XML schema file for the definition of CLUE data model types.

Status of This Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 9, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. Terminology	4
3. XML Schema	4
4. <mediaCaptures>	13
5. <encodings>	13
6. <encodingGroups>	13
7. <captureScenes>	14
8. <simultaneousSets>	14
9. <captureEncodings>	14
10. <mediaCapture>	14
10.1. <capturedMedia>	15
10.2. <captureSceneIDREF>	15
10.3. <encGroupIDREF>	16
10.4. <spatialInformation>	16
10.4.1. <capturePoint>	17
10.4.2. <captureArea>	18
10.5. <nonSpatiallyDefinible>	18
10.6. <description>	19
10.7. <priority>	19
10.8. <lang>	19
10.9. <content>	19
10.10. <switched>	20
10.11. <dynamic>	20
10.12. <composed>	20
10.13. <maxCaptureEncodings>	20
10.14. <relatedTo>	20
10.15. captureID attribute	21
11. Audio captures	21
11.1. <audioChannelFormat>	21
11.2. <micPattern>	22
12. Video captures	22
12.1. <nativeAspectRatio>	23
12.2. <embeddedText>	23
13. Text captures	24
14. <captureScene>	24
14.1. <sceneSpace> (was:<sceneArea>)	25
14.2. <sceneEntries>	26
14.3. sceneID attribute	26
14.4. scale attribute	26
15. <sceneEntry>	27
15.1. <switchingPolicies>	27
15.2. <mediaCaptureIDs>	28
15.3. sceneEntryID attribute	29
15.4. mediaType attribute	29
16. <encoding>	29
16.1. <encodingName>	29
16.2. <maxBandwidth>	29
16.3. encodingID attribute	30

17. Audio encodings	30
18. Video encodings	30
18.1. <maxWidth>	31
18.2. <maxHeight>	31
18.3. <maxFrameRate>	31
19. H26X encodings	31
20. <encodingGroup>	32
20.1. <maxGroupBandwidth>	32
20.2. <maxGroupPps>	33
20.3. <encodingIDList>	33
20.4. encodingGroupID attribute	33
21. <simultaneousSet>	33
21.1. <captureIDREF>	34
21.2. <sceneEntryIDREF>	34
22. <captureEncoding>	34
22.1. <mediaCaptureID>	34
22.2. <encodingID>	34
23. <clueInfo>	34
24. Sample XML file	35
25. Diff with unofficial -02 version	44
26. Diff with -02 version	46
27. Informative References	46

1. Introduction

This document provides an XML schema file for the definition of CLUE data model types.

The schema is based on information contained in [I-D.ietf-clue-framework] and also relates to the data model sketched in [I-D.romanow-clue-data-model]. It encodes information and constraints defined in the aforementioned documents in order to provide a formal representation of the concepts therein presented. The schema definition is intended to be modified according to changes applied to the above mentioned CLUE documents.

The document actually represents a strawman proposal aiming at the definition of a coherent structure for all the information associated with the description of a telepresence scenario.

2. Terminology

[TBD] Copy text from the framework document.

3. XML Schema

This section contains the proposed CLUE data model schema definition.

The element and attribute definitions are formal representation of the concepts needed to describe the capabilities of a media provider and the current streams it is transmitting within a telepresence session.

The main groups of information are:

<mediaCaptures>: the list of media captures available (Section 4)

<encodings>: the list of individual encodings (Section 5)

<encodingGroups>: the list of encodings groups (Section 6)

<captureScenes>: the list of capture scenes (Section 7)

<simultaneousSets>: the list of simultaneous capture sets (Section 8)

<captureEncodings>: the list of instantiated capture encodings (Section 9)

All of the above refers to concepts that have been introduced in [I-D.ietf-clue-framework] and [I-D.romanow-clue-data-model] and

further detailed in threads on the mailing list as well as in the following of this document.

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema
  targetNamespace="urn:ietf:params:xml:ns:clue-info"
  xmlns:tns="urn:ietf:params:xml:ns:clue-info"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="urn:ietf:params:xml:ns:clue-info"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <!-- ELEMENT DEFINITIONS -->
  <xs:element name="mediaCaptures" type="mediaCapturesType"/>
  <xs:element name="encodings" type="encodingsType"/>
  <xs:element name="encodingGroups" type="encodingGroupsType"/>
  <xs:element name="captureScenes" type="captureScenesType"/>
  <xs:element name="simultaneousSets" type="simultaneousSetsType"/>
  <xs:element name="captureEncodings" type="captureEncodingsType"/>

  <!-- MEDIA CAPTURES TYPE -->
  <!-- envelope of media captures -->
  <xs:complexType name="mediaCapturesType">
    <xs:sequence>
      <xs:element name="mediaCapture" type="mediaCaptureType"
        maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- DESCRIPTION element -->
  <xs:element name="description">
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base="xs:string">
          <xs:attribute name="lang" type="xs:language"/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <!-- MEDIA CAPTURE TYPE -->
  <xs:complexType name="mediaCaptureType" abstract="true">
    <xs:sequence>
      <!-- mandatory fields -->
      <xs:element name="capturedMedia" type="xs:string"/>
      <xs:element name="captureSceneIDREF" type="xs:IDREF"/>
      <xs:element name="encGroupIDREF" type="xs:IDREF"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```
<xs:choice>
  <xs:sequence>
    <xs:element name="spatialInformation" type="tns:spatialInformationType"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:element name="nonSpatiallyDefinible" type="xs:boolean" fixed="true"/>
</xs:choice>
<!-- optional fields -->
<xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
<xs:element name="priority" type="xs:integer" minOccurs="0"/>
<xs:element name="lang" type="xs:language" minOccurs="0"/>
<xs:element name="content" type="xs:string" minOccurs="0"/>
<xs:element name="switched" type="xs:boolean" minOccurs="0"/>
<xs:element name="dynamic" type="xs:boolean" minOccurs="0"/>
<xs:element name="composed" type="xs:boolean" minOccurs="0"/>
<xs:element name="maxCaptureEncodings" type="xs:unsignedInt"
  minOccurs="0"/>
<!-- this is in place of "supplementary info": -->
<xs:element name="relatedTo" type="xs:IDREF" minOccurs="0"/>
<xs:any namespace="##other" processContents="lax" minOccurs="0"
  maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="captureID" type="xs:ID" use="required"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- SPATIAL INFORMATION TYPE -->
<xs:complexType name="spatialInformationType">
  <xs:sequence>
    <xs:element name="capturePoint" type="capturePointType"/>
    <xs:element name="captureArea" type="captureAreaType"
      minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- TEXT CAPTURE TYPE -->
<xs:complexType name="textCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- AUDIO CAPTURE TYPE -->
```

```
<xs:complexType name="audioCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
      <xs:sequence>
        <xs:element name="audioChannelFormat" type="audioChannelFormatType"
          minOccurs="0"/>
        <xs:element name="micPattern" type="tns:micPatternType"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- MIC PATTERN TYPE -->
<xs:simpleType name="micPatternType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="uni"/>
    <xs:enumeration value="shotgun"/>
    <xs:enumeration value="omni"/>
    <xs:enumeration value="figure8"/>
    <xs:enumeration value="cardioid"/>
    <xs:enumeration value="hyper-cardioid"/>
  </xs:restriction>
</xs:simpleType>

<!-- AUDIO CHANNEL FORMAT TYPE -->
<xs:simpleType name="audioChannelFormatType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="mono"/>
    <xs:enumeration value="stereo"/>
  </xs:restriction>
</xs:simpleType>

<!-- VIDEO CAPTURE TYPE -->
<xs:complexType name="videoCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
      <xs:sequence>
        <xs:element name="nativeAspectRatio" type="xs:string"
          minOccurs="0"/>
        <xs:element ref="embeddedText" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- EMBEDDED TEXT ELEMENT -->
<xs:element name="embeddedText">
```

```
<xs:complexType>
  <xs:simpleContent>
    <xs:extension base="xs:boolean">
      <xs:attribute name="lang" type="xs:language"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>

<!-- CAPTURE SCENES TYPE -->
<!-- envelope of capture scenes -->
<xs:complexType name="captureScenesType">
  <xs:sequence>
    <xs:element name="captureScene" type="captureSceneType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- CAPTURE SCENE TYPE -->
<xs:complexType name="captureSceneType">
  <xs:sequence>
    <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sceneSpace" type="captureSpaceType" minOccurs="0"/>
    <xs:element name="sceneEntries" type="sceneEntriesType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="sceneID" type="xs:ID" use="required"/>
  <xs:attribute name="scale" type="scaleType" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- SCALE TYPE -->
<xs:simpleType name="scaleType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="millimeters"/>
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="noscale"/>
  </xs:restriction>
</xs:simpleType>

<!-- CAPTURE AREA TYPE -->
<xs:complexType name="captureAreaType">
  <xs:sequence>
    <xs:element name="bottomLeft" type="pointType"/>
    <xs:element name="bottomRight" type="pointType"/>
    <xs:element name="topLeft" type="pointType"/>
    <xs:element name="topRight" type="pointType"/>
  </xs:sequence>
</xs:complexType>
```

```
</xs:sequence>
</xs:complexType>

<!-- CAPTURE SPACE TYPE -->
<xs:complexType name="captureSpaceType">
  <xs:sequence>
    <xs:element name="bottomLeftFront" type="pointType"/>
    <xs:element name="bottomRightFront" type="pointType"/>
    <xs:element name="topLeftFront" type="pointType"/>
    <xs:element name="topRightFront" type="pointType"/>
    <xs:element name="bottomLeftBack" type="pointType"/>
    <xs:element name="bottomRightBack" type="pointType"/>
    <xs:element name="topLeftBack" type="pointType"/>
    <xs:element name="topRightBack" type="pointType"/>
  </xs:sequence>
</xs:complexType>

<!-- POINT TYPE -->
<xs:complexType name="pointType">
  <xs:sequence>
    <xs:element name="x" type="xs:decimal"/>
    <xs:element name="y" type="xs:decimal"/>
    <xs:element name="z" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>

<!-- CAPTURE POINT TYPE -->
<xs:complexType name="capturePointType">
  <xs:complexContent>
    <xs:extension base="pointType">
      <xs:sequence>
        <xs:element name="lineOfCapturePoint" type="tns:pointType"
          minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="pointID" type="xs:ID"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- SCENE ENTRIES TYPE -->
<!-- envelope of scene entries of a capture scene -->
<xs:complexType name="sceneEntriesType">
  <xs:sequence>
    <xs:element name="sceneEntry" type="sceneEntryType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

```
<!-- SCENE ENTRY TYPE -->
<xs:complexType name="sceneEntryType">
  <xs:sequence>
    <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="switchingPolicies" type="switchingPoliciesType"
      minOccurs="0"/>
    <xs:element name="mediaCaptureIDs" type="captureIDListType"/>
  </xs:sequence>
  <xs:attribute name="sceneEntryID" type="xs:ID" use="required"/>
  <xs:attribute name="mediaType" type="xs:string" use="required"/>
</xs:complexType>

<!-- SWITCHING POLICIES TYPE -->
<xs:complexType name="switchingPoliciesType">
  <xs:sequence>
    <xs:element name="siteSwitching" type="xs:boolean" minOccurs="0"/>
    <xs:element name="segmentSwitching" type="xs:boolean"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- CAPTURE ID LIST TYPE -->
<xs:complexType name="captureIDListType">
  <xs:sequence>
    <xs:element name="captureIDREF" type="xs:IDREF"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- ENCODINGS TYPE -->
<xs:complexType name="encodingsType">
  <xs:sequence>
    <xs:element name="encoding" type="encodingType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- ENCODING TYPE -->
<xs:complexType name="encodingType" abstract="true">
  <xs:sequence>
    <xs:element name="encodingName" type="xs:string"/>
    <xs:element name="maxBandwidth" type="xs:integer"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="encodingID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
```

```
<!-- AUDIO ENCODING TYPE -->
<xs:complexType name="audioEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="audio"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- VIDEO ENCODING TYPE -->
<xs:complexType name="videoEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="video"
          minOccurs="0"/>
        <xs:element name="maxWidth" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxHeight" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxFrameRate" type="xs:integer" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- H26X ENCODING TYPE -->
<xs:complexType name="h26XEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:videoEncodingType">
      <xs:sequence>
        <!-- max number of pixels to be processed per second -->
        <xs:element name="maxH26Xpps" type="xs:integer"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- ENCODING GROUPS TYPE -->
<xs:complexType name="encodingGroupsType">
  <xs:sequence>
    <xs:element name="encodingGroup" type="tns:encodingGroupType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```



```
<!-- ENCODING GROUP TYPE -->
<xs:complexType name="encodingGroupType">
  <xs:sequence>
    <xs:element name="maxGroupBandwidth" type="xs:integer"/>
    <xs:element name="maxGroupPps" type="xs:integer"
      minOccurs="0"/>
    <xs:element name="encodingIDList" type="encodingIDListType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="encodingGroupID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>

<!-- ENCODING ID LIST TYPE -->
<xs:complexType name="encodingIDListType">
  <xs:sequence>
    <xs:element name="encIDREF" type="xs:IDREF" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- SIMULTANEOUS SETS TYPE -->
<xs:complexType name="simultaneousSetsType">
  <xs:sequence>
    <xs:element name="simultaneousSet" type="simultaneousSetType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- SIMULTANEOUS SET TYPE -->
<xs:complexType name="simultaneousSetType">
  <xs:sequence>
    <xs:element name="captureIDREF" type="xs:IDREF"
      minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sceneEntryIDREF" type="xs:IDREF"
      minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- CAPTURE ENCODING TYPE -->
<xs:complexType name="captureEncodingType">
  <xs:sequence>
    <xs:element name="mediaCaptureID" type="xs:string"/>
    <xs:element name="encodingID" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<!-- CAPTURE ENCODINGS TYPE -->
```

```
<xs:complexType name="captureEncodingsType">
  <xs:sequence>
    <xs:element name="captureEncoding" type="captureEncodingType"
      maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- CLUE INFO ELEMENT -->
<!-- the <clueInfo> envelope can be seen
      as the ancestor of an <advertisement> envelope -->
<xs:element name="clueInfo" type="clueInfoType"/>

<!-- CLUE INFO TYPE -->
<xs:complexType name="clueInfoType">
  <xs:sequence>
    <xs:element ref="mediaCaptures"/>
    <xs:element ref="encodings"/>
    <xs:element ref="encodingGroups"/>
    <xs:element ref="captureScenes"/>
    <xs:element ref="simultaneousSets"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="clueInfoID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

</xs:schema>
```

Following sections describe the XML schema in more detail.

4. <mediaCaptures>

<mediaCaptures> represents the list of one or more media captures available on the media provider's side. Each media capture is represented by a <mediaCapture> element (Section 10).

5. <encodings>

<encodings> represents the list of individual encodings available on the media provider's side. Each individual encoding is represented by an <encoding> element (Section 16).

6. <encodingGroups>

<encodingGroups> represents the list of the encoding groups organized on the media provider's side. Each encoding group is represented by

a `<encodingGroup>` element (Section 20).

7. `<captureScenes>`

`<captureScenes>` represents the list of the capture scenes organized on the media provider's side. Each capture scene is represented by a `<captureScene>` element. (Section 14).

8. `<simultaneousSets>`

`<simultaneousSets>` contains the simultaneous sets indicated by the media provider. Each simultaneous set is represented by a `<simultaneousSet>` element. (Section 21).

9. `<captureEncodings>`

`<captureEncodings>` is a list of capture encodings. It can represent the list of the desired capture encodings indicated by the media consumer or the list of instantiated captures on the provider's side. Each capture encoding is represented by a `<captureEncoding>` element. (Section 22).

10. `<mediaCapture>`

According to the CLUE framework, a media capture is the fundamental representation of a media flow that is available on the provider's side. Media captures are characterized with a set of features that are independent from the specific type of medium, and with a set of features that are media-specific. We design the media capture type as an abstract type, providing all the features that can be common to all media types. Media-specific captures, such as video captures, audio captures and others, are specialization of that media capture type, as in a typical generalization-specialization hierarchy.

The following is the XML Schema definition of the media capture type:

```

<!-- MEDIA CAPTURE TYPE -->
<xs:complexType name="mediaCaptureType" abstract="true">
  <xs:sequence>
    <!-- mandatory fields -->
    <xs:element name="capturedMedia" type="xs:string"/>
    <xs:element name="captureSceneIDREF" type="xs:IDREF"/>
    <xs:element name="encGroupIDREF" type="xs:IDREF"/>
    <xs:choice>
      <xs:sequence>
        <xs:element name="spatialInformation" type="tns:spatialInformationType"
          maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:element name="nonSpatiallyDefinible" type="xs:boolean" fixed="true"/>
    </xs:choice>
    <!-- optional fields -->
    <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="priority" type="xs:integer" minOccurs="0"/>
    <xs:element name="lang" type="xs:language" minOccurs="0"/>
    <xs:element name="content" type="xs:string" minOccurs="0"/>
    <xs:element name="switched" type="xs:boolean" minOccurs="0"/>
    <xs:element name="dynamic" type="xs:boolean" minOccurs="0"/>
    <xs:element name="composed" type="xs:boolean" minOccurs="0"/>
    <xs:element name="maxCaptureEncodings" type="xs:unsignedInt"
      minOccurs="0"/>
    <!-- this is in place of "supplementary info": -->
    <xs:element name="relatedTo" type="xs:IDREF" minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="captureID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

```

10.1. <capturedMedia>

<capturedMedia> is a mandatory field specifying the media type of the capture ("audio", "video", "text", ...).

10.2. <captureSceneIDREF>

<captureSceneIDREF> is a mandatory field containing the identifier of the capture scene the media capture belongs to. Indeed, each media capture must be associated with one and only capture scene. When a media capture is spatially definible, some spatial information is provided along with it in the form of point coordinates (see Section 10.4). Such coordinates refers to the space of coordinates defined for the capture scene containing the capture.

10.3. <encGroupIDREF>

<encGroupIDREF> is a mandatory field containing the identifier of the encoding group the media capture is associated with.

10.4. <spatialInformation>

Media captures are divided into two categories: non spatially definible captures and spatially definible captures.

Non spatially definible captures are those that do not capture parts of the telepresence room. Capture of this case are for example those related to registrations, text captures, DVDs, registered presentation, or external streams, that are played in the telepresence room and transmitted to remote sites.

Spatially definible captures are those that capture part of the telepresence room. The captured part of the telepresence room is described by means of the <spatialInformation> element.

This is the definition of the spatial information type:

```
<!-- SPATIAL INFORMATION TYPE -->
<xs:complexType name="spatialInformationType">
  <xs:sequence>
    <xs:element name="capturePoint" type="capturePointType"/>
    <xs:element name="captureArea" type="captureAreaType"
      minOccurs="0"/>
    <xs:any namespace="##other" processContents="lax"
      minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other"
    processContents="lax"/>
</xs:complexType>
```

The <capturePoint> contains the coordinates of the capture device that is taking the capture, as well as, optionally, the pointing direction (see Section 10.4.1). It is a mandatory field when the media capture is spatially definible, independently from the media type.

The <captureArea> is an optional field containing four points defining the captured area represented by the capture (see Section 10.4.2).

10.4.1.1. <capturePoint>

The <capturePoint> element is used to represent the position and the line of capture of a capture device. The XML Schema definition of the <capturePoint> element type is the following:

```
<!-- CAPTURE POINT TYPE -->
<xs:complexType name="capturePointType">
  <xs:complexContent>
    <xs:extension base="pointType">
      <xs:sequence>
        <xs:element name="lineOfCapturePoint"
          type="tns:pointType"
          minOccurs="0"/>
      </xs:sequence>
      <xs:attribute name="pointID" type="xs:ID"/>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- POINT TYPE -->
<xs:complexType name="pointType">
  <xs:sequence>
    <xs:element name="x" type="xs:decimal"/>
    <xs:element name="y" type="xs:decimal"/>
    <xs:element name="z" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>
```

The point type contains three spatial coordinates ("x","y","z") representing a point in the space associated with a certain capture scene.

The capture point type extends the point type, i.e., it is represented by three coordinates identifying the position of the capture device, but can add further information. Such further information is conveyed by the <lineOfCapturePoint>, which is another point-type element representing the "point on line of capture", that gives the pointing direction of the capture device.

If the point of capture is not specified, it means the consumer should not assume anything about the spatial location of the capturing device.

The coordinates of the point on line of capture MUST NOT be identical

to the capture point coordinates. If the point on line of capture is not specified, no assumptions are made about the axis of the capturing device.

10.4.2. <captureArea>

<captureArea> is an optional element that can be contained within the spatial information associated with a media capture. It represents the spatial area captured by the media capture.

The XML representation of that area is provided through a set of four point-type element, <bottomLeft>, <bottomRight>, <topLeft>, and <topRight>, as it can be seen from the following definition:

```
<!-- CAPTURE AREA TYPE -->
<xs:complexType name="captureAreaType">
  <xs:sequence>
    <xs:element name="bottomLeft" type="pointType"/>
    <xs:element name="bottomRight" type="pointType"/>
    <xs:element name="topLeft" type="pointType"/>
    <xs:element name="topRight" type="pointType"/>
  </xs:sequence>
</xs:complexType>
```

<bottomLeft>, <bottomRight>, <topLeft>, and <topRight> should be coplanar.

For a switched capture that switches between different sections within a larger area, the area of capture should use coordinates for the larger potential area.

By comparing the capture area of different media captures within the same capture scene, a consumer can determine the spatial relationships between them and render them correctly. If the area of capture is not specified, it means the Media Capture is not spatially related to any other media capture.

10.5. <nonSpatiallyDefinible>

When media captures are non spatially definible, they are marked with the boolean <nonSpatiallyDefinible> element set to "true".

10.6. <description>

<description> is used to provide optionally human-readable textual information. It is used to describe media captures, capture scenes and capture scene entries. A media capture can be described by using multiple <description> elements, each one providing information in a different language. Indeed, the <description> element definition is the following:

```
<!-- DESCRIPTION element -->
<xs:element name="description">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:string">
        <xs:attribute name="lang" type="xs:language"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

As it can be seen, <description> is a string element with an attribute ("lang") indicating the language used in the textual description.

10.7. <priority>

<priority> ([I-D.groves-clue-capture-attr]) is an optional integer field indicating the importance of a media capture according to the media provider's perspective. It can be used on the receiver's side to automatically identify the most "important" contribution available from the media provider.

[edt note: no final consensus has been reached on the adoption of such media capture attribute.]

10.8. <lang>

<lang> is an optional element containing the language used in the capture, if any. The purpose of the element could match the one of the "language" attribute proposed in [I-D.groves-clue-capture-attr].

10.9. <content>

<content> is an optional string element. It contains enumerated values describing the "role" of the media capture according to what

is envisioned in [RFC4796] ("slides", "speaker", "sl", "main", "alt"). The values for this attribute are the same as the media content values for the content attribute in [RFC4796]. This attribute can list multiple values, for example "main, speaker".

[edt note: a better XML Schema definition for that element will soon be defined.]

10.10. <switched>

<switched> is a boolean element which indicates whether or not the media capture represents the most appropriate subset of a "whole". What is "most appropriate" is up to the provider and could be the active speaker, a lecturer or a VIP.

[edt note: :{}]

10.11. <dynamic>

<dynamic> is an optional boolean element indicating whether or not the capture device originating the capture moves during the telepresence session. That optional boolean element has the same purpose of the dynamic attribute proposed in [I-D.groves-clue-capture-attr].

[edt note: There isn't yet final consensus about that element.]

10.12. <composed>

<composed> is an optional boolean element indicating whether or not the media capture is a mix (audio) or composition (video) of streams. This attribute is useful for a media consumer for example to avoid nesting a composed video capture into another composed capture or rendering.

10.13. <maxCaptureEncodings>

The optional <maxCaptureEncodings> contains an unsigned integer indicating the maximum number of capture encodings that can be simultaneously active for the media capture. If absent, this parameter defaults to 1. The minimum value for this attribute is 1. The number of simultaneous capture encodings is also limited by the restrictions of the encoding group the media capture refers to by means of the <encGroupIDREF> element.

10.14. <relatedTo>

The optional <relatedTo> element contains the value of the ID attribute of the media capture it refers to. The media capture

marked with a <relatedTo> element can be for example the translation of a main media capture in a different language. The <relatedTo> element could be interpreted the same manner of the supplementary information attribute proposed in [I-D.groves-clue-capture-attr] and further discussed in <http://www.ietf.org/mail-archive/web/clue/current/msg02238.html>.

[edt note: There isn't yet final consensus about that element.]

10.15. captureID attribute

The "captureID" attribute is a mandatory field containing the identifier of the media capture.

11. Audio captures

Audio captures inherit all the features of a generic media capture and present further audio-specific characteristics. The XML Schema definition of the audio capture type is reported below:

```
<!-- AUDIO CAPTURE TYPE -->
<xs:complexType name="audioCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
      <xs:sequence>
        <xs:element name="audioChannelFormat" type="audioChannelFormatType"
          minOccurs="0"/>
        <xs:element name="micPattern" type="tns:micPatternType"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Audio-specific information about the audio capture is contained in <audioChannelFormat> (Section 11.1) and in <micPattern> (Section 11.2).

11.1. <audioChannelFormat>

The optional <audioChannelFormat> element is a field with enumerated values ("mono" and "stereo") which describes the method of encoding used for audio. A value of "mono" means the audio capture has one channel. A value of "stereo" means the audio capture has two audio channels, left and right. A single stereo capture is different from

two mono captures that have a left-right spatial relationship. A stereo capture maps to a single RTP stream, while each mono audio capture maps to a separate RTP stream.

The XML Schema definition of the `<audioChannelFormat>` element type is provided below:

```
<!-- AUDIO CHANNEL FORMAT TYPE -->
<xs:simpleType name="audioChannelFormatType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="mono"/>
    <xs:enumeration value="stereo"/>
  </xs:restriction>
</xs:simpleType>
```

11.2. `<micPattern>`

The `<micPattern>` element is an optional field describing the characteristic of the mic capturing the audio signal. It contains the enumerated values listed below:

```
<!-- MIC PATTERN TYPE -->
<xs:simpleType name="micPatternType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="uni"/>
    <xs:enumeration value="shotgun"/>
    <xs:enumeration value="omni"/>
    <xs:enumeration value="figure8"/>
    <xs:enumeration value="cardioid"/>
    <xs:enumeration value="hyper-cardioid"/>
  </xs:restriction>
</xs:simpleType>
```

12. Video captures

Video captures, similarly to audio captures, extend the information of a generic media capture with video-specific features, such as `<nativeAspectRatio>` (Section 12.1) and `<embeddedText>` (Section 12.2).

The XML Schema representation of the video capture type is provided in the following:

```
<!-- VIDEO CAPTURE TYPE -->
<xs:complexType name="videoCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
      <xs:sequence>
        <xs:element name="nativeAspectRatio" type="xs:string"
          minOccurs="0"/>
        <xs:element ref="embeddedText" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

12.1. <nativeAspectRatio>

If a video capture has a native aspect ratio (for instance, it corresponds to a camera that generates 4:3 video), then it can be supplied as a value of the <nativeAspectRatio> element, in order to help rendering.

12.2. <embeddedText>

The <embeddedText> element is a boolean element indicating that there is text embedded in the video capture. The language used in such embedded textual description is reported in <embeddedText> "lang" attribute.

The XML Schema definition of the <embeddedText> element is:

```
<!-- EMBEDDED TEXT ELEMENT -->
<xs:element name="embeddedText">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:boolean">
        <xs:attribute name="lang" type="xs:language"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
```

The <embeddedText> element could correspond to the embedded-text attribute introduced in [I-D.groves-clue-capture-attr]

[edt note: no final consensus has been reached yet about the adoption of such element]

13. Text captures

Also text captures can be described by extending the generic media capture information, similarly to audio captures and video captures.

The XML Schema representation of the text capture type is currently lacking text-specific information, as it can be seen by looking at the definition below:

```
<!-- TEXT CAPTURE TYPE -->
<xs:complexType name="textCaptureType">
  <xs:complexContent>
    <xs:extension base="tns:mediaCaptureType">
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

14. <captureScene>

A media provider organizes the available capture in capture scenes in order to help the receiver both in the rendering and in the selection of the group of captures. Capture scenes are made of capture scene entries, that are set of media captures of the same media type. Each capture scene entry represents an alternative to represent completely a capture scene for a fixed media type.

The XML Schema representation of a <captureScene> element is the following:

```
<!-- CAPTURE SCENE TYPE -->
<xs:complexType name="captureSceneType">
  <xs:sequence>
    <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sceneSpace" type="captureSpaceType" minOccurs="0"/>
    <xs:element name="sceneEntries" type="sceneEntriesType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="sceneID" type="xs:ID" use="required"/>
  <xs:attribute name="scale" type="scaleType" use="required"/>
```

```
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

The <captureScene> element can contain zero or more textual <description> elements, defined as in Section 10.6. Besides <description>, there are two other fields: <sceneSpace> (Section 14.1), describing the coordinate space which the media captures of the capture scene refer to, and <sceneEntries> (Section 14.2), the list of the capture scene entries.

14.1. <sceneSpace> (was:<sceneArea>)

The <sceneSpace> describes a bounding volume for the spatial information provided alongside spatially-definible media capture associated with the considered capture scene. Such volume is described as an arbitrary hexahedrons with eight points (<bottomLeftFront>, <bottomRightFront>, <topLeftFront>, <topRightFront>, <bottomLeftBack>, <bottomRightBack>, <topLeftBack>, and <topRightBack>). The coordinate system is Cartesian X, Y, Z with the origin at a spatial location of the media provider's choosing. The media provider must use the same coordinate system with same scale and origin for all media capture coordinates within the same capture scene.

```
<!-- CAPTURE SPACE TYPE -->
<xs:complexType name="captureSpaceType">
  <xs:sequence>
    <xs:element name="bottomLeftFront" type="pointType"/>
    <xs:element name="bottomRightFront" type="pointType"/>
    <xs:element name="topLeftFront" type="pointType"/>
    <xs:element name="topRightFront" type="pointType"/>
    <xs:element name="bottomLeftBack" type="pointType"/>
    <xs:element name="bottomRightBack" type="pointType"/>
    <xs:element name="topLeftBack" type="pointType"/>
    <xs:element name="topRightBack" type="pointType"/>
  </xs:sequence>
</xs:complexType>
```

[edt note: this is just a place holder, the definition of the bounding volume has to be discussed]

14.2. <sceneEntries>

The <sceneEntries> element is a mandatory field of a capture scene containing the list of scene entries. Each scene entry is represented by a <sceneEntry> element (Section 15).

```
<!-- SCENE ENTRIES TYPE -->
<!-- envelope of scene entries of a capture scene -->
<xs:complexType name="sceneEntriesType">
  <xs:sequence>
    <xs:element name="sceneEntry" type="sceneEntryType"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

14.3. sceneID attribute

The sceneID attribute is a mandatory attribute containing the identifier of the capture scene.

14.4. scale attribute

The scale attribute is a mandatory attribute that specifies the scale of the coordinates provided in the capture space and in the spatial information of the media capture belonging to the considered capture scene. The scale attribute can assume three different values:

"millimeters" - the scale is in millimeters. Systems which know their physical dimensions (for example professionally installed telepresence room systems) should always provide those real-world measurements.

"unknown" - the scale is not necessarily millimeters, but the scale is the same for every media capture in the capture scene. Systems which don't know specific physical dimensions but still know relative distances should select "unknown" in the scale attribute of the capture scene to be described.

"noscale" - there is no a common physical scale among the media captures of the capture scene. That means the scale could be different for each media capture.

```
<!-- SCALE TYPE -->
<xs:simpleType name="scaleType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="millimeters"/>
    <xs:enumeration value="unknown"/>
    <xs:enumeration value="noscale"/>
  </xs:restriction>
</xs:simpleType>
```

15. <sceneEntry>

A <sceneEntry> element represents a capture scene entry, which contains a set of media capture of the same media type describing a capture scene.

A <sceneEntry> element is characterized as follows.

```
<!-- SCENE ENTRY TYPE -->
<xs:complexType name="sceneEntryType">
  <xs:sequence>
    <xs:element ref="description" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="switchingPolicies" type="switchingPoliciesType"
      minOccurs="0"/>
    <xs:element name="mediaCaptureIDs" type="captureIDListType"/>
  </xs:sequence>
  <xs:attribute name="sceneEntryID" type="xs:ID" use="required"/>
  <xs:attribute name="mediaType" type="xs:string" use="required"/>
</xs:complexType>
```

One or more optional <description> elements provide human-readable information about what the scene entry contains. <description> is defined as already seen in Section 10.6.

The remaining child elements are described in the following subsections.

15.1. <switchingPolicies>

<switchingPolicies> represents the switching policies the media provider support for the media captures contained inside a scene entry. The <switchingPolicies> element contains two boolean elements:

<siteSwitching>: if set to "true", it means that the media provider supports the site switching policy for the included media captures;

<segmentSwitching>: if set to "true", it means that the media provider supports the segment switching policy for the included media captures.

The "site-switch" policy means all captures are switched at the same time to keep captures from the same endpoint site together.

The "segment-switch" policy means different captures can switch at different times, and can be coming from different endpoints.

```
<!-- SWITCHING POLICIES TYPE -->
<xs:complexType name="switchingPoliciesType">
  <xs:sequence>
    <xs:element name="siteSwitching" type="xs:boolean" minOccurs="0"/>
    <xs:element name="segmentSwitching" type="xs:boolean"
      minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
```

15.2. <mediaCaptureIDs>

The <mediaCaptureIDs> is the list of the identifiers of the media captures included in the scene entry. It is an element of the captureIDListType type, which is defined as a sequence of <captureIDREF> each one containing the identifier of a media capture listed within the <mediaCaptures> element:

```
<!-- CAPTURE ID LIST TYPE -->
<xs:complexType name="captureIDListType">
  <xs:sequence>
    <xs:element name="captureIDREF" type="xs:IDREF"
      maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

15.3. sceneEntryID attribute

The sceneEntryID attribute is a mandatory attribute containing the identifier of the capture scene entry represented by the <sceneEntry> element.

15.4. mediaType attribute

The mediaType attribute contains the media type of the media captures included in the scene entry.

16. <encoding>

The <encoding> element represents an individual encoding, i.e., a way to encode a media capture. Individual encodings can be characterized with features that are independent from the specific type of medium, and with features that are media-specific. We design the individual encoding type as an abstract type, providing all the features that can be common to all media types. Media-specific individual encodings, such as video encodings, audio encodings and others, are specialization of that type, as in a typical generalization-specialization hierarchy.

```
<!-- ENCODING TYPE -->
<xs:complexType name="encodingType" abstract="true">
  <xs:sequence>
    <xs:element name="encodingName" type="xs:string"/>
    <xs:element name="maxBandwidth" type="xs:integer"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="encodingID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
```

16.1. <encodingName>

<encodingName> is a mandatory field containing the name of the encoding (e.g., G711, H264, ...).

16.2. <maxBandwidth>

<maxBandwidth> represent the maximum bitrate the media provider can instantiate for that encoding.

16.3. encodingID attribute

The encodingID attribute is a mandatory attribute containing the identifier of the individual encoding.

17. Audio encodings

Audio encodings inherit all the features of a generic individual encoding and can present further audio-specific encoding characteristics. The XML Schema definition of the audio encoding type is reported below:

```
<!-- AUDIO ENCODING TYPE -->
<xs:complexType name="audioEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="audio"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Up to now the only audio-specific information is the <encodedMedia> element containing the media type of the media captures that can be encoded with the considered individual encoding. In the case of audio encoding, that element is forced to "audio".

18. Video encodings

Similarly to audio encodings, video encodings can extend the information of a generic individual encoding with video-specific encoding features, such as <maxWidth>, <maxHeight> and <maxFrameRate>.

The <encodedMedia> element contains the media type of the media captures that can be encoded with the considered individual encoding. In the case of video encoding, that element is forced to "video".

```
<!-- VIDEO ENCODING TYPE -->
<xs:complexType name="videoEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:encodingType">
      <xs:sequence>
        <xs:element name="encodedMedia" type="xs:string" fixed="video"
          minOccurs="0"/>
        <xs:element name="maxWidth" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxHeight" type="xs:integer" minOccurs="0"/>
        <xs:element name="maxFrameRate" type="xs:integer" minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

18.1. <maxWidth>

<maxWidth> represents the video resolution's maximum width supported by the video encoding, expressed in pixels.

[edt note: not present in -09 version of the framework doc]

18.2. <maxHeight>

<maxHeight> representd the video resolution's maximum heith supported by the video encoding, expressed in pixels.

[edt note: not present in -09 version of the framework doc]

18.3. <maxFrameRate>

<maxFrameRate> provides the maximum frame rate supported by the video encoding for the video capture to be encoded.

[edt note: not present in -09 version of the framework doc]

19. H26X encodings

This is an example of how it is possible to further specialize the definition of a video individual encoding in order to cover encoding specific information. A H26X video encoding can be represented through an element inheriting the video encoding characteristics described above (Section 18) and by adding other information such as <maxH26Xpps>, which represent the maximum number of pixels to be processed per second;.

```
<!-- H26X ENCODING TYPE -->
<xs:complexType name="h26XEncodingType">
  <xs:complexContent>
    <xs:extension base="tns:videoEncodingType">
      <xs:sequence>
        <!-- max number of pixels to be processed per second -->
        <xs:element name="maxH26Xpps" type="xs:integer"
          minOccurs="0"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

[edt note: Need to be checked]

20. <encodingGroup>

The <encodingGroup> element represents an encoding group, which is a set of one or more individual encodings, and parameters that apply to the group as a whole. The definition of the <encodingGroup> element is the following:

```
<!-- ENCODING GROUP TYPE -->
<xs:complexType name="encodingGroupType">
  <xs:sequence>
    <xs:element name="maxGroupBandwidth" type="xs:integer"/>
    <xs:element name="maxGroupPps" type="xs:integer"
      minOccurs="0"/>
    <xs:element name="encodingIDList" type="encodingIDListType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="encodingGroupID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
```

In the following, the contained elements are further described.

20.1. <maxGroupBandwidth>

<maxGroupBandwidth> is an optional field containing the maximum bitrate supported for all the individual encodings included in the encoding group.

20.2. <maxGroupPps>

<maxGroupPps> is an optional field containing the maximum number of pixel per second for all the individual encodings included in the encoding group.

[edt note: Need to be checked]

20.3. <encodingIDList>

<maxGroupBandwidth> is the list of the individual encoding grouped together. Each individual encoding is represented through its identifier contained within an <encIDREF> element.

```
<!-- ENCODING ID LIST TYPE -->
<xs:complexType name="encodingIDListType">
  <xs:sequence>
    <xs:element name="encIDREF" type="xs:IDREF" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

20.4. encodingGroupID attribute

The encodingGroupID attribute contains the identifier of the encoding group.

21. <simultaneousSet>

<simultaneousSet> represents a simultaneous set, i.e. a list of capture of the same type that cab be transmitted at the same time by a media provider. There are different simultaneous transmission sets for each media type.

```
<!-- SIMULTANEOUS SET TYPE -->
<xs:complexType name="simultaneousSetType">
  <xs:sequence>
    <xs:element name="captureIDREF" type="xs:IDREF"
      minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="sceneEntryIDREF" type="xs:IDREF"
      minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
```

[edt note: need to be checked]

21.1. <captureIDREF>

<captureIDREF> contains the identifier of the media capture that belongs to the simultaneous set.

21.2. <sceneEntryIDREF>

<captureIDREF> contains the identifier of the scene entry containing a group of capture that are able to be sent simultaneously with the other capture of the simultaneous set.

22. <captureEncoding>

A <captureEncoding> is given from the association of a media capture and an individual encoding, to form a capture stream. It is defined as an element of the following type:

```
<!-- CAPTURE ENCODING TYPE -->
<xs:complexType name="captureEncodingType">
  <xs:sequence>
    <xs:element name="mediaCaptureID" type="xs:string"/>
    <xs:element name="encodingID" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

22.1. <mediaCaptureID>

<mediaCaptureID> contains the identifier of the media capture that has been encoded to form the capture encoding.

22.2. <encodingID>

<encodingID> contains the identifier of the applied individual encoding.

23. <clueInfo>

The <clueInfo> element has been left within the XML Schema for the sake of convenience when representing a prototype of ADVERTISEMENT message (see the example section).

```
<!-- CLUE INFO ELEMENT -->
<!-- the <clueInfo> envelope can be seen
      as the ancestor of an <advertisement> envelope -->
<xs:element name="clueInfo" type="clueInfoType"/>

<!-- CLUE INFO TYPE -->
<xs:complexType name="clueInfoType">
  <xs:sequence>
    <xs:element ref="mediaCaptures"/>
    <xs:element ref="encodings"/>
    <xs:element ref="encodingGroups"/>
    <xs:element ref="captureScenes"/>
    <xs:element ref="simultaneousSets"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0"
      maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="clueInfoID" type="xs:ID" use="required"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

24. Sample XML file

The following XML document represents a schema compliant example of a CLUE telepresence scenario.

There are 5 video captures:

VC0: the video from the left camera

VC1: the video from the central camera

VC2: the video from the right camera

VC3: the overall view of the telepresence room taken from the
central camera

VC4: the video associated with the slide stream

There are 2 audio captures:

AC0: the overall room audio taken from the central camera

AC1: the audio associated with the slide stream presentation

The captures are organized into two capture scenes:

CS1: this scene contains captures associated with the participants that are in the telepresence room.

CS2: this scene contains captures associated with the slide presentation, which is a pre-registered presentation played within the context of the telepresence session.

Within the capture scene CS1, there are three scene entries available:

CS1_SE1: this entry contains the participants' video captures taken from the three cameras (VC0, VC1, VC2).

CS1_SE2: this entry contains the zoomed-out view of the overall telepresence room (VC3)

CS1_SE3: this entry contains the overall telepresence room audio (AC0)

On the other hand, capture scene CS2 presents two scene entries:

CS2_SE1: this entry contains the presentation audio stream (AC1)

CS2_SE2: this entry contains the presentation video stream (VC4)

There are two encoding groups:

EG0 This encoding groups involves video encodings ENC0, ENC1, ENC2

EG1 This encoding groups involves audio encodings ENC3, ENC4

As to the simultaneous sets, only VC1 and VC3 cannot be transmitted simultaneously since they are captured by the same device. i.e. the central camera (VC3 is a zoomed-out view while VC1 is a focused view of the front participants). The simultaneous sets would then be the following:

SS1 made by VC0, VC1, VC2, VC4, AC0, AC1

SS2 made by VC0, VC3, VC2, VC4, AC0, AC1

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<clueInfo xmlns="urn:ietf:params:xml:ns:clue-info" clueInfoID="prova">
  <mediaCaptures>
    <mediaCapture
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:type="audioCaptureType" captureID="AC1">
  <capturedMedia>audio</capturedMedia>
  <captureSceneIDREF>CS2</captureSceneIDREF>
  <encGroupIDREF>EG1</encGroupIDREF>
  <nonSpatiallyDefinible>true</nonSpatiallyDefinible>
  <description lang="en">presentation audio</description>
  <content>slide</content>
  <audioChannelFormat>mono</audioChannelFormat>
</mediaCapture>
<mediaCapture
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="videoCaptureType" captureID="VC4">
  <capturedMedia>video</capturedMedia>
  <captureSceneIDREF>CS2</captureSceneIDREF>
  <encGroupIDREF>EG0</encGroupIDREF>
  <nonSpatiallyDefinible>true</nonSpatiallyDefinible>
  <description lang="en">presentation video</description>
  <content>slides</content>
</mediaCapture>
<mediaCapture
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="audioCaptureType" captureID="AC0">
  <capturedMedia>audio</capturedMedia>
  <captureSceneIDREF>CS1</captureSceneIDREF>
  <encGroupIDREF>EG1</encGroupIDREF>
  <spatialInformation>
    <capturePoint>
      <x>0.5</x>
      <y>1.0</y>
      <z>0.5</z>
      <lineOfCapturePoint>
        <x>0.5</x>
        <y>0.0</y>
        <z>0.5</z>
      </lineOfCapturePoint>
    </capturePoint>
  </spatialInformation>
  <description lang="en">
    audio from the central camera mic</description>
  <audioChannelFormat>mono</audioChannelFormat>
  <micPattern>figure8</micPattern>
</mediaCapture>
<mediaCapture
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="videoCaptureType" captureID="VC3">
  <capturedMedia>video</capturedMedia>
  <captureSceneIDREF>CS1</captureSceneIDREF>
  <encGroupIDREF>EG0</encGroupIDREF>
```

```

    <spatialInformation>
      <capturePoint>
        <x>1.5</x>
        <y>1.0</y>
        <z>0.5</z>
        <lineOfCapturePoint>
          <x>1.5</x>
          <y>0.0</y>
          <z>0.5</z>
        </lineOfCapturePoint>
      </capturePoint>
      <captureArea>
        <bottomLeft>
          <x>0.0</x>
          <y>3.0</y>
          <z>0.0</z>
        </bottomLeft>
        <bottomRight>
          <x>3.0</x>
          <y>3.0</y>
          <z>0.0</z>
        </bottomRight>
        <topLeft>
          <x>0.0</x>
          <y>3.0</y>
          <z>3.0</z>
        </topLeft>
        <topRight>
          <x>3.0</x>
          <y>3.0</y>
          <z>3.0</z>
        </topRight>
      </captureArea>
    </spatialInformation>
    <description lang="en">
      zoomed out view of the room</description>
  </mediaCapture>
  <mediaCapture
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="videoCaptureType" captureID="VC2">
    <capturedMedia>video</capturedMedia>
    <captureSceneIDREF>CS1</captureSceneIDREF>
    <encGroupIDREF>EG0</encGroupIDREF>
    <spatialInformation>
      <capturePoint>
        <x>2.5</x>
        <y>1.0</y>
        <z>0.5</z>

```

```
        <lineOfCapturePoint>
          <x>2.5</x>
          <y>0.0</y>
          <z>0.5</z>
        </lineOfCapturePoint>
      </capturePoint>
    <captureArea>
      <bottomLeft>
        <x>2.0</x>
        <y>3.0</y>
        <z>0.0</z>
      </bottomLeft>
      <bottomRight>
        <x>3.0</x>
        <y>3.0</y>
        <z>0.0</z>
      </bottomRight>
      <topLeft>
        <x>2.0</x>
        <y>3.0</y>
        <z>3.0</z>
      </topLeft>
      <topRight>
        <x>3.0</x>
        <y>3.0</y>
        <z>3.0</z>
      </topRight>
    </captureArea>
  </spatialInformation>
  <description lang="en">right camera video</description>
</mediaCapture>
<mediaCapture
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="videoCaptureType" captureID="VC1">
  <capturedMedia>video</capturedMedia>
  <captureSceneIDREF>CS1</captureSceneIDREF>
  <encGroupIDREF>EG0</encGroupIDREF>
  <spatialInformation>
    <capturePoint>
      <x>1.5</x>
      <y>1.0</y>
      <z>0.5</z>
      <lineOfCapturePoint>
        <x>1.5</x>
        <y>0.0</y>
        <z>0.5</z>
      </lineOfCapturePoint>
    </capturePoint>
  </spatialInformation>
</mediaCapture>
```

```
<captureArea>
  <bottomLeft>
    <x>1.0</x>
    <y>3.0</y>
    <z>0.0</z>
  </bottomLeft>
  <bottomRight>
    <x>2.0</x>
    <y>3.0</y>
    <z>0.0</z>
  </bottomRight>
  <topLeft>
    <x>1.0</x>
    <y>3.0</y>
    <z>3.0</z>
  </topLeft>
  <topRight>
    <x>2.0</x>
    <y>3.0</y>
    <z>3.0</z>
  </topRight>
</captureArea>
</spatialInformation>
<description lang="en">central camera video</description>
</mediaCapture>
<mediaCapture
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="videoCaptureType" captureID="VC0">
  <capturedMedia>video</capturedMedia>
  <captureSceneIDREF>CS1</captureSceneIDREF>
  <encGroupIDREF>EG0</encGroupIDREF>
  <spatialInformation>
    <capturePoint>
      <x>0.5</x>
      <y>1.0</y>
      <z>0.5</z>
      <lineOfCapturePoint>
        <x>0.5</x>
        <y>0.0</y>
        <z>0.5</z>
      </lineOfCapturePoint>
    </capturePoint>
    <captureArea>
      <bottomLeft>
        <x>0.0</x>
        <y>3.0</y>
        <z>0.0</z>
      </bottomLeft>
```

```
        <bottomRight>
          <x>1.0</x>
          <y>3.0</y>
          <z>0.0</z>
        </bottomRight>
        <topLeft>
          <x>0.0</x>
          <y>3.0</y>
          <z>3.0</z>
        </topLeft>
        <topRight>
          <x>1.0</x>
          <y>3.0</y>
          <z>3.0</z>
        </topRight>
      </captureArea>
    </spatialInformation>
    <description lang="en">left camera video</description>
  </mediaCapture>
</mediaCaptures>
<encodings>
  <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="videoEncodingType" encodingID="ENC0">
    <encodingName>h263</encodingName>
    <maxBandwidth>4000000</maxBandwidth>
    <encodedMedia>video</encodedMedia>
    <maxWidth>1920</maxWidth>
    <maxHeight>1088</maxHeight>
  </encoding>
  <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="videoEncodingType" encodingID="ENC1">
    <encodingName>h263</encodingName>
    <maxBandwidth>4000000</maxBandwidth>
    <encodedMedia>video</encodedMedia>
    <maxWidth>1920</maxWidth>
    <maxHeight>1088</maxHeight>
  </encoding>
  <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="videoEncodingType" encodingID="ENC2">
    <encodingName>h263</encodingName>
    <maxBandwidth>4000000</maxBandwidth>
    <encodedMedia>video</encodedMedia>
    <maxWidth>1920</maxWidth>
    <maxHeight>1088</maxHeight>
  </encoding>
  <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="audioEncodingType" encodingID="ENC3">
    <encodingName>g711</encodingName>
```

```
        <maxBandwidth>64000</maxBandwidth>
        <encodedMedia>audio</encodedMedia>
    </encoding>
    <encoding xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="audioEncodingType" encodingID="ENC4">
        <encodingName>g711</encodingName>
        <maxBandwidth>64000</maxBandwidth>
        <encodedMedia>audio</encodedMedia>
    </encoding>
</encodings>
<encodingGroups>
    <encodingGroup encodingGroupID="EG0">
        <maxGroupBandwidth>12000000</maxGroupBandwidth>
        <encodingIDList>
            <encIDREF>ENC0</encIDREF>
            <encIDREF>ENC1</encIDREF>
            <encIDREF>ENC2</encIDREF>
        </encodingIDList>
    </encodingGroup>
    <encodingGroup encodingGroupID="EG1">
        <maxGroupBandwidth>12000000</maxGroupBandwidth>
        <encodingIDList>
            <encIDREF>ENC3</encIDREF>
            <encIDREF>ENC4</encIDREF>
        </encodingIDList>
    </encodingGroup>
</encodingGroups>
<captureScenes>
    <captureScene scale="unknown" sceneID="CS1">
        <description lang="en">main scene</description>
        <sceneSpace>
            <bottomLeftFront>
                <x>0.0</x>
                <y>3.0</y>
                <z>0.0</z>
            </bottomLeftFront>
            <bottomRightFront>
                <x>3.0</x>
                <y>3.0</y>
                <z>0.0</z>
            </bottomRightFront>
            <topLeftFront>
                <x>0.0</x>
                <y>3.0</y>
                <z>2.0</z>
            </topLeftFront>
            <topRightFront>
                <x>3.0</x>
```

```
        <y>3.0</y>
        <z>2.0</z>
    </topRightFront>
    <bottomLeftBack>
        <x>0.0</x>
        <y>3.0</y>
        <z>0.0</z>
    </bottomLeftBack>
    <bottomRightBack>
        <x>3.0</x>
        <y>3.0</y>
        <z>0.0</z>
    </bottomRightBack>
    <topLeftBack>
        <x>0.0</x>
        <y>3.0</y>
        <z>2.0</z>
    </topLeftBack>
    <topRightBack>
        <x>3.0</x>
        <y>3.0</y>
        <z>2.0</z>
    </topRightBack>
</sceneSpace>
<sceneEntries>
    <sceneEntry mediaType="video" sceneEntryID="SE1">
        <description lang="en">
            participants streams</description>
        <mediaCaptureIDs>
            <captureIDREF>VC0</captureIDREF>
            <captureIDREF>VC1</captureIDREF>
            <captureIDREF>VC2</captureIDREF>
        </mediaCaptureIDs>
    </sceneEntry>
    <sceneEntry mediaType="video" sceneEntryID="SE2">
        <description lang="en">room stream</description>
        <mediaCaptureIDs>
            <captureIDREF>VC3</captureIDREF>
        </mediaCaptureIDs>
    </sceneEntry>
    <sceneEntry mediaType="audio" sceneEntryID="SE3">
        <description lang="en">room audio</description>
        <mediaCaptureIDs>
            <captureIDREF>AC0</captureIDREF>
        </mediaCaptureIDs>
    </sceneEntry>
</sceneEntries>
</captureScene>
```



```
<captureScene scale="noscale" sceneID="CS2">
  <description lang="en">presentation</description>
  <sceneEntries>
    <sceneEntry mediaType="video" sceneEntryID="CS2_SE1">
      <description lang="en">
        presentation video</description>
      <mediaCaptureIDs>
        <captureIDREF>VC4</captureIDREF>
      </mediaCaptureIDs>
    </sceneEntry>
    <sceneEntry mediaType="audio" sceneEntryID="CS2_SE2">
      <description lang="en">
        presentation audio</description>
      <mediaCaptureIDs>
        <captureIDREF>AC1</captureIDREF>
      </mediaCaptureIDs>
    </sceneEntry>
  </sceneEntries>
</captureScene>
</captureScenes>
<simultaneousSets>
  <simultaneousSet setID="SS1">
    <captureIDREF>VC0</captureIDREF>
    <captureIDREF>VC1</captureIDREF>
    <captureIDREF>VC2</captureIDREF>
    <captureIDREF>VC4</captureIDREF>
    <captureIDREF>AC0</captureIDREF>
    <captureIDREF>AC1</captureIDREF>
  </simultaneousSet>
  <simultaneousSet setID="SS2">
    <captureIDREF>VC0</captureIDREF>
    <captureIDREF>VC3</captureIDREF>
    <captureIDREF>VC2</captureIDREF>
    <captureIDREF>VC4</captureIDREF>
    <captureIDREF>AC0</captureIDREF>
    <captureIDREF>AC1</captureIDREF>
  </simultaneousSet>
</simultaneousSets>
</clueInfo>
```

25. Diff with unofficial -02 version

Here the link to the unofficial -02 version:
<http://www.grid.unina.it/Didattica/RetiDiCalcolatori/inf/draft-presta-clue-data-model-schema-02.html>

<mediaCaptures> moved from <sceneEntry> to <clueInfo> elements.

<mediaCaptures> have been moved out from the <captureScene> blob again. Media captures should have identifiers that are valid out of the local scope of capture scenes, since a consumer should be able to require also single captures in the CONFIGURE message. This design choice reflects a bottom up approach where captures are the basis of the data model. In each media capture a reference to the capture scene containing it is provided. It identifies the space the spatial information of the media capture refers to.

XML document example updated A new example, compliant with the updated schema, has been provided.

language attribute added to <mediaCapture> Such optional attribute reflects the language used in the capture, if any. The purpose of the element could match the one of the language attribute proposed in [I-D.groves-clue-capture-attr].

<priority> added to <mediaCapture> The priority element has an integer value helping in specifying a media capture relative importance with respect to the other captures. That element could correspond to the priority attribute introduced in [I-D.groves-clue-capture-attr].

<embeddedText> added to <videoCapture> The element, if present, indicates text embedded in the video capture. The language used in such embedded textual description is also envisioned within the <embeddedText> element itself. That element could correspond to the priority attribute introduced in [I-D.groves-clue-capture-attr]

<relatedTo> added to <mediaCapture> That optional element contains the ID of a capture the capture refers to. This is for supporting cases where there is the translation of a main capture in a different language. Such translation can be marked with a <relatedTo> tag to refer to the main capture. This could be interpreted the same manner of the supplementary information attribute proposed in [I-D.groves-clue-capture-attr] and further discussed in <http://www.ietf.org/mail-archive/web/clue/current/msg02238.html>.

<dynamic> added to <mediaCapture> That optional boolean element has the same purpose of the dynamic attribute proposed in [I-D.groves-clue-capture-attr]. It indicates if the capture device originating the capture moves during the telepresence session.

new element definition for <description> <description> has a new attribute, lang, indicating the language used for the text within <description>. <description> is used to provide human readable information about captures, scene, and scene entries. The definitions of the corresponding XML elements (i.e., <mediaCapture>, <captureScene>, <sceneEntry>) have been updated to make them able to contain more than one <description>. In that way, they can be described in different languages.

text capture added as new type of capture The element is just a place holder, since it is not characterized with any further information up to now.

26. Diff with -02 version

<sceneSpace> of capture space type <sceneSpace> (was:<sceneArea>) describes a bounding volume for the space of a capture scene as an arbitrary hexahedrons with eight points (placeholder solution).

H26X encoding to be checked.

Simultaneous sets The XML Schema definition of the simultaneous sets has changed. A simultaneous set is defined as a list of L media capture identifiers and M capture scene entrie identifiers, where L, M can be 0 or unbounded.

Capture encoding A new XML Schema type has been added to describe capture encodings as the result of the association of a media capture, represented by its identifier, with an individual encoding, represented by its identifier as well.

Clue info The <clueInfo> element has been left within the XML Schema for the sake of convenience when representing a prototype of ADVERTISEMENT message (see the example section).

Data model definitions added For each element of the datamodel a brief description has been reported to foster discussion.

27. Informative References

- [I-D.groves-clue-capture-attr] Groves, C., Yang, W., and R. Even, "CLUE media capture description", draft-groves-clue-capture-attr-01 (work in progress), February 2013.
- [I-D.ietf-clue-framework] Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams",

draft-ietf-clue-framework-09 (work in progress), February 2013.

[I-D.romanow-clue-data-model] Romanow, A. and A. Pepperell, "Data model for the CLUE Framework", draft-romanow-clue-data-model-01 (work in progress), June 2012.

[RFC4796] Hautakorpi, J. and G. Camarillo, "The Session Description Protocol (SDP) Content Attribute", RFC 4796, February 2007.

Authors' Addresses

Roberta Presta
University of Napoli
Via Claudio 21
Napoli 80125
Italy

EMail: roberta.presta@unina.it

Simon Pietro Romano
University of Napoli
Via Claudio 21
Napoli 80125
Italy

EMail: spromano@unina.it

CLUE
Internet-Draft
Intended status: Standards Track
Expires: January 18, 2013

R. Hansen
A. Krishna
A. Romanow
Cisco Systems
July 17, 2012

Call Flow for CLUE
draft-romanow-clue-call-flow-02

Abstract

This draft shows the CLUE call flow demonstrating the use of CLUE in SIP. It also provides information about the message and syntax for CLUE transport establishment and other extensions in SDP. In CLUE participants act as both providers and consumers. The draft includes a detailed example of a typical use case which includes both static and switched captures.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 18, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Background	4
2. Terminology	5
3. Solution Overview	5
4. Transport for carrying CLUE signaling protocol	7
4.1. SCTP over DTLS over UDP or SCTP over UDP	7
5. Initial Call Establishment	8
5.1. First offer-answer exchange	8
5.2. RTP-header extension and SDP support for CLUE calls	9
5.2.1. RTP extension header to associate RTP stream with CLUE capture, CLUE demux id	10
5.2.2. RTP extension header for audio rendering tag	11
5.3. CLUE channel establishment using SDP	12
5.4. Bandwidth considerations for CLUE	14
5.4.1. Allocate as-you-go	14
5.4.2. Pre-allocation of bandwidth	17
5.5. CLUE message exchange	17
5.5.1. Message sequencing	18
5.5.2. Signalling bandwidth allocation	20
5.6. Sending and receiving media	20
5.6.1. RTP	20
5.6.2. RTCP	21
5.7. Interoperability with non-CLUE systems	21
5.7.1. Backward compatibility with non-CLUE endpoints	22
6. Mid-call CLUE messages and feature interactions	24
6.1. CLUE messages triggered due to change in device capabilities	24
6.2. Closure and Re-establishment of the SCTP channel	26
7. Case study: example call flow	27
8. Security Considerations	30
9. Acknowledgements	31
10. IANA Considerations	31
11. References	31
11.1. Normative References	31
11.2. Informative References	31
Appendix A. Example call flow messages	33
A.1. INVITE from Alice	33
A.2. 200 OK from Bob	34
A.3. CLUE advertisement A (from Alice)	34
A.4. CLUE advertisement B (from Bob)	36
A.5. CLUE response A (from Alice)	42
A.6. CLUE response B (from Bob)	43
Appendix B. Changes From Earlier Versions	43

B.1. Changes From Draft -00	43
B.2. Changes From Draft -01	43
Authors' Addresses	43

1. Background

The purpose of this draft is to examine the call flow for implementing the CLUE framework using SIP and other IETF protocols. Although much of the CLUE Framework [I-D.ietf-clue-framework] is reasonably well agreed upon in the CLUE WG, in order to develop a feasible call flow, we had to make assumptions about many aspects of CLUE that have not yet been decided: what the transport will be, what the message format will be, etc. While these assumptions are not definitive final answers, we have tried to make sensible decisions based on existing drafts and common sense. Perhaps by examining in more detail how the call flow could be handled some light will be shed light on some of the proposals.

The following assumptions related to UDP are being made: these are also stated in [I-D.romanow-clue-sdp-usage]

- o For each CLUE media type and content type (audio, video-main, or video-slides) , there will be at most one corresponding SDP media stream ("m=" line). Note: video-main and video-slides are both using SDP "video" media type.
- o Multiple media captures of the same CLUE media and content type, or different encodings of a given media capture will all use the same SDP media stream, i. e., via RTP multiplexing.
- o It is acceptable to use more than one offer/answer exchange to setup the whole Telepresence session.
- o The Telepresence session is established by setting up sets of unidirectional streams (not bidirectional streams).

Today telepresence endpoints actively negotiate audio and video SDP media streams using the SDP offer/answer model during call establishment and during mid-call features, such as hold resume. The CLUE framework adds additional parameters to a telepresence session. Each party in the CLUE conference is usually both a provider and a consumer, whether the conference is point-to-point or multipoint. The CLUE parameter values for one provider may well not be the same values for the other provider. Thus each party needs to advertise its provider encoding parameter values to the other side. This is not the typical communication model for SDP offer/answer, which is more often a bidirectional agreement on parameter values.

As specified in the CLUE framework, parties in a telepresence conference do not negotiate a single value between them; therefore Offer Answer Model with SDP [RFC3264] is not used for the full negotiation of all encoding related values. Rather, we propose, sending CLUE parameter values via a new CLUE signaling protocol which will be negotiated via SIP. The CLUE messages consist of provider advertisements and consumer requests. [Edt. We use consumer request

and consumer configuration interchangeably. The framework uses the term "configuration", we tend to use "request" here because we are talking about the message usually, rather than the concept. Hopefully, it will not cause any confusion.]

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

3. Solution Overview

A number of aspects of CLUE remain to be fully defined, including major issues such as the transport mechanism and message structure. In order to provide illustrative examples, this draft makes a number of assumptions and guesses about the final format of CLUE. Where possible an attempt has been made to use existing drafts (such as for the multiplexing methodology), or similar developments in other working groups (such as the use of SCTP over UDP as a transport from RTCWeb). As such, this draft should not be taken as an attempt to specify such aspects, but merely to illustrate how they would be used to convey the necessary data: were other decisions to be made on aspects such as transport, the mechanisms would be different, but the data needing to be conveyed would remain the same.

We have assumed about the call flow that:

1. SIP is used to establish the telepresence conference, similarly to using SIP for non-CLUE video conferencing.
2. The media parameters for the call are established using SDP.
3. A new and separate CLUE signaling protocol is used for carrying CLUE messages. (Investigation of SDP for CLUE [I-D.romanow-clue-sdp-usage] looked at carrying CLUE messages in SDP and concluded it was infeasible.)
4. Usage of the CLUE protocol is established through SDP
5. The transport for the CLUE protocol is SCTP over UDP, or SCTP over DTLS over UDP, following the RTCWeb specification.
6. The transport for CLUE protocol, SCPT over UDT or SCTP over DTLS over UDP is setup in SDP, as illustrated in the following diagram and discussed below.

The following illustration Figure 1 shows a high level diagram of a call flow between a 3 screen endpoint and an MCU, for example. [Edt. The diagram shows SCTP over UDP, it could also show SCTP over DTLS over UDP.]

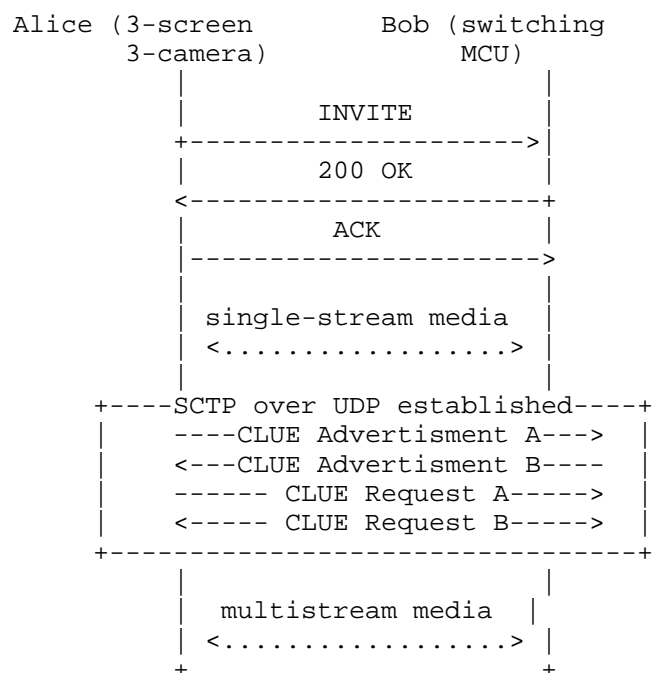


Figure 1: Call setup overview

The next Section 4 discusses the transport for carrying CLUE signaling protocol, following the direction that RTCWeb has taken. Then the telepresence call establishment using SDP is considered in Section 5, including RTP header extension, initial offer answer exchange, and bandwidth considerations. Then Section 5.5 describes CLUE message exchange following the call setup, including provider advertisements and consumer requests. A more detailed coverage of sending and receiving media follows. Interoperability with non-clue implementations is considered in Section 5.7 Then mid-call changes in provider advertisements and consumer requests are considered in Section 6. Section 7 describes a case study which includes both static and dynamic switched captures. Security considerations follows, and there is an appendix with example call flow messages.

4. Transport for carrying CLUE signaling protocol

This section discusses a feasible choice for the transport of CLUE signaling protocol. We are not suggesting the decision on transport be made in this document, but rather we made an assumption based on the requirements and discussion in [I-D.wenger-clue-transport] and what RTCWeb has specified. Most of this section gives a brief overview of the choices made in RTCWeb- SCTP over DTLS over UDP. In this document we have also included the possibility of SCTP over UDP.

4.1. SCTP over DTLS over UDP or SCTP over UDP

There have been several discussion in the CLUE WG suggesting that the transport requirements for CLUE and RTCWeb are sufficiently similar so that CLUE should consider following what RTCWeb has decided. This section briefly reviews RTCWeb's transport specification relevant for CLUE.

RTCWeb WG has reached a general consensus on using SCTP Stream Control Transmission Protocol [RFC4960] encapsulated on DTLS Datagram Transport Layer Security Version 1.2 [RFC6347] encapsulated on UDP for handling non-media data types in the context of RTCWeb. The approach is described in RTCWeb Datagram Connection [I-D.ietf-rtcweb-data-channel].

The transport protocol stack is illustrated in the diagram below Figure 2.

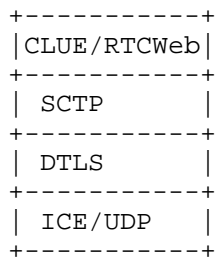


Figure 2: Protocol layers

The services offered by this stack are described in [I-D.ietf-rtcweb-data-channel].

The encapsulation of SCTP over DTLS over ICE over UDP provides a NAT traversal solution together with confidentiality, source authenticated, integrity protected transfers. This data transport

service operates in parallel to the media transports, and all of them can eventually share a single transport-layer port number. SCTP provides multiple streams natively with reliable, unreliable and partially-reliable delivery modes.

DTLS Encapsulation of SCTP Packets for RTCWEB
[I-D.tuexen-tsvwg-sctp-dtls-encaps] describes the encapsulation of SCTP by DTLS.

The Stream Control Transmission Protocol (SCTP) is a transport protocol originally defined to run on top of the network protocols IPv4 or IPv6. This memo document specifies how SCTP can be used on top of the Datagram Transport Layer Security (DTLS) protocol. SCTP over DTLS is used by the RTCWeb protocol suite for transporting non-media data between browsers.

In addition a third RTCWeb related draft, Stream Control Transmission Protocol (SCTP)Based Media Transport in the Session Description Protocol (SDP) [I-D.ietf-mmusic-sctp-sdp] describes in detail how SDP can handle setting up SCTP and DTLS. (The draft does not consider SCTP over UDP.)

SCTP (Stream Control Transmission Protocol) is a transport protocol used to establish associations between two endpoints. This document describes how to express media transport over SCTP in SDP (Session Description Protocol). This document defines the 'SCTP', 'SCTP/DTLS' and 'DTLS/SCTP' protocol identifiers for SDP. We have followed the syntax in this document.

We have followed this syntax here.

5. Initial Call Establishment

This section is not intended to prescribe the flows and messages precisely as they are shown, but rather illustrates the principles.

5.1. First offer-answer exchange

The flow illustrated below shows Alice calling Bob offering SDP parameters that are typical of an offer. The new extensions related to CLUE are highlighted below in Figure 3.

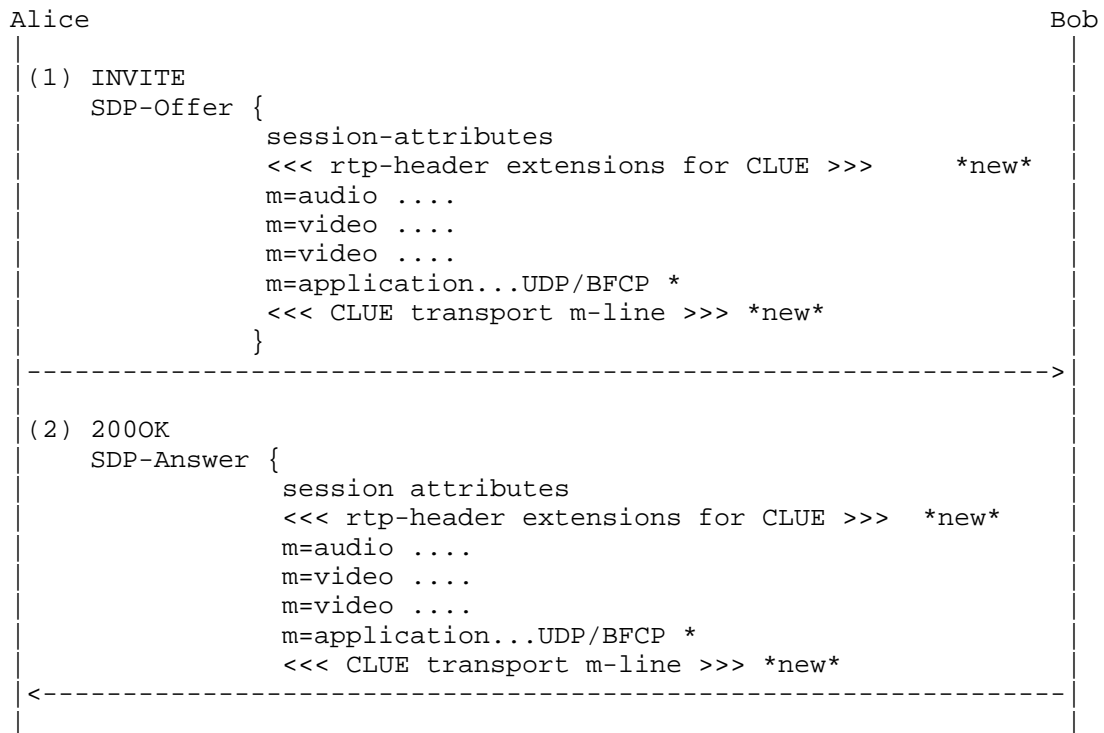


Figure 3: First SDP offer answer exchange

In order to support CLUE using SIP several issues need to be considered during the initial call setup. These include:

1. Extension of the RTP header for telepresence calls. As part of CLUE usage, two such potential requirements are under consideration. Refer to sec 5.2.
2. Establishment of CLUE signaling channel using a separate application m-line. Sec 5.3
3. Bandwidth considerations during initial setup. Sec.5.4

5.2. RTP-header extension and SDP support for CLUE calls

Currently 2 proposals are under consideration for mechanisms that require using RTP extension headers A General Mechanism for RTP Header Extensions [RFC5285]. Although neither proposal has been accepted, we are showing how each would be handled if they are added to the CLUE framework [I-D.ietf-clue-framework].

The first mechanism that requires an RTP header extension is for

associating RTP streams with CLUE captures as described in RTP Usage for Telepresence Sessions [I-D.lennox-clue-rtp-usage]. The second mechanism is using an audio rendering tag to associate audio captures with video captures as described in [I-D.romanow-clue-audio-rendering-tag]. Both use session-level SDP parameters as recommended by [RFC5285].

5.2.1. RTP extension header to associate RTP stream with CLUE capture, CLUE demux id

Telepresence calls multiplex encoded streams from multiple similar media sources on a single RTP session for the same media-type and content-type. For example, Alice has a 3 camera system but in SDP she negotiates a single RTP session for main video. How does she associate these separate RTP streams with the CLUE capture ids to enable Bob, the receiver, to know where to send the audio and video streams? How do the RTP streams get routed to the appropriate decoders and output devices? This is considered in detail in [I-D.lennox-clue-rtp-usage].

The proposal is to extend the RTP header to embed a unique id in each RTP packet, referred to here as the CLUE demux id. This requires negotiating a unique "id" out of band in SDP so that multiple such extensions could co-exist. This is achieved by using the extmap extension in SDP as specified in [RFC5285].

The diagram in Figure 5 illustrates the call flow for the identifier being used to signal the CLUE demux id information in RTP extension header. In this diagram and subsequent ones, we have used the following drawing to represent a camera, by a circle, and a screen, by a square.

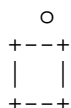


Figure 4: Representation of and endpoint camera and screen

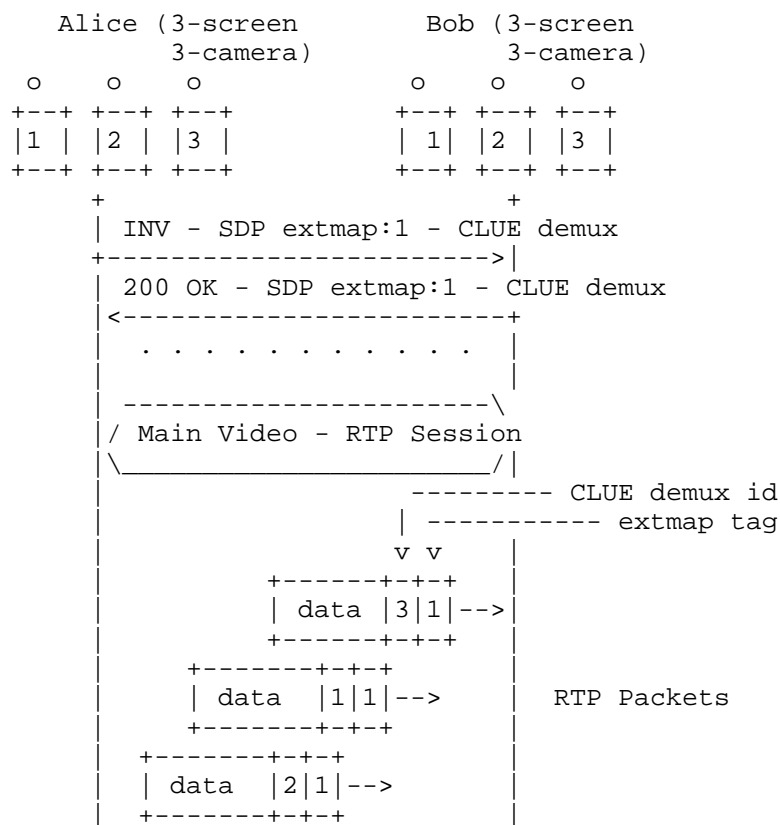


Figure 5: SDP example using extmap id-1

The extmap is a SDP session level attribute in the following example using extmap-1:

```

v=0
o=3ss 40471 40471 IN IP4 10.47.197.103
c=IN IP4 10.47.197.103
t=0 0
a=extmap:1 urn:ietf:params:clue:demux
  
```

5.2.2. RTP extension header for audio rendering tag

As explained in [I-D.romanow-clue-audio-rendering-tag], to support directional audio the receiver needs to know where to render audio in relationship to video captures. In some circumstances the spatial

information is not available in the provider advertisement, so the association cannot be made in that way. The proposal for these cases is that the consumer optionally tells the provider an audio tag value corresponding to each of its chosen video captures, which enables received audio to be associated with the correct video stream, even when the set of audible participants changes.

In the example provided below, the packet shows dual header extensions - one for associating RTP stream with capture as above, and one for associating audio capture with video capture.

The SDP session level attribute extmap id-3 below depicts the identifier being used to signal the audio tag field in the RTP extension header.

```
a=extmap:3 urn:ietf:params:clue:audiotag
```

5.3. CLUE channel establishment using SDP

Most importantly, the initial offer/answer negotiates the port and transport information for establishing a CLUE signaling channel. The new application m-line details the necessary information. As explained in Section 4, the CLUE signaling protocol will ride on top of SCTP transport.

Figure 6 shows an example call flow for establishing an unsecure CLUE channel.

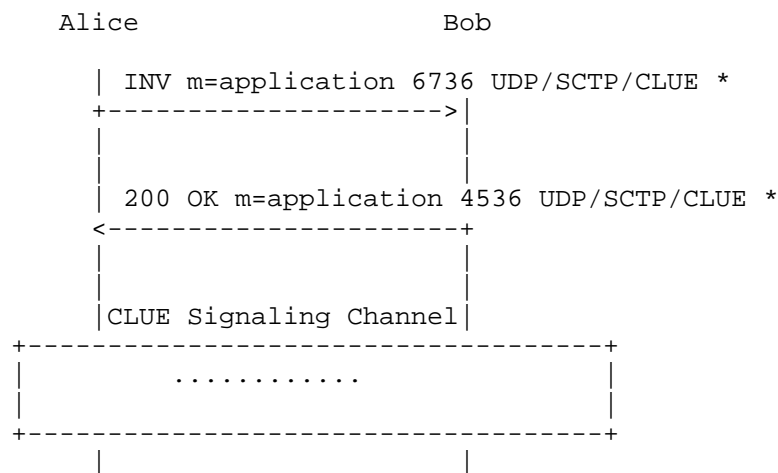


Figure 6: CLUE transport channel establishment

```
SDP Offer:
INVITE bob@biloxi.net SIP/2.0
...
Content-Length: 1000

v=0
....
m=audio...
...
m=video...
...
m=application 6736 UDP/SCTP/CLUE *
a=setup:actpass
a=connection:new
```

```
SDP Answer:
SIP/2.0 200 OK
...
Content-Length: 1000
```

```
v=0
....
m=audio...
m=video...
....
m=application 4534 UDP/SCTP/CLUE *
a=setup:active
a=connection:new
```

The same example encrypted with DTLS shall look as shown in Figure 7. A fingerprint hash attribute is included at the SDP session level in order for the other side to authenticate the identity while exchanging the certificate during the connection setup.

```
SDP Offer:
a=fingerprint:
SHA-1 64:02:2A:20:92:CD:DB:80:9F:68:0D:EF:AC:99:95:34:89:C6:7D:34
...
m=application 6736 UDP/DTLS/SCTP/CLUE *
a=setup:actpass
a=connection:new

SDP Answer:
a=fingerprint:
SHA-1 4B:AC:B7:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:BA
...
m=application 4534 UDP/DTLS/SCTP/CLUE *
a=setup:passive
a=connection:new
```

Figure 7: Secure CLUE channel establishment

In an unencrypted call the 'setup' parameter in SDP is used to negotiate which participant will listen for an incoming SCTP connection, while the other participant initiates the connection. In an encrypted call using DTLS the 'setup' parameter negotiates which participant will establish the DTLS connection (the same participant then establishes SCTP over the DTLS channel once DTLS has been established).

5.4. Bandwidth considerations for CLUE

We discuss two models of bandwidth allocation for CLUE telepresence calls. In the first mechanism bandwidth is allocated as needed through successive re-INVITES. This has the advantage of being bandwidth efficient but may require multiple re-INVITES. In the second approach the maximum allowable bandwidth is allocated initially. This saves on re-INVITES but may allocate bandwidth not needed. If desired the participant can-reinvite with a lower bandwidth.

5.4.1. Allocate as-you-go

The bandwidth initially offered by Alice is set to some default value (say for a single-screen system). The bandwidth requirement is bound to change as the call progresses on learning the remote end capabilities and the selections made thereafter. Any increase in the usage of bandwidth based on the selected captures will result in the telepresence endpoint sending out a new re-INVITE requesting the intermediaries to allocate extra bandwidth from their pool. The call flow is illustrated in Figure 8.

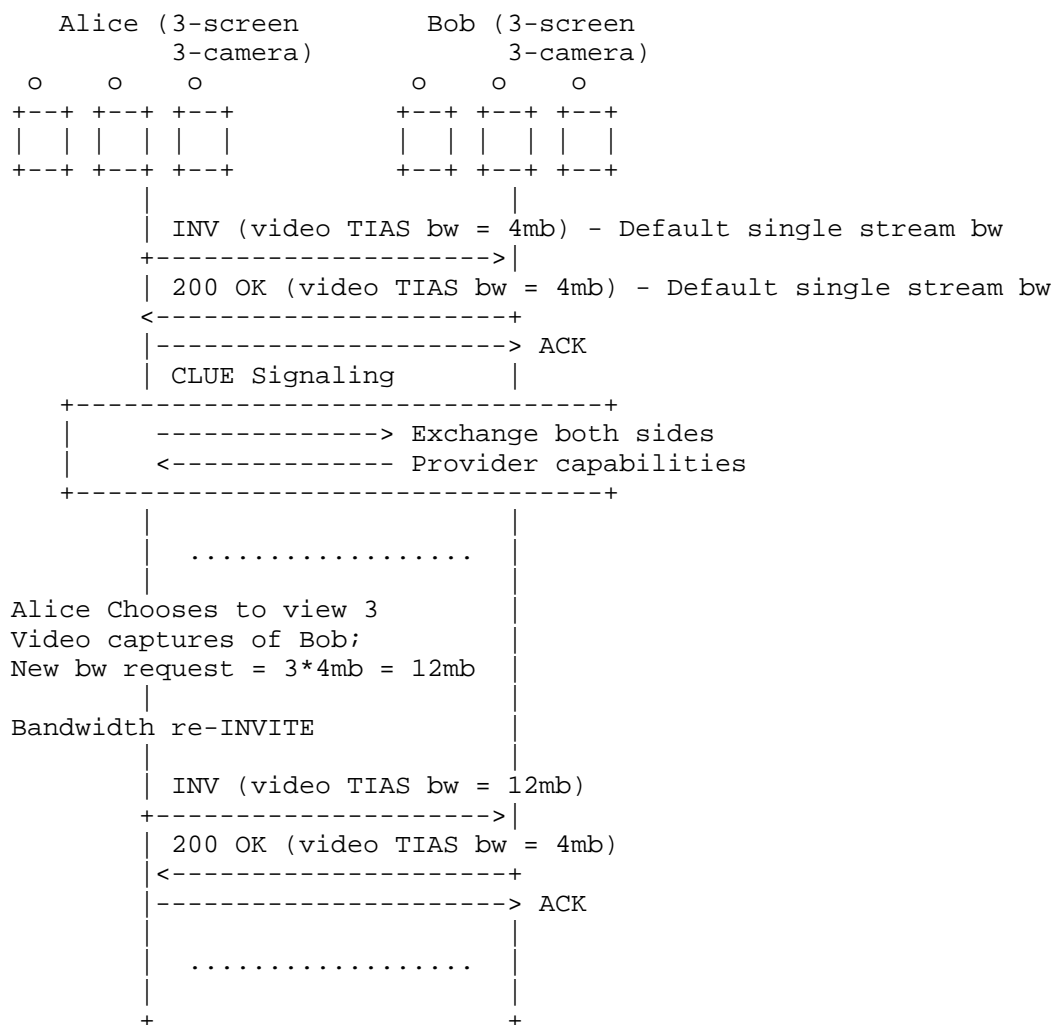


Figure 8: Allocate bandwidth as needed

The advantage of this approach is the usage efficiency, as during the initial call setup the remote endpoint's capabilities are still unknown. Also, even after learning the remote end's capabilities through CLUE, there is no guarantee that Alice shall decide to view all or a subset of the captures thereafter. So the reserved network bandwidth is the amount that is currently being used.

The drawback is the potential need for one or more re-INVITES for re-negotiating new bandwidth numbers from either side if they choose to

view additional captures or choose captures in additional encoding formats. Also due to the asynchronous nature of the consumer requests, the flow is susceptible to glare re-INVITES.

The glare would be more pronounced in the case after the first CLUE provider advertisement exchange. This may be due to these systems being pre-programmed to re-configure themselves immediately after learning each other's capabilities. Figure 9 illustrates the glare condition issue.

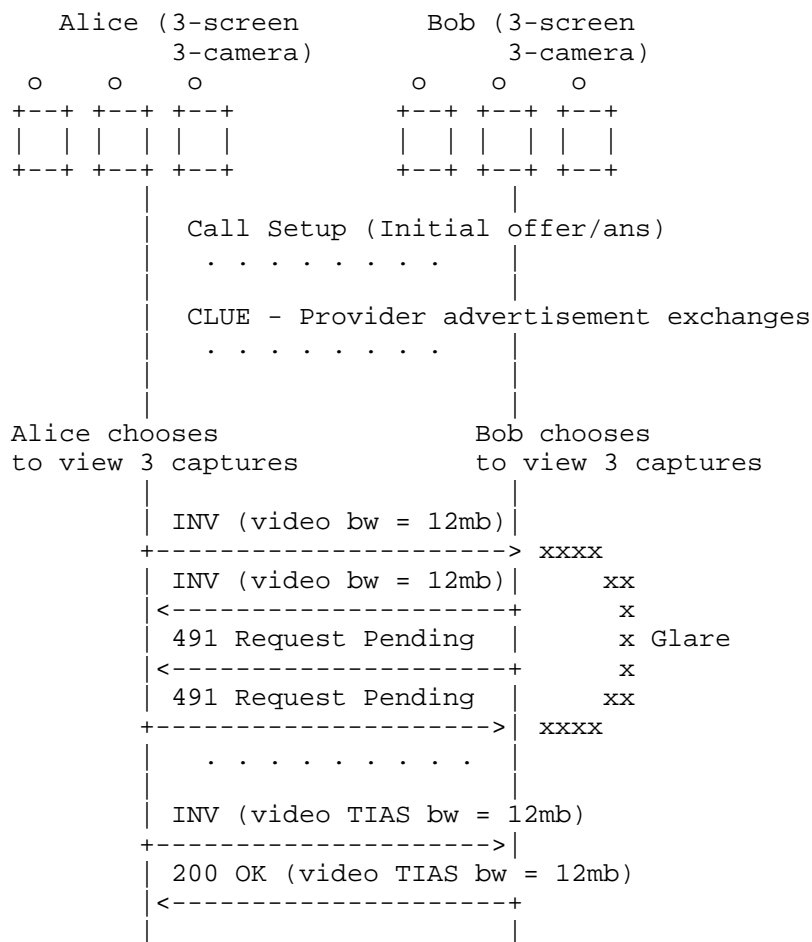


Figure 9: Glare condition in allocate bandwidth as needed

5.4.2. Pre-allocation of bandwidth

An alternate method to avoid the need for multiple re-INVITEs to increase bandwidth is to offer maximum bandwidth size that this device is capable of handling in the initial offer/answer. The call flow is illustrated in Figure 10. However this approach risks unnecessary call failures in the case the network bandwidth is provisioned and policed.

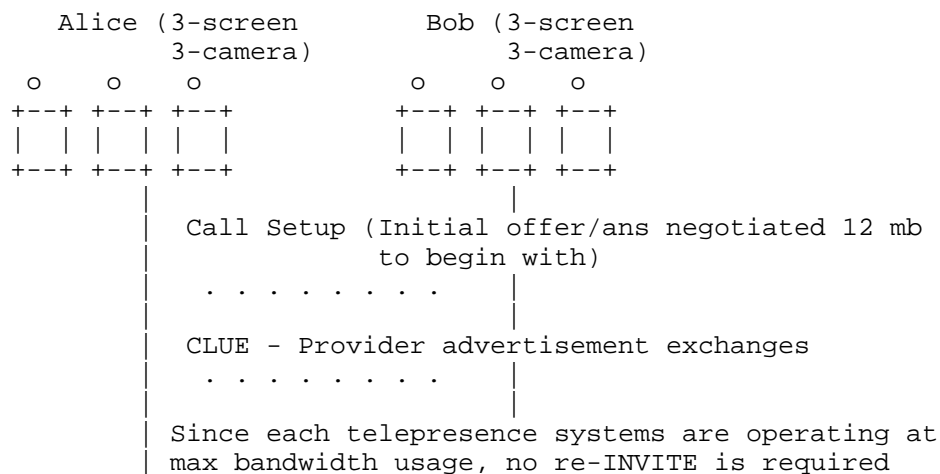


Figure 10: Maximum bandwidth pre-allocation

Implementations must be able to cope with the remote endpoints operating in either mode.

5.5. CLUE message exchange

This section describes the call flow for CLUE messages following call setup.

Following the SIP and SDP call setup, CLUE messages are exchanged. There are only 2 types of CLUE messages: provider advertisements and consumer requests.

Each participant in a CLUE telepresence conference, whether an endpoint or an MCU, is most likely to act as both provider and consumer, as discussed above. Thus each will send provider advertisements and receive consumer requests for those advertisements; and each will also receive provider advertisements

and send consumer requests.

The provider advertisements will consist of the following data elements:

- o captures
- o capture scenes
- o encoding groups
- o encodings
- o simultaneous transmission set

The consumer requests will consist of the following data elements:

- o captures
- o encodings

5.5.1. Message sequencing

In considering the sequencing of messages, either participant could be first to send a provider advertisement. The next message, sent by the other participant, could be either a provider advertisement or a consumer request in response to the provider advertisement. This is illustrated in the 2 examples below. The first example shows Bob's provider advertisement sent before he responds to Alice's provider advertisement. The second example shows Bob sending a consumer request in response to Alice's provider advertisement before sending his own provider advertisement.

The important thing to keep in mind is that these messages are sent asynchronously and can be sent not only at the beginning of a call, but any time throughout the call when a parameter value has changed.

Figure 11 shows an example of CLUE messages with provider advertisements from each side following one another.

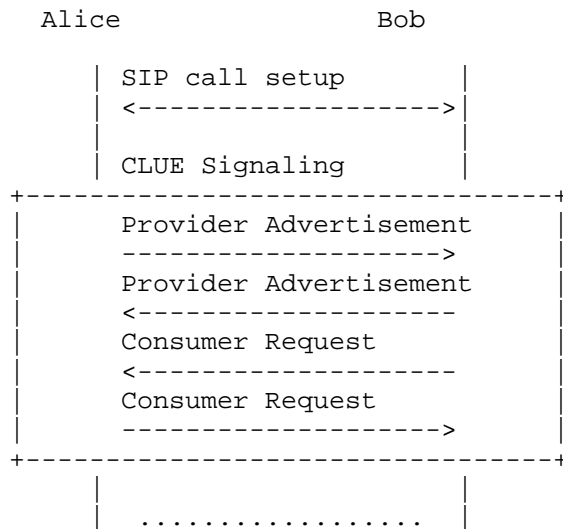


Figure 11: Back-to-back provider advertisements

Figure 12 shows an example of CLUE messages with provider advertisement directly followed by consumer request.

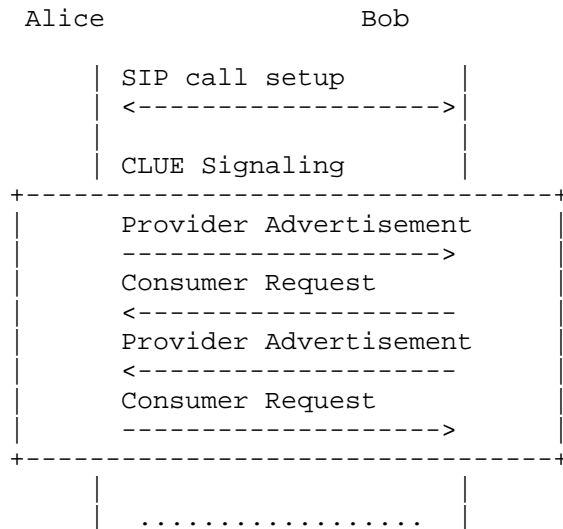


Figure 12: Provider advertisement followed by consumer request

Appendix A shows example provider advertisement and consumer request messages related to Section 7. Keep in mind these are not proposals, just examples using data elements taken from the current versions of the [I-D.romanow-clue-data-model] and [I-D.ietf-clue-framework].

5.5.2. Signalling bandwidth allocation

Call agents and other intermediaries responsible for bandwidth allocation may grant the bandwidth requested in the consumer request. Or, an intermediary may not allow the requested bandwidth. In this case, a feedback mechanism is desirable so that the receiver can adjust accordingly. This is a more generic SIP issue and is thus beyond the scope of this document.

5.6. Sending and receiving media

This section discusses CLUE's use of RTP extension header and RTCP.

5.6.1. RTP

CLUE allows multiple media streams to be multiplexed onto a single RTP session, to reduce the complexity of negotiating independent ports for an asymmetric and variable number of media streams, and aid NAT and SBC traversal [I-D.lennox-clue-rtp-usage]. Demultiplexing these media streams is achieved via the urn:ietf:params:clue:demux RTP header extension negotiated in the SIP messages. The provider MUST include this header extension in all media packets sent due to a successfully processed CLUE consumer request.

The value of the header extension for a given stream MUST match the value of the 'CLUE demux id' element for the associated stream in the consumer request. Figure 13 shows an example of an RTP packet containing an RTP header extension

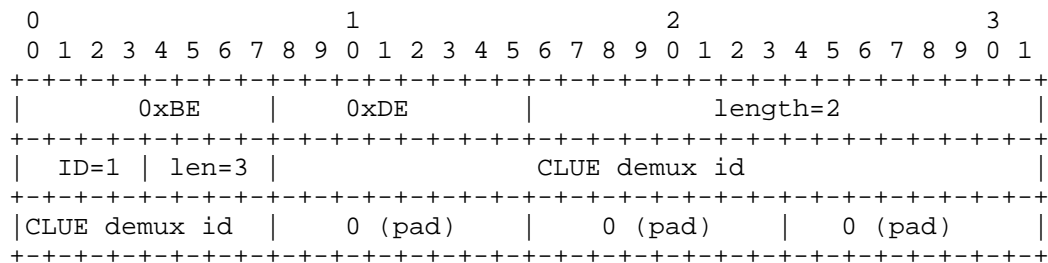


Figure 13: RTP packet showing extension header for CLUE demux id

See [I-D.lennox-clue-rtp-usage] for the specifics of the demultiplexing process.

A receiving device which has sent at least one CLUE consumer request MUST examine RTP packet headers for the presence of this header extension. If it is not present then the packet should be treated as part of a single-stream media stream. If the extension header is present then the value of the CLUE demux id allows the various multiplexed streams to be differentiated. Packets with a CLUE demux id that does not correspond to one of the ids sent in the most recent valid CLUE consumer request MUST be discarded.

5.6.2. RTCP

As described in the preceding session, multiple media streams are multiplexed onto a single RTP session. This RTP session SHOULD be matched by a single set of RTCP messages, sent in a conventional fashion. A device SHOULD send RTCP receiver reports, sender reports, SDPS and other packets as it would in the single-stream case. However, when multiple media streams are present each RTCP packet SHOULD contain multiple chunks; each chunk can be used to identify a specific SSRC and include the information relevant to that media stream. This allows standard statistics, packet loss indications and other RTCP metrics to be used. CNAME, as part of the RTCP SDPS message, can be used to indicate synchronization between multiple multiplexed media streams when they are generated by encoders sharing a common clock, in the same way it can be used to synchronize streams received on different ports.

5.7. Interoperability with non-CLUE systems

CLUE support in a remote device is identified by the presence of a valid SDP media stream advertising the ability to negotiate the SCTP-based CLUE protocol - absence of this requirement indicates that either the remote device does not support CLUE, or that a middle box in the signaling path has suppressed the ability to do so. A device that receives an SDP offer or answer without a valid media stream advertising the CLUE protocol MUST fall back to a single-stream call until such time as new SDP messages are able to negotiate a CLUE protocol media stream between the devices - see Section 5.7 for details on interoperability with non-CLUE devices. If both sides successfully support CLUE then the call flow proceeds as shown in this draft.

Note that as per Offer Answer Model with SDP [RFC3264], both devices MUST be prepared to receive media for any media streams they have advertised as recvonly or sendrecv. The devices SHOULD also begin sending single-stream media once the initial SDP negotiation is

complete, while the SCTP channel is being set up and before any CLUE messages are sent or received. This minimizes the time in which a basic single-stream call is established to match that of a conventional non-CLUE system, and ensures a baseline level of interoperability. Once CLUE messages have been received and processed the call will then be 'upgraded' to multistream.

5.7.1. Backward compatibility with non-CLUE endpoints

One of the requirements of CLUE is that it allows backwards compatibility with devices that are not CLUE-aware, ensuring that a call between a CLUEful device and a conventional non-CLUEful SIP will result in a conventional, single-stream call.

5.7.1.1. Handling of CLUE SDP by conventional SIP devices

The SIP messages sent by CLUEful endpoint are compatible with conventional SIP compliant devices. Audio and video remain on separate ports, which not only allows non-CLUE devices to establish calls as normal, but allows separate QoS settings to be applied to the different media types if desired. All the CLUE-specific changes are in the form of SDP extensions which must either be ignored (new attributes) or answered back appropriately indicating their lack of support (new media line) by the non-CLUE recipient. As such a compliant SIP device without CLUE support will be able to establish a conventional audio-video call on receiving a CLUEful INVITE.

5.7.1.2. CLUE Option Tag usage

It may prove useful to explicitly signal CLUE support; the use of a 'clue' option-tag makes the SIP devices explicit about CLUE support. Endpoints could use this tag to make the registrar aware of their ability to support CLUE, as shown in Figure 14.

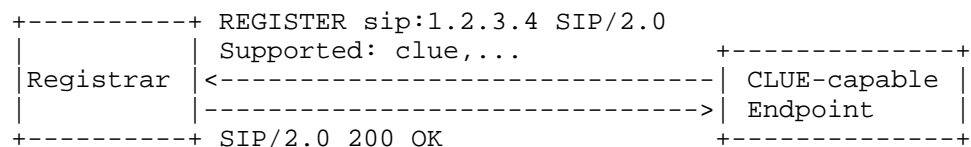


Figure 14: CLUE option tag for registration

Further, the option-tag also allows CLUE endpoints to specify that they do not wish to fall back to single stream behaviour (by including the 'clue' option-tag in a Require field) in cases where

they contact a non-CLUE endpoint:

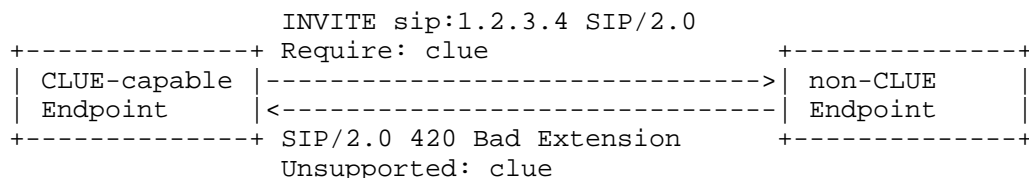


Figure 15: CLUE option tag for require

Finally, the option-tag may be used in an OPTIONS message to signal that a device supports CLUE even without the presence of an SDP, as show in Figure 16.

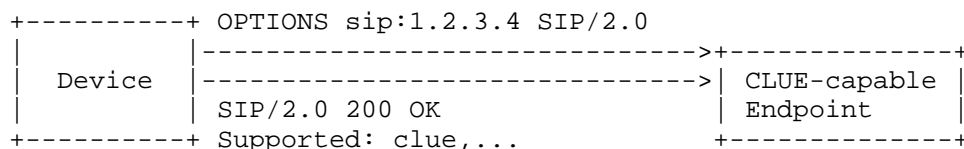


Figure 16: CLUE option tag for options

5.7.1.3. Non-compliant devices and usage of CLUE option-tag

There are non-compliant SIP devices which may react badly when receiving SDP extensions they do not understand, responding with a 400 Bad Request or in some other fashion. While the principal aim of this document is not to work around devices that do not implement specified behaviour, there are a number of potential strategies for dealing with such legacy devices:

1. The calling device may attempt to send a second INVITE without any CLUE parameters.
2. A back-to-back-user-agent in the call path to the legacy device may recognise its lack of CLUE support (via static configuration, lack of a 'Supported: clue' options tag in a REGISTER message or lack of CLUE parameters in the response to an OPTIONS message) and strip CLUE parameters from INVITE messages directed to that device.

6. Mid-call CLUE messages and feature interactions

A provider may send a new advertisement at any time, and a consumer may send a new request at any time (once it has received an initial advertisement). There is not a one-to-one mapping between advertisements and requests - a consumer may send multiple requests to a single advertisement. Advertisements contain a sequence number, while consumer requests contain both a sequence number and a reference to the sequence number of the advertisement to which they correspond - the provider can use these to detect and discard requests relating to a previous, now invalid advertisement.

Consumer re-configuration can occur anytime during the course of a call. Possible triggers include but are not restricted to:

- o New provider advertisement to indicate a changes in captures or encodings.
- o Change in device configuration (such as availability of new screen) or user selection at the consumer end.
- o Invocation of features such as Hold/Resume can potentially alter the call topology/view.

The bandwidth requirement and need for re-INVITE are based on the increase or decrease in usage. Implementations can choose an appropriate bandwidth strategy as discussed previously. However, they must ensure necessary bandwidth is in place before choosing any new captures.

6.1. CLUE messages triggered due to change in device capabilities

The following is a mid-call CLUE provider advertisement from Bob advertising capture-scene with 3 captures as opposed to 2 previously (could be triggered by turning on the 3rd camera. In order to keep the call-flow simple we assume both Alice and Bob allocate a large bandwidth (20Mb). The consumer request is within that bandwidth boundary. The call flow is shown in Figure 17.

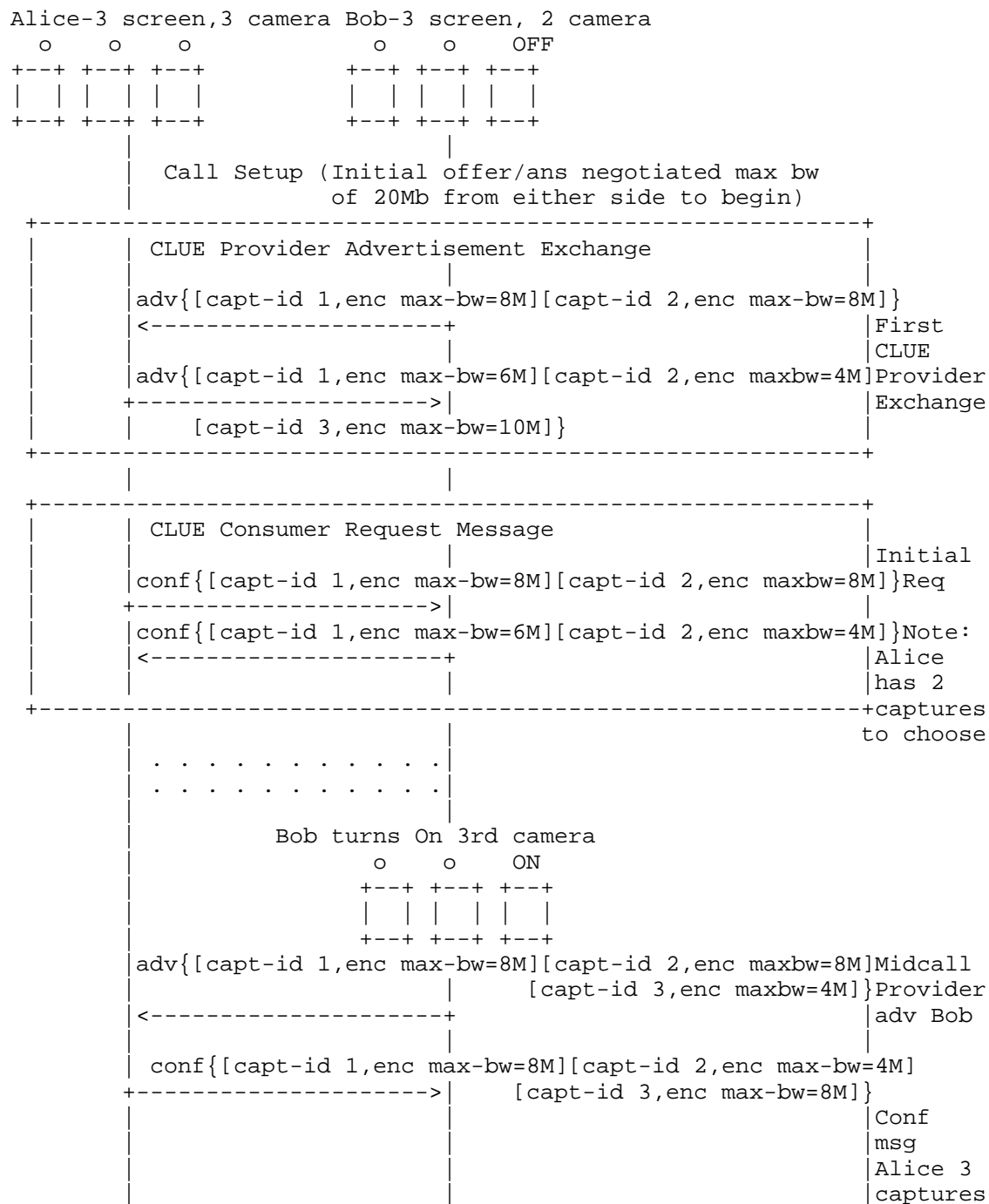


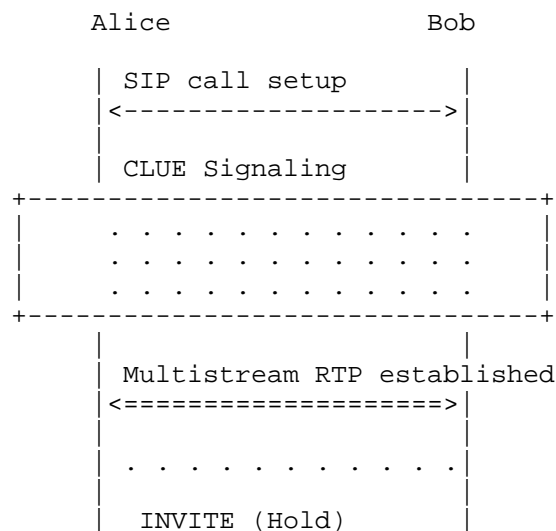
Figure 17: Mid-call change in provider advertisement

6.2. Closure and Re-establishment of the SCTP channel

A mid-call SDP offer or response may result in the closure of the SCTP channel, or a re-establishment of a channel that is either active or inactive. Reasons for these changes may include an endpoint going on or coming off hold, a device re-routing media to a different recipient, or an endpoint no longer wishing to continue to use multistream functionality. If the SCTP channel is reestablished it is not safe to assume that the other end of the SCTP channel has knowledge of the previous state of CLUE messaging - in some cases the new recipient of the channel may be an entirely new individual.

Depending on the intention of the initiator of hold, the CLUE connection could be preserved or re-established. For the shared-line and presence scenarios it is recommended to teardown and re-establish. If the connection was preserved across hold-resume the previous CLUE advertisements and configuration state continues to be valid. The offerer/and or answerer could choose to reuse the existing connection by setting a=connection:existing. If not the connection could be re-established by either piggybacking on the same SDP offer/answer for resume or separately. The CLUE provider advertisement and configuration is exchanged all over again for establishing multistream.

Figure 18 depicts the Hold-Resume in which the CLUE signaling channel is re-established.



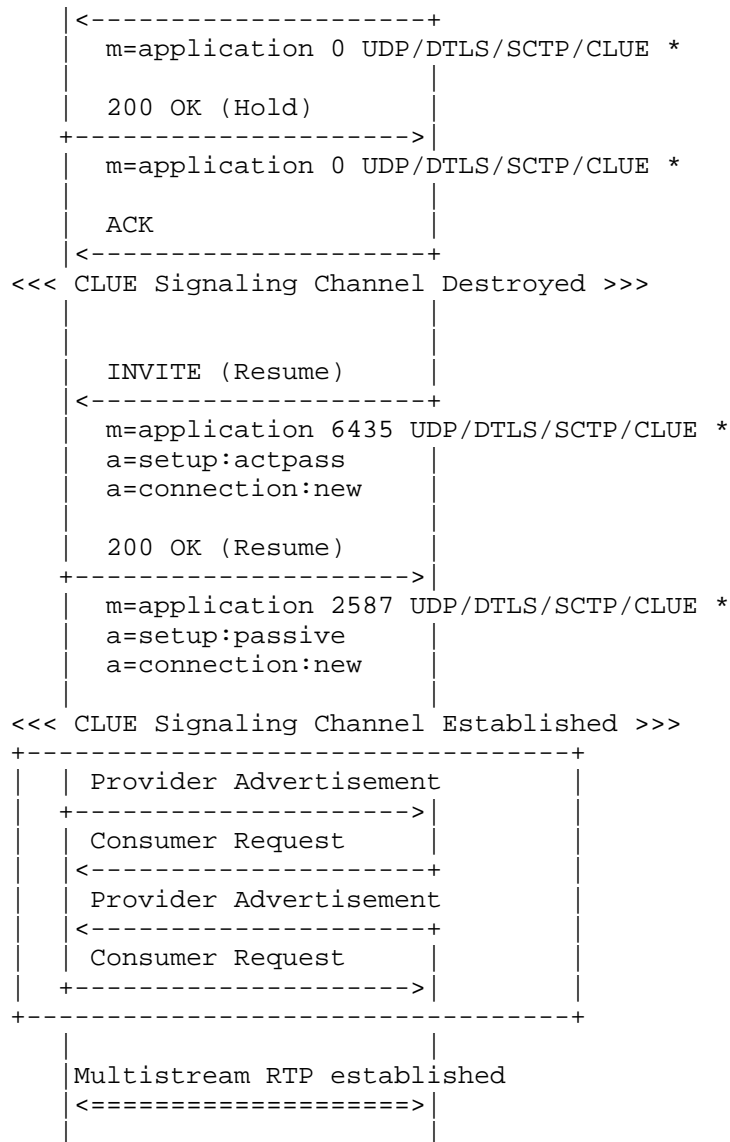


Figure 18: HOLD/RESUME with re-establish

7. Case study: example call flow

To help illustrate how all the aspects of a multistream call using CLUE tie together, a complete set of SIP and CLUE message is provided

for a sample call between two participants. In this example one participant is an endpoint and the other is an MCU. If both participants were endpoints, the call flow would be the same although some of the content may be different.

Alice in this example is a three-screen endpoint, with three cameras. She is able to send all three camera streams individually, or can send a single switched video stream (based on the current active speaker, or most recent speaker if no one in her room is speaking). Her system also has three microphones, but only advertises a single, composed audio stream made up of the streams of three microphones mixed together. Bob, in contrast, is an MCU that switches media between different participants in a conference. Bob advertises up to three video streams and three audio streams, with alternate offerings for one- and two-screen systems. Figure 19 shows an overview of the call flow.

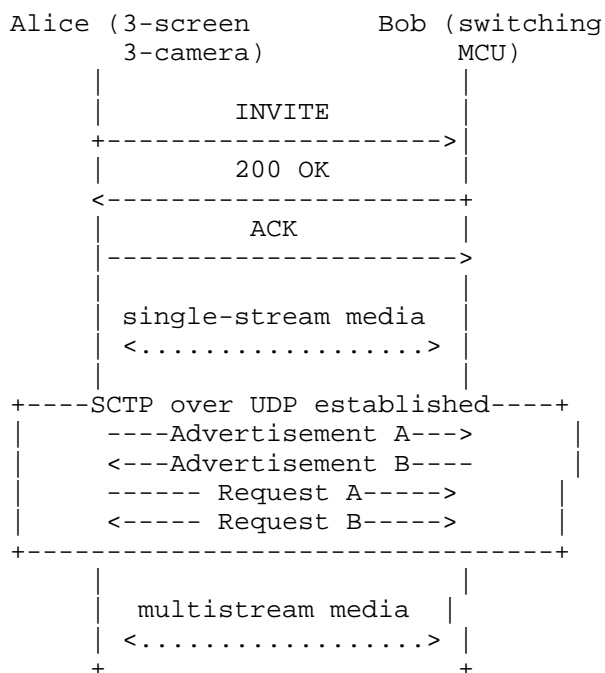


Figure 19: Example call flow

The flow begins with Alice sending an INVITE to Bob (Appendix A.1.) The SDP includes an audio stream, offering AAC and G.711(u), and a video stream offering H264. Alice is limited to receiving 6M of

bandwidth. Alice also includes a media line for CLUE over SCTP over UDP (for simplicity in this example the stream is unencrypted). The attributes for the stream show that this is a new connection, and that Alice is willing to be either active or passive with respect to establishment of the media session. Finally, Alice includes an SDP 'extmap' parameter, advertising that she understands the multiplex id RTP header extension, and that Bob should use an ID of 1 to identify such extensions; this parameter is included at the session level, as it applies to both audio and video media streams.

Bob responds with a 200 OK (Appendix A.2. - any prior lxx messages have been elided for brevity). The SDP also includes an audio stream supporting AAC and G.711(u) and a video stream offering H264. Bob is able to receive up to 10M of media. Bob includes a media line for CLUE over SCTP over UDP that matches Alice's; the 'connection:new' attribute corresponds to hers, and Bob chooses to be passive in establishment of the stream. Bob also includes an SDP 'extmap' parameter, in this case specifying that Alice should send multiplexed media with an RTP extension header with an ID of 3.

Alice sends an ACK, not included in the appendix.

Having completed the initial SIP call setup, Alice and Bob begin sending H264 video and AAC audio in a conventional single-stream fashion. Further, Alice initiates a STCP connection to Bob on port 3782, encapsulated in UDP, as was negotiated in the SDP exchange.

With the CLUE transport established both sides are free to begin sending CLUE messages. The call flow shows Alice sending an advertisement before Bob, but in practice there is no correlation between the sending of these messages; Bob may send his message first, or both may be sent simultaneously. Both sides of the CLUE message exchange are independent of one another.

Alice's CLUE advertisement includes four video captures (Appendix A.3.) The first three are static captures that correspond to the three cameras of the system, while the final one is a switched capture to show the current active speaker. The static captures give the physical coordinates of the area of capture, in millimetres, while the switched capture does not include an area of capture. There is a single, mixed audio capture, again with no area of capture. The entries for the capture scene allow the recipient to understand which captures provide 'equivalent' views. In this case it illustrates that to get a 'complete' view of the scene the recipient should subscribe to either captures 1, 2 and 3, or to capture 4. Finally, the information in the encoding groups shows that while Alice can supply any given video stream at up to 4M, she can only supply a total of 6M of video.

In contrast, as a switching MCU Bob has no static captures (Appendix A.4.) Instead, Bob wants to be able to supply one-, two- and three-screen switched capture views. He does this by advertising six video captures with non-physical area of capture coordinates; these coordinates are used to illustrate the fraction of the overall scene a capture represents (and how to render the captures to ensure their adjacency is correct and no multi-screen systems are rendered back-to-front). A similar approach is taken for audio. This is reinforced in the entries, which illustrate the captures to which a recipient should subscribe to receive a 'correct' view of the conference. The encoding information supplied by Bob shows that he is able to supply up to three video streams, with the only encoding maximum that of the 10M advertised in his SDP for the stream total.

Having received Bob's CLUE advertisement, Alice then sends her consumer request (Appendix A.5.) As she is a three-screen system she requests video captures 4, 5 and 6, which correspond to the switched captures for a three-screen system. She includes a 'max-bandwidth' element, limiting each stream to 2M. Having two speakers she subscribes to audio captures 8 and 9, which correspond to the two-channel audio advertised by Bob.

Having received Alice's CLUE advertisement, Bob sends his initial consumer request (Appendix A.6.) Bob's request is straightforward, requesting the three static camera streams and the mixed audio stream, and adds no additional encoder limitations beyond those already imposed by his SDP.

Both Bob and Alice now send multistream audio and video, the RTP packets including the multiplex extension header with CLUE demux ids specified in the consumer requests. Note again that, though the example illustrates Alice and Bob sending their CLUE messages in order, this is not a requirement; Alice might send her advertisement, receive Bob's request and begin sending multistream media before ever receiving Bob's advertisement. At any time Alice or Bob may send further CLUE advertisement or request messages, which invalidate previous messages. They may also send SIP messages to update or alter media parameters.

8. Security Considerations

This document provides an overview of the call-flow of a CLUE multistream telepresence call, and hence is not an appropriate place to definitively identify and deal with security considerations. However, it should be clear from the above that CLUE does pose a number of security challenges: for example, the transport conveying CLUE messages must be encryptable in a fashion that limits the

potential for man-in-the-middle attacks, while the multiplexing of RTP streams must not interfere with the ability to secure these streams against interception or spoofing. These security considerations will be addressed as part of the specification of these aspects of CLUE, but it is believed that well-understood methods (such as DTLS for the CLUE protocol and SRTP for the media) can be used to secure these channels.

9. Acknowledgements

10. IANA Considerations

This draft includes examples of a number of features which, if agreed on by consensus, would have IANA considerations as part of their specification. The examples used here are purely for illustrative purposes, and do not represent the final format of these features; as such this draft has no IANA considerations.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.
- [RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.
- [RFC5285] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

11.2. Informative References

- [I-D.ietf-clue-framework] Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-06 (work in progress), July 2012.

- [I-D.ietf-mmusic-sctp-sdp]
Loreto, S. and G. Camarillo, "Stream Control Transmission Protocol (SCTP)-Based Media Transport in the Session Description Protocol (SDP)", draft-ietf-mmusic-sctp-sdp-01 (work in progress), March 2012.
- [I-D.ietf-rtcweb-data-channel]
Jesup, R., Loreto, S., and M. Tuexen, "RTCWeb Datagram Connection", draft-ietf-rtcweb-data-channel-00 (work in progress), March 2012.
- [I-D.lennox-clue-rtp-usage]
Lennox, J., Witty, P., and A. Romanow, "Real-Time Transport Protocol (RTP) Usage for Telepresence Sessions", draft-lennox-clue-rtp-usage-04 (work in progress), June 2012.
- [I-D.romanow-clue-audio-rendering-tag]
Romanow, A., Hansen, R., Pepperell, A., and B. Baldino, "The need for audio rendering tag mechanism in the CLUE Framework", draft-romanow-clue-audio-rendering-tag-00 (work in progress), May 2012.
- [I-D.romanow-clue-data-model]
Romanow, A. and A. Pepperell, "Data model for the CLUE Framework", draft-romanow-clue-data-model-01 (work in progress), June 2012.
- [I-D.romanow-clue-sdp-usage]
Romanow, A., Andreasen, F., and A. Krishna, "Investigation of Session Description Protocol (SDP) Usage for Controlling mUltiple streams for tElepresence (CLUE)", draft-romanow-clue-sdp-usage-01 (work in progress), March 2012.
- [I-D.tuexen-tsvwg-sctp-dtls-encaps]
Jesup, R., Loreto, S., Stewart, R., and M. Tuexen, "DTLS Encapsulation of SCTP Packets for RTCWEB", draft-tuexen-tsvwg-sctp-dtls-encaps-01 (work in progress), July 2012.
- [I-D.wenger-clue-transport]
Wenger, S., Eubanks, M., Even, R., and G. Camarillo, "Transport Options for Clue", draft-wenger-clue-transport-02 (work in progress), March 2012.

Appendix A. Example call flow messages

These message examples are entirely illustrative - they are not meant as a proposal for messages. As much as possible they follow the nomenclature in the [I-D.ietf-clue-framework]. However, there are a few elements in the messages which are not in the framework, such as "mixed" for audio, and sequence numbers which will be necessary in any independent signaling protocol. [Edt. Hopefully, these few additions will not be distracting from the benefit of an example.]

A.1. INVITE from Alice

```
INVITE sip:bob@example.com SIP/2.0
Via: SIP/2.0/TCP 10.47.197.7:5060;branch=z9hG4bK7251784nf
Call-ID: 3ebf09e73b83408f@10.47.197.7
CSeq: 1 INVITE
Contact: <sip:alice@example.com>
From: "Alice" <sip:alice@example.com>;tag=6dle0bf6d948f0b8
To: <sip:bob@example.com>
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 525

v=0
o=alice 1 3 IN IP4 10.47.197.7
s=-
c=IN IP4 10.47.197.7
b=AS:6000
t=0 0
a=extmap:1 urn:ietf:params:clue:demux
m=audio 1000 RTP/AVP 101 0
b=TIAS:64000
a=rtpmap:101 MP4A-LATM/90000
a=fmtp:101 profile-level-id=25;object=23;bitrate=64000
a=rtpmap:0 PCMU/8000
a=sendrecv
m=video 1002 RTP/AVP 98
a=rtpmap:98 H264/90000
a=fmtp:98 profile-level-id=42e016;max-mps=35000;max-fs=3600
a=sendrecv
m=application 1004 UDP/SCTP/CLUE *
a=setup:actpass
a=connection:new
```

A.2. 200 OK from Bob

```
SIP/2.0 200 OK
Via: SIP/2.0/TCP 10.47.197.7:5060;branch=z9hG4bK7251784nf
Call-ID: 3ebf09e73b83408f@10.47.197.7
CSeq: 1 INVITE
From: "Alice" <sip:alice@example.com>;tag=6d1e0bf6d948f0b8
To: <sip:bob@example.com>;tag=b91d0ea4
Contact: <sip:bob@example.com>;isfocus
Content-Type: application/sdp
Content-Length: 474

v=0
o=bob 45727 45727 IN IP4 10.47.196.161
s=-
c=IN IP4 10.47.196.161
b=AS:10000
t=0 0
a=extmap:3 urn:ietf:params:clue:demux
m=audio 3778 RTP/AVP 101 0
a=rtpmap:101 MP4A-LATM/90000
a=fmtp:101 profile-level-id=25;object=23;bitrate=64000
a=sendrecv
m=video 3780 RTP/AVP 98 99 34 31
a=rtpmap:98 H264/90000
a=fmtp:98 profile-level-id=42e016;max-mps=244800;max-fs=8160
a=sendrecv
m=application 3782 UDP/SCTP/CLUE *
a=setup:passive
a=connection:new
```

A.3. CLUE advertisement A (from Alice)

```
<advertisement sequence='100'>
  <capture-scene id='1'>
    <spatial-description>
      <scale>millimeters</scale>
    </spatial-description>
    <capture id='1'>
      <media-type>video</media-type>
      <content-type>main</content-type>
      <encoding-group-id>1</encoding-group-id>
      <spatial-description>
        <point-of-capture>
          <point x='0' y='0' z='0' />
        </point-of-capture>
      </spatial-description>
    </capture>
  </capture-scene>
</advertisement>
```

```
</point-of-capture>
<capture-area>
  <point x='-1000' y='0' z='3000' />
  <point x='1000' y='-1125' z='3000' />
  <point x='-1000' y='-1125' z='3000' />
  <point x='1000' y='0' z='3000' />
</capture-area>
</spatial-description>
</capture>
<capture id='2'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <spatial-description>
    <point-of-capture>
      <point x='0' y='0' z='0' />
    </point-of-capture>
    <capture-area>
      <point x='1000' y='0' z='3000' />
      <point x='2600' y='-1125' z='1800' />
      <point x='1000' y='-1125' z='3000' />
      <point x='2600' y='0' z='1800' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='3'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <spatial-description>
    <point-of-capture>
      <point x='0' y='0' z='0' />
    </point-of-capture>
    <capture-area>
      <point x='-1000' y='0' z='3000' />
      <point x='-2600' y='-1125' z='1800' />
      <point x='-1000' y='-1125' z='3000' />
      <point x='-2600' y='0' z='1800' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='4'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
</capture>
<capture id='5'>
```



```

    <media-type>audio</media-type>
    <content-type>main</content-type>
    <encoding-group-id>2</encoding-group-id>
    <mixed>true</mixed>
  </capture>
  <entries>
    <entry>
      <capture id='1' />
      <capture id='2' />
      <capture id='3' />
    </entry>
    <entry>
      <capture id='4' />
    </entry>
    <entry>
      <capture id='5' />
    </entry>
  </entries>
</capture-scene>
<simultaneous-transmission-set />
<encoding-group id='1'>
  <max-bandwidth>6000000</max-bandwidth>
  <encoding id='1'>
    <max-bandwidth>4000000</max-bandwidth>
  </encoding>
  <encoding id='2'>
    <max-bandwidth>4000000</max-bandwidth>
  </encoding>
  <encoding id='3'>
    <max-bandwidth>4000000</max-bandwidth>
  </encoding>
</encoding-group>
<encoding-group id='2'>
  <encoding id='4'>
    <max-bandwidth>64000</max-bandwidth>
  </encoding>
</encoding-group>
</advertisement>

```

A.4. CLUE advertisement B (from Bob)

```

<advertisement sequence='1'>
  <capture-scene id='1'>
    <spatial-description>
      <scale>No Scale</scale>
      <scene-area>

```

```
<point x='0' y='0' z='0' />
<point x='1' y='1' z='0' />
<point x='0' y='1' z='0' />
<point x='1' y='0' z='0' />
</scene-area>
</spatial-description>
<capture id='1'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0' y='0' z='0' />
      <point x='1' y='1' z='0' />
      <point x='0' y='1' z='0' />
      <point x='1' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='2'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0' y='0' z='0' />
      <point x='0.5' y='1' z='0' />
      <point x='0' y='1' z='0' />
      <point x='0.5' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='3'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.5' y='0' z='0' />
      <point x='1' y='1' z='0' />
      <point x='0.5' y='1' z='0' />
      <point x='1' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
```

```
<capture id='4'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0' y='0' z='0' />
      <point x='0.33' y='1' z='0' />
      <point x='0' y='1' z='0' />
      <point x='0.33' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='5'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.33' y='0' z='0' />
      <point x='0.67' y='1' z='0' />
      <point x='0.33' y='1' z='0' />
      <point x='0.67' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='6'>
  <media-type>video</media-type>
  <content-type>main</content-type>
  <encoding-group-id>1</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.67' y='0' z='0' />
      <point x='1' y='1' z='0' />
      <point x='0.67' y='1' z='0' />
      <point x='1' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='7'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
```

```
<capture-area>
  <point x='0' y='0' z='0' />
  <point x='1' y='1' z='0' />
  <point x='0' y='1' z='0' />
  <point x='1' y='0' z='0' />
</capture-area>
</spatial-description>
</capture>
<capture id='8'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0' y='0' z='0' />
      <point x='0.5' y='1' z='0' />
      <point x='0' y='1' z='0' />
      <point x='0.5' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='9'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.5' y='0' z='0' />
      <point x='1' y='1' z='0' />
      <point x='0.5' y='1' z='0' />
      <point x='1' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='10'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0' y='0' z='0' />
      <point x='0.33' y='1' z='0' />
      <point x='0' y='1' z='0' />
      <point x='0.33' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
```

```
</spatial-description>
</capture>
<capture id='11'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.33' y='0' z='0' />
      <point x='0.67' y='1' z='0' />
      <point x='0.33' y='1' z='0' />
      <point x='0.67' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<capture id='12'>
  <media-type>audio</media-type>
  <content-type>main</content-type>
  <encoding-group-id>2</encoding-group-id>
  <switched>true</switched>
  <spatial-description>
    <capture-area>
      <point x='0.67' y='0' z='0' />
      <point x='1' y='1' z='0' />
      <point x='0.67' y='1' z='0' />
      <point x='1' y='0' z='0' />
    </capture-area>
  </spatial-description>
</capture>
<entries>
  <entry>
    <capture id='1' />
  </entry>
  <entry>
    <capture id='2' />
    <capture id='3' />
  </entry>
  <entry>
    <capture id='4' />
    <capture id='5' />
    <capture id='6' />
  </entry>
  <entry>
    <capture id='7' />
  </entry>
  <entry>
    <capture id='8' />
  </entry>
</entries>
```

```
        <capture id='9' />
    </entry>
    <entry>
        <capture id='10' />
        <capture id='11' />
        <capture id='12' />
    </entry>
</entries>
</capture-scene>
<simultaneous-transmission-set />
<encoding-group id='1'>
    <encoding id='1' />
    <encoding id='2' />
    <encoding id='3' />
</encoding-group>
<encoding-group id='2'>
    <encoding id='1' />
    <encoding id='2' />
    <encoding id='3' />
</encoding-group>
</advertisement>
```

A.5. CLUE response A (from Alice)

```
<configure sequence='10' advertise-sequence='1'>
  <capture id='4'>
    <multiplex-id>11</multiplex-id>
    <encoding-id>1</encoding-id>
    <max-bandwidth>2000000</max-bandwidth>
  </capture>
  <capture id='5'>
    <multiplex-id>22</multiplex-id>
    <encoding-id>2</encoding-id>
    <max-bandwidth>2000000</max-bandwidth>
  </capture>
  <capture id='6'>
    <multiplex-id>44</multiplex-id>
    <encoding-id>3</encoding-id>
    <max-bandwidth>2000000</max-bandwidth>
  </capture>
  <capture id='8'>
    <multiplex-id>111</multiplex-id>
    <encoding-id>1</encoding-id>
  </capture>
  <capture id='9'>
    <multiplex-id>222</multiplex-id>
    <encoding-id>2</encoding-id>
  </capture>
</configure>
```

A.6. CLUE response B (from Bob)

```
<configure sequence='1' advertise-sequence='100'>
  <capture id='1'>
    <multiplex-id>1</multiplex-id>
    <encoding-id>1</encoding-id>
  </capture>
  <capture id='2'>
    <multiplex-id>2</multiplex-id>
    <encoding-id>2</encoding-id>
  </capture>
  <capture id='3'>
    <multiplex-id>3</multiplex-id>
    <encoding-id>3</encoding-id>
  </capture>
  <capture id='5'>
    <multiplex-id>1</multiplex-id>
    <encoding-id>4</encoding-id>
  </capture>
</configure>
```

Appendix B. Changes From Earlier Versions

Note to the RFC-Editor: please remove this section prior to publication as an RFC.

B.1. Changes From Draft -00

- o Editorial changes including proper references.

B.2. Changes From Draft -01

- o More editorial changes.
- o Edited mid-call messages and feature interactions section. Introduced option tags.
- o Cleaned up discussion of tags in RTP

Authors' Addresses

Robert Hansen
Cisco Systems
Langley,
UK

Email: rohanse2@cisco.com

Arun Krishna
Cisco Systems
San Jose, CA
USA

Email: arukrish@cisco.com

Allyn Romanow
Cisco Systems
San Jose, CA 95134
USA

Email: allyn@cisco.com

CLUE
Internet-Draft
Intended status: Standards Track
Expires: December 8, 2012

A. Romanow
Cisco Systems
A. Pepperell
Silverflare
June 6, 2012

Data model for the CLUE Framework
draft-romanow-clue-data-model-01

Abstract

This draft is for discussion in the CLUE working group. It proposes a data model for the CLUE Framework.

Status of this Memo

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

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

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 8, 2012.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Overview	4
3.1. Data model format	4
3.2. Data model namespace	4
3.3. Data Model Structure	4
4. Data Model Definition	4
4.1. <CLUE-info>	6
4.2. <capture-description>	6
4.3. <capture-id>	6
4.4. <media-type>	6
4.5. <content-type>	6
4.6. <DERIVED>	6
4.7. <switched>	6
4.8. <spatial-description>	7
4.8.1. <point-of-capture>	7
4.8.2. <capture-area>	7
4.8.3. <NATIVE-ASPECT_RATIO>	7
4.9. <audio-channel-format>	7
4.10. <encoding-group-id>	7
4.11. <future-media-capture-attribute>	7
4.12. <capture-scene>	7
4.12.1. <capture-scene-text>	8
4.12.2. <capture-scene-language>	8
4.12.3. <capture-scene-spatial-description>	8
4.12.4. <future-capture-scene-attribute>	8
4.13. <simultaneous transmission-set>	8
4.14. <stream-description>	9
4.14.1. <encoding>	9
4.14.2. <encoding-group>	10
5. Security Considerations	10
6. IANA Considerations	11
7. References	11
7.1. Normative References	11
7.2. Informative References	11
Authors' Addresses	11

1. Introduction

This document proposes a data model for the CLUE Framework [I-D.ietf-clue-framework].

The previous suggestion for a data model described the 2 CLUE messages, provider advertisement and consumer choice message. The data model proposed here differs in that it describes the basic information, or definitions that are needed. Messages can then be derived from the data model, similarly to what was described in the earlier data model.

The provider advertisement and the consumer choice messages can use or not use any of the data elements defined in the data model, subject only to whether or not the element is deemed mandatory.

The model here follows the example of the xcon data model RFC 6501 [RFC6501].

Initially the data model may seem to describe only provider advertisements. But this is not the case. The consumer choice messages can also be constructed from these elements. Many of the data elements are similar for both messages, but there are also elements defined which are used only in one or the other of the messages.

One of the challenges here is that in addition to proposing a structure for the data model, we would like to also propose some additions or changes to what is in the current framework document. The problem is that any changes from what is in the current framework can detract attention from the basic information model which is the primary focus of this draft initially. We have noted where there are changes from the framework by putting the text in all capital text.

This is a rough draft, its goal is to propose a basic structure and stimulate discussion.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

3. Overview

TEXT

3.1. Data model format

A CLUE object document is an XML [W3C.REC-xml-20081126] document. CLUE object documents MUST be based on XML 1.0 and MUST be encoded using UTF-8.

3.2. Data model namespace

This specification defines a new namespace specification for identifying the elements defined in the data model. This namespace is as follows: urn:ietf:params:xml:ns:clue-info

3.3. Data Model Structure

The information in this data model is structured in the following manner. All the information communicated between consumers and providers in a CLUE session is contained in a <CLUE-info> element. The definitions follow directly from the Framework document.

The <CLUE-info> element contains the following child elements:

- o The <capture-description> element describes the captures. It includes all the elements that have been discussed in the framework document as properties of captures document and it can be extended by adding additional elements describing a capture.
- o The <capture-scene> element is the data structure that describes the captures that the provider is capable of sending in a way that is helpful to the consumer to decide what captures to request.
- o The <simultaneous-transmission-set> element is a data structure that describes the physical simultaneity constraints, as discussed in the framework.
- o The <stream-description> element describes the encoding characteristics of the streams.

4. Data Model Definition

The following non-normative diagram shows the structure of CLUE elements. The symbol "!" preceding an element indicates that the element is REQUIRED in the data model.

```

!<CLUE info>
|
|--<capture-description>
|   |--<capture-id>
|   |--<media-type>
|   |--<content-type>
|   |--<DERIVED>
|   |--<switched>
|   |--<spatial-description>
|       |--<point-of-capture>
|       |--<capture-area>
|       |--<NATIVE-ASPECT-RATIO>
|   |--<audio-channel-format>
|   |--<encoding-group-ID>
|   |--<future-media-capture-attribute>
|--<capture-scene>
|   |--<entry>
|       |--<capture-id>
|       |--<switch-policy>
|   |--<capture-scene-text>
|   |--<capture-scene-language>
|   |--<capture-scene-spatial-description>
|       |--<capture-scene-area>
|       |--<capture-scene-scale>
|   |--<future-capture-scene-attribute>
|--<simultaneous-transmission-set>
|   |--<entry>
|       |--<capture-id>
|--<stream-description>
|   |--<encoding>
|       |--<encoding-id>
|       |--<multiplex-id>
|       |--<AUDIO-RENDERING-ID>
|       |--<max-audio-bandwidth>
|       |--<max-video-bandwidth>
|       |--<max-H264-Mbps>
|       |--<max-width>
|       |--<max-height>
|       |--<max-frame-rate>
|       |--<future-encoding-attribute>
|   |--<encoding-group>
|       |--<encoding-group-id>
|       |--<entry>
|           <video encoding>
|       |--><entry>
|           <audio encoding>
|       |--<max-group-bandwidth>
|       |--<max-group-H264-Mbps>

```

| | |--<future-encoding-group-attribute>

Data Model Definition

The following sections describe these elements in more detail.

4.1. <CLUE-info>

4.2. <capture-description>

The capture-description describes all the aspects of a capture, as discussed in the framework document.

4.3. <capture-id>

This element is a unique numeric value assigned to each capture by the provider. It is used by the consumer in choosing captures and streams.

4.4. <media-type>

This element indicates support by the consumer for a single capture media type, for instance "audio" or "video".

4.5. <content-type>

This per-capture attribute element indicates whether the media capture includes "main media", typically motion video of a real scene, or whether it includes "slides media", typically a computer-generated video stream rather than real people.

This is following the content attribute RFC [ref].

4.6. <DERIVED>

Instead of the boolean composed, we are suggesting an attribute that indicates whether media capture has been changed from its original form. It can take different values. As of now, the only value defined is composed. Composed means a video image it is formed of other, more primary, captures mixed together. For audio it refers to mixed audio.[this needs to follow whatever the WG decides as to whether there is an audio composed, etc.]

4.7. <switched>

Instead of a boolean, we propose this attribute indicates the algorithm or method by which content is chosen from media sources.

The current value is loudest speaker. [this depends entirely on what the WG decides]

4.8. <spatial-description>

Spatial-description is a general element with sub-elements that describe spatial relationships. As of now, the framework defines point-of-capture and capture-area. We propose a third one here-NATIVE-ASPECT-RATIO.

4.8.1. <point-of-capture>

Use framework text.

4.8.2. <capture-area>

Use framework text.

4.8.3. <NATIVE-ASPECT_RATIO>

If a capture has a native aspect ratio (for instance, it corresponds to a camera that generates 4:3 video) then it can be supplied here. This can help rendering.

4.9. <audio-channel-format>

Use framework text.

4.10. <encoding-group-id>

This element is a unique numeric value assigned to each encoding group by the provider. It is included in a capture element because it links the capture to the group of encodings possible to use for the capture.

4.11. <future-media-capture-attribute>

This is a placeholder for media capture attributes that have yet to be defined.

4.12. <capture-scene>

A capture-scene is a description of the captures that the provider is capable of sending. It includes entries which are rows of audio or video captures.

Information concerning the relationship between captures is communicated by the capture-scene, that is, an entry in the capture-

scene signifies a complete representation of the total scene. When there are multiple entries of the same media type in the capture scene, this means that each entry is an alternate representation of the total scene.

Some elements of the capture-scene characterize the scene, as for example, capture-text.

An entry is a list of video captures or a list of audio captures, and an indication of the switching policy. Every <entry> element contains the following child elements:

- o <capture-id> See above.
- o <switch-policy> text from framework.

4.12.1. <capture-scene-text>

Text from framework

4.12.2. <capture-scene-language>

Text from framework

4.12.3. <capture-scene-spatial-description>

4.12.3.1. <capture-scene-area>

Indicates an overall encompassing co-ordinate "bounding box" at a per-capture set level.

4.12.3.2. capture-scene-scale>

On a per-capture scene level, this indicates whether the co-ordinates used for its media captures are in real-world units, relative units, or whether no scale information can be inferred.

4.12.4. <future-capture-scene-attribute>

This is a placeholder for capture set attributes that have yet to be defined.

4.13. <simultaneous transmission-set>

Indicates physical simultaneity constraints. Use framework text. Every <entry> element contains the following child elements:

- o <capture-id> As above

4.14. <stream-description>

4.14.1. <encoding>

4.14.1.1. <encoding-id>

Use framework text. A unique value for the set of encoding parameters that constitute an encoding stream..

4.14.1.2. < MULTIPLEX-ID>

This element is the means by which the consumer, the receiver of a stream, tells the provider which multiplex id to generate that stream with, so that the consumer can demultiplex the stream correctly when receiving it.

4.14.1.3. <AUDIO-RENDERING-TAG>

Optional id for rendering audio supplied by consumer to the provider. The provider then uses this tag to to associate an audio stream with a video stream.

4.14.1.4. <max-audio-bandwidth>

This element gives the bandwidth limit for any audio stream. It can be used in provider advertisements with respect to potential streams, or in consumer request messages, describing a stream choice.

4.14.1.5. <max-video-bandwidth>

This element gives the provider's bandwidth limit for any video stream. It can be used in provider advertisements with respect to potential streams, or in consumer request messages, describing a stream choice.

4.14.1.6. <max-H264-Mbps>

This element gives the provider's maximum macroblock per second rate for any video stream.

4.14.1.7. <max-width>

Framework text

4.14.1.8. <max-height>

Framework text

4.14.1.9. <max-frame-rate>

This element gives a maximum frame rate for a video stream.

4.14.1.10. <future-encoding-attribute>

4.14.2. <encoding-group>

Encoding group is an element used by the provider to describe its capabilities. It groups together the potential encodings that are available for a particular capture. Therefore, a capture has associated with it an encoding-id.

Every <entry> element contains the following child elements:

- o A <video-encoding> is an encoding for video.
- o An <audio-encoding> is an encoding for audio.

4.14.2.1. <encoding-group-id>

A unique id for the group of encodings that can be used for a particular capture.

4.14.2.2. <max-group-bandwidth>

This indicates the maximum bandwidth that the provider can transmit for its combined set of active encodings.

4.14.2.3. <max-group-H264-Mbps>

The maximum number of macroblocks per second that the provider can transmit for its combined set of active encodings.

4.14.2.4. <future-encoding-group-attribute>

This is a placeholder for encoding group attributes that have yet to be defined.

5. Security Considerations

TBD

6. IANA Considerations

TBD

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6501] Novo, O., Camarillo, G., Morgan, D., and J. Urpalainen, "Conference Information Data Model for Centralized Conferencing (XCON)", RFC 6501, March 2012.

7.2. Informative References

- [I-D.ietf-clue-framework]
Romanow, A., Duckworth, M., Pepperell, A., and B. Baldino, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-05 (work in progress), May 2012.
- [I-D.lennox-clue-rtp-usage]
Lennox, J., Witty, P., and A. Romanow, "Real-Time Transport Protocol (RTP) Usage for Telepresence Sessions", draft-lennox-clue-rtp-usage-04 (work in progress), June 2012.

Authors' Addresses

Allyn Romanow
Cisco Systems
San Jose, CA 95134
USA

Email: allyn@cisco.com

Andy Pepperell
Silverflare

Email: andy.pepperell@silverflare.com

