[MS-CONFAV]:
Centralized Conference Control Protocol:
Audio-Video Extensions

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
- **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 may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft's Open Specification Promise (available here: http://www.microsoft.com/interop/osp) or the Community Promise (available here: http://www.microsoft.com/interop/cp/default.mspx). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting iplg@microsoft.com.
- **Trademarks.** The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights.
- **Fictitious Names.** The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events 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 specifically described above, whether by implication, estoppel, or otherwise.

Tools. The Open Specifications do 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 are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial version
04/25/2008	0.2		Revised and edited the technical content
06/27/2008	1.0		Revised and edited the technical content
08/15/2008	1.01		Revised and edited the technical content
12/12/2008	2.0	Major	Revised and edited the technical content
02/13/2009	2.01		Edited the technical content
03/13/2009	2.02		Edited the technical content
07/13/2009	2.03	Major	Revised and edited the technical content
08/28/2009	2.04	Editorial	Revised and edited the technical content
11/06/2009	2.05	Editorial	Revised and edited the technical content
02/19/2010	2.06	Editorial	Revised and edited the technical content
03/31/2010	2.07	Major	Updated and revised the technical content
04/30/2010	2.08	Editorial	Revised and edited the technical content
06/07/2010	2.09	Editorial	Revised and edited the technical content
06/29/2010	2.10	Editorial	Changed language and formatting in the technical content.
07/23/2010	2.10	No change	No changes to the meaning, language, or formatting of the technical content.
09/27/2010	3.0	Major	Significantly changed the technical content.
11/15/2010	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
12/17/2010	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
03/18/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.
06/10/2011	3.0	No change	No changes to the meaning, language, or formatting of the technical content.

Table of Contents

1 Introduction

This document 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-CONFBAS].

1.1 Glossary

The following terms are defined in [MS-GLOS]:

server

The following terms are defined in [MS-OFCGLOS]:

200 OK
Audio/Video Multipoint Control Unit (AVMCU)
codec
conference
data type
dialog
endpoint
endpoint identifier (EPID)
focus
Interactive Connectivity Establishment (ICE)
Internet message
INVITE
MCU-Conference-URI
mixer
Multipoint Control Unit (MCU)
notification
participant
Real-Time Transport Protocol (RTP)
remote endpoint
SDP answer
SDP offer
Session Description Protocol (SDP)
Session Initiation Protocol (SIP)
SIP message
Synchronization Source (SSRC)
Uniform Resource Identifier (URI)
user agent client (UAC)

The following terms are specific to this document:

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

1.2 References

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. Please check the archive site,

http://msdn2.microsoft.com/en-us/library/E4BD6494-06AD-4aed-9823-445E921C9624, as an additional source.

[MS-CONFBAS] Microsoft Corporation, "Centralized Conference Control Protocol: Basic Architecture and Signaling Specification"

[MS-CONFPRO] Microsoft Corporation, "Centralized Conference Control Protocol: Provisioning Specification"

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

[MS-SIPRE] Microsoft Corporation, "Session Initiation Protocol (SIP) Routing Extensions"

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

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", RFC 3261, June 2002, http://www.ietf.org/rfc/rfc3261.txt

[RFC3264] Rosenberg, J., and Schulzrinne, H., "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002, http://www.rfc-editor.org/rfc/rfc3264.txt

[RFC4566] Handley, M., Jacobson, V., and Perkins, C., "SDP: Session Description Protocol", RFC 4566, July 2006, http://www.ietf.org/rfc/rfc4566.txt

[RFC4574] Levin, O., and Camarillo, G., "The Session Description Protocol (SDP) Label Attribute", RFC 4574, August 2006, http://www.rfc-editor.org/rfc/rfc4574.txt

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

1.2.2 Informative References

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary".

[MS-ICE] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions"

[MS-ICE2] Microsoft Corporation, "Interactive Connectivity Establishment (ICE) Extensions 2.0"

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary".

[MS-RTP] Microsoft Corporation, "Real-time Transport Protocol (RTP) Extensions"

[MS-RTPRADEX] Microsoft Corporation, "RTP Payload for Redundant Audio Data Extensions"

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

1.3 Protocol Overview (Synopsis)

The Centralized Conference Control Protocol (C3P) is described in [MS-CONFBAS], which in turn extends [RFC4575] and [RFC4353]. [RFC4575] describes a **Session Initiation Protocol (SIP)** Event Package for conference state. [RFC4353] provides a conceptual description of a framework for conferencing with SIP. [MS-CONFBAS] describes a framework for aggregating multiple instances of a **Multipoint Control Unit (MCU)** in the context of what [RFC4575] section 4 describes as a single

logical conference. [MS-CONFBAS] describes concrete extensions to [RFC4575] that are built on the concepts in [RFC4353].

Within [MS-CONFBAS], 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 (5)** for each user. Each endpoint (5) 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. The conference information that falls within the scope of a single logical MCU entity generally resides in this container. MCU-specific endpoints (5) 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 document location 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 (5) 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 A/V MCU entity is the suite of protocols specified by the following:

Protocol	Description
[MS-SIPRE]	Extensions to SIP
[MS-SDPEXT]	Extensions to SDP
[RFC3264]	An offer/answer Model with SDP

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-CONFBAS]. It does define semantics of some parts of the XML schema and C3P message constructs in more detail than

those described in [MS-CONFBAS], particularly where [MS-CONFBAS] 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-CONFBAS].
- Extends the semantics of **conference-medium-type** relative to [RFC4575].

1.4 Relationship to Other Protocols

In addition to the dependencies described in [MS-CONFBAS] 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-CONFBAS] 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-CONFBAS].

1.7 Versioning and Capability Negotiation

This protocol does not have any additional versioning and capability negotiation constraints beyond those described in [MS-CONFBAS].

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 can define the semantics of other elements and attributes that they depend upon. They can 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] can 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 can 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 (5), 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 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 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 the string literal value "dominant-speaker-switched". This value specifies that the video source is dynamically selected based on the contents of the audio streams received by the MCU. This is the default mode that implementations MUST support.

The **video-mode** element is optional. If this element is not present, it is assumed to have a value of "dominant-speaker-switched".

2.2.1.1.3 capabilities-type*

The XML type **msav:capabilities-type** is intended specifically for A/V conference modalities. It is defined in the avconfinfoextensions namespace that is found here: http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions.

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**. It is defined in the **commonmcuextensions** namespace at http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions.

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

Enabled (xs:boolean): This element specifies whether entry and exit announcements in the **conference** will be played or not.

Modifiable (xs:boolean): This element 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**. It is defined in the **confinfoextensions** namespace at http://schemas.microsoft.com/rtc/2005/08/confinfoextensions.

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 2.2.1.1.1.

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 in the media element with the type equal to "audio".

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**.

2.2.1.1.5.2 initialFilters

The **initialFilters** element specifies the default values that are applied to each **participant's (2)** media element, where type equals "audio", 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 2.2.1.1.1.

2.2.2 MCU Conference Roster Document Format

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

2.2.2.1 MCU endpoint Element Syntax

The model defined in [MS-CONFBAS] 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]. Extensions to [RFC4575] are specified in [MS-CONFBAS]. 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-CONFBAS], 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 notification messages 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.

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 (5) for the stream it sends. This value is available in the SDP offer and SDP answer used to negotiate media, as specified in [RFC3264].

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 2.2.1.1.1.1.

The **media-ingress-filter** element specifies the filter value for media in the direction of a client endpoint (5) 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 2.2.1.1.1.1.

The **media-egress-filter** element specifies the filter value for media in the direction of the MCU to a client endpoint (5).

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.

2.2.2.2 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].

2.2.2.2.1 entity-state Extension Elements

This protocol defines the following extension elements of the **entity-state** element.

2.2.2.2.1.1 media Element Extensions

The Conference Document format defined in [MS-CONFBAS] 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-CONFBAS]. 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-CONFBAS], the semantics specified in [RFC4575] section 5.3.4 remain intact.

2.2.2.2.1.1.1 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.

2.2.2.2.1.1.2 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.

2.2.2.2.1.2 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 .

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 2.2.1.1.5.

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 2.2.3.13 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**.

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 2.2.3.15 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**.

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 2.2.2.1.1.1.

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 (5) 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 MCU Conference Roster state notifications.

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 2.2.3.23, the following rules apply.

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

The Boolean type element **modifiable** can be present. This element is assigned by an MCU and cannot be modified.

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 3.10, the dial-in request can be sent by the client or by the focus itself to the MCU. Clients do not typically 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 3.10.4.1,

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 3.10 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 an **entity-state** element, which 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:

- 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 3.3.5.1.3.
- 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-CONFBAS] section 3.9, A/V-specific **addUser** dial-out requests

MUST conform to the syntax rules specified in section 2.2.3.2.

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 external messages always 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.

Conference Media(XML type "conference-medium-type")	User Media(XML type "media-type")	SDP Media Description("m" line)
label attribute	**label** element	**label** attribute [RFC4574]
type element	**type** element	**m=** field value. For example, m=audio
status element	**status** element	multiple **a=** attribute values: a=sendrecv a=sendonly a=recvonly a=inactive

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:

User media instances	SDPState	SDP media instances
<media id="1">	"Active" or "Rejected"	"m=" line 2
<media id="2">	"Active" or "Rejected"	"m=" line 1
<media id="3">	"NotInstantiated"	(None)
(None)	"Rejected"	"m=" line 3

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 audioor video MCU and initializing its conference state.

In addition to the requirements specified in [MS-CONFBAS] section 3.3.4.3, the following semantics and content apply to bootstrapping an AVMCU.

3.2.3.1.1 Initial Full Conference Notification

The contents of the first 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

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** section.

```
<msmcu:entry-exit-announcements>
    <msmcu:modifiable>true</msmcu:modifiable>
    <msmcu:enabled>false</msmcu:enabled>
</msmcu:entry-exit-announcements>
```

3.2.3.1.1.2.2 mediaFiltersRules element

The **mediaFiltersRules** element MUST be present and MUST be fully populated in the initial 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 in the entity-state section:

```
<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>
```

</msci:mediaFiltersRules>

3.2.3.1.1.2.3 presentation-mode-capable element

The **presentation-mode-capable** element can be present and, if present, MUST adhere to the syntax rules specified in section 2.2.2.2.1.3. The value of this element MUST be "true".

If the presentation-mode capable element is present, then the AVMCU MUST support modification of mediaFiltersRules as discussed in section 3.2.3.1.1.2.2 using the ModifyConference request shown in section 4.5

3.2.3.1.1.2.4 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 2.2.2.2.1.1 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".
- The value of the **media-ingress filter** in **media** elements that have a **type** element value of "audio" is subject to the **mediaFiltersRules** in the entity-state (described in section 2.2.2.2.1.4 and section 3.2.3.1.1.2.2) at all times for all user endpoints that are not of trusted user type (need reference to trusted user definition).

The **conference-view** element package described in [MS-CONFBAS] section 2.2.4.3 is used for conference notifications. When a first notification returned by the **server (2)** 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:

```
<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: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:entity-capabilities>
          <msci:entity-state>
               <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>
</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>
                                        dominant-speaker-switched
                                   </msav:video-mode>
                              </msci:video-parameters>
                         </msci:modal-parameters>
                    </entry>
                    <entry label="panoramic-video">
                         <type>panoramic-video</type>
```

```
                        <status>sendrecv</status>
                    </entry>
                </msci:media>
                <cis:separator/>
                <cis:separator/>
                <msmcu:presentation-mode-capable>true</msmcu:presentation-mode-capable>
                <msmcu:entry-exit-announcements>
                    <msmcu:modifiable>true</msmcu:modifiable>
                    <msmcu:enabled>false</msmcu:enabled>
                </msmcu:entry-exit-announcements>
            </msci:entity-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. That is, 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, **media-ingress-filter** element, and **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.

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 other sequences, 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 proper 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 for possible future use.
- 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 "opposite" 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.

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:

```
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]. This is a "reverse" direction.

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**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 **addUser** dial-out request is used to initiate connecting a participant to an MCU. [MS-CONFBAS] section 3.9 specifies **addUser** dial-out behavior and message processing rules that are common to all MCUs. This section specifies the **addUser** dial-out behavior that is specific to the Audio/Video MCU role.

Unless otherwise specified, all of the message processing and sequencing rules specified in [MS-CONFBAS] section 3.9 for the MCU's role in processing **addUser** dial-out commands apply.

Upon receiving an **addUser** dial-out command, assuming that the prerequisite conditions specified in [MS-CONFBAS] section 3.9 are met, the following additional rules apply.

- The MCU MUST validate the contents of the **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 **addUser** dial-out 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".

Construction of the SIP INVITE message consists of two steps:

- Determining the SIP **URI** to send the message to and construction of the **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 SIP INVITE request that is used to carry the SDP content is constructed as specified in [MS-CONFBAS] section 3.9.4.1.1. This protocol defines only SDP-specific extension rules to the **addUser** dial-out request.

The target SIP URI is obtained using the rules specified in [MS-CONFBAS] section 3.9.4.1.1.

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 **media** element contained in the **endpoint** element of the **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 **media** element contained in the **endpoint** element of the **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 SIP INVITE message containing the constructed SDP offer to the specified target URI.

The MCU SHOULD send an **addUser** "Pending" response at this point.

An example appears in section 4.

3.2.5.3 addUser Dial-in Request

Processing the **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, t

he 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 3.10.5.2.1 apply.

In addition, t

he 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 3.10.5.2.1.

3.2.5.4 modifyEndpointMedia Request

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

- Identifying the targeted user and user media instance that the request applies to.
- 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, the MCU MUST send a **modifyEndpointMedia** error response with the reason "UserDoesntExist".
- If the MCU-specific endpoint (5) 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 (5) 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:
- If the **originator uri** is the same as the **target uri** originator of the request, **Ref originator uri**, the AVMCU MUST retrieve the **mayModifyOwnFilters** rule for the role of the user originating the request and, if the value is "true", the AVMCU can process the request further. If the value of the **mayModifyOwnFilters** rule for the originating user's role 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".
- If the **originator uri** is not the same as the **target uri**, the AVMCU MUST retreive the role of the user who originated the request. If the role is "presenter", the AVMCU MUST process the request and apply the **mayModifyFilterRules** for the "presenter". Additionally, if the originator uri is not present in the AV portion of the conference, the **mayModifyFilterRules** for the "presenter" role are applied. Otherwise, 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".

- 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 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:

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

- If permitted, making changes contained in the request.

The following recommendations apply:

- If the AVMCU receives the **modifyConferenceAnnouncements** request for a non-existent conference, it MUST send a failure response with the reason "ConferenceDoesntExist".
- If the AVMCU is not able to process the **modifyConferenceAnnouncements** request for any other reason, it MUST send a failure response with the reason "OtherFailure".

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

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

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

Upon completion of the previous procedure, the AVMCU 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 AVMCU 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 (2) to an AVMCU. 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, and specifically cases that contain the **mediaFiltersRules** element.

The following rules apply for the AVMCU's handling of **modifyConference** requests:

- The AVMCU 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 AVMCU MUST a **ModifyConference** error response with **reason** equal to "unauthorized" and **modifyConference reason** equal to "OtherFailure".
- If the originator is present in the conference, the AVMCU checks the originating user's role. If the role is not "presenter", the AVMCU MUSTa **ModifyConference** error response with **reason** equal to "unauthorized" and **modifyConference reason** equal to "OtherFailure".

  If the request is valid, the following processing rules apply to all instances of the **media** element of **type** equal to "audio" for non-trusted users, with the exception of the originator of the request.
- The AVMCU MUST update the **mediaFiltersRules** element in the **entity-state** with the **mediaFiltersRules** 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** for a user role is **"blocked"**, the AVMCU MUST update the **media-ingress-filter** in the **media** element of **type** equal to **"**audio**"** of all user endpoints (5) with that user role to "blocked". The

AVMCU 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 AVMCU MUST NOT apply changes to the **media-ingress-filte**r instances in the media element for all instances of user endpoints (5) with that role.

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

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

- The new values within the AVMCU's **entity-state** element.
- The current user state of all users that were affected by the **ModifyConference** request.

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

- The AVMCU MUST update the **mediaFiltersRules** element in the **entity-state** with the **mediaFiltersRules** 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** for a user role is **"blocked"**, the AVMCU MUST update the **media-ingress-filter** in the **media** element of **type** equal to **"**audio**"** of all user endpoints (5) with that user role to "blocked". The AVMCU 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 AVMCU MUST NOT apply changes to the **media-ingress-filte**r instances in the media element for all instances of user endpoints (5) with that role.

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

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

- The new values within the AVMCU's **entity-state** element.
- The current user state of all users that were affected by the **ModifyConference** request.

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

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

- The new values within the AVMCU's **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.

3.2.6 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.

If **GRUU** is present, s

- earch the list for an entry containing a matching value of the **endpoint-uri** attribute of the **endpoint** element within the **user** element.

If **EPID** is present, s

- earch 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 **URI** value contained in the **FROM** header.

If **SIP.INSTANCE** is present, s

- earch 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.

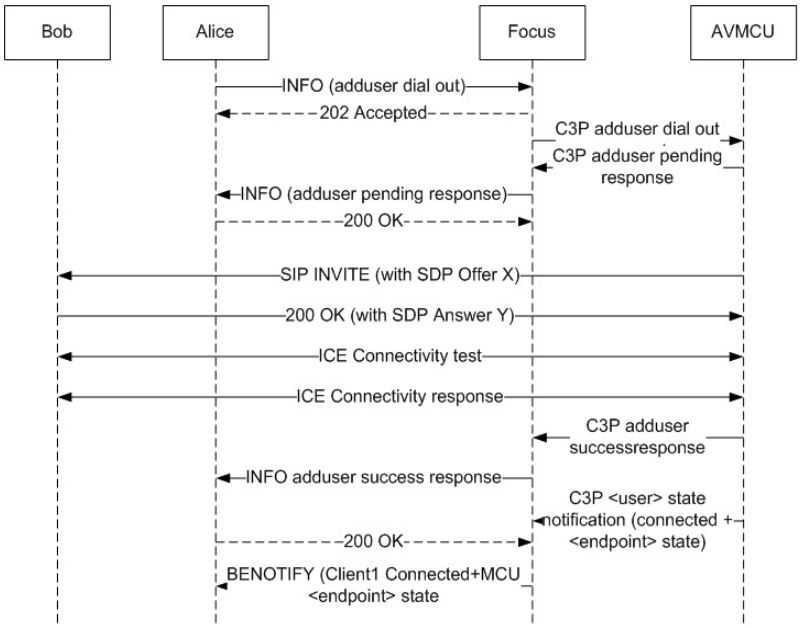

Figure 1: addUser dial-out call flow sequence

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 rspauth="0100000000000000A7C4683CD3C5FC49",
srand="70BB5069", snum="2068", 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:
1650
```

```
From:
<sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=7
F6C0080 Call-ID: 4555a2d175ab44b6ab0e4b5754570d76
CSeq: 5 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="8">
     <users state="partial">
          <user entity="sip:bob@fabrikam.com" state="full">
               <display-text>Bob Freer</display-text>
               <roles>
                    <entry>attendee</entry>
               </roles>
<endpoint entity="{679ACDDB-2171-4176-8414-6ABA41ED881A}" msci:session-type="audio-video"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
                    <joining-method>dialed-out</joining-method>
                    <media id="1">
                         <type>audio</type>
                         <label>main-audio</label>
                         <status>sendrecv</status>
                    </media>
                    <msci:roles>
                         <entry>attendee</entry>
                    </msci:roles>
                    <msci:authMethod>enterprise</msci:authMethod>
                    <msci:accessMethod>internal</msci:accessMethod>
                    <cis:separator/>
                    <msci:languages>en-US</msci:languages>
               </endpoint>
          </user>
     </users>
</conference-info>
```

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.

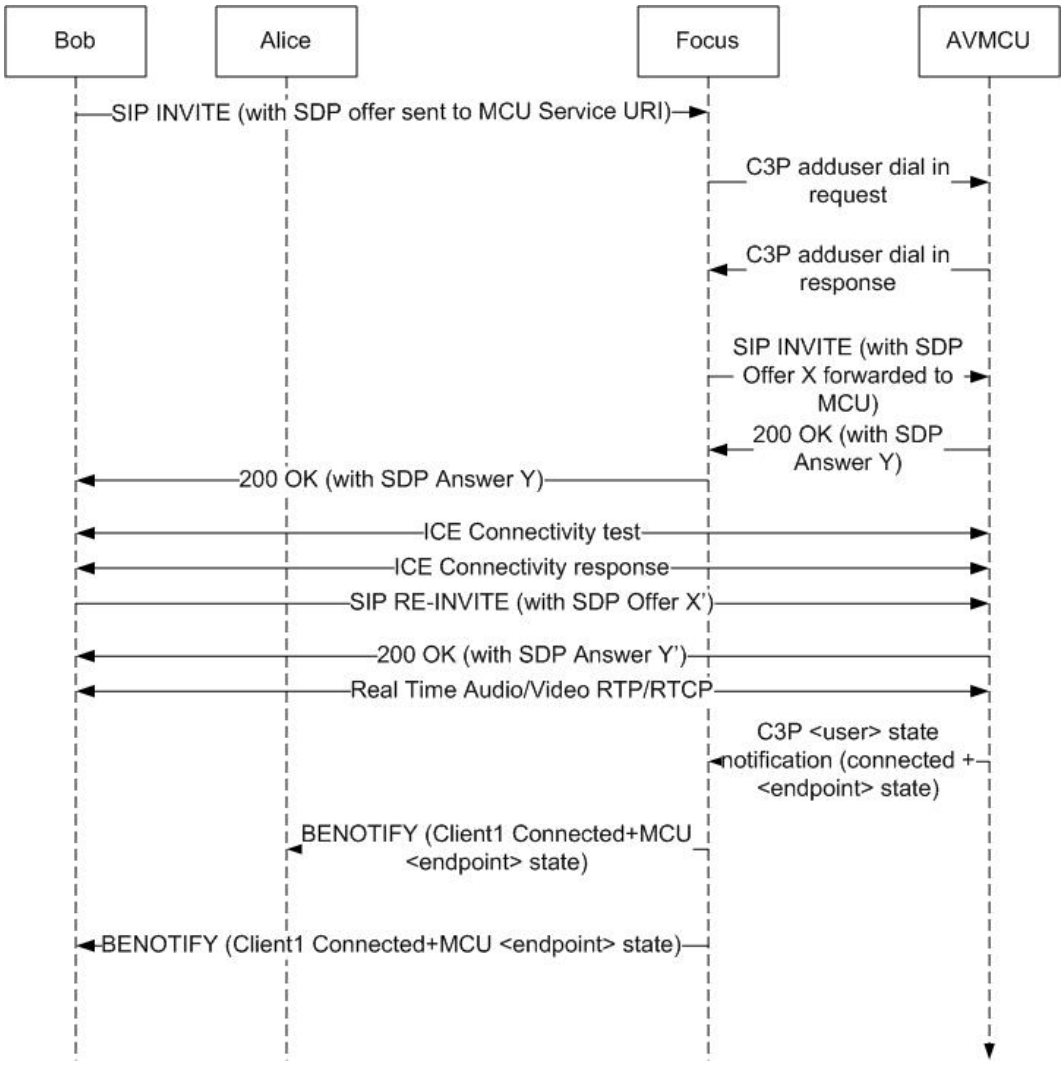

Figure 2: addUser dial-in call flow sequence

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

```
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
 xmlns: 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>
                    <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>
          </conf-uris>
     </conference-description>
     <users state="full">
          <user entity="sip:alice@fabrikam.com" state="full">
               <display-text>Alice Smith</display-text>
               <roles>
                    <entry>presenter</entry>
               </roles>
               <endpoint entity="{0ABEC1D8-039C-4272-ACEE-C0D00057D64E}"
msci:session-type="focus" msci:epid="02e00da898" msci:endpoint-
uri="sip:alice@fabrikam.com;opaque=user:epid:bD8fQE-lq1C1arruZDr3DgAA;gruu">
                    <status>connected</status>
               </endpoint>
          </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: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:entity-capabilities>
               <msci:entity-state>
                     <msci:mediaFiltersRules>
                         <msci:mayModifyOwnFilters>
                            <msci:role>default</msci:role>
                            <msci:value>false</msci:value>
                         </msci:mayModifyOwnFilters>
                         <msci:mayModifyOwnFilters>
                             <msci:role>presenter</msci:role>
                             <msci:value>true</msci:value>
                         </msci:mayModifyOwnFilters>
                         <msci:initialFilters>
                             <msci:role>default</msci:role>
                             <msci:ingressFilter>block</msci:ingressFilter>
                         </msci:initialFilters>
                         <msci:initialFilters>
```

```
                              <msci:role>presenter</msci:role>
                              <msci:ingressFilter>block</msci:ingressFilter>
                         </msci:initialFilters>
                    </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>
                                             dominant-speaker-switched
                                        </msav:video-mode>
                                   </msci:video-parameters>
                              </msci:modal-parameters>
                         </entry>
                         <entry label="panoramic-video">
                              <type>panoramic-video</type>
                              <status>sendrecv</status>
                         </entry>
                    </msci:media>
                    <cis:separator/>
                    <cis:separator/>
                    <msmcu:presentation-mode-capable>true</msmcu:presentation-mode-capable>
                    <msmcu:entry-exit-announcements>
                       <msmcu:modifiable>true</msmcu:modifiable>
                       <msmcu:enabled>false</msmcu:enabled>
                    </msmcu:entry-exit-announcements>
               </msci:entity-state>
          </msci:entity-view>
  <msci:entity-view ci:state="full"
  entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:34D7F8255152F345817A2A6037C579BD"
  >
               <msci:entity-state>
                    <msci:locked>false</msci:locked>
                    <msci:media>
                         <entry label="chat">
                              <type>chat</type>
                         </entry>
                    </msci:media>
               </msci:entity-state>
          </msci:entity-view>
  <msci:entity-view ci:state="full"
  entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:meeting:id:34D7F8255152F345817A2A6037C579
  BD">
  <msci:entity-state application="27877e66-615c-4582-ab88-0cb2ca05d951">
                    <msci:locked>false</msci:locked>
                    <msci:media>
                         <entry label="meeting">
                              <type>meeting</type>
                         </entry>
                    </msci:media>
               </msci:entity-state>
          </msci:entity-view>
     </msci:conference-view>
```

```
</conference-info>.
```

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=7D8A78D161;tag=f8d89ad7d Call-ID: 89fa19a7fc7c4b4d804e56c54f66bde7 CSeq: 5
INVITE Route: <sip:ocs.fabrikam.com:5061;transport=tls;ms-role-rs-from;ms-role-rs-to;ms-ent-
dest;lr;ms-rgs-cid=B1C400;ms-route-sig=bgdDrWFZVoLRuVtQTeFnKJX_Blo5ba1Pe8vY9zXAAA>
Contact: < bob@fabrikam.com;opaque=user:epid:bD8fQE-lq1C1arruZDr3DgAA;gruu>
User-Agent: UCCAPI/2.0.6623.0 OC/2.0.6623.0 (Microsoft Office Communicator)
Supported: ms-dialog-route-set-update Ms-Conversation-ID: Achjsj+npN1Vj6xXR9eQ+niX13+M/Q==
Supported: timer Supported: histinfo Supported: ms-referred-body
Supported: ms-sender
Accept-Language: fr-FR
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="196AFFF9",
crand="50b6d980", cnum="893", targetname="ocs.fabrikam.com",
response="0100000051000000092549dfd3c5fc49"

Content-Type: application/sdp
Content-Length: 1377
v=0
o=- 0 0 IN IP4 10.56.66.199
s=session
c=IN IP4 10.56.66.199
b=CT:99980
t=0 0
m=audio 50000 RTP/SAVP 111 8 0 97 101 a=candidate:5q1qWRkv6z1DFf+07mDwVzlqrMzpYrNkUEeP+0jRSN4
1 alx+3utAJ5f4bRtGxTYmKA UDP 0.850 10.56.66.199 50000
a=candidate:5q1qWRkv6z1DFf+07mDwVzlqrMzpYrNkUEeP+0jRSN4 2 alx+3utAJ5f4bRtGxTYmKA UDP 0.850
10.56.66.199 50016
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:zVTV6wD2sa2x1I7Sj1Z6Hpj2TLDIgJuKN/v0n111|2^31|1:1
a=remote-candidate:XNb7se0dEP6qkMP27aECTauGFohJPE2UDFNAsVfV+zU
a=maxptime:200
a=rtcp:50016
a=rtpmap:111 SIREN/16000
a=fmtp:111 bitrate=16000
a=rtpmap:8 PCMA/8000
a=rtpmap:0 PCMU/8000
a=rtpmap: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=base64:3Vib9sI9U693n3czRc5AZfJOjv1+IilQNNoz9rBeHCsNTAkQsqfMdRBqtRhu
a=candidate:D5zb7FTzC/PeROMqUolbrO8oBQs5Yb6Ycrf4kT/iOJs 1 vAtjb0A2MBDGrGoRuusX3Q UDP 0.860
10.56.66.199 50008
a=candidate:D5zb7FTzC/PeROMqUolbrO8oBQs5Yb6Ycrf4kT/iOJs 2 vAtjb0A2MBDGrGoRuusX3Q UDP 0.860
10.56.66.199 50001
```

```
a=cryptoscale:1 client AES_CM_128_HMAC_SHA1_80
inline:CleJmOuADtS4jc5I/CLM6Vymgf3ZojHVS/3o/2UE|2^31|1:1
a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:5FhbGFSyLth6Y7YBzCN/d8aqf2AYZFiqfqHQhkHv|2^31|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 user notification, as shown in the following sample message.

```
BENOTIFY sip:10.56.66.199:4216;transport=tls;ms-opaque=8a12d4957e;ms-received-cid=B1C400;grid
SIP/2.0Via: SIP/2.0/TLS 10.54.70.62:5061;branch=z9hG4bK1680A441.6AD89D79;branched=FALSE
Authentication-Info: NTLM rspauth="0100000000000000040B3554DD3C5FC49", srand="0852CCEB",
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=02e00da898Content-Length: 1821
From:
<sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=7
F6C0080
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">
     <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-
uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
               </endpoint>
<endpoint entity="{679ACDDB-2171-4176-8414-6ABA41ED881A}" msci:session-type="audio-video"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
                    <joining-method>dialed-in</joining-method>
                    <media id="1">
                         <type>audio</type>
                         <label>main-audio</label>
                         <status>sendrecv</status>
                    </media>
                    <media id="2">
                         <type>video</type>
                         <label>main-video</label>
                         <status>sendrecv</status>
                    </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>
```

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".

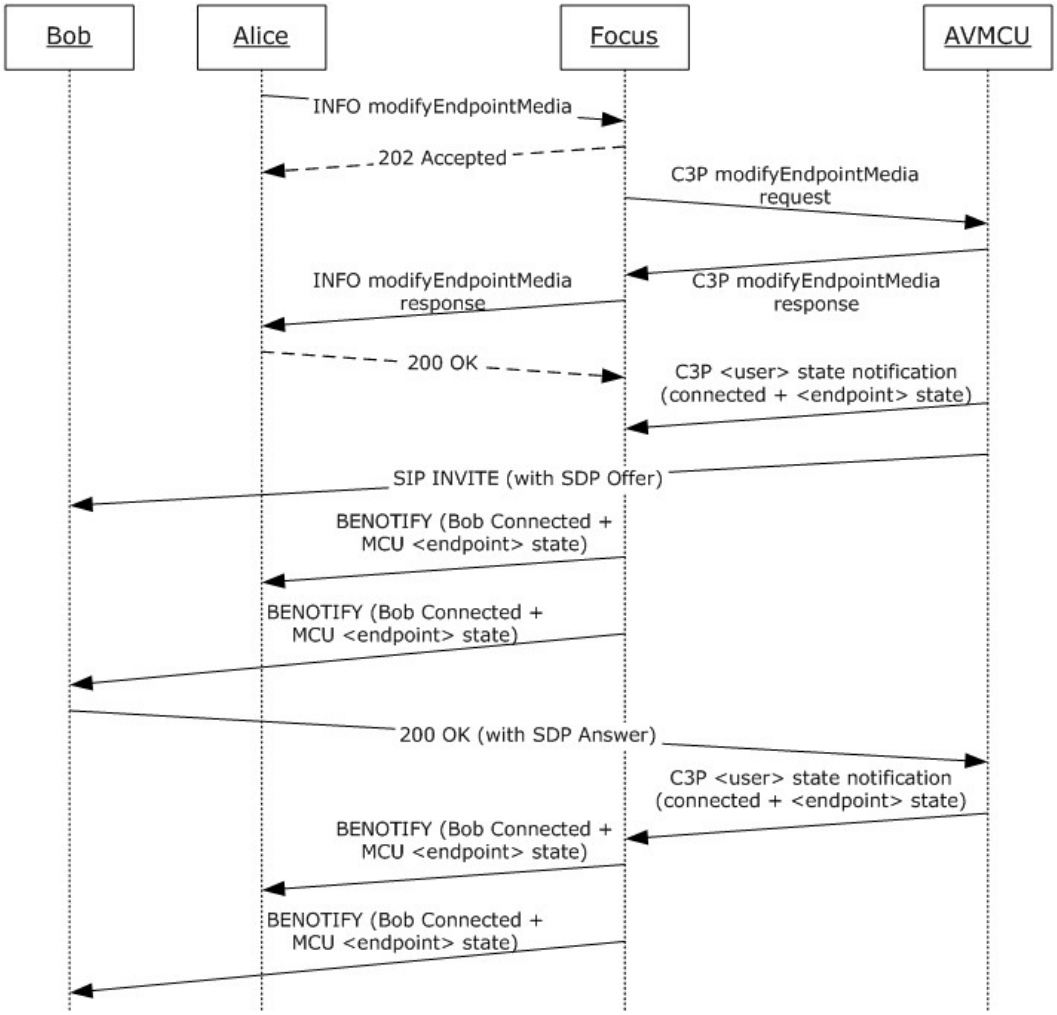

Figure 3: modifyEndpointMedia message flow

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 (5).

```
INFO sip:ocs.fabrikam.com:5061;transport=tls SIP/2.0
Via: SIP/2.0/TLS 10.56.66.199:4216
Max-Forwards: 70
From: <sip:alice@fabrikam.com>;tag=34cbdc7838;epid=02e00da898
To:
<sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C579BD>;tag=D
B3A0080
Call-ID: e5b66a68f7584cf78cd7c8ca762f00ed
CSeq: 2 INFO
User-Agent: UCCAPI/2.0.6623.0 OC/2.0.6623.0 (Microsoft Office
Communicator)
Supported: ms-dialog-route-set-update
Supported: timer
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications
Service", opaque="196AFFF9", crand="c12b73a2", cnum="899",
targetname="ocs.fabrikam.com",
response="01000000000000002f482249d3c5fc49"
Content-Type: application/cccp+xml
Content-Length: 1005
<?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: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:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions">
<mediaKeys
confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:34D7F8255152F345817A2A6037C5
79BD" userEntity="sip:bob@fabrikam.com" endpointEntity="{679ACDDB-2171-4176-8414-
6ABA41ED881A}" mediaId="1"/>
<ci:media xmlns:ci="urn:ietf:params:xml:ns:conference-info" 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="01000000000000040B3554DD3C5FC49",
```

```
srand="0852CCEB", 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=7
F6C0080
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">
     <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-
uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
               </endpoint>
<endpoint entity="{679ACDDB-2171-4176-8414-6ABA41ED881A}" msci:session-type="audio-video"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
                    <joining-method>dialed-in</joining-method>
                    <media id="1">
                         <type>audio</type>
                         <label>main-audio</label>
                         <status>sendrecv</status>
                         <msci:media-ingress-filter>
                         block
                         </msci:media-ingress-filter>
                    </media>
                    <media id="2">
                         <type>video</type>
                         <label>main-video</label>
                         <status>sendrecv</status>
                    </media>
                    <msci:roles>
                         <entry>attendee</entry>
                    </msci:roles>
                    <msci:authMethod>enterprise</msci:authMethod>
                    <msci:accessMethod>internal</msci:accessMethod>
               </endpoint>
          </user>
     </users>
</conference-info>
```

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="0100000000000000040B3554DD3C5FC49",
srand="0852CCEB", 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=7
F6C0080
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">
     <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-
uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
               </endpoint>
<endpoint entity="{679ACDDB-2171-4176-8414-6ABA41ED881A}" msci:session-type="audio-video"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:yIwZ40vZRFqmQ2_rs6oOzQAA;gruu">
                    <status>connected</status>
                    <joining-method>dialed-in</joining-method>
                    <media id="1">
                         <type>audio</type>
                         <label>main-audio</label>
                         <status>recvonly</status>
                         <msci:media-ingress-filter>
                              block
                         </msci:media-ingress-filter>
                    </media>
                    <media id="2">
                         <type>video</type>
                         <label>main-video</label>
                         <status>sendrecv</status>
                    </media>
                    <msci:roles>
                         <entry>attendee</entry>
                    </msci:roles>
                    <msci:authMethod>enterprise</msci:authMethod>
```

```
                    <msci:accessMethod>internal</msci:accessMethod>
                </endpoint>
            </user>
        </users>
    </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.

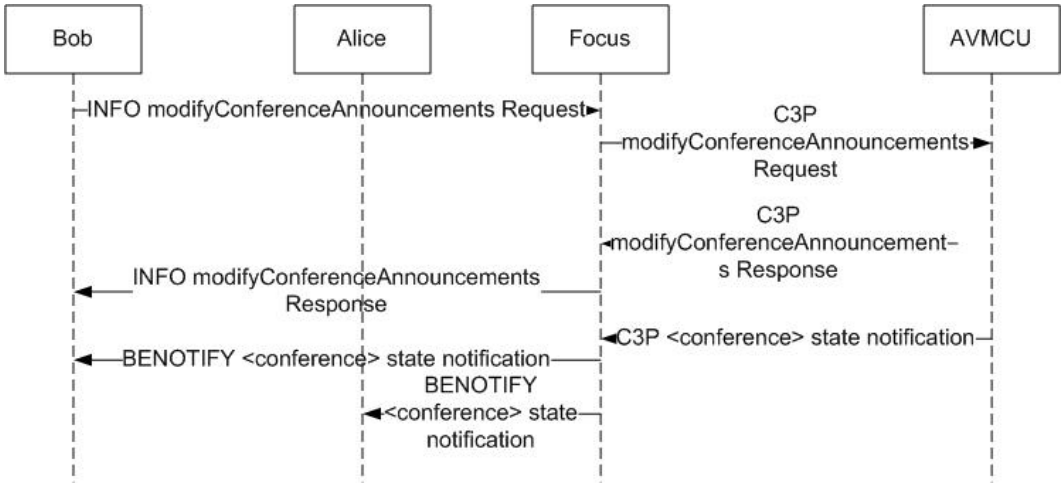

Figure 4: modifyConferenceAnnouncements message flow

A sample **modifyConferenceAnnouncements** request message is as follows:.

```
INFO::Sending Packet - 131.107.106.25:443 (From Local Address: 192.168.1.132:61045) 1986
bytes:
07/23/2010|01:52:06.873 1544:1548 INFO  :: INFO
sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP SIP/2.0
Via: SIP/2.0/TLS 192.168.1.132:61045
Max-Forwards: 70
From: <sip:bob@fabrikam.com>;tag=b03674a9f8;epid=4e91b46850
To:<sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP>;tag=6C3D0080
Call-ID: 24c167a5c0f84f25883d31eeefcc0e13
CSeq: 4 INFO
Route:<sip:sipalt.fabrikam.com:443;transport=tls;opaque=state:Ci.R2dd80300;lr;ms-route-
sig=gtA87hJQ7SENJI6bV7Xu3dXtRx9ZcrjreSazqhB0J82aBI4ycnfADlbgAA>
Route: <sip:ocs.fabrikam.com:5061;transport=tls;ms-fe=
ocs.fabrikam.com;lr;received=10.31.50.6;ms-received-cid=2D616601>
Route: <sip: ocs.fabrikam.com:5061;transport=tls;ms-fe= ocs.fabrikam.com;lr>
Route: <sip:tk5ucdfpl01.exchange.corp.microsoft.com:5061;transport=tls;ms-fe=
ocs.fabrikam.com;opaque=state:T:F;lr>
User-Agent: UCCAPI/4.0.7399.1 OC/4.0.7399.1 (Microsoft Communicator 2010 (Beta Refresh))
Supported: ms-dialog-route-set-update
Supported: timer
```

```
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="69B42DE4", targetname="ocs.fabrikam.com", crand="47d437c6", cnum="1327",
response="254b74c839b682ea0cd74b4e8a1616eeae8f9b6c"
Content-Type: application/cccp+xml
Content-Length: 620

<?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:YAZ75GMP" from="sip:
bob@fabrikam.com.com" requestId="131101736">
<modifyConferenceAnnouncements mscp:mcuUri="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-
video:id:YAZ75GMP" xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions">
<conferenceKeys confEntity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP"/>
<enabled>true</enabled>
</modifyConferenceAnnouncements>
</request>
```

The **modifyConferenceAnnouncements** response is as follows:.

```
07/23/2010|01:52:06.941 1544:1548 INFO  :: Data Received - 131.107.106.25:443 (To Local
Address: 192.168.1.132:61045) 1540 bytes:
07/23/2010|01:52:06.941 1544:1548 INFO  :: INFO sip:157.54.125.148:61045;transport=tls;ms-
opaque=95351d0e04;ms-received-cid=2DD80300;grid SIP/2.0
ms-user-logon-data: RemoteUser
Via: SIP/2.0/TLS
131.107.247.195:443;branch=z9hG4bKC915C06F.7475214D8890BC37;branched=FALSE;ms-internal-
info="beBEF49UJ8CmcekPkXKQsrkErp64KlngWVqMVFdcSYiZ1NIXV0L1tj0AAA"
Max-Forwards: 68
Via: SIP/2.0/TLS 10.31.50.6:5061;branch=z9hG4bK14DBFE2B.0DC70E2A524F0C38;branched=FALSE;ms-
received-port=5061;ms-received-cid=2D616601
Authentication-Info: TLS-DSK qop="auth", opaque="69B42DE4", srand="F37EA865", snum="1335",
rspauth="8b6104ffb947669d553fa21f2649982cacb4b97c", targetname="ocs.fabrikam.com", realm="SIP
Communications Service", version=4
Via: SIP/2.0/TLS
157.54.62.135:53927;branch=z9hG4bK1BE8AA03.A40E13165714BC36;branched=FALSE;ms-received-
port=53927;ms-received-cid=E9E6F100
Content-Length: 388
From: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP>;tag=6C3D0080
To: <sip:bob@fabrikam.com>;tag=b03674a9f8;epid=4e91b46850
Call-ID: 24c167a5c0f84f25883d31eeefcc0e13
CSeq: 191 INFO
Supported: ms-dialog-route-set-update
Content-Type: application/cccp+xml

<response xmlns="urn:ietf:params:xml:ns:cccp" requestId="131101736" C3PVersion="1"
from="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP" to="sip:bob@fabrikam.com"
responder="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:YAZ75GMP" code="success">
<modifyConferenceAnnouncements>
<enabled>true</enabled>
</modifyConferenceAnnouncements>
```

</response>

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.

```
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
xmlns:msas="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
xmlns:msav="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msdata="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP" state="partial"
version="11" static="true">
<msci:conference-view ci:state="full">
<msci:entity-view ci:state="full"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:YAZ75GMP">
<msci:entity-state>
<msci:locked>false</msci:locked>
</msci:entity-state>
</msci:entity-view>
<msci:entity-view ci:state="full"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:applicationsharing:id:YAZ75GMP">
<msci:entity-state>
<msci:locked>false</msci:locked>
<msci:media>
<entry label="applicationsharing">
<type>applicationsharing</type>
</entry>
</msci:media>
<cis:separator/>
<msas:session-ids>
<msas:session-id>1</msas:session-id>
</msas:session-ids>
<cis:separator/>
<msmcu:permissions>
<msmcu:permission-type>
<msmcu:name>AllowUserToScheduleMeetingsWithAppSharing</msmcu:name>
<msmcu:value>True</msmcu:value>
</msmcu:permission-type>
<msmcu:permission-type>
<msmcu:name>AttendeesCanShare</msmcu:name>
<msmcu:value>False</msmcu:value>
</msmcu:permission-type>
</msmcu:permissions>
</msci:entity-state>
</msci:entity-view>
<msci:entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-
video:id:YAZ75GMP">
<msci:entity-capabilities>
<msav:capabilities>
<msav:supports-audio>true</msav:supports-audio>
<msav:supports-video>true</msav:supports-video>
</msav:capabilities>
</msci:entity-capabilities>
<msci:entity-state>
<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:egressFilter>unblock</msci:egressFilter>
</msci:initialFilters>
</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>dominant-speaker-switched</msav:video-mode>
</msci:video-parameters>
</msci:modal-parameters>
</entry>
<entry label="panoramic-video">
<type>panoramic-video</type>
<status>sendrecv</status>
</entry>
</msci:media>
<cis:separator/>
<cis:separator/>
<msmcu:presentation-mode-capable>true</msmcu:presentation-mode-capable>
<msmcu:entry-exit-announcements>
<msmcu:modifiable>true</msmcu:modifiable>
<msmcu:enabled>true</msmcu:enabled>
</msmcu:entry-exit-announcements>
</msci:entity-state>
</msci:entity-view>
<msci:entity-view ci:state="full"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:chat:id:YAZ75GMP">
<msci:entity-state>
<msci:locked>false</msci:locked>
<msci:media>
<entry label="chat">
<type>chat</type>
</entry>
</msci:media>
</msci:entity-state>
</msci:entity-view>
<msci:entity-view ci:state="full" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:data-
conf:id:YAZ75GMP">
<msci:entity-state application="101511fa-90b5-4fa2-9fbf-fa99afef9e9a">
<msci:locked>false</msci:locked>
<msci:media>
<entry label="data-conf">
<type>data-conf</type>
</entry>
</msci:media>
<cis:separator/>
<cis:separator/>
<msdata:data-mcu-state>
<msdata:hasContent>false</msdata:hasContent>
<msdata:hasContentInMeeting>false</msdata:hasContentInMeeting>
<msdata:hasPresentedContent>false</msdata:hasPresentedContent>
</msdata:data-mcu-state>
<msmcu:permission-options>
```

```
<msmcu:permission-option>
<msmcu:name>PptAnnotationsAllowedPresenter</msmcu:name>
<msmcu:value>true</msmcu:value>
<msmcu:mutable>true</msmcu:mutable>
</msmcu:permission-option>
<msmcu:permission-option>
<msmcu:name>PptAnnotationsAllowedAttendee</msmcu:name>
<msmcu:value>true</msmcu:value>
<msmcu:mutable>true</msmcu:mutable>
</msmcu:permission-option>
<msmcu:permission-option>
<msmcu:name>AsynchronousBrowsingAllowedPresenter</msmcu:name>
<msmcu:value>true</msmcu:value>
<msmcu:mutable>true</msmcu:mutable>
</msmcu:permission-option>
<msmcu:permission-option>
<msmcu:name>AsynchronousBrowsingAllowedAttendee</msmcu:name>
<msmcu:value>false</msmcu:value>
<msmcu:mutable>true</msmcu:mutable>
</msmcu:permission-option>
</msmcu:permission-options>
<msmcu:permissions>
<msmcu:permission-type>
<msmcu:name>EnableDataCollaboration</msmcu:name>
<msmcu:value>true</msmcu:value>
</msmcu:permission-type>
<msmcu:permission-type>
<msmcu:name>AllowAnnotations</msmcu:name>
<msmcu:value>true</msmcu:value>
</msmcu:permission-type>
<msmcu:permission-type>
<msmcu:name>AllowExternalUsersToSaveContent</msmcu:name>
<msmcu:value>true</msmcu:value>
</msmcu:permission-type>
<msmcu:permission-type>
<msmcu:name>AllowPolls</msmcu:name>
<msmcu:value>true</msmcu:value>
</msmcu:permission-type>
<msmcu:permission-type>
<msmcu:name>EnableFileTransfer</msmcu:name>
<msmcu:value>true</msmcu:value>
</msmcu:permission-type>
</msmcu:permissions>
</msci:entity-state>
</msci:entity-view>
</msci:conference-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 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.

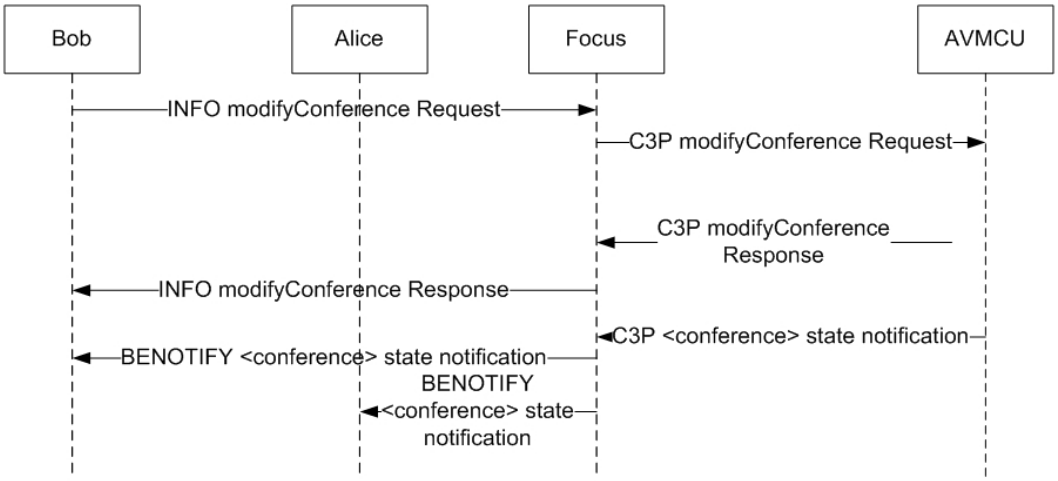

Figure 5: modifyConference message flow

An example **modifyConference** request message 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:
08/10/2010|19:05:32.231 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=dccb9664e6;epid=4e91b46850
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL>;tag=A6260080
Call-ID: 09c2c6a6f7c3406e9f5a69f1b688260d
CSeq: 6 INFO
Route: <sip:ocs.fabrikam.com:443;transport=tls;opaque=state:Ci.R252de00;lr;ms-route-
sig=ebWOPBtTv8VmQPKR96i-y9T-nrXyP6KnHAfFjrrX2WOcYNoSXHh6fZPAAA>
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;opaque=state:T:F;lr>
User-Agent: UCCAPI/4.0.7400.0 OC/4.0.7400.0 (Microsoft Communicator 2010 (Beta Refresh))
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="e925a5fec2ac140d0bf7825584944ae1611b9512"
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">
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" state="partial">
```

```
<msci:conference-view
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions" ci:state="partial"
xmlns:ci="urn:ietf:params:xml:ns:conference-info">
<msci:entity-view ci:state="partial" entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-
video:id:7Y7DGPBL">
      <msci: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>block</msci:ingressFilter>
                  </msci:initialFilters>
                  <msci:initialFilters>
                        <msci:role>default</msci:role>
                        <msci:ingressFilter>block</msci:ingressFilter>
                  </msci:initialFilters>
            </msci:mediaFiltersRules>
       </msci:entity-state>
</msci:entity-view>
</msci:conference-view>
</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) 1693 bytes:
08/10/2010|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=z9hG4bK4E0FEA28.3ABF8C10F2D5614E;branched=FALSE;ms-internal-
info="aakIJgqfpRkMR_gFmEX3Ylpn99KG7U0YsuRdw-n1VDi1cQjL86Q9YfOAAA"
Max-Forwards: 68
Via: SIP/2.0/TLS 10.31.50.6:5061;branch=z9hG4bK1839F509.8C995EBE3319414D;branched=FALSE;ms-
received-port=5061;ms-received-cid=2513B00
Authentication-Info: TLS-DSK qop="auth", opaque="54574323", srand="28B50009", snum="2412",
rspauth="b00dea42e6be9750a6679b9e1f8289767ff35dd5", 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
To: <sip:bob@fabrikam.com>;tag=dccb9664e6;epid=4e91b46850
```

```
Call-ID: 09c2c6a6f7c3406e9f5a69f1b688260d
CSeq: 117 INFO
Supported: ms-dialog-route-set-update
Content-Type: application/cccp+xml

<response xmlns="urn:ietf:params:xml:ns:cccp" xmlns:ci="urn:ietf:params:xml:ns:conference-
info" requestId="164518168" C3PVersion="1"
from="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" to="sip:bob@fabrikam.com"
responder="sip:bob@fabrikam.com;gruu;opaque=app:conf:audio-video:id:7Y7DGPBL" code="success">
<modifyConference>
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" state="partial"/>
</modifyConference>

</response>
```

08/10/2010|19:05:32.257 1A0C:1A10 INFO :: End of Data Received - 157.54.27.30:80 (To Local Address: 10.80.20.229:54075) 1693 bytes

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:

```
<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
xmlns:msas="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
xmlns:msav="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msdata="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:7Y7DGPBL" state="partial"
version="19" static="false">
<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">
<msci:entity-capabilities>
<msav:capabilities>
<msav:supports-audio>true</msav:supports-audio>
<msav:supports-video>true</msav:supports-video>
</msav:capabilities>
</msci:entity-capabilities>
<msci:entity-state>
<msci:mediaFiltersRules>
<msci:mayModifyOwnFilters>
<msci:role>default</msci:role>
<msci:value>false</msci:value>
</msci:mayModifyOwnFilters>
<msci:mayModifyOwnFilters>
<msci:role>presenter</msci:role>
<msci:value>true</msci:value>
</msci:mayModifyOwnFilters>
<msci:initialFilters>
<msci:role>default</msci:role>
<msci:ingressFilter>block</msci:ingressFilter>
```

```
</msci:initialFilters>
<msci:initialFilters>
<msci:role>presenter</msci:role>
<msci:ingressFilter>block</msci:ingressFilter>
</msci:initialFilters>
</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>dominant-speaker-switched</msav:video-mode>
</msci:video-parameters>
</msci:modal-parameters>
</entry>
<entry label="panoramic-video">
<type>panoramic-video</type>
<status>sendrecv</status>
</entry>
</msci:media>
<cis:separator/>
<cis:separator/>
<msmcu:presentation-mode-capable>true</msmcu:presentation-mode-capable>
<msmcu:entry-exit-announcements>
<msmcu:modifiable>true</msmcu:modifiable>
<msmcu:enabled>false</msmcu:enabled>
</msmcu:entry-exit-announcements>
</msci:entity-state>
</msci:entity-view>
</msci:conference-view>
</conference-info>
```

5 Security

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: application/conference-info+xml schema reference

6.1 conference-info Namespace (urn:ietf:params:xml:ns:conference-info)

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

```
<?xml version="`1.0 encoding="utf-9"?>

<xs:schema

targetNamespace="urn:ietf:params:xml:ns:conference-info"

xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"

xmlns:xs="http://www.w3.org/2001/XMLSchema"

xmlns="urn:ietf:params:xml:ns:conference-info"

xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"

xmlns:ms="urn:microsoft-cpp-xml-serializer"

elementFormDefault="qualified"

attributeFormDefault="unqualified">

<!--

This imports the standard separator

-->

<xs:import namespace="urn:ietf:params:xml:ns:conference-info-separator" schemaLocation="ms-ci-separator.xsd"/>

<!--

This import brings in the MS Conference Package extensions

-->

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions" 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"/>

<!--

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 ref="msci:conference-media-states" minOccurs="0"/>

<xs:element ref="msci:conference-view" minOccurs="0" ms:propertyName="ConferenceView"/>

<xs:sequence minOccurs="0" ms:propertyName="Extension1"
ms:className="CC3PConfTypeExtension1">

<xs:element ref="cis:separator"/>

<xs:element ref="msci:trusted-entities" minOccurs="0"
ms:propertyName="TrustedEntityCollection"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>

</xs:sequence>

</xs:sequence>

</xs:sequence>
```

```
<xs:attribute ref="msci:conference-id" ms:propertyName="ConferenceId"/>

<xs:attribute name="entity" type="xs:anyURI" use="required"
ms:propertyName="ConferenceUri"/>

<xs:attribute name="state" type="state-type" use="optional" default="full"
ms:propertyName="ConferenceState"/>

<xs:attribute name="version" type="xs:unsignedInt" use="optional"
ms:propertyName="RosterVersion"/>

<xs:attribute name="static" type="xs:boolean" use="optional" ms:propertyName="Static"/>

<xs:attribute name="deactivation-secs" type="xs:int" use="optional"
ms:propertyName="DeactivationSecs"/>

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

</xs:complexType>

<!--

STATE TYPE

-->

<xs:simpleType name="state-type">

<xs:restriction base="xs:string">

<xs:enumeration value="full"/>

<xs:enumeration value="partial"/>

<xs:enumeration value="deleted"/>

</xs:restriction>

</xs:simpleType>

<!--

CONFERENCE DESCRIPTION TYPE

-->

<xs:complexType name="conference-description-type"
ms:className="CC3PConferenceDescription">

<xs:sequence>

<xs:element name="display-text" type="xs:string" minOccurs="0"
ms:propertyName="Description"/>

<xs:element name="subject" type="xs:string" minOccurs="0" ms:propertyName="Subject"/>

<xs:element name="free-text" type="xs:string" minOccurs="0"/>

<xs:element name="keywords" type="keywords-type" minOccurs="0"/>
```

```
<xs:element name="conf-uris" type="uris-type" minOccurs="0" ms:propertyName="ConfUris"/>

<xs:element name="service-uris" type="uris-type" minOccurs="0"
ms:propertyName="ServiceUris"/>

<xs:element name="maximum-user-count" type="xs:unsignedInt" minOccurs="0"
ms:propertyName="MaximumUserCount"/>

<xs:element name="available-media" type="conference-media-type" minOccurs="0"/>

<xs:element ref="msci:disclaimer" minOccurs="0" ms:propertyName="DisclaimerBody"/>

<xs:element ref="msci:organizer" minOccurs="0" ms:propertyName="Organizer"/>

<xs:element ref="msci:conference-id" minOccurs="0" ms:propertyName="ConferenceId"/>

<xs:element ref="msci:conference-key" minOccurs="0" ms:propertyName="ConferenceKey"/>

<xs:element ref="msci:last-update" minOccurs="0" ms:propertyName="LastUpdate"/>

<xs:element ref="msci:last-activate" minOccurs="0" ms:propertyName="LastActivate"/>

<xs:element ref="msci:is-active" minOccurs="0" ms:propertyName="IsActive"/>

<xs:element ref="msci:expiry-time" minOccurs="0" ms:propertyName="ExpiryTime"/>

<xs:element ref="msci:admission-policy" minOccurs="0" ms:propertyName="AdmissionPolicy"/>

<xs:element ref="msci:organizer-roaming-data" minOccurs="0"
ms:propertyName="OrganizerData"/>

<xs:element ref="msci:notification-data" minOccurs="0" ms:propertyName="NotificationData"/>

<!-- TODO: Remove the next element -->

<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 minOccurs="0" ms:propertyName="Extension2"
ms:className="CC3PConfDescriptionExtension2">

<xs:element ref="cis:separator"/>

<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 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:pstn-lobby-bypass-allowed" minOccurs="0"
ms:propertyName="PstnLobbyBypassAllowed"/>

<xs:element ref="msci:disclaimer-title" minOccurs="0" ms:propertyName="DisclaimerTitle"/>

<xs:element ref="msci:recording-allowed" minOccurs="0"
ms:propertyName="RecordingAllowed"/>

<xs:element ref="msci:externaluser-recording-allowed" minOccurs="0"
ms:propertyName="ExternalUserRecordingAllowed"/>

<xs:element ref="msci:server-mode" minOccurs="0" ms:propertyName="ServerMode"/>

<xs:element ref="msci:recording-notification" minOccurs="0"
ms:propertyName="RecordingNotification"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>

</xs:sequence>

</xs:sequence>

</xs:sequence>

</xs:sequence>

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

</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:element name="uris" type="uris-type" minOccurs="0"/>

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

</xs:sequence>
```

```
<xs:anyAttribute 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"/>
<xs:element name="active" type="xs:boolean" minOccurs="0"/>
<xs:element name="locked" type="xs:boolean" minOccurs="0" ms:propertyName="Locked"/>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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:anyAttribute namespace="##other" processContents="lax"/>
</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:sequence>
<xs:attribute name="label" type="xs:string" use="required"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
URIs TYPE
-->
<xs:complexType name="uris-type" ms:className="CC3PUrisType">
<xs:sequence>
<xs:element name="entry" type="uri-type" maxOccurs="unbounded" ms:propertyName="Entry"/>
</xs:sequence>
<xs:attribute name="state" type="state-type" use="optional" default="full"
ms:propertyName="State"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
URI TYPE
-->
<xs:complexType name="uri-type" ms:className="CC3PUriType">
<xs:sequence>
<xs:element name="uri" type="xs:anyURI" ms:propertyName="Uri"/>
<xs:element name="display-text" type="xs:string" minOccurs="0"
ms:propertyName="DisplayText"/>
<xs:element name="purpose" type="xs:string" minOccurs="0" ms:propertyName="Purpose"/>
<xs:element name="modified" type="execution-type" minOccurs="0"/>
```

```
<xs:element ref="msci:hash-code" minOccurs="0"/>

<xs:sequence minOccurs="0" ms:propertyName="Extension1"
ms:className="CC3PUriTypeExtension1">

<xs:element ref="cis:separator"/>

<xs:element ref="msci:encrypted-uri" minOccurs="0" ms:propertyName="EncryptedUri"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>

</xs:sequence>

</xs:sequence>

</xs:sequence>

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

</xs:complexType>

<!--

KEWORDS TYPE

-->

<xs:simpleType name="keywords-type">

<xs:list itemType="xs:string"/>

</xs:simpleType>

<!--

USERS TYPE

-->

<xs:complexType name="users-type" ms:className="CC3PUserCollection">

<xs:sequence>

<xs:element name="user" type="user-type" minOccurs="0" maxOccurs="unbounded"
ms:propertyName="Users"/>

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

</xs:sequence>

<xs:attribute name="state" type="state-type"

use="optional" default="full" ms:propertyName="UsersState"/>

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

```
</xs:complexType>

<!--

USER TYPE

-->

<xs:complexType name="user-type" ms:className="CC3PUser">

<xs:sequence>

<xs:element name="display-text" type="xs:string" minOccurs="0"
ms:propertyName="DisplayText"/>

<xs:element name="associated-aors" type="uris-type" minOccurs="0"/>

<xs:element name="roles" type="user-roles-type" minOccurs="0"
ms:propertyName="RoleCollection"/>

<xs:element name="languages" type="user-languages-type" minOccurs="0"/>

<xs:element name="cascaded-focus" type="xs:anyURI" minOccurs="0"/>

<xs:element name="endpoint" type="endpoint-type" minOccurs="0" maxOccurs="unbounded"
ms:propertyName="EndpointCollection"/>

<xs:element ref="msci:designated-presenter" minOccurs="0"/>

<xs:sequence minOccurs="0" ms:propertyName="Extension1"
ms:className="CC3PUserExtension1">

<xs:element ref="cis:separator"/>

<xs:element ref="msci:trusted" minOccurs="0"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>

</xs:sequence>

</xs:sequence>

</xs:sequence>

<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: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"/>

<xs:element name="referred" type="execution-type" minOccurs="0"/>

<xs:element name="status" type="endpoint-status-type" minOccurs="0"
ms:propertyName="Status"/>

<xs:element name="joining-method" type="joining-type" minOccurs="0"
ms:propertyName="JoiningMethod"/>

<xs:element name="joining-info" type="execution-type" minOccurs="0"/>

<xs:element name="disconnection-method" type="disconnection-type" minOccurs="0"/>

<xs:element name="disconnection-info" type="execution-type" minOccurs="0"/>

<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"/>

<xs:element ref="msci:authMethod" 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"/>

<xs:element ref="msci:pstnRole" minOccurs="0"/>

<xs:element ref="msci:pstnLeaderPasscode" minOccurs="0"/>

<xs:element ref="msci:endpoint-capabilities" minOccurs="0"/>

<xs:element ref="msci:is-robot" minOccurs="0"/>

<xs:element ref="msci:current-sidebar" minOccurs="0"/>

<xs:sequence minOccurs="0" ms:propertyName="Extension1"
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="InConferencingServicesCollection"/>

<xs:element ref="msci:languages" minOccurs="0" ms:propertyName="Languages"/>

<xs:element ref="msci:is-pstn-endpoint" minOccurs="0" ms:propertyName="IsPstnEndpoint"/>

<xs:sequence minOccurs="0" ms:propertyName="Extension2"
ms:className="CC3PEndpointExtension2">

<xs:element ref="cis:separator"/>

<xs:element ref="msci: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>

</xs:sequence>

</xs:sequence>

<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="msci:session-type" use="optional"/>

<xs:attribute ref="msci:epid" use="optional"/>

<xs:attribute ref="msci:sip-instance" use="optional"/>

<xs:attribute ref="msci:endpoint-uri" use="optional"/>

<xs:attribute ref="msci:refer-to-uri" use="optional"/>

<xs:attribute ref="msci:asserted-identity" use="optional"/>

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

</xs:complexType>

<!--

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: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-id-type" ms:propertyName="SIPDialogId"/>

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

<!--

TODO: syounis: Filed a bug #135103 - 'C3PWrapperGenerator tool barfs when a complex-type has xs:sequence embedded inside the xs:choice'

The bug needs to be fixed before we ship W13 or we won't have extensibility story for call-type for w14

<xs:element ref="msci:session-description"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

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

</xs:sequence>

-->

</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>

<xs:element ref="msci:media-egress-filter" minOccurs="0"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:element ref="msci:to-mixer" minOccurs="0"/>

<xs:element ref="msci:from-mixer" minOccurs="0"/>

<xs:element ref="msci:media-state" minOccurs="0"/>

<xs:element ref="msci:session-id" minOccurs="0"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>
```

```
<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>
<xs:element ref="msci:conf-media-filter" minOccurs="0"/>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
</xs:sequence>
</xs:sequence>
<xs:attribute name="id" type="xs:string" use="required" ms:propertyName="Id" />
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
MEDIA STATUS TYPE
-->
<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>
<!--
```

```
SIDEBARS BY VAL TYPE
-->
<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:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
</xs:schema>
```

6.2 conference-info-extensions Namespace (http://schemas.microsoft.com/rtc/2005/08/confinfoextensions)

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

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
targetNamespace="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msacp="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
xmlns:msav="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
xmlns:msas="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
xmlns:msdata="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
xmlns:msim="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
```

```
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"

xmlns:msendpt="http://schemas.microsoft.com/rtc/2008/12/endpointinfoextensions">

<!-- Bring in standard conferencing package separator -->

<xs:import namespace="urn:ietf:params:xml:ns:conference-info-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

-->

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
schemaLocation="acpmcusettings.xsd"/>

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
schemaLocation="avmcusettings.xsd"/>

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
schemaLocation="datamcusettings.xsd"/>

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
schemaLocation="immcusettings.xsd"/>

<xs:import namespace="http://schemas.microsoft.com/rtc/2008/12/endpointinfoextensions"
schemaLocation="endptsettings.xsd"/>

<xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
schemaLocation="asmcusettings.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-key-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: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="externaluser-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"/>
<!--
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 minOccurs="0">

<xs:element ref="cis:separator"/>

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

</xs:sequence>

</xs:sequence>

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

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

</xs:sequence>

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

</xs:complexType>

<!--

ORGANIZER ROAMING DATA TYPE

-->

<xs:complexType name="organizer-roaming-data-type"
ms:className="CC3POrganizerDataType">

<xs:sequence>

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

```
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
NOTIFICATION DATA TYPE
-->
<xs:complexType name="notification-data-type" ms:className="CC3PNotificationDataType">
<xs:sequence>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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:enumeration value="anonymous"/>
</xs:restriction>
</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="CC3PPstnAccessType">

<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"
ms:propertyName="AccessNumbers"/>

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

</xs:sequence>

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

</xs:complexType>

<xs:complexType name="pstn-access-numbers-type"
ms:className="CC3PPstnAccessNumbersType">

<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:element ref="msci2:default-region" minOccurs="0" ms:propertyName="DefaultRegion"/>

<xs:sequence minOccurs="0">

<xs:element ref="cis:separator"/>

<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>

</xs:sequence>

</xs:sequence>

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

</xs:complexType>

<xs:complexType name="pstn-access-number-region-type"
ms:className="CC3PPstnAccessNumberRegionType">

<xs:sequence>

<xs:element name="access-number" type="tns:pstn-access-number-type" minOccurs="0"
maxOccurs="unbounded" ms:propertyName="AccessNumberList"/>
```

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

</xs:sequence>

<xs:attribute name="name" type="xs:string" use="required" ms:propertyName="Name"/>

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

</xs:complexType>

<xs:complexType name="pstn-access-number-type"
ms:className="CC3PPstnAccessNumberType">

<xs:sequence>

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

<xs:element name="number" type="xs:string" ms:propertyName="Number"/>

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

</xs:sequence>

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

</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"/>

<xs:element name="entity-settings" type="entity-settings-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"/>

<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

TODO: This, all references, needs to be removed

-->

<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

TODO: This, all references, needs to be removed

-->

<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
TODO: Remove this and all references to it
-->
<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>
<!--
ENTITY CAPABILITIES TYPE
-->
<xs:complexType name="entity-capabilities-type" ms:className="CC3PEntityCapabilitiesType">
<xs:sequence>
<xs:choice minOccurs="0">
<xs:element ref="msacp:capabilities"/>
<xs:element ref="msav:capabilities"/>
<!-- <xs:element ref="msdata:capabilities"/> -->
<!-- <xs:element ref="msim:capabilities"/> -->
</xs:choice>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
ENTITY POLICY TYPE
-->
<xs:complexType name="entity-policy-type">
<xs:sequence>
```

```
<xs:element name="guid" type="xs:anyURI" minOccurs="0"/>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute ref="msci2:policyAssignment" use="optional"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
ENTITY SETTINGS TYPE
-->
<xs:complexType name="entity-settings-type" ms:className="CC3PEntitySettingsType">
<xs:sequence>
<xs:element name="mediaFiltersRules" type="tns:media-filters-rules-type" minOccurs="0" />
<xs:element name="media" type="ci:conference-media-type" minOccurs="0" />
<xs:choice minOccurs="0">
<xs:element ref="msacp:settings" ms:propertyName="AcpMcuSettings"/>
<xs:element ref="msav:settings" ms:propertyName="AvMcuSettings"/>
<xs:element ref="msdata:settings" ms:propertyName="DataMcuSettings"/>
<xs:element ref="msim:settings" ms:propertyName="ImMcuSettings"/>
</xs:choice>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
ENTITY STATE TYPE
-->
<xs:complexType name="entity-state-type" ms:className="CC3PEntityStateType">
```

```
<xs:sequence>
<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="msdata:data-mcu-state" minOccurs="0" />
<xs:element ref="msmcu:permission-options" minOccurs="0"/>
<xs:element ref="msmcu:permissions" minOccurs="0"/>
<xs:element ref="msmcu:presentation-mode-capable" minOccurs="0"/>
<xs:element ref="msmcu:entry-exit-announcements" minOccurs="0"/>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
</xs:sequence>
</xs:sequence>
<xs:attribute name="application" type="xs:string"/>
```

```
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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:element ref="msav:shared-data"/> -->
<!-- <xs:element ref="msdata:shared-data"/> -->
<!-- <xs:element ref="msim:shared-data"/> -->
</xs:choice>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
```

```
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<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>
<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"/>
<xs:element name="egressFilter" type="tns:media-filter-type" minOccurs="0"/>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<!--
MS Authentication Type - LCS Extension
-->
<xs:element name="authMethod" type="tns:auth-method-type" ms:ignore="true"/>
<xs:simpleType name="auth-method-type">
<xs:restriction base="xs:string">
<xs:enumeration value="enterprise"/>
<xs:enumeration value="anonymous"/>
<xs:enumeration value="federated"/>
</xs:restriction>
</xs:simpleType>
```

```
<!--
MS Role Type - LCS Extension
-->
<xs:simpleType name="ms-role-type">
<xs:restriction base="xs:string">
<xs:enumeration value="attendee"/>
<xs:enumeration value="presenter"/>
</xs:restriction>
</xs:simpleType>
<xs:element name="pstnRole" type="tns:ms-role-type" ms:ignore="true"/>
<!--
USER ACCESS TYPE
-->
<xs:element name="accessMethod" type="tns:access-method-type" ms:ignore="true"/>
<xs:simpleType name="access-method-type">
<xs:restriction base="xs:string">
<xs:enumeration value="external"/>
<xs:enumeration value="internal"/>
</xs:restriction>
</xs:simpleType>
<!--
CLIENT INFO TYPE
-->
<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"/>
<xs:element name="thread-id" type="xs:string" minOccurs="0"/>
<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" />
<xs:element ref="msci2:lobby-capable" minOccurs="0" ms:propertyName="LobbyCapable"/>
<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: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"/>
<xs:element name="display-name" type="xs:string" minOccurs="0"/>
<xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</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:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
```

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

</xs:sequence>

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

</xs:complexType>

<xs:complexType name="in-conferencing-services-entry-type"
ms:className="CC3PInConferencingServicesEntry">

<xs:sequence>

<xs:element name="active" type="xs:boolean" minOccurs="1"/>

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

</xs:sequence>

<xs:attribute name="type" type="tns:in-conferencing-service-types-ex"/>

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

</xs:complexType>

<xs:simpleType name="in-conferencing-service-types-ex">

<xs:union memberTypes="tns:in-conferencing-service-types xs:string"/>

</xs:simpleType>

<xs:simpleType name="in-conferencing-service-types">

<xs:restriction base="xs:string">

<xs:enumeration value="personalVirtualAssistant"/>

<xs:enumeration value="entryExitAnnouncements"/>

</xs:restriction>
```

```
</xs:simpleType>
<!--
CLIENT RECORDING
-->
<xs:complexType name="client-recording-type" ms:className="CC3PClientRecording">
<xs:sequence>
<xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded" processContents="lax"/>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<xs:element name="client-recording" type="tns:client-recording-type" ms:ignore="true"/>
<!--
MEDIA FILTER
-->
<xs:element name="media-filter" type="tns:media-filter-type" ms:ignore="true"/>
<xs:element name="media-ingress-filter" type="tns:media-filter-type" ms:ignore="true"/>
<xs:element name="media-egress-filter" type="tns:media-filter-type" ms:ignore="true"/>
<xs:simpleType name="media-filter-type">
<xs:restriction base="xs:string">
<xs:enumeration value="block" />
<xs:enumeration value="unblock" />
</xs:restriction>
</xs:simpleType>
<xs:element name="conf-media-filter" type="tns:conf-media-filter-type" ms:ignore="true"/>
<xs:complexType name="conf-media-filter-type">
<xs:attribute name="filter" type="tns:media-filter-type" use="required"
ms:propertyName="Filter"/>
<xs:attribute name="duration" type="xs:int" use="optional" ms:propertyName="Duration"/>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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>
<xs:choice minOccurs="0">
<xs:element ref="msacp:endpoint-capabilities"/>
<xs:element ref="msav:endpoint-capabilities"/>
<xs:element ref="msdata:endpoint-capabilities"/>
<xs:element ref="msim:endpoint-capabilities"/>
<xs:element ref="msendpt:endpoint-capabilities"/>
</xs:choice>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator" />
<xs:choice minOccurs="0">
<xs:element ref="msmcu:session-capabilities"/>
</xs:choice>
<xs:sequence minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
```

```
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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 minOccurs="0">
<xs:element ref="cis:separator"/>
<xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
</xs:sequence>
</xs:sequence>
<xs:anyAttribute namespace="##other" processContents="lax"/>
</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: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"/>

<!-- 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"/>
<!-- 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 -->
<xs:attribute name="asserted-identity" type="xs:string"/>
<!--
SESSION DESCRIPTION TYPE: carries SDP for now just a string
-->
<xs:element name="session-description" type="tns:session-description-type" ms:ignore="true"/>
<xs:complexType name="session-description-type">
<xs:sequence>
<xs:element name="sdp-string" type="xs:string" minOccurs="0" />
```

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

</xs:sequence>

</xs:complexType>

<!--

MEDIA STATE TYPE: This provides the state of the media

-->

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

<xs:simpleType name="media-state-type-ex">

<xs:union memberTypes="tns:media-state-type xs:string"/>

</xs:simpleType>

<xs:simpleType name="media-state-type">

<xs:restriction base="xs:string">

<xs:enumeration value="pending"/>

<xs:enumeration value="disconnected"/>

<xs:enumeration value="offering"/>

<xs:enumeration value="trying"/>

<xs:enumeration value="proceeding"/>

<xs:enumeration value="ringing"/>

<xs:enumeration value="joining"/>

<xs:enumeration value="connected"/>

<xs:enumeration value="hold"/>

<xs:enumeration value="forwarding"/>

<xs:enumeration value="transferring"/>

</xs:restriction>

</xs:simpleType>

</xs:schema>


   ">
```

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

Following is the schema for the avconfinfoextensions namespace.

```
<?xml version="1.0" encoding="utf-8"?>

<xs:schema

targetNamespace="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"

elementFormDefault="qualified"

attributeFormDefault="unqualified"

version="1.0"

xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"

xmlns:ms="urn:microsoft-cpp-xml-serializer"

xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"

xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"

xmlns:xs="http://www.w3.org/2001/XMLSchema">

<!--

This import brings in the standard Conference Package Standard Separator definitions

-->

<xs:import namespace="urn:ietf:params:xml:ns:conference-info-separator" schemaLocation="ms-ci-separator.xsd" />

<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"/>
<xs:element name="video" type="tns:video-settings-type" minOccurs="0"/>
<xs:element ref="msmcu:entry-exit-announcements" minOccurs="0"/>
<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:sequence>
<xs:attribute name="empty" type="xs:boolean" use="optional"/>
</xs:complexType>
```

```
<xs:complexType name="dominant-speaker-source-type">
<xs:sequence>
<xs:element name="entry" type="xs:anyURI" minOccurs="0" />
</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: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>
<!-- TODO: remove -->
<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:sequence>

</xs:complexType>

<!--

TODO: Remove this element and all references to this element

-->

<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"/>

<!--

Media Routing Type

Controls Parameters type

Route Parameters type

Wire Parameters type

-->

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

</xs:sequence>

</xs:complexType>

<xs:complexType name="route-parameters-type">

<xs:sequence>

<xs:element name="wire" type="tns:wire-parameters-type" minOccurs="0"
maxOccurs="unbounded"/>

<xs:element name="unwire" type="tns:wire-parameters-type" minOccurs="0"
maxOccurs="unbounded"/>

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

</xs:sequence>

</xs:complexType>

<xs:complexType name="wire-parameters-type">

<xs:sequence>

<xs:element name="filter" type="tns:filter-type" minOccurs="0" maxOccurs="1"/>

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

</xs:sequence>

<xs:attribute name="userEntity" type="xs:anyURI" use="required" ms:propertyName="UserUri"/>

<xs:attribute name="endpointEntity" type="xs:string" use="required"
ms:propertyName="EndpointUri"/>

<xs:attribute name="mediaId" type="xs:string" use="required"/>

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

</xs:complexType>

<xs:simpleType name="filter-type">
```

```
<xs:restriction base="xs:string">
<xs:enumeration value="DTMF"/>
</xs:restriction>
</xs:simpleType>
</xs:schema>
```

7 Appendix B: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs:

- 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

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product 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.

8 Change Tracking

No table of changes is available. The document is either new or has had no changes since its last release.

9 Index

N

O

P

R

S

T

U

V