Centralized Conference Control Protocol: Audio-Video Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation ("this documentation") for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.

- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.

- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.

- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft Open Specifications Promise or the Microsoft Community Promise. If you would prefer a written license, or if the technologies described in this documentation are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.

- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the Patent Map.

- **Trademarks.** The names of companies and products contained in this documentation might be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit www.microsoft.com/trademarks.

- **Fictitious Names.** The example companies, organizations, products, domain names, email addresses, logos, people, places, and events that are depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights.** All other rights are reserved, and this notice does not grant any rights other than as specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications documentation does not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments, you are free to take advantage of them. Certain Open Specifications documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact dochelp@microsoft.com.
## Revision Summary

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision History</th>
<th>Revision Class</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/4/2008</td>
<td>0.1</td>
<td>New</td>
<td>Initial version</td>
</tr>
<tr>
<td>4/25/2008</td>
<td>0.2</td>
<td>Minor</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/27/2008</td>
<td>1.0</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>8/15/2008</td>
<td>1.01</td>
<td>Minor</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>12/12/2008</td>
<td>2.0</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>2/13/2009</td>
<td>2.01</td>
<td>Minor</td>
<td>Edited the technical content</td>
</tr>
<tr>
<td>3/13/2009</td>
<td>2.02</td>
<td>Minor</td>
<td>Edited the technical content</td>
</tr>
<tr>
<td>7/13/2009</td>
<td>2.03</td>
<td>Major</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>8/28/2009</td>
<td>2.04</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>11/6/2009</td>
<td>2.05</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>2/19/2010</td>
<td>2.06</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>3/31/2010</td>
<td>2.07</td>
<td>Major</td>
<td>Updated and revised the technical content</td>
</tr>
<tr>
<td>4/30/2010</td>
<td>2.08</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/7/2010</td>
<td>2.09</td>
<td>Editorial</td>
<td>Revised and edited the technical content</td>
</tr>
<tr>
<td>6/29/2010</td>
<td>2.10</td>
<td>Editorial</td>
<td>Changed language and formatting in the technical content.</td>
</tr>
<tr>
<td>7/23/2010</td>
<td>2.10</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/27/2010</td>
<td>3.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>11/15/2010</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>12/17/2010</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/18/2011</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>6/10/2011</td>
<td>3.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>1/20/2012</td>
<td>4.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>4/11/2012</td>
<td>4.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/16/2012</td>
<td>4.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/8/2012</td>
<td>4.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/11/2013</td>
<td>4.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>Date</td>
<td>Revision History</td>
<td>Revision Class</td>
<td>Comments</td>
</tr>
<tr>
<td>--------------</td>
<td>------------------</td>
<td>----------------</td>
<td>----------------------------------------------------</td>
</tr>
<tr>
<td>7/30/2013</td>
<td>4.1</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>11/18/2013</td>
<td>4.2</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>2/10/2014</td>
<td>4.2</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/30/2014</td>
<td>4.3</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
<tr>
<td>7/31/2014</td>
<td>4.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>10/30/2014</td>
<td>4.3</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>3/30/2015</td>
<td>5.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>9/4/2015</td>
<td>5.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>7/15/2016</td>
<td>5.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/14/2016</td>
<td>5.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>9/29/2016</td>
<td>5.0</td>
<td>None</td>
<td>No changes to the meaning, language, or formatting of the technical content.</td>
</tr>
<tr>
<td>4/27/2018</td>
<td>6.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>7/24/2018</td>
<td>7.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>8/28/2018</td>
<td>8.0</td>
<td>Major</td>
<td>Significantly changed the technical content.</td>
</tr>
<tr>
<td>12/11/2018</td>
<td>8.1</td>
<td>Minor</td>
<td>Clarified the meaning of the technical content.</td>
</tr>
</tbody>
</table>
# Table of Contents

1 Introduction .................................................................................................................. 7  
1.1 Glossary ...................................................................................................................... 7  
1.2 References ................................................................................................................... 9  
1.2.1 Normative References ............................................................................................. 9  
1.2.2 Informative References ............................................................................................ 9  
1.3 Overview ..................................................................................................................... 10  
1.3.1 Overview of Conceptual Conference Document Structure .................................... 10  
1.3.2 Scope ....................................................................................................................... 11  
1.4 Relationship to Other Protocols ................................................................................ 12  
1.5 Prerequisites/Preconditions ....................................................................................... 12  
1.6 Applicability Statement .............................................................................................. 12  
1.7 Versioning and Capability Negotiation ....................................................................... 12  
1.8 Vendor-Extensible Fields ........................................................................................... 13  
1.9 Standards Assignments .............................................................................................. 13  

2 Messages ........................................................................................................................... 14  
2.1 Transport .................................................................................................................... 14  
2.2 Message Syntax .......................................................................................................... 14  
2.2.1 Extension Semantics of application/conference-info+xml Document Format .......... 14  
2.2.1.1 XML Schema Types used in A/V Conference Modalities .................................. 14  
2.2.1.1.1 Media Filter Types .............................................................................................. 14  
2.2.1.1.2 video-parameters-type* ....................................................................................... 15  
2.2.1.1.3 capabilities-type* ................................................................................................. 15  
2.2.1.1.4 entry-exit-announcements type .............................................................. 16  
2.2.1.1.5 media-filters-rules-type ...................................................................................... 16  
2.2.1.1.5.1 mayModifyOwnFilters ................................................................................. 17  
2.2.1.1.5.2 initialFilters ..................................................................................................... 18  
2.2.1.1.5.3 type ................................................................................................................ 18  
2.2.2 MCU Conference Roster Document Format ......................................................... 19  
2.2.2.1 MCU endpoint Element Syntax ........................................................................... 19  
2.2.2.1.1 endpoint Element Semantic Extensions ...................................................... 19  
2.2.2.1.1.1 media Element Instances .............................................................................. 19  
2.2.2.1.2 endpoint Element Extension Elements ....................................................... 20  
2.2.2.1.2.1 media-ingress-filter Element ........................................................................ 20  
2.2.2.1.2.1 media-ingress-filter Element ........................................................................ 20  
2.2.2.1.2.3 media-source-id Element ............................................................................ 20  
2.2.2.1.2.4 source-name Element .................................................................................. 20  
2.2.2.2 MCU conference-view Element Syntax ............................................................... 20  
2.2.2.2.1 entity-state Extension Elements ....................................................................... 20  
2.2.2.2.1.1 media Element Extensions ........................................................................... 21  
2.2.2.2.2 entry-exit-announcements .............................................................................. 21  
2.2.2.2.2.1 presentation-mode-capable ........................................................................... 21  
2.2.2.2.2.2 mediaFiltersRules ......................................................................................... 22  
2.2.2.2.2.3 multi-view-capable Element ....................................................................... 22  
2.2.2.2.2.4 video-presentation-mode-capable Element ................................................. 22  
2.2.2.2.2.5 conf-media-filters-rules ............................................................................... 22  
2.2.3 C3P request/response Document Content ............................................................ 23  
2.2.3.1 addUser Dial-out Request Document Syntax ..................................................... 23  
2.2.3.1.1 endpoint Element ............................................................................................... 23  
2.2.3.1.2 media Element ................................................................................................. 23  
2.2.3.2 addUser Dial-in Request Document Syntax ........................................................ 23
3 Protocol Details ........................................................................................................26
  3.1 Client Details ........................................................................................................26
    3.1.1 Abstract Data Model .........................................................................................26
    3.1.2 Timers ..............................................................................................................26
    3.1.3 Initialization ......................................................................................................26
    3.1.4 Higher-Layer Triggered Events ........................................................................26
    3.1.5 Message Processing Events and Sequencing Rules .........................................26
      3.1.5.1 Constructing the Outgoing addUser Dial-in Request .................................26
      3.1.5.2 Constructing the Outgoing SIP INVITE Dial-in Request ............................26
      3.1.5.2.1 Constructing the SDP Offer in the Outgoing SIP INVITE Message ........27
      3.1.5.3 Constructing the Outgoing addUser Dial-out Request ...............................27
    3.1.6 Timer Events ....................................................................................................27
    3.1.7 Other Local Events ..........................................................................................27
  3.2 Server Details .........................................................................................................28
    3.2.1 Abstract Data Model .........................................................................................28
    3.2.1.1 Correlation of Media Parameters .................................................................28
    3.2.1.2 Correlation of Media Instances .....................................................................29
    3.2.2 Timers ..............................................................................................................30
    3.2.3 Initialization ......................................................................................................30
    3.2.3.1 Conference Activation (MCU Bootstrap) ......................................................30
      3.2.3.1.1 Initial Full Conference Notification .........................................................30
        3.2.3.1.1.1 entity-capabilities Element .................................................................30
        3.2.3.1.1.2 Child Elements of the entity-state Element .........................................30
          3.2.3.1.1.2.1 entry-exit-announcements element ................................................30
          3.2.3.1.1.2.2 mediaFiltersRules element ..............................................................31
          3.2.3.1.1.2.3 presentation-mode-capable element ...............................................31
          3.2.3.1.1.2.4 media Element ...............................................................................31
          3.2.3.1.1.2.5 multi-view-capable Element ............................................................32
          3.2.3.1.1.2.6 video-presentation-mode-capable Element .....................................32
          3.2.3.1.1.2.7 conf-media-filters-rules Element ......................................................32
    3.2.4 Higher-Layer Triggered Events ........................................................................34
    3.2.5 Message Processing Events and Sequencing Rules .........................................34
      3.2.5.1 Common Rules for Processing SDP Offers and Answers ............................34
        3.2.5.1.1 Generating an Initial SDP Offer ...............................................................34
        3.2.5.1.2 Correlation of Offered SDP Media Instances .........................................35
        3.2.5.1.3 Processing a Received SDP Offer .............................................................36
        3.2.5.1.4 Processing a Received SDP Answer ........................................................37
      3.2.5.2 addUser Dial-out Request .........................................................................38
        3.2.5.2.1 Constructing the Outgoing SIP INVITE Request ....................................38
        3.2.5.2.2 Construction of SDP Contents .................................................................38
      3.2.5.3 addUser Dial-in Request .............................................................................38
        3.2.5.3.1 Constructing the addUser Dial-in Response ...........................................39
      3.2.5.4 modifyEndpointMedia Request ................................................................39
      3.2.5.5 modifyConferenceAnnouncements Request ...........................................41
      3.2.5.6 modifyConference Request .....................................................................41
        3.2.5.6.1 Handling media-filters-rules type in modifyConference Request ........42
        3.2.5.6.2 Handling video-parameters-type in modifyConference Request ............43
    3.2.6 Timer Events ....................................................................................................43
    3.2.7 Other Local Events .........................................................................................44
      3.2.7.1 User signaling (SIP dialog) Events .............................................................44
        3.2.7.1.1 Receipt of an Initial SDP Answer in SIP 200-OK Message Sent as Response to addUser Dial-out INVITE .......................................................44
        3.2.7.1.2 Receipt of Initial SIP INVITE Messages (Dial-in User join) ....................44
1 Introduction

The Centralized Conference Control Protocol: Audio-Video Extensions protocol specifies proprietary extensions to the Centralized Conference Control Protocol that can be used to integrate audio and video conference modes within the framework described in [MS-CONFBS].

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

1.1 Glossary

This document uses the following terms:

200 OK: A response to indicate that the request has succeeded.

Audio/Video Multipoint Control Unit (AVMCU): A Multipoint Control Unit (MCU) that supports audio-video (AV) conferencing.

codec: An algorithm that is used to convert media between digital formats, especially between raw media data and a format that is more suitable for a specific purpose. Encoding converts the raw data to a digital format. Decoding reverses the process.

conference: A Real-Time Transport Protocol (RTP) session that includes more than one participant.

data type: A property of a field that defines the kind of data that is stored in the field, or defines the kind of data returned by an expression when the expression is evaluated.

dialog: A peer-to-peer Session Initiation Protocol (SIP) relationship that exists between two user agents and persists for a period of time. A dialog is established by SIP messages, such as a 2xx response to an INVITE request, and is identified by a call identifier, a local tag, and a remote tag.

endpoint: A device that is connected to a computer network.

endpoint identifier (EPID): A unique identifier of a Session Initiation Protocol (SIP) endpoint. It is formed by combining the value of an epid parameter in a From or To header field with the address-of-record in the corresponding header field.

focus: A single user agent that maintains a dialog and Session Initiation Protocol (SIP) signaling relationship with each participant, implements conference policies, and ensures that each participant receives the media that comprise the tightly coupled conference.

Interactive Connectivity Establishment (ICE): A methodology that was established by the Internet Engineering Task Force (IETF) to facilitate the traversal of network address translation (NAT) by media.

Internet message: A message, such as an email message, that conforms to the syntax that is described in [RFC2822].

INVITE: A Session Initiation Protocol (SIP) method that is used to invite a user or a service to participate in a session.

MCU-Conference-URI: A literal that specifies a URI that can be used to access conferencing services in the context of a Multipoint Control Unit (MCU).

Media Source ID (MSI): A 32-bit identifier that uniquely identifies an audio or video source in a conference.
**mixer**: An intermediate system that receives a set of media streams of the same type, combines the media in a type-specific manner, and redistributes the result to each participant.

**Multipoint Control Unit (MCU)**: A server endpoint that offers mixing services for multiparty, multiuser conferencing. An MCU typically supports one or more media types, such as audio, video, and data.

**notification**: A process in which a subscribing Session Initiation Protocol (SIP) client is notified of the state of a subscribed resource by sending a NOTIFY message to the subscriber.

**participant**: A user who is participating in a conference or peer-to-peer call, or the object that is used to represent that user.

**Real-Time Transport Protocol (RTP)**: A network transport protocol that provides end-to-end transport functions that are suitable for applications that transmit real-time data, such as audio and video, as described in [RFC3550].

**remote endpoint**: See peer.

**request message**: A Traversal Using Relay NAT (TURN) message that is sent from a protocol client to a protocol server.

**SDP answer**: A Session Description Protocol (SDP) message that is sent by an answerer in response to an offer that is received from an offerer.

**SDP offer**: A Session Description Protocol (SDP) message that is sent by an offerer.

**server**: A replicating machine that sends replicated files to a partner (client). The term "server" refers to the machine acting in response to requests from partners that want to receive replicated files.

**Session Description Protocol (SDP)**: A protocol that is used for session announcement, session invitation, and other forms of multimedia session initiation. For more information see [MS-SDP] and [RFC3264].

**Session Initiation Protocol (SIP)**: An application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. SIP is defined in [RFC3261].

**SIP message**: The data that is exchanged between Session Initiation Protocol (SIP) elements as part of the protocol. An SIP message is either a request or a response.

**Synchronization Source (SSRC)**: A 32-bit identifier that uniquely identifies a media stream in a Real-Time Transport Protocol (RTP) session. An SSRC value is part of an RTP packet header, as described in [RFC3550].

**Uniform Resource Identifier (URI)**: A string that identifies a resource. The URI is an addressing mechanism defined in Internet Engineering Task Force (IETF) Uniform Resource Identifier (URI): Generic Syntax [RFC3986].

**user agent client (UAC)**: A logical entity that creates a new request, and then uses the client transaction state machinery to send it. The role of UAC lasts only for the duration of that transaction. In other words, if a piece of software initiates a request, it acts as a UAC for the duration of that transaction. If it receives a request later, it assumes the role of a user agent server (UAS) for the processing of that transaction.

**XML**: The Extensible Markup Language, as described in [XML1.0].

**XML element**: An XML structure that typically consists of a start tag, an end tag, and the information between those tags. Elements can have attributes and can contain other elements.
**XML schema**: A description of a type of XML document that is typically expressed in terms of constraints on the structure and content of documents of that type, in addition to the basic syntax constraints that are imposed by XML itself. An XML schema provides a view of a document type at a relatively high level of abstraction.

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT**: These terms (in all caps) are used as defined in [RFC2119]. All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

### 1.2 References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the Errata.

#### 1.2.1 Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact dochelp@microsoft.com. We will assist you in finding the relevant information.

- [MS-SDPEXT] Microsoft Corporation, "Session Description Protocol (SDP) Version 2.0 Extensions".

#### 1.2.2 Informative References

- [MS-ICE2] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions 2.0".
- [MS-ICE] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions".
- [MS-RTPRADEX] Microsoft Corporation, "RTP Payload for Redundant Audio Data Extensions".
- [MS-RTP] Microsoft Corporation, "Real-time Transport Protocol (RTP) Extensions".
1.3 Overview


Within [MS-CONFBAS] section 2.2.2.4, centralized processing of conference media content is delegated to specialized media-type-specific MCUs. For example, a multiparty conference that simultaneously encompasses sending Internet messages, data and application sharing, and audio-video media types is processed by three separate logical MCU entities: one for Internet messages, one for data and application sharing, and one for audio and video.

This document specifies extensions to [MS-CONFBAS] that relate to audio and video media content that is transferred using the Real-Time Transport Protocol (RTP) and Interactive Connectivity Establishment (ICE).

To put the scope of the extensions specified in this document in perspective, it is helpful to start with a conceptual view of how the extensions described in [MS-CONFBAS] define the effective scope of the separate logical MCU entities with respect to the contents of the Conference Document.

1.3.1 Overview of Conceptual Conference Document Structure

[MS-CONFBAS] describes extensions to the XML schema of the Conference Document that were originally described in [RFC4575]. Central to those extensions is the representation of separate logical focus, or MCU, entities in the structure of the Conference Document.

In general:

- Each MCU independently maintains a list of users, with exactly one endpoint for each user. Each endpoint represents a media-specific communication session between the MCU and one user.

- Separate containers represented by entity-view elements exist for each logical MCU entity as described in [MS-CONFBAS] section 2.2.2.7. The conference information that falls within the scope of a single logical MCU entity generally resides in this container. MCU-specific endpoints are the main exception, as noted in the preceding paragraph).

[MS-CONFBAS] redefines how conference media is represented relative to [RFC4575]. [MS-CONFBAS] and [RFC4575] use essentially the same underlying XML data types for conference and user media instances.

The Conference Document structure described in [MS-CONFBAS] does not place the conference media elements in the same location in the Conference Document as described by [RFC4575]. As a result, the definitions of the media-related data elements in [RFC4575] sections 5.3.4 and 5.8 are interpreted in the context of [MS-CONFBAS].

This interpretation can be summarized as follows:

- Where [RFC4575] describes one container for all conference-wide media, [MS-CONFBAS] describes separate MCU-specific containers. Each [MS-CONFBAS]-defined container is limited in scope to only the conference media instances processed by the designated MCU.
[RFC4575] section 5.3.4 defines the available-media element as the container for conference media. Definitions of endpoint media instances, as described in [RFC4575] section 5.8, refer back to the available-media element.

[MS-CONFBAS] deprecates use of the available-media element in all messages.

[MS-CONFBAS] defines media elements under the MCU-specific entity-view container hierarchy. Each media element is a container for one MCU's conference media instances.

The general form of the underlying XML data types used to represent conference media is the same in [MS-CONFBAS] as it is in [RFC4575]. The XML type conference-medium-type, described in [RFC4575], is the basis for describing a single Conference Media instance.

The semantics of the elements and attributes of conference-medium-type are described in [RFC4575].

The XML schema introduced in [MS-CONFBAS] describes XML schema extensions to conference-medium-type. This protocol defines semantics of the extension elements that are specific to audio/video (A/V) media types.

The collection of all MCU-specific media elements, when taken as a whole, replaces the available-media element described in [RFC4575].

Where [RFC4575] refers to the available-media element in the semantic definition of any element, the reference is to the MCU-specific media element instead.

### 1.3.2 Scope

This protocol defines extensions to [MS-CONFBAS] that enable SIP/Session Description Protocol (SDP)/RTP-based audio and video conference modalities and features within the multiple-MCU architecture that is described in [MS-CONFBAS].

The framework described in [MS-CONFBAS] calls for MCU entities to maintain separate, media type-specific communication sessions with each client. This protocol assumes that the communication protocol for signaling and media handshakes between clients and the logical AV MCU entity is the suite of protocols specified by the following:

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[MS-SIPRE]</td>
<td>Extensions to SIP</td>
</tr>
<tr>
<td>[MS-SDPEXT]</td>
<td>Extensions to SDP</td>
</tr>
<tr>
<td>[RFC3264]</td>
<td>An offer/answer Model with SDP</td>
</tr>
</tbody>
</table>

This protocol defines the necessary interactions between the preceding protocols and the Centralized Conference Control Protocol (C3P), as described in [MS-CONFBAS]. For example, some of the interactions specified in this protocol are as follows:

- Correlation of the C3P conference state and message element and attribute values with SIP URIs and SIP header values.
- Correlation of the C3P conference state and message element and attribute values with the values of standard SDP attributes, as described in [RFC4566] and [RFC3264].
- Processing of C3P commands that result in an action on one or more SIP dialogs between the server, or MCU, and the client or clients.
- Changes in SIP dialog states between the MCU and the client that result in conference state changes and C3P notifications.
Logic rules based on C3P conference state that are factored into media-negotiation behavior whenever a client or MCU formulates an **SDP offer** or responds with an **SDP answer**.

This protocol does not specify any **XML schema** extensions beyond that of [MS-CONFBASES]. It does define semantics of some parts of the XML schema and C3P message constructs in more detail than those described in [MS-CONFBASES], particularly where [MS-CONFBASES] obsoletes, replaces, or deprecates parts of [RFC4575].

Therefore, this protocol does the following:

- Defines semantics for the XML schema extensions to **conference-medium-type** that are described in [MS-CONFBASES].
- Extends the semantics of **conference-medium-type** relative to [RFC4575].

### 1.4 Relationship to Other Protocols

In addition to the dependencies described in [MS-CONFBASES] section 1.4, the following protocols are required components of a complete implementation:

- [MS-SDPEXT]
- [MS-SIPRE]
- [MS-ICE]
- [MS-RTP]
- [MS-RTPRADEX]
- [MS-ICE2]

Note that each of these protocols can be extended independently.

### 1.5 Prerequisites/Preconditions

In addition to the prerequisites and preconditions described in [MS-CONFBASES] section 1.5 and the protocol dependencies specified in section 1.4, this protocol assumes that the client and the **server**:

- Support mutually-interoperable implementations of all of the protocols listed in section 1.4.
- Support at least one **RTP** audio or video payload format in common.

When the client and server meet these requirements, they are able to negotiate a viable, bidirectional RTP channel between them using the standard protocols listed in section 1.4.

### 1.6 Applicability Statement

The extensions specified in this protocol apply when both of the following are true:

- The client and **server** both meet the prerequisites and preconditions in section 1.5.
- The client and server both intend to implement audio or video, or both types, of conference communication modes within the framework and architecture described in [MS-CONFBASES].

### 1.7 Versioning and Capability Negotiation

This protocol does not have any additional versioning and capability negotiation constraints beyond those described in [MS-CONFBASES].
1.8 Vendor-Extensible Fields
None.

1.9 Standards Assignments
None.
2 Messages

2.1 Transport

This protocol does not introduce a new transport to exchange messages. The constraints and conditions for exchanging messages are specified in [MS-CONFBAS].

2.2 Message Syntax

This protocol does not introduce new message formats outside of the encapsulating message structures and envelopes specified in [MS-CONFBAS]. All messages within this section conform to the message syntax specification in [MS-CONFBAS] section 2.2.

Extensions to message content and the associated syntax are discussed in this and subsequent sections.

2.2.1 Extension Semantics of application/conference-info+xml Document Format

The application/conference-info+xml document format details the data model for a conference specified in [RFC4575]. Extensions to [RFC4575] are also specified in [MS-CONFBAS]. This protocol does not introduce any new extensions to the underlying XML schema that is defined in [MS-CONFBAS]. This protocol further extends the conference data model by specifying the semantics of XML elements and attributes that are relevant to the message syntaxes defined in this protocol.

Extensions to this protocol MAY define the semantics of other elements and attributes that they depend upon. They MAY also introduce new extensions to this data model.

Note that not all of the extension element semantics defined by this protocol are exclusively limited to audio and video conference modalities. Unless otherwise specified, extensions to this protocol or other extensions to [RFC4575] and to [MS-CONFBAS] MAY define media type-specific semantics, MCU-specific semantics, or generic media-type-agnostic semantics that are broader and more general in scope than those defined in this protocol.

The cardinality of each extension element is specified in the XML schema using standard minOccurs and maxOccurs XML schema conventions. Similarly, the cardinality of each extension attribute is specified in the XML schema using standard required or optional attributes. Similarly, the namespace of each extension attribute or element is specified in the XML schema using standard conventions and is omitted here for brevity.

This section defines only the general semantics of extension XML data types out of the context of any C3P message. C3P requests, responses, and notifications MAY further define or restrict this data model. Any such restrictions are specified by their message syntaxes as needed.

2.2.1.1 XML Schema Types used in A/V Conference Modalities

Message elements and attributes that have specific semantics with respect to A/V media are specified here. However, it is important to note that not all of the schema extension semantics specified in this protocol are exclusive to A/V media. They are emphasized in this protocol to define them as they apply to RTP audio and RTP video media types. This emphasis also defines the minimal common C3P profile for control of multiparty RTP audio and RTP video conferences.

This section of this protocol defines only the XML constructs and the generic semantics of the XML schema types that this protocol introduces, unless otherwise specified. The data types in this protocol that are intended exclusively for A/V conference modalities are indicated by an asterisk (*) immediately following the element or data type name in each document sub-heading. Those not
indicated by (*) can be used in other conferencing modalities. However, such usage is beyond the scope of this protocol.

### 2.2.1.1.1 Media Filter Types

The following XML types are used to construct rules for allowing or preventing media flow to or from remote **endpoints**, or clients.

#### 2.2.1.1.1.1 Media-Filter-Type

The type **media-filter-type** is a simple enumeration type with two possible values, "block", and "unblock". Elements of this type appear in several other types in the C3P message schema. The schema of the **media-filter-type** is as follows:

```xml
<xs:simpleType name="media-filter-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="block" />
    <xs:enumeration value="unblock" />
  </xs:restriction>
</xs:simpleType>
```

The general semantics of the enumeration values are as follows:

- "block": Media MUST NOT be propagated. This value takes precedence over all other protocol state and behavior specifications.
- "unblock": Allow propagation of media data within any other constraints that are specified by the suite of protocols used in the implementation.

#### 2.2.1.1.2 video-parameters-type*

The XML type **video-parameters-type** is intended specifically for video media types. It is defined in the avconfinfoextensions namespace: http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions.

The schema of **video-parameters-type** is as follows:

```xml
<xs:element name="video-parameters" type="tns:video-parameters-type" ms:ignore="true"/>
<xs:complexType name="video-parameters-type">
  <xs:sequence>
    <xs:element name="video-mode" type="xs:string" minOccurs="0" />
    <xs:element name="intended-primary-presenter-source" type="tns:contributing-sources-type" minOccurs="0" />
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
```

The child elements of the **video-parameters-type** are as follows:

**video-mode**: (*) This element specifies the video processing mode of the **MCU**. The XML type of this element is **xs:string**. The schema does not constrain the value to an enumeration. The base profile defines two literal strings values: "dominant-speaker-switched" and "manual-switched". The "dominant-speaker-switched" value specifies that the video source has not been restricted to a specific source in the conference. The "manual-switched" value specifies that the video source is restricted to either no sources or one source for the **conference**. These are the default modes that implementations SHOULD support. The **video-mode** element is optional. If this element is not present, it is assumed to have a value of "dominant-speaker-switched".
The **intended-primary-presenter-source** is optional. It SHOULD only be present if the **video-mode** is present and has a value of "manual-switched".

If the client supports Video Source Requests as defined in [MS-RTP] section 2.2.12.2, the MCU SHOULD NOT select the video source for that client.

If the client does not support Video Source Requests as defined in [MS-RTP] section 2.2.12.2, the MCU SHOULD dynamically select the video source based on the contents of the audio streams received by the MCU when in "dominant-speaker-switched" mode or select the source specified by **intended-primary-presenter-source** when in "manual-switched" mode.

### 2.2.1.1.2.1 contributing-sources-type

The XML type **contributing-sources-type** is intended specifically for video media types when **video-mode** is "manual-switched". It is defined in the avconfinfoextensions namespace:

http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions

The schema **contributing-sources-type** is as follows:

```xml
<xs:complexType name="contributing-sources-type">
  <xs:sequence>
    <xs:element name="entry" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
  <xs:attribute name="empty" type="xs:boolean" use="optional"/>
</xs:complexType>
```

The elements and attributes of **contributing-sources-type** are as follows:

**entry**: Element, optional, xml type **xs:anyURI**. If present, it identifies the user in the **conference** that will be the only video source available for the conference.

**empty**: Attribute, optional, xml type **xs:Boolean**. If present and has a value of "true", it identifies that the video source specified by the **entry** element is not in the conference. If present and has a value of "false", it identifies that the video source specified by the **entry** element is in the conference. In the context of C3P request messages, this attribute is ignored.

### 2.2.1.1.3 capabilities-type*

The XML type **capabilities-type** is intended specifically for A/V conference modalities. It is defined in the avconfinfoextensions namespace:


The schema for **capabilities-type** is as follows:

```xml
<xs:complexType name="capabilities-type">
  <xs:sequence>
    <xs:element name="supports-audio" type="xs:boolean" />
    <xs:element name="supports-video" type="xs:boolean" />
  </xs:sequence>
</xs:complexType>
```

The child elements of the **capabilities-type** are as follows:

**supports-audio**: A Boolean value that specifies whether or not audio is supported.

**supports-video**: A Boolean value that specifies whether or not video is supported.
2.2.1.1.4 entry-exit-announcements type

The XML type **entry-exit-announcements-type** is intended for the **entity-state-type**. The description of **entity-state-type** is in [MS-CONF BAS] section 2.2.2.7. The **entry-exit-announcements-type** is defined in the commonmcuextensions namespace:


The schema for **entry-exit-announcements-type** is as follows:

```xml
<xs:complexType name="entry-exit-announcements-type">
  <xs:sequence>
    <xs:element name="modifiable" type="xs:boolean" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="enabled" type="xs:boolean" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

The child elements of the **entry-exit-announcements-type** are as follows:

**Enabled** (xs:boolean): A Boolean value that specifies whether entry and exit announcements in the conference will be played or not.

**Modifiable** (xs:boolean): A Boolean value that specifies whether the Enabled element of the **entry-exit-announcements-type** can be changed during the conference or not. If the value is "true", the value of the element Enabled can be changed.

2.2.1.1.5 media-filters-rules-type

The XML type **media-filters-rules-type** is intended for the **entity-state-type**. The description of **entity-state-type** is in [MS-CON FBAS] section 2.2.2.7. The **media-filters-rules-type** is defined in the confinfoextensions namespace:


The schema for **media-filters-rules-type** is as follows:

```xml
<xs:complexType name="media-filters-rules-type">
  <xs:sequence>
    <xs:element name="mayModifyOwnFilters" type="tns:boolean-role-rule-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="initialFilters" type="tns:media-filters-role-rule-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="msav:type" minOccurs="0"/>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:sequence>
</xs:complexType>
```

The following XML types are used to construct and express rules for allowing and preventing the flow of media in the conference by manipulating the media filter types described in section 2.2.1.1.

2.2.1.1.5.1 mayModifyOwnFilters
The `mayModifyOwnFilters` element specifies rules for each user role in a conference with regard to permission to modify the ingress filters for the media element specified by the type element defined in section 2.2.1.5.3.

The XML type for `mayModifyOwnFilters` is `boolean-role-rule-type`, which is comprised of an element role of XML type `xs:string`, paired with an element value of XML type `xs:boolean`.

The XML type `boolean-role-rule-type` is defined in the confinfoextensions namespace:


The schema for `boolean-role-rule-type` is as follows:

```
<xs:complexType name="boolean-role-rule-type">
  <xs:sequence>
    <xs:element name="role" type="xs:string"/>
    <xs:element name="value" type="xs:boolean"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

### 2.2.1.5.2 initialFilters

The `initialFilters` element specifies the default values that are applied to each participant with the media element specified by the type element defined in section 2.2.1.5.3, to be applied to different roles in the conference.

The XML type for `initialFilters` is `media-filters-role-rule-type`, which is comprised of an element role of XML type `xs:string`, and elements `ingressFilter`, and `egressFilter` of XML type `media-filter-type`, which is defined in section 2.2.1.1.

The XML type `media-filters-role-rule-type` is defined in the confinfoextensions namespace:


The schema for `media-filters-role-rule-type` is as follows:

```
<xs:complexType name="media-filters-role-rule-type">
  <xs:sequence>
    <xs:element name="role" type="xs:string"/>
    <xs:element name="ingressFilter" type="tns:media-filter-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="egressFilter" type="tns:media-filter-type" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```

### 2.2.1.5.3 type

The `type` element specifies the conference-media-type as defined in [RFC4575] section 5.8.2 that the `mayModifyOwnFilters` and `initialFilters` are applied to. The XML type is defined in the avconfinfoextensions namespace:


The schema for `type` is as follows:
The schema does not constrain the value to an enumeration. The base profile defines 3 literal string values: "audio", "video", and "panoramic-video".

The type element is optional. If this element is not present, it is assumed to have a value of "audio".

### 2.2.2 MCU Conference Roster Document Format

This section specifies extensions to the MCU Conference Roster Document Format specified in [MS-CONF BAS] section 2.2.5.

#### 2.2.2.1 MCU endpoint Element Syntax

The model defined in [MS-CONF BAS] section 2.2.2.6 specifies the role of MCU entities in generating and maintaining MCU-specific endpoint elements. This section specifies extended message syntax and semantics for A/V-specific endpoint elements.

The XML schema for the type endpoint-type and the semantics of the elements it contains were originally specified in [RFC4575] in section 5.7. Extensions to [RFC4575] are specified in [MS-CONF BAS] section 2.2.2.6. This protocol further extends the semantics of the elements within endpoint-type relative to [RFC4575], and defines additional extension elements.

#### 2.2.2.1.1 endpoint Element Semantic Extensions

The following extension semantics are defined relative to [RFC4575] section 5.7. Unless extension semantics are explicitly defined in this section or in [MS-CONF BAS], the semantics specified in [RFC4575] remain intact.

##### 2.2.2.1.1.1 media Element Instances

The media element is semantically extended as follows:

**label:** The label element has message-specific semantics. This element MUST be present in MCU Conference Roster Document sent by an MCU. In the context of C3P request messages, this element is optional. If present, the value of this element MUST be equal to the value of the label attribute of the corresponding media stream entry element in the MCU-specific media container element in the MCU-specific entity-state container element.

Matching label elements tie an endpoint's media stream instance with a specific conference media instance.

The schema for label element is as follows:

```
<xs:element name="label" type="xs:string" minOccurs="0"/>
```

**src-id:** The src-id element carries the identifier of the actual source of RTP-based media. This element is optional. If present, the value MUST contain the RTP Synchronization Source (SSRC) identifier value generated by the endpoint for the stream it sends. This value is available in the SDP offer and SDP answer used to negotiate media, as specified in [RFC3264].

The schema for src-id element is as follows:

```
<xs:element name="src-id" type="xs:string" minOccurs="0"/>
```
2.2.2.1.2 endpoint Element Extension Elements

This protocol defines the following extension elements of the media element within the endpoint element.

2.2.2.1.2.1 media-ingress-filter Element

The media-ingress-filter element is of the XML type media-filter-type, following the semantic definition specified in section 2.2.1.1.1.

The media-ingress-filter element specifies the filter value for media in the direction of a client endpoint to the MCU.

The media-ingress-filter element is optional. This element has message-specific semantics that are specified in the context of containing message semantics detailed in subsequent sections.

2.2.2.1.2.2 media-egress-filter Element

The media-egress-filter element is of the XML type media-filter-type, following the semantic definition specified in section 2.2.1.1.1.

The media-egress-filter element specifies the filter value for media in the direction of the MCU to a client endpoint.

The media-egress-filter element is optional. This element has message-specific semantics that are specified in the context of containing message semantics detailed in subsequent sections.

2.2.2.1.2.3 media-source-id Element

The media-source-id element SHOULD be present in the MCU Conference Roster Document sent by an MCU. The value represents the Media Source ID (MSI) as defined in [MS-SDPEXT] section 3.1.5.32. In the context of C3P request messages, this element is ignored.

The schema for media-source-id element is as follows:

```xml
<xs:element name="media-source-id" type="xs:unsignedInt" ms:ignore="true"/>
```

2.2.2.1.2.4 source-name Element

The source-name element SHOULD be present in the MCU Conference Roster Document sent by an MCU. The value might be available in the SDP offer or SDP answer as the a=x-source attribute as defined in [MS-SDPEXT] section 3.1.5.33. If the value is not present in the SDP offer or SDP answer, then the label SHOULD be used. In the context of C3P request messages, this element is ignored.

The schema for source-name element is as follows:

```xml
<xs:element name="source-name" type="xs:string" ms:ignore="true"/>
```

2.2.2.2 MCU conference-view Element Syntax

This section specifies semantics for the notification message elements that reside within the MCU-specific entity-view element within the conference-view element defined in [MS-CONFBAS] section 2.2.2.7.

2.2.2.2.1 entity-state Extension Elements
This protocol defines the following extension elements of the entity-state element.

### 2.2.2.2.1 media Element Extensions

The Conference Document format defined in [MS-CONFBASE] specifies the role of MCU entities in generating and maintaining MCU-specific entity-view elements and subelements. This section specifies extended message syntax and semantics for MCU-specific entry elements within the media element within the entity-state element of the MCU-specific entity-view element.

The XML schema for the type conference-media-type and the semantics of the elements it contains are originally established in [RFC4575]. Extensions to [RFC4575] are specified in [MS-CONFBASE]. This protocol further extends the semantics of the elements within conference-media-type relative to [RFC4575] and defines additional extension elements.

The following extension semantics are defined relative to [RFC4575] section 5.3.4. Unless extension semantics are explicitly defined in this section or in [MS-CONFBASE], the semantics specified in [RFC4575] section 5.3.4 remain intact.

#### 2.2.2.2.1.1 media entry Element Semantic Extensions

The media entry element semantic extensions are specified as follows:

**label**: The label attribute is the identifier for the MCU-centric view of an instance of conference media. For example, this attribute can identify an instance of an audio mixer within the MCU. This identification is made by finding the label attribute of the conference media instance that equals the label element of the media instances of the user within the endpoint element's media element.

The label attribute is assigned by the MCU and MUST be unique within the containing entity-state/media element container. The value of this attribute is the same as the SDP label media attribute defined in [RFC4574] section 1.

#### 2.2.2.2.1.2 media entry Element Extension Elements

This protocol defines the following extension elements of the entry element child of the media element.

**modal-parameters**: The modal-parameters element can be present within the entry element when the entry element describes a video type. This is true when the type element contains the value "video".

The modal-parameters element can be present within the entry element when the entry element describes a media type other than video. This protocol defines no semantics or behavior associated with the modal-parameters element for types other than video. A message containing this element is considered to be syntactically valid; however, the contents of the modal-parameters element are ignored for media types other than video.

**audio-parameters**: This protocol defines no schema extensions, semantics, or content for audio-parameters. Extensions to this protocol can define schema extensions and related semantics and message processing rules.

**video-parameters**: The video-parameters element is of the XML type video-parameters-type, following the semantic definition specified in section 2.2.1.1.2.

#### 2.2.2.2.1.3 entry-exit-announcements

**entry-exit-announcements**: The entry-exit-announcements element is of the XML type entry-exit-announcements-type, following the semantic definition in section 2.2.1.1.4.

#### 2.2.2.2.1.3 presentation-mode-capable
Presentation-mode-capable*: The presentation-mode-capable element is of the XML type xs:boolean following the semantic definition in section 3.2.3.1.1.2.3. The schema for presentation-mode-capable element is as follows:

```xml
<xs:element name="presentation-mode-capable" type="xs:boolean" ms:ignore="true"/>
```

2.2.2.2.1.4 mediaFiltersRules

mediaFiltersRules *: The mediaFiltersRules element is of the XML type media-filters-rules-type, following the semantic definition in section 2.2.1.5.

2.2.2.2.1.5 multi-view-capable Element

The multi-view-capable element SHOULD<6> be present in the MCU Conference Roster Document sent by an MCU. The value represents whether the conference supports sending multiple streams for the conference-media-type with a label value equal to "main-video" for each client. In the context of C3P request messages, this element is ignored.

It is defined in the commonmcuextensions namespace found here:

The schema for multi-view-capable-type is as follows:

```xml
<xs:element name="multi-view-capable" type="xs:boolean" ms:ignore="true"/>
```

2.2.2.2.1.6 video-presentation-mode-capable Element

The video-presentation-mode-capable element SHOULD<7> be present in the MCU Conference Roster Document sent by an MCU. The value represents whether the conference supports role-based restrictions specified by media-filters-rules-type in section 2.2.1.5, with type value equal to "video" or "panoramic-video". In the context of C3P request messages, this element is ignored.

It is defined in the commonmcuextensions namespace found here:

The schema for video-presentation-mode-capable is as follows:

```xml
<xs:element name="video-presentation-mode-capable" type="xs:boolean" ms:ignore="true"/>
```

2.2.2.2.1.7 conf-media-filters-rules

The conf-media-filters-rules element SHOULD<8> be present in the MCU Conference Roster Document sent by an MCU. The conf-media-filters-rules element is of the XML type media-filters-rules-type, following the semantic definition in section 2.2.1.5. The value represents the media-filters-rules-type that is applied to the conference.

It is defined in the commonmcuextensions namespace found here:

The schema for conf-media-filters-rules is as follows:
2.2.3 C3P request/response Document Content

2.2.3.1 addUser Dial-out Request Document Syntax

In addition to the syntax rules given in [MS-CONFBAS] section 2.2.3.15 for addUser dial-out requests, additional rules apply to the elements specified in the following subsections.

2.2.3.1.1 endpoint Element

The following rules apply to the endpoint element.

- One single endpoint element MUST be present inside the user element. The user element MUST NOT contain more than one endpoint element.
- The epid attribute can be specified. If specified, it MUST be a valid endpoint identifier (EPID), as specified in [MS-SIPRE] section 3.2.
- The attributes epid, endpoint-uri, and sip-instance are mutually exclusive. While any one of them can be specified, entities MUST NOT specify more than one in an addUser dial-out request document.
- If the languages element is present it MUST NOT contain more than one language. The language specified will be used, if supported, to play the entry and exit announcements to this particular endpoint element.

2.2.3.1.2 media Element

Instances of the media element of the endpoint element can be present. If present, instances MUST conform to the syntax specified in section 2.2.2.1.1.1. If instances of the media element are present in an addUser dial-out request, the contents are interpreted as a constraint on the default SDP offer that the MCU sends the called party in the SIP INVITE.

The specific rules for processing the addUser dial-out request are specified in section 3.2.5.2.

2.2.3.2 addUser Dial-in Request Document Syntax

In addition to the syntax rules given in [MS-CONFBAS] section 2.2.3.17 for addUser dial-in requests, the additional rules in the following subsections apply.

2.2.3.2.1 endpoint Element

- Exactly one endpoint element MUST be present inside the user element. The user element MUST NOT contain more than one endpoint element.
- The epid attribute can be specified. If specified, it MUST be a valid EPID, as specified in [MS-SIPRE] section 3.2.
- One and only one of the attributes epid, endpoint-uri, and sip-instance MUST be specified.
- If the languages element is present it MUST contain exactly one language. The language specified will be used, if supported, to play the entry and exit announcements to this particular endpoint element.

2.2.3.2.2 media Element
Instances of the media element of the endpoint element can be present inside the endpoint element. If present, instances MUST conform to the media element syntax specified in section 2.2.2.1.1.

This protocol does not define any processing rules or behavior related to endpoint media elements in addUser dial-in request messages. If instances of the media element are present in an addUser dial-in request, they are ignored.

2.2.3.3 modifyEndpointMedia Request Syntax

This section defines A/V-specific semantics for the modifyEndpointMedia request message.

In addition to the syntax rules for C3P request document formats specified in [MS-CONFBAS] section 2.2.3, the following additional syntax rules apply.

The mediaKeys element MUST be present, and it MUST contain the following attributes:

- userEntity: Specifies the user to which the request applies.
- endpointEntity: Specifies the user’s endpoint to which the request applies.
- mediaId: Specifies the user media instance to which the request applies.

The previous attributes SHOULD contain values that reference current MCU Conference Roster Document contents. In other words, these attributes SHOULD correlate with values that were reflected in the most recent notification that contains the MCU Conference Roster Document.

The schema for media-keys-type element is as follows:

```xml
<xs:complexType name="media-keys-type" ms:className="CC3PMediaKeysType">
  <xs:attribute name="confEntity" type="xs:anyURI" ms:propertyName="ConfUri"/>
  <xs:attribute name="userEntity" type="xs:anyURI" ms:propertyName="UserUri"/>
  <xs:attribute name="endpointEntity" type="xs:string" ms:propertyName="EndpointUri"/>
  <xs:attribute name="mediaId" type="xs:string"/>
  <xs:anyAttribute namespace="#other" processContents="lax"/>
</xs:complexType>
```

The label element can be present. If present, it MUST contain the pre-existing value. This element is assigned by an MCU and cannot be modified.

The media-ingress-filter element is optional. If present, it specifies a new media filter direction of client-endpoint-to-MCU. If it is not present, the existing value of the filter is unaffected by the request.

The media-egress-filter element is optional. If present, it specifies a new media filter direction of MCU-to-client-endpoint. If it is not present, the existing value of the filter is unaffected by the request.

The status element can be present. Extensions to this protocol can specify additional syntax and processing rules for modifying the value of the status element.

The src-id element SHOULD NOT be present. This element is assigned by an MCU and cannot be modified. If this element is present in a modifyEndpointMedia request, its value is ignored.

The type element can be present. This element is assigned by an MCU and cannot be modified. If this element is present in a modifyEndpointMedia request, its value is ignored.

2.2.3.4 modifyConferenceAnnouncements Request Syntax

This section defines A/V-specific semantics for the modifyConferenceAnnouncements request message.
In addition to the syntax rules for C3P request document formats specified in [MS-CONFBAS] section 2.2.3.25, the following rules apply.

The Boolean type element **enabled** MUST be present.

The schema for **modify-conference-announcements-type** is as follows:

```xml
<xs:complexType name="modify-conference-announcements-type"
    ms:className="CC3PModifyConfAnnouncementsRequestType">
    <xs:sequence>
        <xs:element name="conferenceKeys" type="tns:conference-keys-type"
            ms:propertyName="ConferenceKeys"/>
        <xs:element name="enabled" type="xs:boolean"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional" ms:propertyName="McuUri"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
```
3 Protocol Details

3.1 Client Details

3.1.1 Abstract Data Model

None.

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

None.

3.1.5 Message Processing Events and Sequencing Rules

3.1.5.1 Constructing the Outgoing addUser Dial-in Request

As specified in [MS-CONFBAS] section 3.11, the dial-in request can be sent by the client or by the focus itself to the MCU. Clients do not need to explicitly send an addUser dial-in request to join MCU-specific conference modes. This assumes that they have followed the conference subscription rules specified in [MS-CONFBAS]. Specifically, the clients have extracted the MCU-Conference-URI of the desired MCU from the received conference notifications. A SIP INVITE message sent to the MCU-Conference-URI of a specific MCU results in an implicit addUser dial-in message being sent from the focus to the MCU.

In addition to the rules specified in [MS-CONFBAS] section 3.11.4.1, an A/V specific addUser dial-in request MUST conform to the syntax rules specified in section 2.2.3.2.

3.1.5.2 Constructing the Outgoing SIP INVITE Dial-in Request

This section specifies A/V media specific SDP content rules for SIP INVITE messages associated with an addUser dial-in. In addition to the rules specified in [MS-CONFBAS] section 3.11 for outgoing SIP INVITE requests, the following rules apply to SIP INVITE messages that follow A/V-specific addUser dial-in requests.

If the client specifies an Accept-Language header, as specified in [RFC3261] section 20.3, the conference MUST use the specified language to play all entry and exit announcements if the language is available. The behavior SHOULD be the same as if the languages element were specified in the addUser dial-in request, as explained in section 2.2.3.2.1. If present, the header MUST contain exactly one language and MUST NOT contain any accept-param header parameters.

It is assumed that clients have subscribed to conference notifications and have followed all of the rules and recommendations specified in [MS-CONFBAS]. Therefore, the client is aware of the following information before constructing the SIP INVITE message and the SDP offer content contained within it.
The **MCU-Conference-URI** of the Audio/Video Multipoint Control Unit (AVMCU) that is extracted from the **conf-uris** element of the Conference Document, whose child **purpose** element contains the value "audio-video", as specified in [MS-CONFBAS] section 2.2.2.4.

The contents of the AVMCU-specific **entity-view** element, and its subelements, within the **conference-view** element. The **conference-view** element can contain zero or more MCU-specific **entity-view** elements. Each **entity-view** element contains a **media** element. The **media** element can contain zero or more **entry** elements, each representing one conference media instance. Therefore, the client has a list of the A/V-specific conference media instances.

**3.1.5.2.1 Constructing the SDP Offer in the Outgoing SIP INVITE Message**

When constructing the **SDP offer**, the offered media instances are expected to correlate with the **conference** media instances that are reflected in conference state **notifications**. Therefore, the following rules apply:

- If the conference notification sent by the **MCU** does not contain a **multi-view-capable** element with a value of "true", the client **SHOULD NOT** offer a greater number of **SDP** media instances of a given media type than are reflected in conference media instances. For example, if there is one conference media instance of type "audio" and one conference media instance of type "video", the client's SDP offer **SHOULD NOT** contain more than one **m=audio** and one **m=video** instance.

  Note that if the client SDP offer contains more SDP media instances than are reflected in conference media instances, the MCU will reject those media instances using the conventional "port=0" semantics specified in [RFC3264] section 8.2 for removing a media stream.

- If the conference notification sent by the MCU contains a **multi-view-capable** element with a value of "true", the client can offer multiple SDP media instances of type "video" for one instance of conference media instance of type "video". For the other media instance types, the client **SHOULD** behave as described above, wherein the client **SHOULD NOT** offer a greater number of SDP media instances of a given media type than are reflected in conference media instances. When offering such media instances, the client **SHOULD** follow the descriptions specified in [MS-SDPEXT] section 3.1.5.34 and [MS-SDPEXT] section 3.1.5.35 to construct the SDP and construct a MIME structure as defined in [MS-SIPRE] section 3.15.4.1.

  The client can supplement each SDP **m=** line with the **label** attribute where the value of the **label** attribute is equal to the value of the **label** element that is reflected in conference state notifications.

**3.1.5.3 Constructing the Outgoing addUser Dial-out Request**

In addition to the rules specified in [MS-CONF BAS] section 3.10, A/V-specific **addUser** dial-out requests **MUST** conform to the syntax rules specified in section 2.2.3.1.

**3.1.6 Timer Events**

None.

**3.1.7 Other Local Events**

None.
3.2 Server Details

3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model as long as their external behavior is consistent with that described in this document.

Because all external messages relate in some way to the Conference Document structure that is described in section 1.3, it is convenient to use that as the conceptual data model. In other words, the abstract data model is represented by the structure defined by the XML schema for the conference-info element and its entire hierarchy of subelements.

Using the Conference Document structure as the basis for representing abstract state allows interim processing steps to be described in terms of data-modification operations made directly on a copy of the Conference Document. Where externally-visible C3P messages contain parts and fragments of the conference document, descriptions of the interim steps are used in subsequent sections to illustrate how the externally-visible state changes are realized.

Note that the actual data model can be implemented using a variety of techniques. An implementation is at liberty to represent such data in any way convenient.

3.2.1.1 Correlation of Media Parameters

The message processing and sequencing rules specified for the server role have common requirements for correlating media information across Conference Media instances, user endpoint element media instances, and SDP, [RFC4566], and [MS-SDPEXT] media descriptions contained in the application/SDP section of SIP messages.

The message processing and sequencing rules for the server role are as follows:

- A conference media instance is described using the XML type conference-medium-type in an instance of an entry element within the media element within the MCU-specific entity-state element.
- A user media instance is described using the XML type media-type in an instance of a media element within the endpoint element.
- Media instances are represented in SDP by the m= line.

The following table specifies the extent of possible correlation relationships across the three separate constructs.

<table>
<thead>
<tr>
<th>Conference Media (XML type &quot;conference-medium-type&quot;)</th>
<th>User Media (XML type &quot;media-type&quot;)</th>
<th>SDP Media Description (&quot;m&quot; line)</th>
</tr>
</thead>
<tbody>
<tr>
<td>label attribute</td>
<td>label element</td>
<td>label attribute [RFC4574]</td>
</tr>
<tr>
<td>type element</td>
<td>type element</td>
<td>m= field value. For example, m=audio</td>
</tr>
<tr>
<td>status element</td>
<td>status element</td>
<td>multiple a= attribute values: a=sendrecv a=sendonly a=recvonly</td>
</tr>
</tbody>
</table>
The *type* element can have the value of "audio" or "video". Other values MUST be ignored.

### 3.2.1.2 Correlation of Media Instances

A relationship exists between user media instances and **SDP** media instances. When correlated, user media instances and SDP media instances are paired together.

Note that there are cases where correlation is not possible. For example:

- There are more SDP media instances than user media instances.
- There are more user media instances than SDP media instances.
- An SDP media instance exists for which there is no matching user media instance that has not already been paired with a different SDP media instance.

The following abstract data model can be used to maintain the correlated relationships and state that is referenced when processing messages.

Note that an implementation is at liberty to represent the media instance relationships in any way that is convenient.

The conceptual model is a two-dimensional table containing all user media instances and all SDP media instances, whether correlated or not. For each user media instance in the table, there is also a corresponding state variable that controls message processing behavior. The user media instances in the correlation table are uniquely identified by the *id* attribute value. SDP media instances in the table are uniquely identified by their positional order within the body of the "application/SDP" content specified in [RFC3264].

For example:

<table>
<thead>
<tr>
<th>User media instances</th>
<th>SDPState</th>
<th>SDP media instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;media id=&quot;1&quot;&gt;</td>
<td>&quot;Active&quot; or &quot;Rejected&quot;</td>
<td>&quot;m=&quot; line 2</td>
</tr>
<tr>
<td>&lt;media id=&quot;2&quot;&gt;</td>
<td>&quot;Active&quot; or &quot;Rejected&quot;</td>
<td>&quot;m=&quot; line 1</td>
</tr>
<tr>
<td>&lt;media id=&quot;3&quot;&gt;</td>
<td>&quot;NotInstantiated&quot;</td>
<td>(None)</td>
</tr>
<tr>
<td>(None)</td>
<td>&quot;Rejected&quot;</td>
<td>&quot;m=&quot; line 3</td>
</tr>
</tbody>
</table>

In the example shown in the preceding table:

- The user media instance `<media id="1">` is correlated with the SDP media instance that appears second in the "application/SDP" content.
- The user media instance `<media id="2">` is correlated with the SDP media instance that appears first in the "application/SDP" content.
- The user media instance `<media id="3">` is not correlated with any SDP media instance.
- The third SDP media instance is not correlated with a user media instance.

The **SDPState** variable has the following state values and behaviors associated with each state:
"NotInstantiated": Indicates that the user media instance has not been correlated with an SDP media instance. User media instances in this state are not included in MCU Roster (user) notifications.

"Active": Indicates that the SDP media instance refers to a negotiated RTP stream. User media instances in this state are included in MCU Roster notifications for the user.

"Rejected": Indicates that the SDP media instance has been rejected using the "port=0" semantics specified in [RFC3264]. User media instances in this state are included in MCU Roster notifications for the user.

3.2.2 Timers

None.

3.2.3 Initialization

3.2.3.1 Conference Activation (MCU Bootstrap)

This section defines the message content and semantics for bootstrapping an AVMCU and initializing its conference state.

In addition to the requirements specified in [MS-CONFBAS] section 3.4.4.3, the following semantics and content apply to bootstrapping an AVMCU.

3.2.3.1.1 Initial Full Conference Notification

The contents of the initial full conference notification sent by an AVMCU are specified in the content rules contained in the following subsections.

3.2.3.1.1.1 entity-capabilities Element

The entity-capabilities element SHOULD be present. The contents of the entity-capabilities element SHOULD be fully populated and initialized as follows.

The msav:capabilities element SHOULD be present. If present, its child elements, supports-audio and supports-video, MUST be valid XML according to the defined schema. In other words, both elements MUST be present and MUST contain either "true" or "false".

3.2.3.1.1.2 Child Elements of the entity-state Element

3.2.3.1.1.2.1 entry-exit-announcements element

This section follows the product behavior described in endnote.<9>

The entry-exit-announcements element MUST be present and MUST be fully populated. The following rules apply:

- If the entry-exit-announcements is included in the entity-settings upon conference activation the AVMCU MUST copy the received value of entry-exit-announcements to the corresponding instance in entity-state element.

- Otherwise, the AVMCU MUST populate the following default value for the entry-exit-announcements in the entity-state element.

<msmcu:entry-exit-announcements>
<msmcu:modifiable>true</msmcu:modifiable>
3.2.3.1.2.2 mediaFiltersRules element

This section follows the product behavior described in endnote. <10>

The mediaFiltersRules element MUST be present and MUST be fully populated in the initial full conference notification.

The following rules apply:

- If the AVMCU receives a mediaFiltersRules element in the entity-settings section of the initial addConference request during conference bootstrap, the AVMCU MUST copy the received value of mediaFiltersRules to the corresponding instance in the entity-state element.

- Otherwise, the AVMCU MUST populate the following default values for the mediaFiltersRules element with type value equal to "audio" in the entity-state element:

  <msci:mediaFiltersRules>
  <msci:mayModifyOwnFilters>
    <msci:role>default</msci:role>
    <msci:value>true</msci:value>
  </msci:mayModifyOwnFilters>
  <msci:initialFilters>
    <msci:role>default</msci:role>
    <msci:ingressFilter>unblock</msci:ingressFilter>
  </msci:initialFilters>
  <msav:type>audio</msav:type>
  </msci:mediaFiltersRules>

3.2.3.1.2.3 presentation-mode-capable element

This section follows the product behavior described in endnote. <11>

The presentation-mode-capable element can be present and, if present, MUST adhere to the syntax rules specified in section 2.2.2.1.3. The value of this element MUST be "true".

If the presentation-mode-capable element is present, then the AVMCU MUST support modification of the mediaFiltersRules or conf-media-filters-rules elements with a type of value equal to "audio" as discussed in section 3.2.3.1.2.2 using the ModifyConference request shown in section 4.5.

3.2.3.1.2.4 media Element

The media element MUST be present and MUST be fully populated to contain the descriptions of the AVMCU's full complement of conference media instances, following the semantic rules specified in section 2.2.2.1 for each entry element within the media element.

The following rules also apply:

- If the capabilities element of the entity-capabilities element is also present, and the value of its supports-video element is "false", the media elements MUST NOT contain any instances that have a type element value of "video".

- If the capabilities element of entity-capabilities is also present, and the value of its supports-audio element is "false", the media elements MUST NOT contain any instances that have a type element value of "audio".
3.2.3.1.2.5 multi-view-capable Element

The multi-view-capable element SHOULD be present and be fully populated. If the multi-view-capable element is set to "true", then the MCU MUST support negotiating multiple video streams with label "main-video" as defined in [MS-SDPEXT] section 3.1.5.35.

3.2.3.1.2.6 video-presentation-mode-capable Element

The video-presentation-mode-capable element SHOULD be present and, if present, MUST adhere to the syntax rules specified in section 2.2.2.1.6.

If the video-presentation-mode-capable element is set to "true", then the MCU MUST support modification of the mediaFiltersRules or conf-media-filters-rules elements with a type (section 2.2.1.5.3) of value equal to "video" or "panoramic-video", as discussed in section 3.2.3.1.2.2, using the ModifyConference request shown in section 4.5.

3.2.3.1.2.7 conf-media-filters-rules Element

The conf-media-filters-rules element SHOULD be present and if present, MUST be fully populated in the initial full conference notification.

The following rules apply:

- If the AVMCU receives a conf-media-filters-rules element in the entity-settings section of the initial addConference request during conference bootstrap, the AVMCU SHOULD reflect the received value of conf-media-filters-rules to the corresponding instance in the entity-state element.

- Otherwise, the AVMCU SHOULD populate the following default values for the conf-media-filters-rules element with type value equal to "video" or "panoramic-video" in the entity-state element:

```xml
<msmcu:conf-media-filters-rules>
  <msci:mayModifyOwnFilters>
    <msci:role>default</msci:role>
    <msci:value>true</msci:value>
  </msci:mayModifyOwnFilters>
  <msci:initialFilters>
    <msci:role>default</msci:role>
    <msci:ingressFilter>unblock</msci:ingressFilter>
  </msci:initialFilters>
  <msav:type>video</msav:type>
</msmcu:conf-media-filters-rules>
```

The conference-view element package described in [MS-CONFBAS] section 2.2.4.4 is used for conference notifications. When a first notification returned by the server has state equal to "full", it is called the initial full conference notification. An example of a conference-view element package containing the initial full conference notification for an AVMCU is as follows:

```xml
<msmcu:conference-view>
  <msav:full>full</msav:full>
</msmcu:conference-view>
```
<msci:conference-view ci:state="full" entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD">
  <msci:entity-view ci:state="full" entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:34D7F8255152F345817A2A6037C579BD">
    <msci:entity-capabilities>
      <msav:capabilities>
        <msav:supports-audio>true</msav:supports-audio>
        <msav:supports-video>true</msav:supports-video>
      </msav:capabilities>
      <msci:mediaFiltersRules>
        <msci:mayModifyOwnFilters>
          <msci:role>default</msci:role>
          <msci:value>true</msci:value>
        </msci:mayModifyOwnFilters>
        <msci:initialFilters>
          <msci:media>
            <entry label="main-audio">
              <type>audio</type>
              <status>sendrecv</status>
            </entry>
            <entry label="main-video">
              <type>video</type>
              <status>sendrecv</status>
            </entry>
            <entry label="panoramic-video">
              <type>panoramic-video</type>
              <status>sendrecv</status>
            </entry>
          </msci:media>
        </msci:initialFilters>
        <msav:type>audio</msav:type>
      </msci:mediaFiltersRules>
    </msci:entity-capabilities>
    <msci:state>
      <msci:locked>false</msci:locked>
    </msci:state>
  </msci:entity-view>
</msci:conference-view>
3.2.4 Higher-Layer Triggered Events

None.

3.2.5 Message Processing Events and Sequencing Rules

Unless otherwise specified, the message processing rules defined in this protocol assume that the commands documented herein are executed without error. Implementations of AVMCU services can have operating modes and application logic that prohibit certain commands from being carried out to their normal conclusion. For example, an implementation's configuration or policy could result in certain commands being deliberately denied or rejected with error responses. Such implementation decisions are beyond the scope of this protocol, and not necessary for illustrating the protocol extensions.

3.2.5.1 Common Rules for Processing SDP Offers and Answers

This section specifies common rules that apply to any handling of audio/video-specific SDP offers or answers within the MCU entity. These rules are referred to by the message processing rules defined in this section.

The rules specified in this section assume that the following conditions exist:

- The MCU entity has an internal representation of the current state of conference media, so it could generate a full conference notification. In other words, bootstrap has occurred.

- The MCU entity has an internal representation of user media instances, so it could construct at least the contents of the media element, the media-ingress-filter element, and the media-egress-filter elements within the endpoint element as it would appear in the MCU Conference Roster Document Format specified in section 2.2.2.1.

Note that the previous condition can be met by generating a temporary working copy of user media instances on demand.

- If the MCU has set the multi-view-capable attribute to "true" as described in section 2.2.2.1.5, then the MCU SHOULD have multiple instances of user media instances with label "main-video" for its internal representation, but SHOULD NOT include more than one media instance in the full conference notification. The single instance of the user media instance with label "main-video" SHOULD be the instance which the client might contribute its video source.

3.2.5.1.1 Generating an Initial SDP Offer
This section specifies the processing rules and behavior for generating an initial SDP offer, as is typical when preparing to send a SIP INVITE as part of addUser dial-out processing. It is assumed that an implementation is independently capable of SDP negotiation using [MS-SDPEXT] before considering the extensions specified in this protocol.

The following conceptual steps specify the requirements for generating an initial SDP offer. The steps can be performed in another order from the one given here, as long as the same end results are achieved.

For each instance of the media element within the endpoint element:

- If the media-ingress-filter element or the media-egress-filter elements are present within the media element, apply them to the initial value of the status element.
  - Note that the semantics of the media-ingress-filter element and media-egress-filter element are referenced from the point of view of the MCU. The status element is referenced from the point of view of the user.

  For example, if the initial value of the status element is "sendrecv" and the value of media-ingress-filter element is "block", the resulting value of the status element is "recvonly".

- Generate a valid and fully-formed SDP m= line of the media type specified in the type element of the user media instance. Referring to the table and the special case specified in section 3.2.1.1:
  - If the value of the type element is audio, the m= line begins with "m=audio".
  - If the value of the type element is video, the m= line begins with "m=video".
  - If the value of the type element is another value, the media instance MUST be ignored; they are not used by a user agent client (UAC) and are reserved.

- The directional attribute, such as "a=sendrecv", MUST be set to reflect the directionality of the MCU's point of view based on the value specified in the status element of the user media instance, with the following exception. When the direction is "sendrecv", the MCU SHOULd exclude the direction attribute in the SDP offer, because a default of "a=sendrecv" SHOULD be assumed in cases where a direction attribute is not explicitly specified, as specified in [RFC3264]. The status element reflects the user's point of view and, following the conventions established in [RFC3264], the offered directional attribute is the opposite of the directional attribute of the remote endpoint. For example, if the user media instance directional attribute is "recvonly", the MCU's offered directional attribute is "sendonly".

- An MCU SHOULD supplement the SDP media description with the label attribute, setting the value of the label attribute to the value contained in the media instance's label element.

- This protocol specifies only the media type, either "audio" or "video", the label element value, and the status element value. The remainder of the SDP media description (m line) is implementation-specific and beyond the scope of this protocol. This protocol assumes that an implementation is first capable of operating independently of the protocol extensions specified in this protocol before the extensions are applied. For more information, see section 1.5.

- Using the conceptual data model described in section 3.2.1.2, create and initialize a table of correlated media instances for reference when processing subsequent messages.

- If the MCU internally has multiple instances of a media element of type "video" with label "main-video", the video streams SHOULD be multiplexed as defined in [MS-SDPEXT] section 3.1.5.35.

3.2.5.1.2 Correlation of Offered SDP Media Instances
This section specifies processing rules for establishing the initial correlation between user media instances and newly-offered SDP media instances. This typically occurs when the first SIP INVITE is received, such as in the case of an addUser dial-in. It can also occur in subsequent re-INVITE messages, regardless of the direction of the original SDP offer.

The following rules assume use of the conceptual data model described in section 3.2.1.2 and the media parameter correlations specified in section 3.2.1.1:

- A user media instance and an SDP media instance can be considered to correlate only if the following conditions are met:
  - If the SDP media instance specifies "m=audio", the value of the type element of the user media instance MUST be "audio".
  - If the SDP media instance specifies "m=video", the value of the type element of the user media instance MUST be "video".
- Once correlation between a user media instance and an SDP media instance has been established, that correlation relationship MUST NOT change. Thus, only the user media instances having an SDPState of "NotInstantiated" are eligible for correlation with a new SDP media instance.

Other than the previously mentioned rules, this protocol does not mandate use of any specific algorithm to correlate media instances. An implementation can use any convenient mechanism. In typical cases, all that is necessary is to apply the previously mentioned rules in a simple search loop.

The following pseudocode illustrates correlating media instances:

```plaintext
for each new SDP media instance that does not already exist in the table
{
    for each row in the table that contains a user media instance
    {
        if the preceding correlating conditions are met
        {
            add the correlating SDP media instance to the row
            change the SDPState value of the row from "NotInstantiated" to "Active"
            continue on to the next new SDP media instance
        }
    }
    if no correlating user media instance was found
    append the table with a new row containing only the SDP media instance
}
```

If the label attribute is present in the SDP-offered media instances, the MCU can implement correlation logic that considers the label attribute value of the SDP media instance and the label element value of the user media instance, particularly when multiple media instances of the same media type are present.

### 3.2.5.1.3 Processing a Received SDP Offer

This section specifies processing rules for processing an SDP offer. The rules specified in this section assume that the correlation relationships between SDP media instances and user media instances have been established. In other words, the conceptual correlation table described in section 3.2.1.2 has been initialized using the rules specified in section 3.2.5.1.2.

The following conceptual steps specify the requirements for processing a received SDP offer. The steps can be performed in other sequences, as long as the same end results are achieved.

For each SDP media instance that is correlated with a user media instance:

1. If the SDP offer specifies that a previously negotiated media stream has been removed, as specified in [RFC3264] section 8.2, the MCU MUST omit the user media instance from subsequent
MCU Roster notifications for the user and continue on to the next media instance. Using the conceptual correlation table described in section 3.2.1.2, this is accomplished by changing the **SDPState** value from "Active" to "Rejected".

2. If the SDP offer specifies that a previously rejected or removed media stream has been re-instantiated using the same SDP media "slot", as specified in [RFC3264] section 8.1, the MCU MUST include the user media instance in subsequent MCU Roster notifications for the user and continue on to the next step. Using the conceptual correlation table described in section 3.2.1.2, this is accomplished by changing the **SDPState** value from "Rejected" to "Active".

3. The MCU MUST intersect the values of the **status** element and the offered SDP directional attribute contained in an **a=** line to determine the effective directionality, and apply the resulting value to the **status** element. For example, if the original **status** element value is "sendrecv" and the SDP offer specifies "a=recvonly", the resulting value of the **status** element is "recvonly".

4. If the media capabilities of the implementation do not support the parameters of the offered media instance, such as when the offered **codec** is not supported, the MCU MUST reject the media instance using the conventional "port=0" semantics specified in [RFC3264]. In addition, the MCU MUST omit the user media instance from subsequent MCU Roster notifications for the user and continue on to the next media instance. Using the conceptual correlation table described in section 3.2.1.2, this is accomplished by changing the **SDPState** value from "Active" to "Rejected".

5. When constructing the **SDP answer** for this media instance, the MCU MUST specify the intersected directionality using the standard conventions specified in [RFC3264].

6. The remainder of the SDP media description, or **m** line, is implementation-specific, and beyond the scope of this protocol. This protocol assumes that an implementation is first capable of operating independently of the protocol extensions specified in this protocol before the extensions are applied. For more information, see section 1.5.

7. For each SDP media instance that is not correlated with a user media instance, the MCU MUST reject the media instance using the conventional "port=0" semantics specified in [RFC3264].

### 3.2.5.1.4 Processing a Received SDP Answer

This section specifies processing rules for processing an **SDP answer**. The rules specified in this section assume that the correlation relationships between **SDP** media instances and user media instances have been established. In other words, the conceptual correlation table described in section 3.2.1.2 has been initialized using the rules specified in section 3.2.5.1.2.

In addition, the full contents of the SDP answer are assumed to conform to the requirements and recommendations specified in [RFC3264] and, therefore, any uncorrelated SDP media instances have been previously removed or rejected using the conventions specified in [RFC3264].

The following conceptual steps specify the requirements for processing a received SDP answer to a previously sent offer. The steps can be performed in a different sequence, as long as the same end results are achieved.

For each SDP media instance that is correlated with a user media instance:

1. If the SDP answer specifies that a media instance has been rejected, as specified in [RFC3264] section 6, the MCU MUST omit the user media instance from subsequent MCU Roster notifications for the user and continue on to the next media instance. Using the conceptual correlation table described in section 3.2.1.2 this is accomplished by changing the **SDPState** value from "Active" to "Rejected".

2. The MCU MUST intersect the values of the **status** element and the SDP **a=** element to determine the effective directionality, and apply the resulting value to the **status** element. For example, if the original **status** element value is "sendrecv" and the SDP answer specifies "a=recvonly", the resulting value of the **status** element is "recvonly".
3. The remainder of the SDP media description, which is contained in an \texttt{m=} line, is implementation-specific, and beyond the scope of this protocol. This protocol assumes that an implementation is first capable of operating independently of the protocol extensions specified in this protocol before the extensions are applied. For more information, see section 1.5.

### 3.2.5.2 addUser Dial-out Request

The \textit{addUser} dial-out request is used to initiate connecting a participant to an MCU. [MS-CONFABS] section 3.10 specifies \textit{addUser} dial-out behavior and message processing rules that are common to all MCUs. This section specifies the \textit{addUser} dial-out behavior that is specific to the AVMCU role.

Unless otherwise specified, all of the message processing and sequencing rules specified in [MS-CONFABS] section 3.10 for the MCU's role in processing \textit{addUser} dial-out commands apply.

Upon receiving an \textit{addUser} dial-out command, assuming that the prerequisite conditions specified in [MS-CONFABS] section 3.10 are met, the following additional rules apply.

- The MCU MUST validate the contents of the \textit{addUser} dial-out request for conformance to the syntax rules specified in section 2.2.3.1. If the syntax is not valid, the MCU MUST respond to the \textit{addUser} dial-out request with an \textit{addUser} "error" response. The "error" response contains the response code "Request Malformed", and the body of the \textit{addUser} contains the \texttt{reason} attribute with the value "OtherFailure".

Construction of the \texttt{SIP INVITE} message consists of two steps:

- Determining the \texttt{SIP URI} to send the message to and construction of the \texttt{SIP message} envelope including all headers.
- Construction of the media-specific, application/SDP, content.

#### 3.2.5.2.1 Constructing the Outgoing SIP INVITE Request

The \texttt{SIP INVITE} request that is used to carry the SDP content is constructed as specified in [MS-CONFABS] section 3.10.4.1.1. This protocol defines only SDP-specific extension rules to the \textit{addUser} dial-out request.

The target \texttt{SIP URI} is obtained using the rules specified in [MS-CONFABS] section 3.10.4.1.1.

#### 3.2.5.2.2 Construction of SDP Contents

The following conceptual interim steps facilitate construction of SDP content and subsequent C3P messages related to the added user.

- If the \texttt{media} element contained in the \texttt{endpoint} element of the \textit{addUser} message is present and populated with media instances, given the contents of initialized user media instances, the SDP offer content is constructed using the rules defined in section 3.2.5.1.1.

- If the \texttt{media} element contained in the \texttt{endpoint} element of the \textit{addUser} message is not present or is empty, user media instances MUST be constructed and initialized with a one-to-one relationship to conference media instances.

After completion of the previous step, the MCU MUST send a \texttt{SIP INVITE} message containing the constructed SDP offer to the specified target \texttt{URI}.

The MCU SHOULD send an \textit{addUser} "Pending" response at this point.

#### 3.2.5.3 addUser Dial-in Request

Processing the \textit{addUser} dial-in message consists of three steps:
Validating the message syntax and contents.

Saving a record of the message contents for later reference when processing SIP INVITE messages.

Constructing and sending the addUser dial-in response message.

On receipt of the addUser dial-in message, the MCU first validates the message syntax.

In addition to the message processing and sequencing rules specified in [MS-CONFBAS] for the MCU's role in processing the addUser dial-in, the MCU MUST validate the contents of the addUser dial-in request for conformance to the syntax rules specified in section 2.2.3.2. If the syntax is not valid, the MCU MUST respond to the addUser dial-in request with an addUser "error" response. The "error" response contains the response code "Request Malformed", and the body of the addUser contains the reason attribute with the value "OtherFailure".

Once the request is deemed valid, the MCU saves the contents of the message for later reference when processing the received SIP INVITE message for this user. Because there can be several addUser dial-in operations in progress at any given time, the MCU SHOULD implement and maintain a per-conference list of pending addUser dial-in request contents.

The data elements of the addUser dial-in request that are used later for search comparisons when processing received SIP INVITE messages are as follows:

- The entity attribute of the user element.
- The epid attribute of the endpoint element within the user element.
- The sip-instance attribute of the endpoint element within the user element.
- The endpoint-uri attribute of the endpoint element within the user element.

3.2.5.3.1 Constructing the addUser Dial-in Response

When constructing and sending the addUser dial-in response, the rules specified in [MS-CONFBAS] section 3.11.5.2.1 apply.

In addition, the MCU SHOULD populate the connection-info element with key-value pairs using the key values for mcu-server-uri and "mcu-conference-uri", as specified in [MS-CONFBAS] section 3.11.5.2.1.

3.2.5.4 modifyEndpointMedia Request

This section specifies processing rules for the modifyEndpointMedia request. Processing the modifyEndpointMedia request consists of two steps:

1. Identifying the targeted user and user media instance to which the request applies.

2. Processing the media-instance-specific modification specified by the body of the request. In other words, this modification is specified by the subelements of the media element.

The rules for identifying the targeted user and user media instance are specified as follows:

- If the user specified by the userEntity attribute of the C3P request message does not exist in the current MCU Conference Roster Document, the MCU MUST send a modifyEndpointMedia error response with the reason "UserDoesntExist".

- If the MCU-specific endpoint specified by the endpointEntity attribute of the C3P request message does not exist in the current MCU Conference Roster, the MCU MUST send a modifyEndpointMedia error response with the reason "EndpointDoesntExist".
If the user media instance specified by the `mediaId` attribute of the C3P request message does not exist under the specified user and the endpoint in the current MCU Conference Roster, the MCU MUST send a `modifyEndpointMedia` error response with the reason "OtherFailure".

If the media element in the `modifyEndpointMedia` request contains a value of "unblock" for the `media-ingress-filter` element, the following additional rules apply:

- The AVMCU MUST retrieve the `mayModifyOwnFilters` rule for the role of the user originating the request specified by the `Ref originator uri`. If the originator uri is not present in the AV portion of the conference, the `mayModifyFilterRules` for the "presenter" role are applied. If the value of the `mayModifyOwnFilters` rule is "true", the AVMCU can process the request further. If the value of the `mayModifyOwnFilters` rule is "false", the AVMCU MUST respond to the request with a `modifyEndpointMedia` error response with the CCCP response code "unauthorized" and `addEndpointMedia reason` equal to "OtherFailure".

SDP media renegotiation occurs when the MCU sends a SIP re-INVITE containing a modified SDP offer on its pre-established SIP session between itself and the client. The modification of a user media instance can result in SDP media renegotiation. An MCU SHOULD take the current state of that session into account before proceeding.

The following recommendations apply:

- If the SIP session is not in a connected state, for example when the MCU has just received a SIP BYE but has not yet reflected the pending state change in the MCU conference roster, the MCU SHOULD send a `modifyEndpointMedia` error response with the reason "OtherFailure".

- If SDP media renegotiation is in progress—for example, when the MCU has previously sent a SIP INVITE containing a modified SDP offer but has not yet received an SDP answer in response—an implementation SHOULD consider one of the following options:
  
  - The MCU can send a `modifyEndpointMedia` error response with the reason "OtherFailure".\[15\]
  - The MCU can delay processing of the `modifyEndpointMedia` request until the prior SDP media renegotiation is complete. However, delaying processing does not guarantee that the client will not send a SIP BYE in the next message it sends to the MCU. During this delay, the MCU SHOULD send a `modifyEndpointMedia` "pending" response, as described in [MS-CONFBAS] section 2.2.3.2.1.
  - The MCU can send a `modifyEndpointMedia` "failure" response if the INVITE from the MCU containing the modified SDP offer receives a SIP Failure response (4xx, 5xx, or 6xx).
  - The MCU can process the `modifyEndpointMedia` request before the prior SDP media renegotiation is complete. However, upon completion of the prior renegotiations, the MCU SHOULD determine if another renegotiation is necessary based on the steps below.\[16\]

The following steps complete the processing of the `modifyEndpointMedia` request.

1. If the `media-ingress-filter` or `media-egress-filter` elements are present within the `media` element of the `modifyEndpointMedia` request, apply (or copy) their values to the `media-ingress-filter` and `media-egress-filter` elements within the targeted user media instance.

2. Save a temporary copy of the value of the `status` element of the targeted user media instance. In other words, remember the currently-negotiated SDP directional attribute for this media instance.

3. Apply the new values of the `media-ingress-filter` and `media-egress-filter` elements of the targeted media instance to the current value of the `status` element.

   - Note that the semantics of the `media-ingress-filter` and `media-egress-filter` elements are from the perspective of the MCU, and the `status` element is from the perspective of the user.
4. If the new value of the **status** element is different from the previous value, SDP media renegotiation is required.

5. If SDP media renegotiation is required, the MCU SHOULD initiate SDP renegotiation by sending a SIP INVITE message to the client, specifying the full complement of previously-negotiated SDP media instances, following the model specified in [RFC3264] section 1.

Upon completion of the previous procedure, the MCU MUST send a **modifyEndpointMedia** "success" response, which is characterized by a "success" response code for the **modifyEndpointMedia** request. If the new values of the **media- ingress-filter** and **media- egress-filter** elements are different from their previous values, the MCU MUST send a MCU conference roster **notification** containing the updated **user** element state to the **focus**.

Note that sending a SIP INVITE message with a new SDP offer to renegotiate media results in subsequent receipt of a **200 OK** message containing the SDP answer. The rules specified in section 3.2.7.1.3 apply at that time.

### 3.2.5.5 modifyConferenceAnnouncements Request

This section specifies processing rules for the **modifyConferenceAnnouncements** request. Processing the **modifyConferenceAnnouncements** request consists of two steps:

1. Validating whether the changes contained in the request are permitted within the context of the **conference** the request is targeted to.

2. If permitted, making changes contained in the request.

The following recommendations apply:

- If the **MCU** receives the **modifyConferenceAnnouncements** request for a non-existent conference, it MUST send a failure response with the reason "ConferenceDoesntExist".

The following steps complete the processing of the **modifyConferenceAnnouncement** request:

1. If the **modifiable** attribute of the **entry-exit-announcements** element under the **entity-view** is set to "false", the MCU **modifyConferenceAnnouncements** MUST send a failure response with the reason "notSupported".

2. If the **modifiable** attribute of the **entry-exit-announcements** element under the **entity-view** is set to "true", the MCU SHOULD update its value of the **enabled** attribute of the **entry-exit-announcements** element.

Upon completion of the previous procedure, the MCU MUST send a **modifyConferenceAnnouncements** success response, which is characterized by a "success" response code for the **modifyConferenceAnnouncements** request. If the new values of the **enabled** attribute of the **entry-exit-announcements** element is different from its previous values, the MCU MUST send an MCU Conference Roster **notification** containing the updated **entry-exit-announcements** element state to the **focus**.

### 3.2.5.6 modifyConference Request

The **modifyConference** request is used to initiate changes to the **conference** properties by a **participant** to an **MCU**. Unless specified otherwise, the rules common to all MCUs for processing **modifyConference** requests specified in [MS-CONFPRO] section 3.1.4.2, [MS-CONFPRO] section 3.1.5.4, and [MS-CONFPRO] section 3.2.5.3 apply.

This section specifies the **modifyConference** behavior that is specific to the AVMCU role.

The following rules apply to the MCU handling of **modifyConference** requests:
• The MCU MUST check if the originator URI in the request is present in the conference.

• If the originator of the request is not present in the conference, the MCU MUST send a ModifyConference error response equal to "unauthorized" and reason equal to "OtherFailure".

• If the originator is present in the conference, the MCU checks the originating user's role. If the role is not "presenter", the AVMCU MUST send a ModifyConference error response equal to "unauthorized" and reason equal to "OtherFailure".

  If the request contains changes to conference-media-type and media-Filters-Rules type, the MCU MUST send a ModifyConference error response equal to "unauthorized" and reason equal to "OtherFailure".

### 3.2.5.6.1 Handling media-filters-rules type in modifyConference Request

This section specifically describes the cases that contain the media-Filters-Rules type in either the mediaFiltersRules element or conf-media-filters-rules element.

If the request is valid, the following processing rules apply to all instances of the media element of type equal to "audio", "video", or "panoramic-video" for non-trusted users, with the exception of the originator of the request.

• The MCU SHOULD update the media-Filters-Rules type in the entity-state with the mediaFiltersRules or conf-media-filters-rules received in the modifyConference request. If any existing elements within the entity-state are omitted from the content in the modifyConference request, this implies no change from their current state.

• In the mediaFilterRules element within the entity-state, if the value of media-ingress-filter specified in media-filters-role-rule-type for a user role is "blocked", the MCU MUST update the media-ingress-filter in the media element, specified by the type element in the mediaFilterRules, of all user endpoints with that user role to "blocked". The MCU MUST also update the status element under the media state wherever changes are made to maintain the mapping specified in section 3.2.1. Note that these changes during modifyConference processing are not followed by a SIP RE-INVITE from the MCU to renegotiate the SDP media attributes as is the case in the processing of modifyEndpointMedia requests described in section 3.2.5.4.

• If the media-ingress-filter value for a user role in mediaFiltersRules within the entity-state is "unblocked", the MCU MUST NOT apply changes to the media-ingress-filter instances in the media element for all instances of user endpoints (5) with that role.

If the modified media type is "video" or "panoramic-video", then the following additional steps SHOULD be followed:

• A media-filters-role-rule-type with a media-ingress-filter value of "blocked" SHOULD only be supported for "video-mode" of value "dominant-speaker-switched".

• If the "video-mode" is "manual-switched" and the mediaFilterRules element includes a media-ingress-filter rule for any role with a value of "blocked", the MCU SHOULD switch to "dominant-speaker-switched". This implies that the previous video restriction specified by the intended-primary-presenter-source has been removed and has been replaced by the role base filter specified by media-filters-role-rule-type.

Following these steps, the MCU MUST send a modifyConference "Success" response to the focus.

After processing the modifyConference request, the MCU MUST send a conference notification containing the following:

• The new values within the MCU entity-state element.
The current user state of all users that were affected by the ModifyConference request.

Example contents of a ModifyConference request, response, and the resulting conference notifications are shown in section 4.5.

3.2.5.6.2 Handling video-parameters-type in modifyConference Request

This section describes how the MCU SHOULD handle a modifyConference request that contains video-parameters-type as described in section 2.2.1.1.2.

If video-mode in the request is set to "manual-switched":

- If the source specified in the intended-primary-presenter-source type does not exist in the conference, MCU SHOULD send a ModifyConference error response equal to "OtherFailure" and reason equal to "OtherFailure".

- If the source specified in the intended-primary-presenter-source is a video contributor in the conference, then the MCU MUST NOT allow any media to propagate into the conference from any other video source with the same type and label as in the request.

If the video-mode has changed from "dominant-speaker-switched" to "manual-switched", then the AVMCU SHOULD perform the following additional step:

- If a media-filters-rules-type exists for the same type as in the request in either mediaFiltersRules or conf-media-filters-rules, the MCU SHOULD update the media-ingress-filter value for all user roles to "unblocked" and the mayModifyFilterRules value for all user roles to "true".

Following these steps, the MCU MUST send a modifyConference "Success" response to the focus.

If modifyConference operation was successful and there is a change to the conference, a notification containing the following elements MUST be sent:

- The new values within the entity-state element reflecting the new video mode and new video source.

- If a video source is not specified, the MCU SHOULD add an intended-primary-presenter-source type with the empty attribute set to "true".

If the video-mode in the request is set to "dominant-speaker-switched" and initiates a change in video mode from "manual-switched":

- The MCU SHOULD allow all video sources to contribute video into the conference as specified by "dominant-speaker-switched" mode in section 2.2.1.1.2, unless a client's media instance was previously "blocked" by a media-ingress-filter as specified in section 2.2.2.1.2.1.

If modifyConference operation was successful and there is a change to the conference, a notification containing the following elements MUST be sent:

- The new values within the MCU entity-state element reflecting the new video mode as "dominant-speaker-switched".

3.2.6 Timer Events

None.
3.2.7 Other Local Events

3.2.7.1 User signaling (SIP dialog) Events

This section specifies the A/V-specific extension actions taken when the MCU processes SIP messages and events.

3.2.7.1.1 Receipt of an Initial SDP Answer in SIP 200-OK Message Sent as Response to addUser Dial-out INVITE

When the MCU receives the first non-provisional 200 OK message containing an SDP answer in response to a sent INVITE, the following conceptual interim step prepares for subsequent C3P messages related to the invited user.

- The received SDP answer content is processed using the rules defined in section 3.2.5.1.4.

At this point, the MCU MUST perform the following steps:

- Send an addUser "success" response to the focus.
- Send a MCU Conference Roster notification to the focus containing the "full" user state.

3.2.7.1.2 Receipt of Initial SIP INVITE Messages (Dial-in User join)

This section describes processing rules for received SIP INVITE messages that are not already associated with an existing SIP dialog. Receipt of this message occurs during normal addUser dial-in sequences. Receipt of this message is normally preceded by the receipt and processing of the C3P addUser dial-in message, as specified in section 3.2.5.3.

There are two steps to process the received SIP INVITE message. The first step is to match or associate the INVITE message to a previously received C3P addUser dial-in message. The second step is to process A/V-specific SDP content within the INVITE message body.

The following steps describe processing rules for validating the routing and addressing information in the INVITE message and for matching the INVITE message to a previously-received C3P addUser dial-in message.

The rules for matching the INVITE message to a previously-received C3P addUser dial-in message use the following information from the INVITE message:

- The URI value contained in the FROM header.
- The following endpoint identification attributes. The format is defined in [MS-SIPRE]:
  - EPID
  - SIP.INSTANCE
  - GRUU

The contents of previously-received C3P addUser dial-in messages are stored in the MCU’s list of pending addUser dial-in requests. The MCU searches the contents of the list for a matching entry using the following comparisons, in order, until a match is found or all of the comparisons have been attempted.

The MCU MUST NOT accept the INVITE message if a matching entry is not found using only the following comparisons, and MUST apply the comparisons in the following order of precedence.

1. If GRUU is present, search the list for an entry containing a matching value of the endpoint-uri attribute of the endpoint element within the user element.

[MS-CONFAV] - v20181211
Centralized Conference Control Protocol: Audio-Video Extensions
Copyright © 2018 Microsoft Corporation
Release: December 11, 2018
2. If **EPID** is present, search the list for an entry containing a matching value of the **epid** attribute of the **endpoint** element within the **user** element AND that also has a value of the **entity** attribute of the **user** element that matches the **FROM** header.

3. If **SIP.INSTANCE** is present, search the list for an entry containing a matching value of the **sip-instance** attribute of the **endpoint** element within the **user** element.

If no match is found, the MCU MUST respond to the INVITE message with a SIP 404 Not Found message.

Once a matching **addUser** dial-in entry is found, it is removed from the list of pending **addUser** dial-in requests, and the SDP Media Negotiation steps begin.

### 3.2.7.1.2.1 Construction of SDP Answer Contents

User media instances are constructed and initialized with a one-to-one relationship to conference media instances. This step facilitates the construction of SDP answer contents and subsequent C3P messages related to the added user.

At this point, it is necessary to correlate the newly-offered SDP media instances and user media instances. The **MCU** MUST attempt to correlate all offered SDP media instances with user media instances, applying the rules and constraints specified in section 3.2.5.1.2. If none of the offered media instances can be correlated with a user media instance, the **MCU** MUST respond to the **INVITE** message with a **SIP** 488 Not Acceptable Here response, and do no further processing.

The **MCU** MUST apply the rules specified in section 3.2.5.1.3 when constructing the SDP answer. If the resulting SDP answer would reject all offered media instances, the **MCU** MUST respond to the **INVITE** message with a **SIP** 488 Not Acceptable Here response, and do no further processing.

### 3.2.7.1.2.2 Accepting the Initial INVITE

The **MCU** MUST send a **SIP** 200 OK message containing the **SDP answer** in response to the received **INVITE** message.

After completing the procedure in the previous section, the **MCU** MUST send an MCU Conference Roster notification for the user to the **focus** containing the "full" user state for the user that has just sent the INVITE message.

### 3.2.7.1.3 Receipt of Subsequent SIP Re-INVITE Message

This section describes processing rules for received **SIP** re-INVITE messages that occur on existing **SIP** dialogs. The processing rules guide the media renegotiation process.

When **SIP** re-INVITE messages are received, only the SDP-content-related content needs to be processed. It is assumed that a reasonable implementation would preserve the correlated relationships between media instances that were established or constructed during processing of the initially received **SDP offer**, as specified in section 3.2.7.1.2.1, or the initially constructed SDP offer, as specified in section 3.2.5.2.2, and thus those steps do not have to be repeated.

The rules for processing, or performing media-renegotiation of, re-INVITE messages:

- If the **SDP offer** contains new media instances through **m=** lines that have not previously appeared in any **SDP offer**, the new instances **MUST** be correlated with user media instances using the rules specified in section 3.2.5.1.2.

- The **SDP answer** content **MUST** be constructed using the aligned media instances and the rules defined in section 3.2.5.1.4.

- The **MCU** **MUST** send a **SIP** 200 OK message containing the **SDP answer** in response to the received **INVITE** message.
After completing the previous steps, if any of the media element instance values have changed, the MCU MUST send an MCU Conference Roster notification to the focus containing the "full" user state for the user that has just sent the INVITE message.
4 Protocol Examples

4.1 addUser Dial-out

The following example shows a typical call-flow sequence for an addUser dial-out. In the example, the user "Alice" (alice@fabrikam.com) is assumed to have already joined and subscribed to the conference.

Figure 1: addUser dial-out call flow sequence

When the focus receives the MCU user element state notification, it notifies the existing conference participant, "Alice", that the user, "Bob", has joined the conference. The user element state notification contains the MCU endpoint element and the elements it contains.

BENOTIFY sip:10.56.66.199:4216;transport=tls;ms-opaque=8a12d4957e;ms-received-cid=B1C400;grid SIP/2.0
Via: SIP/2.0/TLS 10.54.70.62:5061;branch=z9hG4bKAB66CC9D.EB5CD874;branched=FALSE
Authentication-Info: NTLM rsapath="0100000000000000A7C4683CD3C5FC49", srand="70BB5069", snum="2068", opaque="196AFFF9", qop="auth",
targetname="ocs.fabrikam.com",
realm="SIP"

4.2 **addUser Dial-in**

The following example shows a typical call-flow sequence for an **addUser** dial-in. In the example, the user "Alice" (alice@fabrikam.com) is assumed to have already joined and subscribed to the conference. The user "Bob" is assumed to have joined and subscribed to the conference and has obtained the **MCU service URI** for "audio-video". The dial-in flow begins with Bob joining the A/V conference modality by sending a **SIP INVITE** message to the service URI.
Figure 2: addUser dial-in call flow sequence

Following is a sample conference state notification package showing the state prior to Bob sending the INVITE message:

```xml
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
                 xmlns:snipped="snipped">
  entity="sip:alice@fabrikam.com;gruu;opaque-app:conf:focus:id:34D7F8255152F345817A2A6037C579BD"
  state="full" version="4">
    <conference-description>
      <conf-uris>
        <entry>
          <uri>
            sip:alice@fabrikam.com;gruu;opaque-app:conf:chat:id:34D7F8255152F345817A2A6037C579BD
          </uri>
          <display-text>chat</display-text>
        </entry>
      </conf-uris>
    </conference-description>
  </conference-info>
```

[MS-CONFAV] - v20181211
Centralized Conference Control Protocol: Audio-Video Extensions
Copyright © 2018 Microsoft Corporation
Release: December 11, 2018
<purpose>chat</purpose>
</entry>

<entry>
<uri>
sip:alice@fabrikam.com;gruu;opaque-app:conf:meeting:id:34D7F8255152F345817A2A6037C579BD
</uri>
<display-text>meeting</display-text>
<purpose>meeting</purpose>
</entry>

<entry>
<uri>
sip:alice@fabrikam.com;gruu;opaque-app:conf:audio-video:id:34D7F8255152F345817A2A6037C579BD
</uri>
<display-text>audio-video</display-text>
<purpose>audio-video</purpose>
</entry>
</conference-description>

<users state="full">
  <user entity="sip:alice@fabrikam.com" state="full">
    <display-text>Alice Smith</display-text>
  </user>
</users>

<msci:conference-view ci:state="full">
  <msci:entity-view ci:state="full" entity="sip:alice@fabrikam.com;gruu;opaque-app:conf:focus:id:34D7F8255152F345817A2A6037C579BD">
    <msci:entity-state>
      <msci:locked>false</msci:locked>
    </msci:entity-state>
  </msci:entity-view>
</msci:conference-view>
<entry>
<type>chat</type>
</entry>

<entry label="main-video">
<type>video</type>
</entry>

<entry label="panoramic-video">
<type>panoramic-video</type>
</entry>

<entry label="chat">
<type>chat</type>
</entry>

<entry label="full">
<type>config</type>
</entry>
Note that the **MCU-Conference-URI** for audio-video is "sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:34D7F8255152F345817A2A6037C579BD".

The following sample message shows the INVITE message sent from Bob's client.

```
INVITE sip:ocs.fabrikam.com:5063;transport=Tls SIP/2.0 Via: SIP/2.0/TLS 10.56.66.199:4216
Max-Forwards: 70 From: <bob@fabrikam.com>;tag=0467d088cd;epid=02e00da898
To: <sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:34D7F8255152F345817A2A6037C579BD>
epid=7D8A78D161tag=f8d9ad7d Call-ID: 89fa19a7fc7c4b4d80e56c4d6bde7 CSeq: 5
INVITE Route: <sip:ocs.fabrikam.com:5061;transport=tls;ms-role-rs-from;ms-role-rs-to;ms-ent-dest;lr;m-rgs-cid=B1C400;ms-route-sig=bgdDrWFZv0LRuVtQtFePnKJY_Blo5ba1Pe8vY9zXAAA>
Contact: <bob@fabrikam.com;opaque=app:user;epid=bD8fQE-lqIClarruZDr3dGRA;gruu>
Proxy-User: UCCAPI/2.0.6623.0 OC/2.0.6623.0 (Microsoft Office Communicator)
Supported: ms-dialog-route-set-update Ms-Conversation-ID: Achjaj+npNLVj6xXRI9Q+nLI133/M/Q--
Supported: timer Supported: histinfo Supported: ms-referred-body
Supported: ms-sender
Accept-Language: fr-FR
ms-keep-alive: UAC;hop=hop=yes
ms-sdp: UAC;hop=hop=yes
Proxy-authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="196AFF99", crand="50b6d980", cnum="893", targetname="ocs.fabrikam.com", response="01000005010000092549d3dc5c49"

Content-Type: application/sdp
Content-Length: 1377
v=0
o-- 0 0 IN IP4 10.56.66.199
s=Session
i=IN IP4 10.56.66.199
b=CT:99980
m=audio 50000 RTP/SAVP 111 8 0 97 101
a=candidate:5q1qWRkv6z1DFf+07mDwVz1grMrzpYrNKUEeP+0JRSN4 1 alx+3utAJ5f4bRTGxTYmKA UDP 0.850 10.56.66.199 50000
a=candidate:5q1qWRkv6z1DFf+07mDwVz1grMrzpYrNKUEeP+0JRSN4 2 alx+3utAJ5f4bRTGxTYmKA UDP 0.850 10.56.66.199 50016
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80 inline:zVT6V5d2zaX1175j11z6hpj2TLDIGJxJvN/v0n111]2311:1
a=remote-candidate:XXb5se0dEp6qKMP27aECTauGFohJPE2UDFNAevFy+zU
a=maxptime:200
a=rtcpmap:1000 SIREN/16000
a=fmt:1 bitrate=16000
a=rtcpmap:18 PCMA/8000
a=rtcpmap:0 PCMU/8000
a=rtcpmap:97 RED/8000
```
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=encryption:required
m=video 50008 RTP/SAVP 121 34
k=base64:3Vib9a190693n3czRc5AzfJOjv1+I1lQNNOz9rBeHCSNTAkQaqfMDRBqtRhu
a=candidate:DSzb7F7zc/FeROMQyObbr0080Bqa5Yb6Ycrf4kt/I0Jz 1 vAtjb0AZMBdGrGoRuusX3 Q UDP 0.860 10.56.66.199 50008
a=candidate:DSzb7F7zc/FeROMQyObbr0080Bqa5Yb6Ycrf4kt/I0Jz 2 vAtjb0AZMBdGrGoRuusX3Q UDP 0.860 10.56.66.199 50001
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:CleJmOuADtS4jc5I/CLM6Vymgf3ZojHVS/3c72UE1^2|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80
inline:5FhbGF5yLth6Y7YBzCN/d8aqZ2AYZF4qfHQkHv|^2|1:1
a=maxptime:200
a=rtcp:50001
a=rtpmap:121 x-rtvc1/90000
a=rtpmap:34 H263/90000
a=encryption:required

After the received **SDP answer** is processed in response to the previous INVITE message, the MCU sends a MCU Conference Roster notification, as shown in the following sample message.

```
BENOTIFY sip:10.56.66.199:4216;transport-tls;ms-opaque=8a12d4957e;ms-received-cid=BlC400;grid SIP/2.0Via: SIP/2.0/TLS 10.54.70.62:5061;branch=z9hG4bK1680A441.6AD89D79;branched=FALSE
Authentication-Info: NTLM rspauth="010000000000000040B3554DD3C5FC49", srand="0852CCEB", snnum="2097", opaque="196AFFF9", qop="auth", targetname="ocs.fabrikam.com", realm="SIP Communications Service"
Max-Forwards: 70
To: <sip:alice@fabrikam.com>;tag=ab7629db57;epid=02e00da898Content-Length: 1821
From: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=7F6C0080
Call-ID: 4555a2d175ab44b6ab0e4b544570d76
CSeq: 13 BENOTIFY
Content-Type: application/conference-info+xml
Event: conference
subscription-state: active,expires=3600
<conference-info xmlns="snipped" state="partial" version="16">
  <user entity="sip:bob@fabrikam.com" state="full">
    <display-text>Bob Freer</display-text>
    <roles>
      <entry>attendee</entry>
    </roles>
  </user>
  <endpoint entity="{BB7A3FD2-6467-4A42-88D6-4D0488D74A9D}" msci:session-type="focus"
           msci:epid="b7b1d3a6e6" msci:endpoint-uri="sip:bob@fabrikam.com;opaque:user:epid:YwZ40vZRFqmQ2_ra60OzQA;gruu">
    <status>connected</status>
  </endpoint>
</conference-info>
```
4.3 modifyEndpointMedia

The modifyEndpointMedia request can be used to change the state of media for a user. The following diagram shows a sample message flow for modifyEndpointMedia, where the user "Alice" is muting the user "Bob".

```xml
<media id="2">
  <type>video</type>
  <label>main-video</label>
  <status>sendrecv</status>
  <cis:separator/>
  <cis:separator/>
  <cis:separator/>
  <msci:media-source-id>2000256</msci:media-source-id>
  <msci:source-name>main-video</msci:source-name>
</media>
<msci:roles>
  <entry>attendee</entry>
</msci:roles>
<msci:authMethod>enterprise</msci:authMethod>
<msci:accessMethod>internal</msci:accessMethod>
<cis:separator/>
<msci:languages>fr-FR</msci:languages>
</endpoint>
</user>
</users>
</conference-info>
```
Figure 3: modifyEndpointMedia message flow

A sample modifyEndpointMedia request message appears next.

Note that the mediaKeys element, userEntity, endpointEntity, and mediaId attributes specify the Audio media instance of Bob's endpoint.
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="196AFF9", crand="c12b73a2", cnum="899", targetname="ocs.fabrikam.com", response="0100000000000002f082249d3c5fc49"
Content-Type: application/cccp+xml
Content-Length: 1005

<?xml version="1.0"?>
C3PVersion="1"
to="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD"
from="sip:alice@fabrikam.com" requestId="96451088">
<modifyEndpointMedia
mscp:mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:34D7F8255152F345817A2A6037C579BD"
xmlns="urn:ietf:params:xml:ns:cccp"
xmlns:ci="urn:ietf:params:xml:ns:conference-info">
<mediaKeys
confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD"
userEntity="sip:bob@fabrikam.com" endpointEntity="{679ACDD8-2171-4176-8414-6ABA41ED881A}" mediaId="1"/
</mediaKeys>
<ci:media
id="1">
<ci:type>audio</ci:type>
<ci:status>sendrecv</ci:status>
<media-ingress-filter>
block
</media-ingress-filter>
</ci:media>
</modifyEndpointMedia>
</request>

When the user element notification is sent by the MCU to the focus, the contents are forwarded by the focus to conference subscribers. The following sample shows a typical notification message sent by the focus when MCU user element state notifications are processed.

BENOTIFY sip:10.56.66.199:4216;transport=tls;ms-opaque=8a12d4957e;ms-received-cid=B1C400;grid SIP/2.0
Via: SIP/2.0/TLS 10.54.70.62:5061;branch=z9hG4bK1680A441.6AD89D79;branched=FALSE
Authentication-Info: NTLM rspauth="0100000000000000040B3554DD3C5FC49", srand="0852CCEB", snum="2097", opaque="196AFF9", qop="auth", targetname="ocs.fabrikam.com", realm="SIP Communications Service"
Max-Forwards: 70
To: <sip:alice@fabrikam.com>;tag=ab7629db57;epid=02e00da898
Content-Length: 1821
From: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=7F6C0080
Call-ID: 4555a2d175ab44b6ab0e4b5754570d76
CSeq: 13 BENOTIFY
Content-Type: application/conference-info+xml
Event: conference subscription-state: active; expires=3600
<conference-info entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD" state="partial" version="16">
<users state="partial">
<user entity="sip:bob@fabrikam.com" state="full">
<display-text>Bob Freer</display-text>
<roles>
<entry>attendee</entry>
</roles>
<endpoint entity="{BB7A3FD2-6467-4A42-88D6-4D0488D74A9D}" msci:session-type="focus" msci:epid="b7b13da6d6" msci:endpoint-url="sip:bob@fabrikam.com;opaque=app:epid:yiW240vZRFqmQ2_r560O2QA;gruu">

[MS-CONFAV] - v20181211
Centralized Conference Control Protocol: Audio-Video Extensions
Copyright © 2018 Microsoft Corporation
Release: December 11, 2018
Note that in the previous example, the value of the audio media instance’s status element is still "sendrecv".

Subsequently, after the MCU processes the received SDP answer, it sends another notification containing the updated media state to the focus. The following example shows that the status element value of the "audio" media type has changed to "recvonly" as a result of the SDP renegotiation.

```
BENOTIFY sip:10.56.66.199:4216;transport=tls;ms-opaque=8a12d4957e;ms-received-cid=B1C400;grid SIP/2.0
Via: SIP/2.0/TLS 10.54.70.62:5061;branch=z9hG4bK1680A441.6AD89D79;branched=FALSEAuthentication-Info: NTLM
    rspauth="010000000000000040B3554DD3C5FC49", srandom="08522C68", snum="2097", opaque="196AFFF9", qop="auth",
    targetname="ocs.fabrikam.com", realm="SIP Communications Service"
Max-Forwards: 70
To: <sip:alice@fabrikam.com>;tag=ab7629db57;epid=02e00da898
Content-Length: 1821
From: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=7F6C0080
Call-ID: 4555a2d175ab44b6ab0e4b5754570d76
CSeq: 13 BENOTIFY
Content-Type: application/conference-info+xml
Event: conference subscription-state: active;expires=3600
<conference-info xmlns:snipped entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD" state="partial" version="16">
    <user entity="sip:bob@fabrikam.com" state="full">
        <display-text>Bob Freer</display-text>
        <roles>
            <entry>attendee</entry>
        </roles>
    </user>
</conference-info>
```
4.4 modifyConferenceAnnouncements

The modifyConferenceAnnouncements request can be used to change the value of the enabled attribute of the entry-exit-announcements element under the entity-view for a conference. The following diagram shows a sample message flow for modifyConferenceAnnouncements, where the user "Bob" is issuing the request.

Figure 4: modifyConferenceAnnouncements message flow
A sample `modifyConferenceAnnouncements` request message is as follows:

```xml
```

The `modifyConferenceAnnouncements` response is as follows:

```xml
```
When the **entity-view** element **notification** is sent by the **MCU** to the **focus**, the notification is forwarded by the focus to all the conference subscribers. The following example shows a typical notification that is sent by the focus when the MCU **entity** notifications are processed.

```xml
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP" state="partial"
version="1" static="true">
  <entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP">
    <entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:applicationsharing:id:YAZ75GMP">
      <entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:YAZ75GMP">
        <capabilities>
          <audio>
            <supports>true</supports>
          </audio>
          <video>
            <supports>true</supports>
          </video>
        </capabilities>
      </entity-view>
    </entity-view>
  </entity-view>
</conference-info>
```
4.5 modifyConference

In the context of Audio Video Conferencing, the modifyConference request can be used to change the value of the mediaFiltersRules element, conf-media-filters-rules element or video-parameters-type element under the entity-state for a conference. The following diagram shows a sample message flow for modifyConference, where the user "Bob" is issuing the request.

![modifyConference message flow](image)

An example modifyConference request message that modifies a mediaFiltersRules element is provided below. Modifications to conf-media-filters-rules element are supposed to be constructed in an analogous manner:

```
08/10/2010|19:05:32.231 1A0C:1A10 INFO :: Sending Packet - 157.54.27.30:80 (From Local Address: 10.80.20.229:54075) 2820 bytes:
08/10/2010|19:05:32.231 1A0C:1A10 INFO :: INFO sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DG8PBL SIP/2.0
Via: SIP/2.0/TLS 10.80.20.229:54075
Max-Forwards: 70
From: <sip:bob@fabrikam.com>;tag=dccb9664e6;epid=4e91b46850
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DG8PBL>;tag=A6260080
Call-ID: 09c2c6a6f7c3406e9f5a69f1b688260d
CSeq: 6 INFO
Route: <sip:ocs.fabrikam.com:443;transport=tls;opaque-state:Cl.R252de00;lr;ms-route=sig<sip:perm18vmQPKP961-y9T-nrXyP6KnHAFFJrrX2WccYNo5XHb6fFPAAA>
Route: <sip:ocs.fabrikam.com:5061;transport-tls;ms-fe=ocs.fabrikam.com;lr;received=10.31.50.6;ms-received-cid=2513B00>
Route: <sip:ocs.fabrikam.com:5061;transport-tls;ms-fe=ocs.fabrikam.com;lr>
Route: <sip:ocs.fabrikam.com:5061;transport-tls;ms-fe-ocs.fabrikam.com;lr>
User-Agent: UCCAPI/4.0.7400.0 OC/4.0.7400.0 (Microsoft Communicator 2010)
Supported: ms-dialog-route-set-update
Supported: timer
```
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service", opaque="54574323", targetname="ocs.fabrikam.com", crand="f835ac40", cnum="2400", response="e925a5fec2ac14d0dfb7825584944ae1611b9512"

Content-Type: application/cccp+xml

Content-Length: 1456

<?xml version="1.0"?>
<request xmlns="urn:ietf:params:xml:ns:cccp" xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions" C3PVersion="1" to="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" from="sip:bob@fabrikam.com" requestId="164518168">
<modifyConference mscp:mcuUri="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">

<mcsi:entity-view entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">
<mcsi:entity-state>
<msci:mediaFiltersRules>
<msci:mayModifyOwnFilters>
<msci:role>presenter</msci:role>
<msci:value>true</msci:value>
</msci:mayModifyOwnFilters>
<msci:mayModifyOwnFilters>
<msci:role>default</msci:role>
<msci:value>false</msci:value>
</msci:mayModifyOwnFilters>
<msci:initialFilters>
<msci:role>presenter</msci:role>
<msci:ingressFilter>unblock</msci:ingressFilter>
</msci:initialFilters>
<msci:initialFilters>
<msci:role>default</msci:role>
<msci:ingressFilter>block</msci:ingressFilter>
</msci:initialFilters>

<msav:type>audio</msav:type>
</mcsi:mediaFiltersRules>
</mcsi:entity-state>
</mcsi:entity-view>
</mcsi:conference-view>
</mcsi:conference-info>

</modifyConference>
</request>

The modifyConference response is as follows:

8/10/2010|19:05:32.257 1A0C:1A10 INFO :: Data Received - 157.54.27.30:80 (To Local Address: 10.80.20.229:54075) 1693 bytes:
8/10/2010|19:05:32.257 1A0C:1A10 INFO :: INFO sip:157.54.125.148:48044;transport=tls;ms-opaque=e519aad283;ms-received-cid=252DE00;grid SIP/2.0
ms-user-logon-data: RemoteUser
Via: SIP/2.0/TLS 131.107.247.195:443;branch=z9hG4bK1839F509.8C995EBC3319414D;branched=FALSE;ms-internal-info="aakIjgfrKk8fQrMe3Ylpn99K07UOYseRd-n1Vd1g1QjL8QYfOAAM"
Max-Forwards: 68
Via: SIP/2.0/TLS 10.31.50.6:5061;branch=z9hG4bK1839F509.8C995EBC3319414D;branched=FALSE;ms-received-cid=2513B00
Authentication-Info: TLS-DSK qop="auth", opaque="54574323", srnd="28B50009", snum="2412", rpauth="b00ed42e6e0e750a6679b9e1f8529767ff35d55", targetname="ocs.fabrikam.com", realm="SIP Communications Service", version=4
Via: SIP/2.0/TLS 157.54.62.135:56647;branch=z9hG4bK22686BD2.440CE3C7ED1AA14E;branched=FALSE;ms-received-port=56647;ms-received-cid=4E3D4600
Content-Length: 543
From: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL>;tag=A6260080
When the entity-view element notification is sent by the MCU to the focus, the notification is forwarded by the focus to all the conference subscribers. The following example shows a typical notification that is sent by the focus when the MCU entity notifications are processed:

```
  <mci:conference-view ci:state="full">
    <mci:entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">
      <mci:entity-capabilities>
        <msav:capabilities>
          <msav:supports-video>true</msav:supports-video>
        </msav:capabilities>
        <msav:supports-audio>true</msav:supports-audio>
      </mci:entity-capabilities>
      <msav:capabilities>
      </mci:entity-capabilities>
    </mci:entity-view>
  </mci:conference-view>
  <mci:mediaFiltersRules>
    <mci:mayModifyOwnFilters>
      <mci:role>default</mci:role>
      <mci:value>false</mci:value>
    </mci:mayModifyOwnFilters>
    <mci:mayModifyOwnFilters>
      <mci:role>presenter</mci:role>
      <mci:value>true</mci:value>
    </mci:mayModifyOwnFilters>
    <mci:initialFilters>
      <mci:ingressFilter>block</mci:ingressFilter>
    </mci:initialFilters>
    <mci:initialFilters>
      <mci:ingressFilter>unblock</mci:ingressFilter>
    </mci:initialFilters>
  </mci:mediaFiltersRules>
  <msav:state>sendrecv</msav:state>
</conference-info>
```
The **conf-media-filters-rules** element is supposed to be modified in the same manner. The request is supposed to contain the new values for a **conf-media-filters-rules** value. After a successfully handling the request, as defined in section 3.2.5.6.1, a successful response is supposed to be followed by a notification that contain the updated **conf-media-filters-rules**.

An example of **modifyConference** request message that modifies a **video-parameters-type** is as follows:

08/10/2010|19:05:32.231 1A0C:1A10 INFO :: Sending Packet = 157.54.27.30:80 (From Local Address: 10.80.20.229:54075) 2820 bytes:
The `modifyConference` response is as follows:

```
08/10/2010|19:05:32.257 1A0C:1A10 INFO :: INFO sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL SIP/2.0
Via: SIP/2.0/TLS 10.80.20.229:54075
Max-Forwards: 70
From: <sip:bob@fabrikam.com>;tag=dcbb9664e6;epid=4e91b46850
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL>;tag=A6260080
Call-ID: 09c2c6a67c3406e95a69f1b68b260d
CSeq: 6 INFO
Route: <sip:ocs.fabrikam.com:443;transport=tlste:state:Ci.R252de00;lr;ms-route-sig=ebWPbtTv8WqF96T-y9T-nrXYH6khvFJr4XW0cYNko5XHh6fZPAAA>
Route: <sip:ocs.fabrikam.com:5061;transport=tlste:state=0c.fabrikam.com;lr;received=10.31.50.6;ms-received-cid=2513800>
Route: <sip:ocs.fabrikam.com:5061;transport=tlste:state=ocs.fabrikam.com;lr>
Route: <sip:ocs.fabrikam.com:5061;transport=tlste:state=ocs.fabrikam.com;opaque=state:Ti:F;lr>
User-Agent: UCCAPI/4.0.7400.0 OC/4.0.7400.0 (Microsoft Communicator 2010)
Supported: ms-dialog-route-set-update
Supported: timer
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service", opaque="e925a5fec2ac140d0bf7849444e16b9512"
Content-Type: application/cccp+xml
Content-Length: 2756
<?xml version="1.0"?>
to="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" from="sip:bob@fabrikam.com" requestId="164518168">
  <modifyConference mscp:mcuUri="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">
    <conference-info xmlns="urn:ietf:params:xml:ns:conference-info" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" state="partial">
      <mcci:entity-view ci:state="partial" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">
        <mcci:entity-view>
          <mcci:media>
            <mcci:entry label="main-video">
              <mcci:type>video</mcci:type>
              <mcci:modal-parameters>
                <mcci:video-parameter>msav:intended-primary-presenter-source</mcci:video-parameter>
                <mcci:video-parameter>msav:entry=sip:alice@fabrikam.com</mcci:video-parameter>
              </mcci:modal-parameters>
            </mcci:entry>
          </mcci:media>
          <mcci:entity-view>
          </mcci:entity-state>
        </mcci:entity-view>
      </mcci:entity-view>
    </mcci:conference-info>
  </modifyConference>
</request>
```

The `modifyConference` response is as follows:

```
08/10/2010|19:05:32.257 1A0C:1A10 INFO :: Data Received = 157.54.27.30:80 (To Local Address: 10.80.20.229:54075) 1639 bytes:
08/10/2010|19:05:32.257 1A0C:1A10 INFO :: INFO sip:157.54.125.148:48044;transport=tls;opaque=e519aad283;ms-received-cid=252DE00;grid SIP/2.0
From: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL>;tag=A6260080
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL>;tag=A6260080
Call-ID: 09c2c6a67c3406e95a69f1b68b260d
CSeq: 6 INFO
Max-Forwards: 70
Via: SIP/2.0/TLS 10.80.20.229:54075
Via: SIP/2.0/TLS 10.31.50.6:443;branch=z9hG4bK4E0FEA28.3ABF8C10F2D5614E;branched=FALSE;ms-user-logon-data: RemoteUser
```

67 / 107
When the entity-view element notification is sent by the MCU to the focus, the notification is forwarded by the focus to all the conference subscribers. The following example shows a typical notification that is sent by the focus when the MCU entity notifications are processed:

```
  <msci:conference-view ci:state="full">
    <msci:entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL">
      <msav:capabilities>
        <msav:supports-audio>true</msav:supports-audio>
        <msav:supports-video>true</msav:supports-video>
      </msav:capabilities>
    </msci:entity-view>
    <msci:mediaFiltersRules>
      <msci:mayModifyOwnFilters>
        <msci:role>default</msci:role>
        <msci:value>true</msci:value>
        <msci:mayModifyOwnFilters>
          <msci:role>presenter</msci:role>
          <msci:value>true</msci:value>
          <msci:initialFilters>
            <msci:role>default</msci:role>
            <msci: ingressFilter>unblock</msci: ingressFilter>
            <msci: initialFilters>
```

[MS-CONFAV] - v20181211
Centralized Conference Control Protocol: Audio-Video Extensions
Copyright © 2018 Microsoft Corporation
Release: December 11, 2018
<msci:initialFilters>
  <msci:role>presenter</msci:role>
  <msci:ingressFilter>unblock</msci:ingressFilter>
</msci:initialFilters>
<msav:type>audio</msav:type>
<msci:mediaFiltersRules>
  <msci:media>
    <entry label="main-audio">
      <type>audio</type>
      <status>sendrecv</status>
    </entry>
    <entry label="main-video">
      <type>video</type>
      <status>sendrecv</status>
      <msci:modal-parameters>
        <msci:video-parameters>
          <msav:video-mode>manual-switched</msav:video-mode>
          <msav:intended-primary-presenter-source>
            <msav:entry>sip:alice@fabrikam.com</msav:entry>
          </msav:intended-primary-presenter-source>
        </msci:video-parameters>
      </msci:modal-parameters>
    </entry>
    <entry label="panoramic-video">
      <type>panoramic-video</type>
      <status>sendrecv</status>
      <msci:modal-parameters>
        <msci:video-parameters>
          <msav:video-mode>dominant-speaker-switched</msav:video-mode>
        </msci:video-parameters>
      </msci:modal-parameters>
    </entry>
  </msci:media>
</msci:mediaFiltersRules>
<msav:type>video</msav:type>
<msci:mediaFiltersRules>
  <msci:media>
    <entry label="main-video">
      <type>video</type>
      <status>sendrecv</status>
      <msci:modal-parameters>
        <msci:video-parameters>
          <msav:video-mode>manual-switched</msav:video-mode>
        </msci:video-parameters>
      </msci:modal-parameters>
    </entry>
    <entry label="panoramic-video">
      <type>panoramic-video</type>
      <status>sendrecv</status>
      <msci:modal-parameters>
        <msci:video-parameters>
          <msav:video-mode>dominant-speaker-switched</msav:video-mode>
        </msci:video-parameters>
      </msci:modal-parameters>
    </entry>
  </msci:media>
</msci:mediaFiltersRules>
A video-parameters-type that modifies a video type of "panoramic-video" is supposed to be handled in the same way. After a successfully handling the request, as defined in section 3.2.5.6.2, a successful response is supposed to be followed by a notification that contain the updated video-parameters-type.
5 Security

5.1 Security Considerations for Implementers
None.

5.2 Index of Security Parameters
None.
6  Appendix A: Full XML Schema


The schema for the conference-info namespace is based on [RFC4575] with extensions specified in namespaces defined subsequently.

```xml
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
targetNamespace="urn:ietf:params:xml:ns:conference-info"
xmlns:elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <!--
  This imports the standard separator
  -->
schemaLocation="ms-ci-separator.xsd"/>
  <!--
  This import brings in the MS Conference Package extensions
  -->
schemaLocation="ms-ci-ext.xsd"/>
  <!-- ELEMENTs and Attributes for CCCP definitions
  -->
  <xs:attribute name="state" type="state-type"/>
  <xs:element name="media" type="media-type"/>
  <xs:element name="endpoint" type="endpoint-type"/>
  <xs:element name="user-roles" type="user-roles-type"/>
  <xs:element name="user" type="user-type"/>
  <xs:element name="dialog-id" type="sip-dialog-id-type"/>
  <!--
  CONFERENCE ELEMENT
  -->
  <xs:element name="conference-info" type="conference-type"
ms:className="C3PConferenceInfo"/>
  <!--
  CONFERENCE TYPE
  -->
  <xs:complexType name="conference-type" ms:className="CC3PConferenceType">
    <xs:sequence>
      <xs:element name="conference-description" type="conference-description-type"
minOccurs="0" ms:propertyName="ConferenceDescription"/>
      <xs:element name="host-info" type="host-type" minOccurs="0"/>
      <xs:element name="conference-state" type="conference-state-type" minOccurs="0"
ms:propertyName="State"/>
      <xs:element name="users" type="users-type" minOccurs="0"
ms:propertyName="UserCollection"/>
      <xs:element name="sidebars-by-ref" type="uris-type" minOccurs="0"/>
      <xs:element name="sidebars-by-val" type="sidebars-by-val-type" minOccurs="0"/>
      <xs:element name="msci:conference-media-states" minOccurs="0"/>
      <xs:element name="msci:conference-view" minOccurs="0"
ms:propertyName="ConferenceView"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```
<xs:element ref="msci:notification-data" minOccurs="0" ms:propertyName="NotificationData"/>
<xs:element ref="msci:conference-mcu-policies" minOccurs="0" maxOccurs="unbounded">
    <xs:sequence minOccurs="0" ms:propertyName="Extension1" ms:className="CC3PConfDescriptionExtension1">
        <xs:element ref="cis:separator"/>
        <xs:element ref="msci:pstn-access" minOccurs="0" ms:propertyName="PstnAccess"/>
    </xs:sequence>
</xs:element>
<xs:element ref="msci:conference-extension" minOccurs="0" ms:propertyName="Extension2" ms:className="CC3PConfDescriptionExtension2">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msci:pstn-lobby-capable" minOccurs="0" ms:propertyName="PstnLobbyBypass"/>
</xs:element>
<xs:element ref="msci:lobby-capable" minOccurs="0" ms:propertyName="LobbyCapable"/>
<xs:element ref="msci:anonymous-type-allowed" minOccurs="0" ms:propertyName="Anonymous"/>
<xs:element ref="msci:join-url" minOccurs="0" ms:propertyName="JoinUrl"/>
</xs:element>
<xs:element ref="msci:autopromote" minOccurs="0" ms:propertyName="Autopromote"/>
<xs:element ref="msci:autopromote-allowed" minOccurs="0" ms:propertyName="AutopromoteAllowedValues"/>
<xs:element ref="msci:pstn-lobby-bypass" minOccurs="0" ms:propertyName="PstnLobbyBypass"/>
<xs:element ref="msci:disclaimer-title" minOccurs="0" ms:propertyName="DisclaimerTitle"/>
<xs:element ref="msci:recording-allowed" minOccurs="0" ms:propertyName="RecordingAllowed"/>
</xs:element>
<xs:element ref="msci:external-user-booking-allowed" minOccurs="0" ms:propertyName="ExternalUserBookingAllowed"/>
<xs:element ref="msci:server-mode" minOccurs="0" ms:propertyName="ServerMode"/>
</xs:element>
<xs:element ref="msci:recording-notification" minOccurs="0" ms:propertyName="RecordingNotification"/>
<xs:sequence minOccurs="0" ms:propertyName="Extension3" ms:className="CC3PConfDescriptionExtension3">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msci:custom-invite" minOccurs="0" ms:propertyName="CustomInvite"/>
    <xs:element ref="msci:anonymous-dial-out-allowed" minOccurs="0" ms:propertyName="AnonymousDialoutAllowed"/>
</xs:sequence>
<xs:element ref="msci:endorse-allowed" minOccurs="0" ms:propertyName="EndorseAllowed"/>
</xs:element>
<xs:element ref="msci:main-video-mute-allowed" minOccurs="0" ms:propertyName="MainVideoMuteAllowed"/>
</xs:element>
<xs:element ref="msci:pano-video-mute-allowed" minOccurs="0" ms:propertyName="PanoVideoMuteAllowed"/>
<xs:element ref="msci:is-large-meeting" minOccurs="0" ms:propertyName="IsLargeMeeting"/>
</xs:element>
<xs:element ref="cis:separator"/>
<xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
</xs:complexType>

<!--
HOST TYPE
-->
<xs:complexType name="host-type">
<xs:sequence>
    <xs:element name="display-text" type="xs:string" minOccurs="0"/>
    <xs:element name="web-page" type="xs:anyURI" minOccurs="0"/>
</xs:sequence>
</xs:complexType>
<xs:element name="uris" type="uris-type" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:any namespace="#other" processContents="lax"/>
</xs:complexType>

<!--
  CONFERENCE STATE TYPE
-->  
<xs:complexType name="conference-state-type" ms:className="CC3PConferenceStateType">
  <xs:sequence>
    <xs:element name="user-count" type="xs:unsignedInt" minOccurs="0" ms:propertyName="Locked"/>
    <xs:element name="active" type="xs:boolean" minOccurs="0" ms:propertyName="Locked"/>
  </xs:sequence>
</xs:complexType>

<!--
  CONFERENCE MEDIA TYPE
-->  
<xs:complexType name="conference-media-type">
  <xs:sequence>
    <xs:element name="entry" type="conference-medium-type" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!--
  CONFERENCE MEDIUM TYPE
-->  
<xs:complexType name="conference-medium-type">
  <xs:sequence>
    <xs:element name="display-text" type="xs:string" minOccurs="0"/>
    <xs:element name="type" type="xs:string"/>
    <xs:element name="status" type="media-status-type" minOccurs="0"/>
    <xs:element ref="msci:modal-parameters" minOccurs="0"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:any namespace="#other" processContents="lax" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="label" type="xs:string" use="required" ms:propertyName="Locked"/>
  </xs:sequence>
</xs:complexType>

<!--
  URIs TYPE
-->  
<xs:complexType name="uris-type" ms:className="CC3PUrirsType">
  <xs:sequence>
    <xs:element name="entry" type="uri-type" maxOccurs="unbounded" ms:propertyName="Locked"/>
  </xs:sequence>
  <xs:attribute name="state" type="state-type" use="optional" default="full" ms:propertyName="Locked"/>
</xs:complexType>

<!--
  URI TYPE
-->  
<xs:complexType name="uri-type" ms:className="CC3PUriType">
  <xs:sequence>
    <xs:element name="uri" type="xs:anyURI" ms:propertyName="Locked"/>
  </xs:sequence>
</xs:complexType>
<xs:attribute name="entity" type="xs:anyURI" ms:propertyName="UserUri"/>
<xs:attribute ref="msci:smtp-address"/>
<xs:attribute name="state" type="state-type" use="optional" default="full" ms:propertyName="UserState"/>
<xs:attribute ref="msci:endorser" use="optional" ms:propertyName="Endorser"/>
<xs:attribute ref="msci:endorser-display-name" use="optional" ms:propertyName="EndorserDisplayName"/>
<xs:attribute name="device-type" use="optional" ms:propertyName="DeviceType"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- USER ROLES TYPE -->
<xs:complexType name="user-roles-type" ms:className="CC3PRoleCollection">
<xs:sequence>
<xs:element name="entry" type="xs:string" maxOccurs="unbounded" ms:propertyName="UserRoles"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- USER LANGUAGES TYPE -->
<xs:simpleType name="user-languages-type">
<xs:list itemType="xs:language"/>
</xs:simpleType>

<!-- ENDPOINT TYPE -->
<xs:complexType name="endpoint-type" ms:className="CC3PEndpoint">
<xs:sequence>
<xs:element name="display-text" type="xs:string" minOccurs="0" ms:propertyName="Status"/>
<xs:element name="referred" type="execution-type" minOccurs="0" ms:propertyName="JoiningMethod"/>
<xs:element name="status" type="endpoint-status-type" minOccurs="0" ms:propertyName="DisconnectionInfo"/>
<xs:element name="joining-method" type="joining-type" minOccurs="0" ms:propertyName="JoiningMethod"/>
<xs:element name="disconnection-method" type="disconnection-type" minOccurs="0" ms:propertyName="DisconnectionInfo"/>
<xs:element name="media" type="media-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="MediaCollection"/>
<xs:element name="call-info" type="call-type" minOccurs="0" ms:propertyName="CallInfo"/>
<xs:element ref="msci:roles" minOccurs="0" ms:propertyName="AuthMethod"/>
<xs:element ref="msci:accessMethod" minOccurs="0" ms:propertyName="AccessMethod"/>
<xs:element ref="msci:clientInfo" minOccurs="0" ms:propertyName="ClientInfo"/>
<xs:element ref="msci:post-dial" minOccurs="0" ms:propertyName="IncomingCalling"/>
<xs:element ref="msci:is-endpoint-capabilities" minOccurs="0" ms:propertyName="IsPstnEndpoint"/>
<xs:element ref="msci:領導erPasscode" minOccurs="0" ms:propertyName="LeaderPasscode"/>
<xs:element ref="msci:is-robot" minOccurs="0" ms:propertyName="IsRobot"/>
<xs:element ref="msci:current-sidebar" minOccurs="0" ms:propertyName="CurrentSidebar"/>
<xs:sequence minOccurs="0" ms:className="CC3PEndpointExtension1">
<xs:element ref="cis:separator"/>
<xs:element ref="msci:session-on-behalf-of" minOccurs="0" ms:propertyName="SessionOnBehalfOf"/>
<xs:element ref="msci:in-conferencing-services" minOccurs="0" ms:propertyName="InConferenceServicesCollection"/>
<xs:element ref="msci:can-endpoint" minOccurs="0" ms:propertyName="Languages"/>
<xs:element ref="msci:is-pstn-endpoint" minOccurs="0" ms:propertyName="IsPstnEndpoint"/>
</xs:sequence>
</xs:complexType>
<?xml version="1.0" encoding="UTF-8"?>
<xs:sequence minOccurs="0" ms:propertyName="Extension2">
  <xs:element ref="cis:separator"/>
  <xs:element ref="masci:client-recording" minOccurs="0" ms:propertyName="ClientRecording"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:sequence>

<!--
ENDPOINT STATUS TYPE
-->
<xs:simpleType name="endpoint-status-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="pending"/>
    <xs:enumeration value="dialing-out"/>
    <xs:enumeration value="dialing-in"/>
    <xs:enumeration value="alerting"/>
    <xs:enumeration value="on-hold"/>
    <xs:enumeration value="connected"/>
    <xs:enumeration value="muted-via-focus"/>
    <xs:enumeration value="disconnecting"/>
    <xs:enumeration value="disconnected"/>
  </xs:restriction>
</xs:simpleType>

<!--
JOINING TYPE
-->
<xs:simpleType name="joining-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="dialed-in"/>
    <xs:enumeration value="dialed-out"/>
    <xs:enumeration value="focus-owner"/>
  </xs:restriction>
</xs:simpleType>

<!--
DISCONNECTION TYPE
-->
<xs:simpleType name="disconnection-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="departed"/>
    <xs:enumeration value="booted"/>
    <xs:enumeration value="failed"/>
    <xs:enumeration value="busy"/>
  </xs:restriction>
</xs:simpleType>

<!--
EXECUTION TYPE
-->
<xs:complexType name="execution-type" ms:className="CC3PExecutionType">
  <xs:attribute name="entity" type="xs:string" ms:propertyName="EndpointEntity"/>
  <xs:attribute name="state" type="state-type" use="optional" default="full" ms:propertyName="EndpointState"/>
  <xs:attribute ref="masci:session-type" use="optional"/>
  <xs:attribute ref="masci:epid" use="optional"/>
  <xs:attribute ref="masci:sip-instance" use="optional"/>
  <xs:attribute ref="masci:endpoint-uri" use="optional"/>
  <xs:attribute ref="masci:refer-to-uri" use="optional"/>
  <xs:attribute ref="masci:asserted-identity" use="optional"/>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<xs:sequence>
  <xs:element name="when" type="xs:dateTime"
    minOccurs="0"/>
  <xs:element name="reason" type="xs:string"
    minOccurs="0"/>
  <xs:element name="by" type="xs:anyURI"
    minOccurs="0"/>
</xs:sequence>

<xs:anyAttribute namespace="#other" processContents="lax"/>

<xs:complexType>
  <!-- CALL TYPE -->
  <xs:complexType name="call-type">
    <xs:choice>
      <xs:element name="sip" type="sip-dialog-type" ms:propertyName="SIPDialogId"/>
      <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:choice>
    <xs:anyAttribute namespace="#other" processContents="lax"/>
  </xs:complexType>

  <!-- SIP DIALOG ID TYPE -->
  <xs:complexType name="sip-dialog-id-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"/>
      <xs:element name="call-id" type="xs:string" ms:propertyName="CallId"/>
      <xs:element name="from-tag" type="xs:string" ms:propertyName="FromTag"/>
      <xs:element name="to-tag" type="xs:string" ms:propertyName="ToTag"/>
      <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="#other" processContents="lax"/>
  </xs:complexType>

  <!-- MEDIA TYPE -->
  <xs:complexType name="media-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"/>
      <xs:element name="type" type="xs:string" minOccurs="0" ms:propertyName="Type"/>
      <xs:element name="label" type="xs:string" minOccurs="0"/>
      <xs:element name="src-id" type="xs:string" minOccurs="0"/>
      <xs:element name="status" type="media-status-type" minOccurs="0"
        ms:propertyName="Status"/>
      <xs:element ref="msci:media-ingress-filter" minOccurs="0"/>
      <xs:annotation>
        <xs:documentation>
          If this element is not present, a value of 'unblock'
          should be assumed
        </xs:documentation>
      </xs:annotation>
      <xs:element ref="msci:media-egress-filter" minOccurs="0"/>
      <xs:sequence minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
<xs:element ref="msci:media-capabilities" minOccurs="0">
  <xs:annotation>
    <xs:documentation>
      This element corresponds to the media level capabilities from an SDP
    </xs:documentation>
  </xs:annotation>
  <xs:element ref="msci:conf-media-filter" minOccurs="0"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msci:media-source-id" minOccurs="0"/>
    <xs:element ref="msci:source-name" minOccurs="0"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:sequence>
  </xs:sequence>
</xs:element>

<!--
-->
<xs:simpleType name="media-status-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="recvonly"/>
    <xs:enumeration value="sendonly"/>
    <xs:enumeration value="sendrecv"/>
    <xs:enumeration value="inactive"/>
  </xs:restriction>
</xs:simpleType>

<!--
-->
<xs:complexType name="sidebars-by-val-type">
  <xs:sequence>
    <xs:element name="entry" type="conference-type" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="state" type="state-type" use="optional" default="full"/>
</xs:complexType>
</xs:schema>

6.2 conference-info-extensions Namespace

Following is the schema for the conference-info-extensions namespace.

<?xml version="1.0" encoding="utf-8"?>
xmlns:ns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msci2="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ms="urn:microsoft-cpp-xml-serializer"
xmlns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"

<!-- Bring in standard conferencing package separator -->
schemaLocation="ms-ci-separator.xsd"/>

<!-- Bring in standard conferencing package -->
<xs:import namespace="urn:ietf:params:xml:ns:conference-info" schemaLocation="ms-ci.xsd"/>

<!-- Bring in later extensions to this package -->
<xs:import namespace="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions"
schemaLocation="ms-ci-ext2.xsd"/>

<!-- This import brings the MCU settings definitions -->
schemaLocation="acpmcusettings.xsd"/>
schemaLocation="avmcuscusings.xsd"/>
schemaLocation="datamcusettings.xsd"/>
schemaLocation="immcususcings.xsd"/>
schemaLocation="asmcususcings.xsd"/>
<xs:import namespace="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
schemaLocation="mcucommon.xsd"/>

<xs:element name="to-mixer" type="msav:media-routing-type" ms:ignore="true"/>
<xs:element name="from-mixer" type="msav:media-routing-type" ms:ignore="true"/>
<xs:element name="session-id" type="xs:string" ms:ignore="true"/>
<xs:element name="disclaimer" type="xs:string" ms:ignore="true"/>
<xs:element name="designated.presenter" type="xs:boolean" ms:ignore="true"/>
<xs:element name="trusted" type="xs:boolean" ms:ignore="true"/>
<xs:attribute name="conference-id" type="xs:string" ms:ignore="true"/>
<xs:element name="conference-id" type="xs:string" ms:ignore="true"/>
<xs:element name="conference-key" type="tns:conference-key-type" ms:ignore="true"/>
<xs:element name="current-sidebar" type="xs:anyURI" ms:ignore="true"/>
<xs:element name="last-update" type="xs:dateTime" ms:ignore="true"/>
<xs:element name="last-activate" type="xs:dateTime" ms:ignore="true"/>
<xs:element name="is-active" type="xs:boolean" ms:ignore="true"/>
<xs:element name="expiry-time" type="xs:dateTime" ms:ignore="true"/>
<xs:element name="organizer-roaming-data" type="tns:organizer-roaming-data-type" ms:ignore="true"/>
<xs:element name="notification-data" type="tns:notification-data-type" ms:ignore="true"/>
<xs:element name="encryption-key" type="tns:encryption-key-type" ms:ignore="true"/>
<xs:element name="opaque" type="tns:encryption-opaque-type" ms:ignore="true"/>
<xs:attribute name="mcu-type" type="xs:string" ms:ignore="true"/>
<xs:element name="roles" type="ci:user-roles-type" ms:ignore="true"/>
<xs:attribute name="smtp-address" type="xs:anyURI" ms:ignore="true"/>
<xs:attribute name="endorser" type="xs:anyURI" ms:ignore="true"/>
<xs:attribute name="endorser-display-name" type="xs:string" ms:ignore="true"/>
<xs:element name="trusted-entities" type="ci:users-type" ms:ignore="true"/>
<xs:element name="languages" type="ci:user-languages-type" ms:ignore="true"/>
<xs:element name="encrypted-uri" type="tns:encrypted-content-type" ms:ignore="true"/>
<xs:element name="is-pstn-endpoint" type="xs:boolean" ms:ignore="true"/>
<xs:element name="anonymous-type-allowed" type="xs:boolean" ms:ignore="true"/>
<xs:element name="lobby-capable" type="xs:boolean" ms:ignore="true"/>
<xs:element name="join-url" type="xs:anyURI" ms:ignore="true"/>
<xs:element name="autopromote-allowed" type="tns:autopromote-type" ms:ignore="true"/>
<xs:element name="autopromote" type="tns:autopromote-type" ms:ignore="true"/>
<xs:simpleType name="autopromote-type">
  <xs:restriction base="xs:unsignedInt"/>
</xs:simpleType>
<xs:element name="pstn-lobby-bypass-allowed" type="xs:boolean" ms:ignore="true"/>
<xs:element name="pstn-lobby-bypass" type="xs:boolean" ms:ignore="true"/>
<xs:element name="disclaimer-title" type="xs:string" ms:ignore="true"/>
<xs:element name="recording-allowed" type="xs:boolean" ms:ignore="true"/>
<xs:element name="external-user-recording-allowed" type="xs:boolean" ms:ignore="true"/>
<xs:element name="recording-notification" type="xs:boolean" ms:ignore="true"/>
<xs:element name="server-mode" type="xs:unsignedInt" ms:ignore="true"/>
<xs:element name="default-entry-exit-announcements" type="xs:boolean" ms:ignore="true"/>
<xs:element name="anonymous-dialout-allowed" type="xs:boolean" ms:ignore="true"/>
<!--
  Device Type Enumeration
-->
<xs:restriction base="xs:string">
  <xs:enumeration value="meetingRoom"/>
</xs:restriction>
</xs:simpleType>

<!-- CUSTOM INVITE TYPE -->
<xs:element name="custom-invite" type="tns:custom-invite-type" ms:ignore="true"/>
<xs:complexType name="custom-invite-type" ms:className="CC3PCustomInviteType">
  <xs:sequence>
    <xs:element name="logo-url" type="xs:anyURI" minOccurs="0" ms:propertyName="LogoUrl"/>
    <xs:element name="legal-url" type="xs:anyURI" minOccurs="0" ms:propertyName="LegalUrl"/>
    <xs:element name="help-url" type="xs:anyURI" minOccurs="0" ms:propertyName="HelpUrl"/>
    <xs:element name="custom-footer-text" type="xs:string" minOccurs="0" maxOccurs="unbounded" ms:propertyName="CustomFooterText"/>
  </xs:sequence>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:complexType>

<!-- Number of users in a meeting -->
<xs:attribute name="participant-count" type="xs:unsignedInt" ms:ignore="true"/>

<!-- Indicates if the server supports the Endorse command -->
<xs:element name="endorse-allowed" type="xs:boolean" ms:ignore="true"/>

<!-- Indicates server support for Main Video Hard Mute -->
<xs:element name="main-video-mute-allowed" type="xs:boolean" ms:ignore="true"/>

<!-- Server support for Pano Video Hard Mute -->
<xs:element name="pano-video-mute-allowed" type="xs:boolean" ms:ignore="true"/>

<!-- Indicates if the current conference runs with large meetings optimizations. -->
<xs:element name="is-large-meeting" type="xs:boolean" ms:ignore="true"/>

<!-- ENCRYPTION KEY OPAQUE TYPE -->
<xs:complexType name="encryption-key-opaque-type">
  <xs:sequence>
    <xs:element name="issuing-server" type="xs:string" ms:propertyName="IssuingServer"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- ENCRYPTION KEY TYPE -->
<xs:complexType name="encryption-key-type" ms:className="CC3PEncryptionKey">
  <xs:sequence>
    <xs:element name="x509-certificate" type="xs:base64Binary" ms:propertyName="X509Cert"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
  CONFERENCE KEY TYPE
-->
<xs:complexType name="conference-key-type" ms:className="CC3PConferenceKey">
  <xs:sequence>
    <xs:element name="cms-data" type="xs:base64Binary" ms:propertyName="CmsData"/>
    <xs:element name="opaque" type="tns:encryption-key-opaque-type" minOccurs="0" ms:propertyName="Opaque"/>
    <xs:element ref="msci2:optional" minOccurs="0" ms:propertyName="IsOptional"/>
  </xs:sequence>
  <xs:attribute namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:complexType>

<!--
  ENCRYPTED CONTENT TYPE
-->
<xs:complexType name="encrypted-content-type" ms:className="CC3PEncryptedContent">
  <xs:sequence>
    <xs:element name="cms-data" type="xs:base64Binary" ms:propertyName="CmsData"/>
  </xs:sequence>
  <xs:attribute namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:complexType>

<!--
  ORGANIZER ROAMING DATA TYPE
-->
<xs:complexType name="organizer-roaming-data-type" ms:className="CC3POrganizerDataType">
  <xs:sequence>
    <xs:attribute namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!--
  NOTIFICATION DATA TYPE
-->
<xs:complexType name="notification-data-type" ms:className="CC3PNotificationDataType">
  <xs:sequence>
    <xs:attribute namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!--
  Admission policy for the conference
-->
<xs:element name="admission-policy" type="tns:admission-policy-type" ms:ignore="true"/>
<xs:simpleType name="admission-policy-type">
  <xs:restriction base="xs:string">
    <xs:enumeration value="closedAuthenticated"/>
    <xs:enumeration value="openAuthenticated"/>
  </xs:restriction>
</xs:simpleType>
<xs:enumeration value="anonymous/>
</xs:simpleType>

<!--[PSTN bridging access information for the conference -->
<xs:element name="pstn-access" type="tns:pstn-access-type" ms:ignore="true"/>
<xs:simpleType name="pstn-meeting-id-type">
<xs:restriction base="xs:string">
<xs:pattern value="[1-9][0-9]*"/>
</xs:restriction>
</xs:simpleType>

<xs:complexType name="pstn-access-type" ms:className="CC3PstnAccessType">
<xs:sequence>
<xs:element name="id" type="tns:pstn-meeting-id-type" minOccurs="0" ms:propertyName="Id"/>
<xs:element name="access-numbers" type="tns:pstn-access-numbers-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="AccessNumbers"/>
<xs:element ref="msci2:default-region" minOccurs="0" ms:propertyName="DefaultRegion"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="pstn-access-numbers-type" ms:className="CC3PstnAccessNumbersType">
<xs:sequence>
<xs:element name="internal-url" type="xs:anyURI" minOccurs="0" ms:propertyName="InternalUrl"/>
<xs:element name="external-url" type="xs:anyURI" minOccurs="0" ms:propertyName="ExternalUrl"/>
<xs:element name="region" type="tns:pstn-access-number-region-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="RegionList"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="pstn-access-number-region-type" ms:className="CC3PstnAccessNumberRegionType">
<xs:sequence>
<xs:element name="access-number" type="tns:pstn-access-number-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="AccessNumberList"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="pstn-access-number-type" ms:className="CC3PstnAccessNumberType">
<xs:sequence>
<xs:element name="language" type="tns:pstn-access-number-language-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="Languages"/>
</xs:sequence>
</xs:complexType>

<xs:complexType name="pstn-access-number-language-type"
   ms:className="CC3PPstnAccessNumberLanguageType">
   <xs:sequence>
     <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute name="tag" type="xs:language" use="required" ms:propertyName="Tag"/>
   <xs:attribute name="lcid" type="xs:nonNegativeInteger" use="optional" ms:propertyName="LCID"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
 CONFERENCE VIEW TYPE
 -->
<xs:element name="conference-view" type="tns:conference-view-type" ms:ignore="true"/>
<xs:complexType name="conference-view-type" ms:className="CC3PConferenceViewType">
   <xs:sequence>
     <xs:element name="entity-view" type="entity-view-type" minOccurs="0" maxOccurs="unbounded" ms:propertyName="EntityView"/>
     <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute ref="ci:state" default="full" ms:propertyName="State"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
 ENTITY VIEW TYPE
 -->
<xs:complexType name="entity-view-type" ms:className="CC3PEntityViewType">
   <xs:sequence>
     <xs:element name="entity-capabilities" type="entity-capabilities-type" minOccurs="0" ms:propertyName="EntityCapabilities"/>
     <xs:element name="entity-policy" type="entity-policy-type" minOccurs="0" ms:propertyName="EntitySettings"/>
     <xs:element name="entity-state" type="entity-state-type" minOccurs="0" ms:propertyName="EntityState"/>
     <xs:element name="entity-shared-data" type="entity-shared-data-type" minOccurs="0" ms:propertyName="EntitySharedData"/>
     <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
   <xs:attribute ref="ci:state" default="full" ms:propertyName="State"/>
   <xs:attribute name="entity" type="xs:anyURI" ms:propertyName="Entity"/>
   <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:element name="modal-parameters" type="tns:modal-parameters-type" ms:ignore="true"/>
<xs:complexType name="modal-parameters-type">
   <xs:sequence>
     <xs:choice>
       <xs:element name="audio-parameters" type="msav:audio-parameters-type" minOccurs="0"/>
       <xs:element name="video-parameters" type="msav:video-parameters-type" minOccurs="0"/>
     </xs:choice>
     <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
   </xs:sequence>
</xs:complexType>

<!--
 MS CONFERENCE MEDIA STATES
 -->

<xs:element name="conference-media-states" type="tns:conference-media-states-type" ms:ignore="true"/>
<xs:complexType name="conference-media-states-type">
    <xs:sequence>
        <xs:element name="entry" type="conference-media-state-type" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute ref="ci:state" use="required"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- MS CONFERENCE MEDIA STATE TYPE per MCU -->
<xs:element name="conference-media-state" type="tns:conference-media-state-type" ms:ignore="true"/>
<xs:complexType name="conference-media-state-type">
    <xs:sequence>
        <xs:element name="displayText" type="xs:string" minOccurs="0"/>
        <xs:element name="webPage" type="xs:anyURI" minOccurs="0"/>
        <xs:element name="userCount" type="xs:unsignedInt" minOccurs="0"/>
        <xs:element name="active" type="xs:boolean" minOccurs="0"/>
        <xs:element name="locked" type="xs:boolean" minOccurs="0"/>
        <xs:element name="recording" type="xs:boolean" minOccurs="0"/>
        <xs:element name="mixing" type="xs:boolean" minOccurs="0"/>
        <xs:element name="allMuted" type="xs:boolean" minOccurs="0"/>
        <xs:element name="mediums" type="ci:conference-media-type" minOccurs="0"/>
        <xs:element ref="msav:audio-video-media-state" minOccurs="0"/>
        <xs:sequence minOccurs="0">
            <xs:element ref="cis:separator"/>
            <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:sequence>
    <xs:attribute ref="ci:state" use="required"/>
    <xs:attribute name="entity" type="xs:anyURI"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!-- MCU POLICIES TYPE per MCU -->
<xs:element name="conference-mcu-policies" type="tns:mcu-policies-type" ms:ignore="true"/>
<xs:complexType name="mcu-policies-type">
    <xs:annotation>
        <xs:documentation>TBD</xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="guid" type="xs:anyURI" minOccurs="0"/>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="entity" type="xs:anyURI" use="required"/>
    <xs:attribute name="version" type="xs:string" use="required">
        <xs:annotation>
            <xs:documentation>Identifies the version of MCU settings specified.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
</xs:complexType>
<xs:element name="displayText" type="xs:string" minOccurs="0"/>
<xs:element name="userCount" type="xs:unsignedInt" minOccurs="0"/>
<xs:element name="active" type="xs:boolean" minOccurs="0"/>
<xs:element name="locked" type="xs:boolean" minOccurs="0"/>
<xs:element name="mediaFiltersRules" type="tns:media-filters-rules-type" minOccurs="0"/>
<xs:element name="recording" type="recording-type" minOccurs="0" />
<xs:element name="media" type="ci:conference-media-type" minOccurs="0"/>
<xs:choice minOccurs="0">
  <xs:element ref="msacp:state"/>
  <xs:element ref="msav:state"/>
</xs:choice>
<xs:sequence minOccurs="0">
  <xs:element ref="cis:separator"/>
  <xs:element ref="msas:session-ids" minOccurs="0"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msas:session-ids" minOccurs="0"/>
  </xs:sequence>
  <xs:element ref="cis:separator"/>
  <xs:element ref="msas:session-ids" minOccurs="0"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msas:session-ids" minOccurs="0"/>
  </xs:sequence>
  <xs:element ref="cis:separator"/>
  <xs:attribute name="application" type="xs:string"/>
  <xs:anyAttribute namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
</xs:complexType>
<!--
 ENTITY SHARED DATA TYPE
 -->
<xs:complexType name="entity-shared-data-type">
<xs:sequence>
  <xs:choice minOccurs="0">
    <xs:element ref="msacp:shared-data"/>
  </xs:choice>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="msacp:shared-data"/>'
  </xs:sequence>
  <xs:element ref="cis:separator"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:attribute name="application" type="xs:string"/>
  </xs:sequence>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:attribute name="application" type="xs:string"/>
  </xs:sequence>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="cis:separator"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:element ref="cis:separator"/>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"/>
        <xs:element ref="cis:separator"/>
      </xs:sequence>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"/>
        <xs:element ref="cis:separator"/>
      </xs:sequence>
    </xs:sequence>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:element ref="cis:separator"/>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"/>
        <xs:element ref="cis:separator"/>
      </xs:sequence>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"/>
        <xs:element ref="cis:separator"/>
      </xs:sequence>
    </xs:sequence>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:sequence>
</xs:complexType>
<!--
 Media filters rules
 -->
<xs:complexType name="media-filters-rules-type">
<xs:sequence>
  <xs:element name="mayModifyOwnFilters" type="tns:boolean-role-rule-type" minOccurs="0" maxOccurs="unbounded"/>
  <xs:element name="initialFilters" type="tns:media-filters-role-rule-type" minOccurs="0" maxOccurs="unbounded"/>
  <xs:element ref="msav:type" minOccurs="0"/>
  <xs:sequence minOccurs="0">
    <xs:element ref="cis:separator"/>
    <xs:element ref="cis:separator"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:element ref="cis:separator"/>
    </xs:sequence>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"/>
      <xs:element ref="cis:separator"/>
    </xs:sequence>
  </xs:sequence>
</xs:sequence>
</xs:complexType>
<!--
<xs:element name="clientInfo" type="tns:client-info-type" ms:ignore="true"/>

<xs:complexType name="client-info-type" ms:className="CC3PClientInfoType">
  <xs:sequence>
    <xs:element name="conversation-id" type="xs:string" minOccurs="0" ms:className="CC3PClientInfoExtension1"/>
    <xs:element name="thread-id" type="xs:string" minOccurs="0" ms:className="CC3PClientInfoExtension1"/>
    <xs:sequence minOccurs="0" ms:propertyName="Extension1" ms:className="CC3PClientInfoExtension1">
      <xs:element ref="cis:separator"/>
      <xs:element ref="msci2:user-agent" minOccurs="0" maxOccurs="1" ms:propertyName="LobbyCapable"/>
      <xs:element ref="msci2:lobby-capable" minOccurs="0" ms:propertyName="LobbyCapable"/>
    </xs:sequence>
  </xs:sequence>
</xs:complexType>

<!-- ORGANIZER TYPE -->

<xs:element name="organizer" type="tns:organizer-type" ms:ignore="true"/>

<xs:complexType name="organizer-type">
  <xs:sequence>
    <xs:element name="entity" type="xs:anyURI" minOccurs="0" ms:propertyName="Extension1"/>
    <xs:element name="display-name" type="xs:string" minOccurs="0" ms:propertyName="LobbyCapable"/>
    <xs:sequence minOccurs="0" ms:propertyName="Extension1" ms:className="CC3POrganizerExtension1">
      <xs:element ref="cis:separator"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:sequence>
</xs:complexType>

<!-- SESSION-ON-BEHALF-OF TYPE -->

<xs:element name="session-on-behalf-of" type="tns:session-on-behalf-of-type" ms:ignore="true"/>

<xs:complexType name="session-on-behalf-of-type" ms:className="CC3PSessionOnBehalfOf">
  <xs:sequence>
    <xs:element name="entity" type="xs:anyURI" minOccurs="1" ms:propertyName="Entity"/>
    <xs:sequence minOccurs="0" ms:propertyName="Extension1" ms:className="CC3POrganizerExtension1">
      <xs:element ref="cis:separator"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:sequence>
</xs:complexType>

<!-- IN-CONFERENCING-SERVICES TYPE -->

<xs:element name="in-conferencing-services" type="tns:in-conferencing-services-type" ms:ignore="true"/>

<xs:complexType name="in-conferencing-services-type" ms:className="CC3PINConferencingServices">
  <xs:sequence>
    <xs:element name="entry" type="tns:in-conferencing-services-entry-type" minOccurs="1" maxOccurs="unbounded" ms:propertyName="InConferencingServicesEntry"/>
  </xs:sequence>
</xs:complexType>
<!-- Post dial strings -->
<xs:element name="post-dial" type="xs:string" ms:ignore="true"/>

<!-- pstnLeaderPasscode -->
<xs:element name="pstnLeaderPasscode" type="xs:string" ms:ignore="true"/>

<!-- endpoint capabilities -->
<xs:element name="endpoint-capabilities" type="endpoint-capabilities-type" ms:ignore="true"/>

<xs:complexType name="endpoint-capabilities-type">
  <xs:sequence minOccurs="0">
    <xs:element ref="msacp:endpoint-capabilities" />
    <xs:element ref="msav:endpoint-capabilities" />
    <xs:element ref="msdata:endpoint-capabilities" />
  </xs:sequence>
</xs:complexType>

<!-- media capabilities -->
<xs:element name="media-capabilities" type="tns:media-capabilities-type" ms:ignore="true"/>

<xs:complexType name="media-capabilities-type" ms:className="CC3PMediaCapabilitiesType">
  <xs:sequence>
    <xs:element ref="msas:media-capabilities" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- IS ROBOT ELEMENT -->
<xs:element name="is-robot" type="xs:boolean" ms:ignore="true"/>

<!-- Hash code string -->
<xs:element name="hash-code" type="xs:string" ms:ignore="true"/>

<!-- RECORDING TYPE -->
<xs:complexType name="recording-type">
  <xs:sequence>
    <xs:element name="active" type="xs:boolean" minOccurs="0"/>
    <xs:element name="error" type="error-type" minOccurs="0"/>
    <xs:element name="paused" type="xs:boolean" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="recorded-media" type="recorded-media-entry-collection-type" minOccurs="0"/>
</xs:sequence>
<xs:attribute name="id" type="xs:string" use="required"/>
</xs:complexType>
<!-- RECORDING ENTITIES ELEMENT -->
<xs:element name="recording-entities" type="recording-entity-collection-type" ms:ignore="true"/>
</xs:complexType>
<!-- RECORDING ENTITY COLLECTION TYPE -->
<xs:complexType name="recording-entity-collection-type">
<xs:sequence>
<xs:element name="recording-entity" type="recording-entity-type" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<!-- RECORDING ENTITY TYPE -->
<xs:complexType name="recording-entity-type">
<xs:sequence>
<xs:element name="recorded-media" type="recorded-media-entry-collection-type"/>
</xs:sequence>
<xs:attribute name="entity" type="xs:anyURI" use="required"/>
</xs:complexType>
<!-- RECORDED MEDIA ENTRY COLLECTION TYPE -->
<xs:complexType name="recorded-media-entry-collection-type">
<xs:sequence>
<xs:element name="entry" type="recorded-media-entry-type" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>
<!-- RECORDED MEDIA ENTRY TYPE -->
<xs:complexType name="recorded-media-entry-type">
<xs:sequence>
<xs:element name="error" type="error-type" minOccurs="0"/>
</xs:sequence>
<xs:attribute name="type" type="xs:string" use="required"/>
</xs:complexType>
<!-- ERROR ELEMENT -->
<xs:element name="error" type="error-type" ms:ignore="true"/>
</xs:complexType>
<!-- ERROR TYPE -->
<xs:complexType name="error-type">
<xs:sequence>
<xs:element name="code" type="xs:string"/>
<xs:element name="description" type="xs:string" minOccurs="0"/>
</xs:sequence>
</xs:complexType>
<!-- session-type string -->
<xs:attribute name="session-type" type="xs:string"/>
<!-- epid -->
<xs:attribute name="epid" type="xs:string"/>
<!-- sip-instance-->  
<xs:attribute name="sip-instance" type="xs:string"/>
<!-- endpoint-uri uri -->
<xs:attribute name="endpoint-uri" type="xs:anyURI"/>
<!-- refer-to-uri uri -->
<xs:attribute name="refer-to-uri" type="xs:anyURI"/>
<!-- asserted-identity -->
6.3 avconfinfoextensions

Namespace(http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions)

Schema

Following is the schema for the avconfinfoextensions namespace.

```xml
"/>"
<!-- This import brings in the standard Conference Package Standard Separator definitions -->
<xs:import namespace="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions" schemaLocation="mcucommon.xsd" />
<xs:complexType name="capabilities-type">
  <xs:sequence>
    <!-- The following MSAV settings are actually derived from policy, and appear here to reflect a view of "capabilities" that are of interest to some clients. These are read-only. The effective value can only be changed via policy (entity-policy element of entity-view) -->
    <xs:element name="supports-audio" type="xs:boolean" />
    <xs:element name="supports-video" type="xs:boolean" />
  </xs:sequence>
</xs:complexType>
<xs:element name="capabilities" type="tns:capabilities-type" />
<xs:complexType name="audio-settings-type">
  <xs:sequence>
    <xs:any namespace="#any" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
<xs:complexType name="video-settings-type">
  <xs:sequence>
    <xs:any namespace="#any" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
<xs:complexType name="settings-type">
  <xs:sequence>
    <xs:element name="audio" type="tns:audio-settings-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="video" type="tns:video-settings-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element ref="msmcu:entry-exit-announcements" minOccurs="0" maxOccurs="unbounded"/>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator" />
      <xs:any namespace="#other" processContents="lax" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:sequence>
</xs:complexType>
<xs:element name="settings" type="tns:settings-type" ms:ignore="true" />
<xs:complexType name="contributing-sources-type">
  <xs:sequence>
    <xs:element name="entry" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded" />  
    <xs:attribute name="empty" type="xs:boolean" use="optional" />
  </xs:sequence>
</xs:complexType>
<xs:complexType name="dominant-speaker-source-type">
  <xs:sequence>
    <xs:element name="entry" type="xs:anyURI" minOccurs="0" maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>
<xs:element name="video-parameters" type="tns:video-parameters-type" ms:ignore="true" />
<xs:complexType name="video-parameters-type">
  <xs:sequence>
    <!-- The video switching mode. Supported values are "dominant-speaker-switched" and "manual-switched" -->
  </xs:sequence>
</xs:complexType>
-->  
<xs:element name="video-mode" type="xs:string" minOccurs="0"/>

<!-- The desired video source in manual switched mode -->
<xs:element name="intended-primary-presenter-source" type="tns:contributing-sources-type" minOccurs="0"/>

<!-- The intended-secondary-presenter-source is reserved for future use. -->
<xs:element name="intended-secondary-presenter-source" type="tns:contributing-sources-type" minOccurs="0"/>

<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:element name="audio-parameters" type="tns:audio-parameters-type" ms:ignore="true"/>
<xs:complexType name="audio-parameters-type">
<xs:sequence>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="state-type" ms:className="C3PAvMcuStateType">
<xs:sequence>
<xs:element name="video-mode" type="xs:string" minOccurs="0"/>
<xs:element name="dominant-speaker-source" type="tns:dominant-speaker-source-type" minOccurs="0"/>
<xs:element name="intended-prime-presenter-source" type="tns:contributing-sources-type" minOccurs="0"/>
<xs:element name="intended-secondary-presenter-source" type="tns:contributing-sources-type" minOccurs="0"/>

<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:element name="audio-video-media-state" type="tns:state-type" ms:ignore="true"/>
<xs:element name="state" type="tns:state-type" ms:ignore="true"/>
<xs:complexType name="endpoint-capabilities-type" ms:className="C3PAvMcuEndpointCapabilitiesType">
<xs:sequence>
<xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<xs:element name="endpoint-capabilities" type="tns:endpoint-capabilities-type" ms:ignore="true"/>
<xs:complexType name="media-routing-type">
<xs:sequence>
<xs:element name="controls" type="tns:controls-parameters-type" minOccurs="0"/>

<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<xs:complexType name="controls-parameters-type">
<xs:sequence>
<xs:element name="route" type="tns:route-parameters-type" minOccurs="0"/>
</xs:sequence>
</xs:complexType>
6.4 commonmcuextensions Namespace

(http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions) Schema

Following is the schema for the conference-info-extensions namespace.

<?xml version="1.0" encoding="utf-8"?>
<xs:schema version="1.0"
  targetNamespace="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
  xmlns:tns="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
  xmlns:ms="urn:microsoft-cpp-xml-serializer"
  xmlns="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
  xmlns:ci="urn:ietf:params:xml:ns:conference-info"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">
  <!-- C3P schema extension for elements common to multiple MCUs -->
  <!-- This imports the standard separator -->
    schemaLocation="ms-ci-separator.xsd" />
</xs:schema>
This import brings in the MS Conference Package extensions.

```xml
```

```xml
<!--
Sequence of possible capabilities, to be inserted (with
reference to the capability-value-type type) as they are defined in the
future. The structure of this XML should look like media-capabilities
(see ms-ci-ext.xsd)
-->
<xs:complexType name="session-capabilities-type" ms:className="C3PmcuCommonSessionCapabilitiesType">
<xs:sequence>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="value" type="tns:capability-value-type"/>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<xs:element name="session-capabilities" type="tns:session-capabilities-type" ms:ignore="true"/>
<xs:complexType name="capability-value-type">
<xs:annotation>
  <xs:documentation>
  Possible capability values (for both endpoint-capabilities and media-capabilities)
  </xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
  <xs:enumeration value="sendrecv"/>
  <xs:enumeration value="sendonly"/>
  <xs:enumeration value="recvonly"/>
  <xs:enumeration value="none"/>
</xs:restriction>
</xs:simpleType>
<xs:complexType name="media-capability-type">
<xs:sequence>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="value" type="tns:capability-value-type"/>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:simpleType name="media-capability-type"/>
<xs:element name="presentation-mode-capable" type="xs:boolean" ms:ignore="true"/>
<xs:element name="multi-view-capable" type="xs:boolean" ms:ignore="true"/>
<xs:element name="video-presentation-mode-capable" type="xs:boolean" ms:ignore="true"/>
<xs:complexType name="permission-options-type">
<xs:element name="permission-options" type="tns:permission-options-type" ms:ignore="true"/>
</xs:complexType>"
<xs:sequence>
  <xs:element name="permission-option" type="tns:permission-option-type"
    minOccurs="0" maxOccurs="unbounded"/>
  <xs:element name="name" type="xs:string" maxOccurs="unbounded"/>
  <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
  <xs:element name="mutable" type="xs:boolean" maxOccurs="unbounded"/>
  <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
</xs:complexType>

<!-- PERMISSION OPTION TYPE -->
<xs:complexType name="permission-option-type">
  <xs:sequence>
    <xs:element name="name" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="mutable" type="xs:boolean" maxOccurs="unbounded"/>
    <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- PERMISSIONS TYPE -->
<xs:element name="permissions" type="tns:permissions-type" ms:ignore="true"/>
<xs:complexType name="permissions-type">
  <xs:sequence>
    <xs:element name="permission-type" type="tns:permission-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="name" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="mutable" type="xs:boolean" maxOccurs="unbounded"/>
    <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- PERMISSION TYPE -->
<xs:element name="permissions-type" ms:ignore="true"/>
<xs:complexType name="permissions-type">
  <xs:sequence>
    <xs:element name="permission-type" type="tns:permission-type" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="name" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="value" type="xs:string" maxOccurs="unbounded"/>
    <xs:element name="mutable" type="xs:boolean" maxOccurs="unbounded"/>
    <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- ENTRY-EXIT ANNOUNCEMENTS TYPE -->
<xs:element name="entry-exit-announcements" type="tns:entry-exit-announcements-type"
  ms:ignore="true"/>
<xs:complexType name="entry-exit-announcements-type">
  <xs:sequence>
    <xs:element name="modifiable" type="xs:boolean" minOccurs="0" maxOccurs="unbounded"/>
    <xs:element name="enabled" type="xs:boolean" minOccurs="0" maxOccurs="unbounded"/>
    <xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- CONFDとなっています - v20181211
Centralized Conference Control Protocol: Audio-Video Extensions
Copyright © 2018 Microsoft Corporation
Release: December 11, 2018

DataMcu state/settings elements added after W14
<xs:element name="hasWacConnectivity" type="xs:boolean" ms:ignore="true"/>
</xs:schema>
Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include updates to those products.

- Microsoft Office Communications Server 2007
- Microsoft Office Communications Server 2007 R2
- Microsoft Office Communicator 2007
- Microsoft Office Communicator 2007 R2
- Microsoft Lync Server 2010
- Microsoft Lync 2010
- Microsoft Lync Server 2013
- Microsoft Lync Client 2013/Skype for Business
- Microsoft Skype for Business 2016
- Microsoft Skype for Business Server 2015
- Microsoft Skype for Business 2019
- Microsoft Skype for Business Server 2019

Exceptions, if any, are noted in this section. If an update version, service pack or Knowledge Base (KB) number appears with a product name, the behavior changed in that update. The new behavior also applies to subsequent updates unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms "SHOULD" or "SHOULD NOT" implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term "MAY" implies that the product does not follow the prescription.

<1> Section 2.2.1.1.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: "manual-switched" is not a supported value for video-mode.

<2> Section 2.2.1.1.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: Video Source Requests are not supported and the video source is selected by the MCU.

<3> Section 2.2.1.1.5.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The type element is not supported and ignored if present in a request or notification. It is assumed to have a value of "audio".

<4> Section 2.2.2.1.2.3: Office Communicator 2007, Office Communications Server 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2, Lync Server 2010, Lync 2010: The media-source-id element is not supported. It is ignored if present in a notification.

<5> Section 2.2.2.1.2.4: Office Communicator 2007, Office Communications Server 2007, Office Communications Server 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The source-name element is not supported. It is ignored if present in a notification.
<6> Section 2.2.2.2.1.5: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The multi-view-capable element is not supported. It is ignored if present in either a request or a notification.

<7> Section 2.2.2.2.1.6: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The video-presentation-mode-capable element is not supported. It is ignored if present in either a request or a notification.

<8> Section 2.2.2.2.1.7: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The conf-media-filters-rules element is not supported. It is ignored if present in either a request or a notification.

<9> Section 3.2.3.1.1.2.1: Office Communications Server 2007, Office Communications Server 2007 R2: This behavior is not supported.

<10> Section 3.2.3.1.1.2.2: Office Communications Server 2007, Office Communications Server 2007 R2: This behavior is not supported.

<11> Section 3.2.3.1.1.2.3: Office Communications Server 2007, Office Communications Server 2007 R2: This behavior is not supported.

<12> Section 3.2.3.1.1.2.5: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The multi-view-capable element is not supported. It is ignored if present in either a request or a notification.

<13> Section 3.2.3.1.1.2.6: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The video-presentation-mode-capable element is not supported. It is ignored if present in either a request or a notification.

<14> Section 3.2.3.1.1.2.7: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The conf-media-filters-rules element is not supported. It is ignored if present in either a request or a notification.

<15> Section 3.2.5.4: Applicable to Office Communications Server 2007 and Office Communications Server 2007 R2.

<16> Section 3.2.5.4: Applicable to Lync Server 2010.

<17> Section 3.2.5.6.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: The modification of media-filters-rule-type is not supported.

<18> Section 3.2.5.6.1: Lync Server 2010, Lync 2010: The modification of media-filters-rule-type is only supported for mediaFiltersRules element and the type is assumed to be "audio".

<19> Section 3.2.5.6.1: Office Communicator 2007, Office Communications Server 2007, Office Communicator 2007 R2, Office Communications Server 2007 R2, Lync 2010, Lync Server 2010: The modification of media-filters-rule-type for media type "video" is not supported.

8 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

- A document revision that incorporates changes to interoperability requirements.
- A document revision that captures changes to protocol functionality.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

The changes made to this document are listed in the following table. For more information, please contact dochelp@microsoft.com.

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Revision class</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.5</td>
<td>modifyConference</td>
<td>Minor</td>
</tr>
<tr>
<td></td>
<td>Removed instances of a normative.</td>
<td></td>
</tr>
</tbody>
</table>
9 Index

A

Abstract data model
  client 26
  server 28
  correlation of media instances 29
  correlation of media parameters 28
addUser dial-in request
  client
    message processing 26
    sequencing rules 26
    example 48
  server
    message processing 38
    sequencing rules 38
addUser dial-in request document syntax 23
addUser dial-out request
  client
    message processing 27
    sequencing rules 27
    example 47
  server
    message processing 38
    sequencing rules 38
addUser dial-out request document syntax 23
Applicability 12
avconfinfoextensions namespace
  schema 95

C

C3P Request and Response Document Formats
message
  getConference response document (section 2.2.1.1.5.3 18, section 2.2.2.1.2.3 20, section 2.2.2.1.2.4 20, section 2.2.2.1.2.5 22, section 2.2.2.1.6 22, section 2.2.2.1.7 22, section 3.2.3.1.1.2.1 30, section 3.2.3.1.1.2.2 31, section 3.2.3.1.1.2.3 31, section 3.2.3.1.1.2.5 32, section 3.2.3.1.1.2.6 32, section 3.2.3.1.1.2.7 32)
C3P request/response Document Content
  addUser dial-in request document syntax 23
  addUser dial-out request document syntax 23
  modifyConferenceAnnouncements request syntax 24
  modifyEndpoint media request syntax 24
Capability negotiation 12
Change tracking 104
Client
  abstract data model 26
  higher-layer triggered events 26
  initialization 26
  message processing
    addUser dial-in request 26
    addUser dial-out request 27
    SIP INVITE dial-in request 26
    SIP INVITE dial-out request 26
  other local events 27
  sequencing rules
    addUser dial-in request 26
    addUser dial-out request 27
  SIP INVITE dial-in request 26
  timer events 27
  timers 26
  Conceptual conference document structure 10
  Conference activation 30
  conference-info namespace
    schema 72
  conference-info-extensions namespace
    schema (section 6.2 80, section 6.4 98)
  Correlation of media instances 29
  Correlation of media parameters 28

D

Data model - abstract
  client 26
  server 28
  correlation of media instances 29
  correlation of media parameters 28

E

Examples
  addUser dial-in request 48
  addUser dial-out request 47
  modifyConference request 63
  modifyConferenceAnnouncements request 58
  modifyEndpointMedia request 54
Extension Semantics of application/conference-info+xml Document Format message 14
XML schema types 14

F

Fields - vendor-extensible 13

G

Glossary 7

H

Higher-layer triggered events
  client 26
  server 34

I

Implementer - security considerations 71
Index of security parameters 71
Informative references 9
Initialization
  client 26
  server
    conference activation 30
Introduction 7

M

MCU bootstrap 30
modifyConference request 41
modifyConferenceAnnouncements request 41
modifyEndpointMedia request 39

timer events 43
timers 30
SIP dialog events 44
Standards assignments 13
Syntax 14

T

Timer events
  client 27
  server 43
Timers
  client 26
  server 30
Tracking changes 104
Transport 14
Triggered events - higher-layer
  client 26
  server 34

U

User signaling events 44

V

Vendor-extensible fields 13
Versioning 12