[MS-CONFBAS]:
Centralized Conference Control Protocol:
Basic Architecture and Signaling Specification

Revision Summary

Date	Revision History	Revision Class	Comments
04/04/2008	0.1		Initial Availability
04/25/2008	0.2	Editorial	Revised and edited the technical content
06/27/2008	1.0	Major	Revised and edited the technical content
08/15/2008	1.01	Editorial	Revised and edited the technical content
12/12/2008	1.01	Editorial	Revised and edited the technical content
02/13/2009	2.01	Major	Revised and edited the technical content
03/13/2009	2.02	Editorial	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	Minor	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.

Table of Contents

1 Introduction

This document specifies the Centralized Conference Control Protocol (C3P), which can be used to activate, modify, deactivate, and control conferences. This protocol specifies extensions to the general conferencing framework described in [RFC4353]. It also defines extensions to the Session Initiation Protocol (SIP) conference state event package. Furthermore, this document describes how the client can be invoked to join a conference via a web browser interface using the conferencing join web URL, which is also referred to as a Simple Join. The protocols defined in the document are used by SIP clients and servers to support conferencing scenarios.

1.1 Glossary

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

fully qualified domain name (FQDN)
globally unique identifier (GUID)
Hypertext Transfer Protocol (HTTP)
Transmission Control Protocol (TCP)
UTF-8
X.509

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

200 OK
202 Accepted
403 Forbidden
Audio/Video Multipoint Control Unit (AVMCU)
BENOTIFY (Best Effort NOTIFY)
conference
conference control command
conference control request
conference URI (conference-URI)
dialog
endpoint
endpoint identifier (EPID)
event package
focus
Focus Factory
Globally Routable User Agent URI (GRUU)
IM MCU
in-band provisioning
INVITE
lobby
MCU-Conference-URI
MCU-Type
MIME (Multipurpose Internet Mail Extensions)
mixer
Multipoint Control Unit (MCU)
notification
NOTIFY
organizer
participant
public switched telephone network (PSTN)
SERVICE
Session Description Protocol (SDP)

Session Initiation Protocol (SIP)
SIP message
SIP request
SIP response
SUBSCRIBE
subscription
Transport Layer Security (TLS)
URI (Uniform Resource Identifier)
URL (Uniform Resource Locator)
user agent client (UAC)
user agent server (UAS)

The following terms are specific to this document:

conference store: A database that stores all of the conference-related information for an organization.

final response: A Session Initiation Protocol (SIP) response that terminates an SIP transaction. All 2xx, 3xx, 4xx, 5xx, and 6xx responses are final.

first-party request: A conference control request that modifies the state of the sending participant only.

third-party request: A conference control request that modifies the state of participants other than the participant who sent the request.

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-CONFPRO] Microsoft Corporation, "Centralized Conference Control Protocol: Provisioning Specification", June 2008.

[MS-CONMGMT] Microsoft Corporation, "Connection Management Protocol Specification", June 2008.

[MS-OCER] Microsoft Corporation, "Client Error Reporting Protocol Specification", June 2008.

[MS-PSOM] Microsoft Corporation, "PSOM Shared Object Messaging Protocol Specification", March 2010.

[MS-SIP] Microsoft Corporation, "Session Initiation Protocol Extensions", August 2007.

[MS-SIPAE] Microsoft Corporation, "Session Initiation Protocol (SIP) Authentication Extensions", June 2008.

[MS-SIPRE] Microsoft Corporation, "Session Initiation Protocol (SIP) Routing Extensions", June 2008.

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

[RFC2976] Donovan, S., "The SIP INFO Method", RFC 2976, October 2000, http://www.ietf.org/rfc/rfc2976.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

[RFC3265] Roach, A. B., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002, http://www.ietf.org/rfc/rfc3265.txt

[RFC3280] Housley, R., Polk, W., Ford, W., and Solo, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002, http://www.ietf.org/rfc/rfc3280.txt

[RFC3311] Rosenberg, J., "The Session Initiation Protocol (SIP) UPDATE Method", RFC 3311, September 2002, http://www.ietf.org/rfc/rfc3311.txt

[RFC3629] Yergeau, F., "UTF-8, A Transformation Format of ISO 10646", STD 63, RFC 3629, November 2003, http://www.ietf.org/rfc/rfc3629.txt

[RFC4028] Donovan, S., Rosenberg, J., "Session Timers in the Session Initiation Protocol (SIP)", RFC 4028, April 2005, http://www.ietf.org/rfc/rfc4028.txt

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

[RFC4575] Rosenberg, J., et al., "A Session Initiation Protocol (SIP) Event Package for Conference State", RFC 4575, August 2006, http://www.ietf.org/rfc/rfc4575.txt

[XML10] World Wide Web Consortium, "Extensible Markup Language (XML) 1.0 (Third Edition)", February 2004, http://www.w3.org/TR/REC-xml

1.2.2 Informative References

[IETFDRAFT-SIPSOAP-00] Deason, N., "SIP and SOAP", draft-deason-sip-soap-00, June 30 2000, http://www.softarmor.com/wgdb/docs/draft-deason-sip-soap-00.txt

[MS-GLOS] Microsoft Corporation, "Windows Protocols Master Glossary", March 2007.

[MS-OFCGLOS] Microsoft Corporation, "Microsoft Office Master Glossary", June 2008.

[MS-SIPREGE] Microsoft Corporation, "Session Initiation Protocol (SIP) Registration Extensions", June 2008.

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

[RFC3325] Jennings, C., Peterson, J., Watson, M., "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002, http://www.ietf.org/rfc/rfc3325.txt

[RFC4579] Johnston, A. and Levin, O., "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents", BCP 119, RFC 4579, August 2006, http://www.ietf.org/rfc/rfc4579.txt

1.3 Protocol Overview (Synopsis)

1.3.1 Architecture and Background

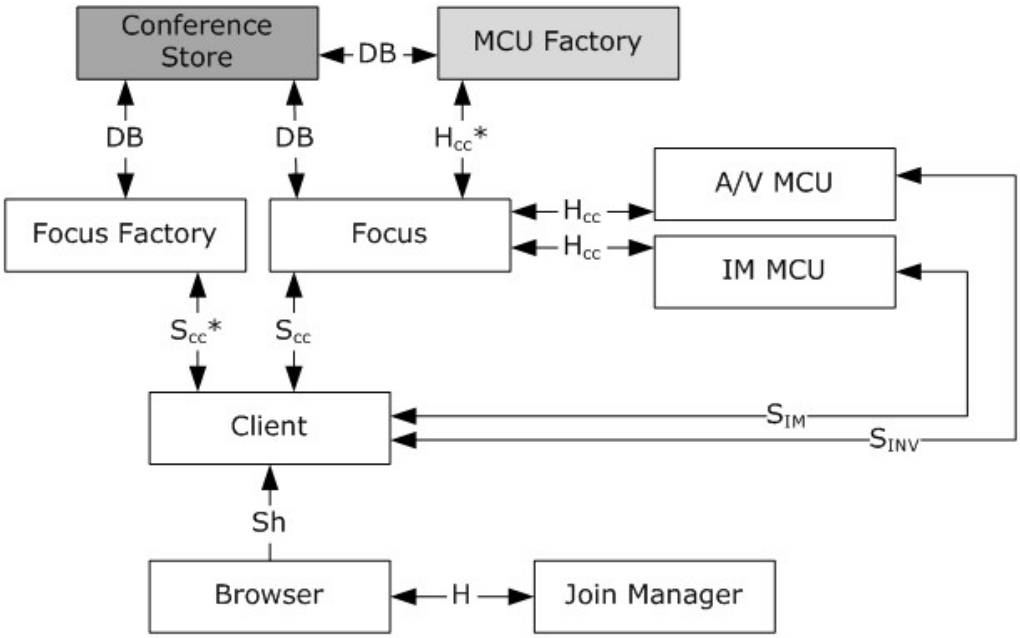
The conferencing system defined in this document extends the tightly coupled **conference** architecture described in [RFC4353] by incorporating the following:

- Support for multiple occurrences of a **Multipoint Control Unit (MCU)**, as opposed to plain **mixers**, to participate in a conference and provide richer control and media conferencing services. MCUs maintain and publish conference state to the **focus** using the conference data model described in [RFC4575].
- A conference provisioning component called the **Focus Factory** which is used to create, delete , modify, and query conference information. Conference provisioning is described in [MS-CONFPRO].
- Implements the conference policy and **notification** services described in [RFC4353] and also performs various runtime conference management tasks, such as conference activation and deactivation, allocating and managing MCUs, and performing conference roster aggregation. The focus notification service implements the conference **event package** notifications described in [RFC4575].

Conference state is the current participant roster and status and configuration of all individual conference components in the collection of components in a conference. Conference state is encapsulated by a full conference state event package, as defined in section 2.2.2.2.

Simple Join is made possible by the Join Manager. The Join Manager is responsible for parsing the conferencing join web **URL (Uniform Resource Locator)** and determining the **conference URI (conference-URI)** that is required by the client to join the conference. Join Manager is also responsible for detecting the presence of the client and starting it.

This architecture and the inter-component protocols are shown in the following figure. Note that this is just one possible way of implementing a conferencing system and is used to illustrate the protocol for readability purposes. Implementers can adopt other architectures, as long as they keep the external protocol consistent with this specification.



LEGEND

H	HTTP
Sh	Shell execute
S_{cc}	C3P over SIP (INVITE dialog + C3P in INFO) + conference event package
S_{cc}*	C3P over SIP (no dialog + C3P in SERVICE) + no conference event package
H_{cc}	C3P over HTTP + conference event package over HTTP
H_{cc}*	C3P over HTTP + no conference event package
S_{IM}	SIMPLE-based IM (INVITE dialog + session-based IM with MESSAGE)
S_{INV}	SIP SDP RT A/V media negotiation (INVITE dialog)
DB	Interface to Conference Store (e.g., SQL)

Figure 1: Conferencing architecture

In the preceding figure, the client uses the Focus Factory to schedule a conference. During conference creation, the client supplies the conference meta-data and the modalities of interest. Each modality in turn is comprised of several media types, such as meeting, audio, video, and chat. The conference-URI and other metadata for the created conference are returned by the Focus Factory to the client. This architecture assumes a common shared **conference store** between the Focus Factory and the focus.

The client then joins a conference by communicating with the focus. At conference startup, the focus retrieves the conference metadata from the conference store and then bootstraps all of the MCUs needed for the conference. It then admits the participant into the conference and notifies all watchers of conference state changes. The client can then perform various conference control operations through the focus.

After an MCU is bootstrapped, it publishes its initial state and capabilities through the conference roster. Participants discover the MCUs through the conference roster and can then join the MCUs of interest by sending the appropriate command to the focus. After a participant joins the MCU, it can exchange media with the MCU, as well as perform appropriate media control. Unlike mixers, described in [RFC4353], MCUs in the conferencing system defined by this specification are capable

of receiving **conference control commands** from the focus and performing changes to the conference state. Each MCU maintains its conference state, and publishes it to the focus through the general conference notification mechanism described in [RFC4575].

The focus aggregates the conference state received from the MCUs with the conference state that it maintains into a single conference document and publishes it to its watchers. This is an extension to the specifications in both [RFC4575] and [RFC4353].

Note that the client entity can be one or more clients. In addition to the **organizer** of the conference, whose joining sequence is described earlier, other users who request to participate in the conference can do so via Simple Join. When the user clicks on the conferencing join web URL, Join Manager receives the request. Upon receipt of the request, Join Manager parses the conferencing join web URL and returns back in as a HTTP response a Java-script that contains logic and contents of an xml document which has the conference-URI embedded in it. The logic in the script is to enumerate the candidate applications installed on the client computer, create the XML document in the appropriate location with an appropriate extension and pick one of the client applications to be bootstrapped with the conference document to join the meeting.

1.3.2 End-to-End Call Flow Overview

This section describes a basic end-to-end call flow of the signaling steps involved in establishing a conference.

The following figure shows a typical call flow for establishing a conference.

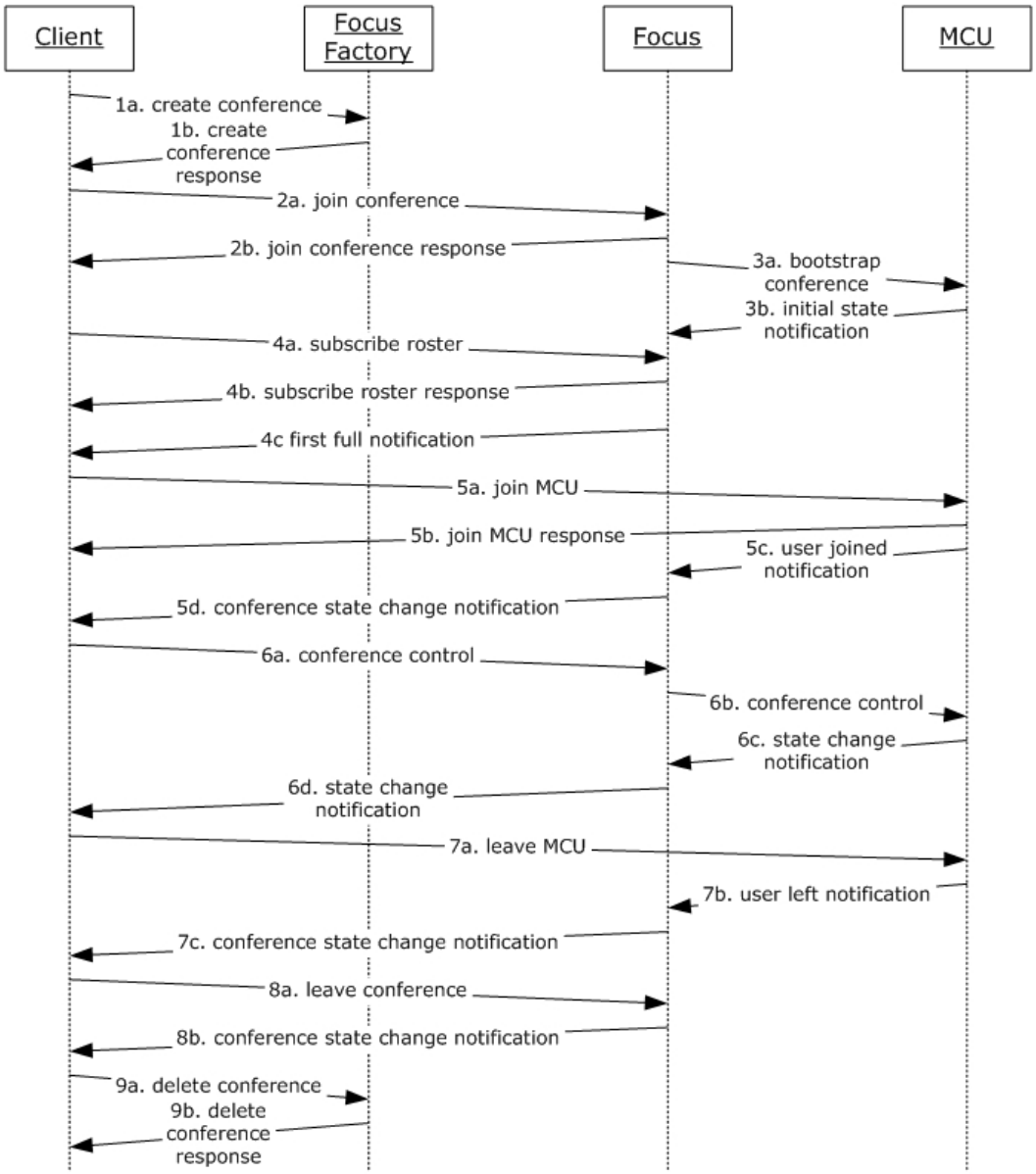


Figure 2: Conference signaling call flow

The call flow in the preceding figure is explained in this section. It is useful to note the following:

- The client entity can be one or more clients participating in a conference, including the organizer of the conference.
- The Focus Factory and the focus can be collocated. They are shown as distinct entities for readability.
- The MCU entity can be one or more MCUs servicing the conference. The focus and the MCU can be collocated. They are shown as distinct entities for readability.

Unless otherwise specified, the following steps are usually temporally correlated.

Step 1: The organizer contacts the Focus Factory to provision a new conference. The provisioning protocol is described in [MS-CONFPRO]. The client obtains its Focus Factory **Uniform Resource Identifier (URI)** using **in-band provisioning**, as described in [MS-SIPREGE]. At the end of this step, the client has a conference-URI and associated provisioning data, such as the conference subject and conference expiration time.

Step 2: The client joins the conference by sending a signaling request to the focus. At the end of this step, the client gets a response from the focus and it has joined this conference.

Step 3: A conference is activated in the focus when the first client joins the conference. As part of this process, the focus bootstraps the MCUs needed for the conference. On successful bootstrap, the MCUs notify the focus of their initial conference state. Note that step 3 does not have temporal correlation with step 2, but is a prerequisite for step 4.

Step 4: The client subscribes to the conference roster, after which it can receive all the conference state change notifications. At the end of this step, the client receives the first full conference state document, with which it can construct initial conference state.

Step 5: The client joins one or more MCUs available in the conference. This step involves signaling and media handshakes that are MCU-specific. This signaling step involves sending a **Session Initiation Protocol (SIP) INVITE** to the MCU and then negotiating media. At the end of this step, the MCU notifies the focus that the user has joined the MCU, which in turn is propagated to all watchers.

Step 6: The client performs various conference control operations. Examples of conference control include locking a conference, ejecting users, promoting users, and muting a user's media. Conference control can span multiple components, such as MCUs and the focus. This step can also result in state change notifications, which are sent to all watchers.

Step 7: The conference ends and the client leaves the MCU by performing an appropriate MCU-specific signaling handshake. The MCU notifies the focus that the user has left and this, in turn, is propagated to all watchers. At this point, the client could still be part of the conference, but is not participating in the media types represented by the MCU.

Step 8: The client leaves the conference completely by sending a signaling request to the focus. At this point, the client is no longer part of the conference.

Step 9: The organizer sends a request to the Focus Factory to de-provision the conference.

1.3.3 Inter-Component Protocols

The protocols used between the client and the server components are given in the following subsections.

1.3.3.1 Browser to Join Manager

To enable Simple Join, the Join Manager returns a Java-script in **Hypertext Transfer Protocol (HTTP)** response to detect the presence of the client. The Java-Script runs at the client end in the browser and detects and picks the appropriate client to launch with the help of an Active-X control or a Firefox Plug-in. The Active-X control or plug-in then creates a document with a pre-defined schema that is provided by the Java-script and executes this document to launch the appropriate client.

1.3.3.2 Client to Focus Factory

The client uses the application/cccp+xml document format to exchange conference provisioning commands with the Focus Factory. The SIP **SERVICE** method is used as the carrier protocol for these documents, as described in [IETFDRAFT-SIPSOAP-00].

1.3.3.3 Client to Focus

The client extends the procedures described in [RFC4353] to communicate with the focus. Specifically, it establishes an INVITE **dialog** with the focus and then uses the application/cccp+xml document format to exchange conference control commands with the focus. The SIP INFO method is used as the carrier protocol for these documents.

The client establishes a **SUBSCRIBE** dialog with the focus and then uses the conference event package to receive conference notifications, as described in [RFC4575].

1.3.4 Scope of this document

This specification defines extensions to the conference event package data model described in [RFC4575] to support the architecture described earlier. These documents are also referred to as application/conference-info+xml documents.

This specification defines the Centralized Conference Control Protocol (C3P), which can be used to exchange conference control commands between various conferencing entities. These are also referred to as application/cccp+xml documents.

This specification also defines the mechanism for using SIP as the carrier protocol for C3P commands between clients and the focus.

This specification also defines the roster aggregation algorithm used by the focus to aggregate the conference state published by multiple MCUs in the conference.

This specification treats the focus and the MCUs as one homogeneous "server" entity as presented to the client. The protocol used between the focus and the MCUs to exchange messages if they are not collocated is outside the scope of this specification. However, there are some intercomponent protocol behaviors that all implementations are required to adhere to so that a holistic behavior can be presented to the client. These behaviors are specified in this document.

Extensions to this specification can specify the protocol used between the client and the MCUs for negotiating media sessions.

1.4 Relationship to Other Protocols

This specification depends on the SIP. It defines additional SIP message body formats and XML schemas to support the protocols defined in this document. This specification also depends on the conference event package data model described in [RFC4575] and defines extensions to it.

1.5 Prerequisites/Preconditions

This protocol assumes that both the clients and the server support SIP, and that they implement the extensions specified in the following extension specifications as needed:

- Session Initiation Protocol Extensions [MS-SIP]
- Session Initiation Protocol Routing Extensions [MS-SIPRE]

This protocol assumes that conferences are provisioned using the protocol described in [MS-CONFPRO].

1.6 Applicability Statement

This protocol is applicable when clients and the server support SIP and intend to use one or more features of the conferencing functionality defined by this protocol.

1.7 Versioning and Capability Negotiation

This protocol uses the **C3PVersion** attribute to indicate the version of the Centralized Conference Control Protocol messages. The currently defined protocol version is "1".

Explicit capability negotiation can be done for these messages using standard SIP-based mechanisms, such as the SIP **Supported** header.

1.8 Vendor-Extensible Fields

There are no vendor-extensible fields specific to this protocol. Standard XML extension mechanisms can be used to extend Centralized Conference Control Protocol commands. Similarly, standard SIP extension mechanisms can be used to extend the signaling messages specified in this protocol, as needed.

1.9 Standards Assignments

None.

2 Messages

2.1 Transport

2.1.1 HTTP Transport

Simple Join uses HTTP transport.

2.1.2 SIP Transport

This specification does not introduce a new transport to exchange messages but is capable of being used with **Transmission Control Protocol (TCP)** and **Transport Layer Security (TLS)** as transports for SIP.

2.2 Message Syntax

This conferencing specification does not introduce a new message format for SIP. It relies on the **SIP message** format specified in [RFC3261] section 7, the authentication extensions defined in [MS-SIPAE], the routing protocol extensions defined in [MS-SIPRE], and the connection management protocol defined in [MS-CONMGMT].

Conferencing messages defined by this specification can be categorized as follows:

- Signaling messages exchanged between the client and the focus for conference signaling dialog establishment or teardown.
- Subscription messages exchanged between the client and the focus for conference **subscription** dialog establishment or teardown.
- Notification messages from the focus to the client for conference notifications.
- Control messages exchanged between the client and the focus or MCUs, or both, for the purposes of conference control.

These message formats are discussed in subsequent sections.

2.2.1 Focus Signaling Messages

2.2.1.1 Common Signaling Header Formats

The following common header formats and rules are applicable to all of the messages defined in this section.

The requests and responses SHOULD contain authentication headers, as defined in [MS-SIPAE].

The **Content-Type** header field MUST be set to "application/cccp+xml".

The content body MUST conform to the request or response format defined in section 2.2.2.11.

The **ms-keep-alive** header is optional. If specified, follow the specifications in [MS-CONMGMT].

Unless specified otherwise in the following sections, all unknown headers SHOULD be ignored by the processing entities.

2.2.1.2 Format of the Signaling Dialog Establishment Message

The INVITE request is used by a participant to establish a dialog with the focus. It relies on the SIP message formats specified in [RFC3261]. The sender is a **user agent client (UAC)**, as defined in [RFC3261].

The SIP Uniform Resource Identifier (URI) in the **To** header field of the request MUST be set to the conference-URI.

The client MUST add a valid **Contact** header that can be used as the remote target URI of the SIP dialog route set, as specified in [RFC3261] section 12.

A **Supported** header field with the option tag **timer** is optional in INVITE requests and, if specified, follows the specifications in [RFC4028]. The **Session-Expires** header field SHOULD be present in requests.

The body of the request MUST be set to a C3P **addUser** Request Body for a focus INVITE request, as specified in section 2.2.3.3.

The INVITE response is used to indicate the success or failure of the INVITE request. It conforms to the SIP message formats specified in [RFC3261]. The focus receiving the INVITE request and generating the response behaves as a **user agent server (UAS)**, as defined in [RFC3261].

The body of the response MUST be set to a C3P **addUser** Response Body for a focus INVITE response, as specified in section 2.2.3.4.

The **200 OK** response to INVITE requests MUST have a valid **Contact** header that can be used as the remote target URI of the SIP dialog route set, as specified in [RFC3261] section 12. It SHOULD have the **isfocus** feature parameter, as described in [RFC4579].

The **Session-Expires** header SHOULD be present in a response if the corresponding request indicated support for session timers by the presence of a **Supported** header field with the option tag **timer**. In such a case, the **Session-Expires** header value is computed as specified in [RFC4028], with the exception that the refresher parameter MUST always be set to "uac". In addition, such a response MUST include a **Supported** header field with the option tag **timer**, as well as a **Require** header field with the option tag **timer**.

The INVITE response MUST contain an **Allow** header that lists the SIP verbs supported by the focus. Typically, this list consists of INVITE, ACK, BYE, CANCEL, UPDATE, and INFO.

Require headers can be present in the request.

2.2.1.3 Format of the Signaling Dialog Update Message

The UPDATE request is used to extend a dialog. This request relies on the SIP UPDATE message format defined in [RFC3311].

The message body MUST be empty for both requests and responses.

The **Supported** header field with the option tag **timer** MUST be present in the request.

Require headers can be present in the request.

2.2.1.4 Format of the Signaling Dialog Teardown Message

The BYE request is used to tear down a dialog. This request relies on the SIP message formats specified in [RFC3261]. The sender is a UAC, as defined in [RFC3261].

The BYE response is used to respond to a BYE request. It relies on the SIP message formats specified in [RFC3261].

The message body SHOULD be empty for both requests and responses.

Require headers can be present in the request.

2.2.1.5 Conference Control Messages

INFO **SIP requests** and responses are used to exchange conference control messages. These messages rely on the SIP message formats specified in [RFC2976]. Unless otherwise specified, these INFO messages are exchanged within the INVITE dialog specified previously.

The INFO request MUST contain a valid request message body as defined in section 2.2.2.11.

A body MUST be present for INFO responses. If a body is present, it MUST be a valid response body, as defined in section 2.2.2.11.

Require headers can be present in the INFO request or response.

2.2.2 Focus Subscription Messages

The focus subscription dialog SHOULD follow the conference-event package dialog establishment procedures described in [RFC4575].

2.2.2.1 Subscription Establishment Messages

Subscription establishment SHOULD use the SUBSCRIBE request and response defined in [RFC3265] with the conference event package defined in [RFC4575].

The SIP URI in the **To** header field of the request MUST be set to the conference-URI.

The client MUST add a valid **Contact** header that can be used as the remote target URI of a SIP dialog route set, as specified in [RFC3261] section 12.

The following extensions are specified for SUBSCRIBE requests:

- The **Accept** header, if present, SHOULD be populated with a value of 'application/conference-info+xml'. Another **Accept** header SHOULD NOT be present.
- The **Supported** header field with the option tag **ms-benotify** can be present. If present, it SHOULD follow the **BENOTIFY (Best Effort NOTIFY)** extensions defined in [MS-SIP].
- The **Supported** header field with the option tag **ms-piggyback-first-notify** SHOULD be present. If present, it SHOULD follow the piggyback notification mechanism defined in [MS-SIP].
- **Require** headers can be present in the request.

2.2.2.2 Extensions to the application/conference-info+xml Document Format

The application/conference-info+xml document format, as specified in [RFC4575], specifies the data model for a conference. This specification extends the conference data model with additional elements.

All of the extension elements that are used in messages defined by this specification are defined subsequently. Extensions to this document can define the semantics of other elements and attributes on which they depend. They can also introduce new extensions to this data model.

The cardinality of each extension element is specified in the XML schema using standard **minOccurs** and **maxOccurs** XSD conventions. Similarly, the cardinality of each extension attribute is specified in the XML schema using standard "required" or "optional" attribute values. Similarly, the namespace of each extension attribute or element is specified in the XML schema using standard conventions and is omitted here for brevity. The **conference-description** element is defined in the XML schema in section 8.1.

Requests and responses can further restrict this data model. Any such restrictions are specified by their message syntaxes as needed.

The following is the data model hierarchy for a conference. Elements in the data model are identified by the following markings:

- Extensions added by this protocol are marked with a plus sign (+).
- Elements marked with an asterisk (*) SHOULD NOT be used in conferencing commands or notifications because they are not used by the conferencing system.
- Elements marked with a hyphen (-) have additional semantics beyond those defined in [RFC4575].

```
|
|-- conference-description
|   |-display-text (*)
|   |-subject
|   |-free-text (*)
|   |-keywords (*)
|   |-conf-uris (-)
|   |-service-uris (*)
|   |-maximum-user-count (*)
|   |-available-media (*)
|   |-disclaimer (+)
|   |-organizer (+)
|   |-conference-id (+)
|   |-conference-key (+)
|   |-last-update (+)
|   |-last-activate (+)
|   |-is-active (+)
|   |-expiry-time (+)
|   |-admission-policy (+)
|   |-organizer-roaming-data (+)
|   |-notification-data (+)
|   |-pstn-access (+)
|   |-lobby-capable (+)
|   |-anonymous-type-allowed (+)
|   |-autopromote (+)
|   |-autopromote-allowed (+)
|   |-disclaimer-title (+)
|   |-recording-allowed (+)
|   |-externaluser-recording-allowed (+)
|
|
|-- host-info (*)
|
|-- conference-state (*)
|
|-- users
|     |-- user
```

```
|    |    |--display-text
|    |    |--associated-aors (*)
|    |    |--roles (-)
|    |    |--languages (*)
|    |    |--cascaded-focus (*)
|    |    |-- endpoint
|    |        |--display-text  (*)
|    |        |--referred (*)
|    |        |--status
|    |        |--joining-method
|    |        |--joining-info
|    |        |--disconnection-info (*)
|    |        |--media
|    |            |--display-text
|    |            |--type
|    |            |--label
|    |            |--src-id
|    |            |--status
|    |            |--media-ingress-filter (+)
|    |            |--media-egress-filter (+)
|    |
|    |            |
|    |        |--call-info
|    |        |--roles (+)
|    |        |--authMethod (+)
|    |        |--accessMethod (+)
|    |        |--clientInfo (+)
|    |        |--post-dial (+)
|    |        |--pstnRole (+)
|    |        |--pstnLeaderPassCode (+)
|    |        |--endpoint-capabilities (+)
|    |        |--is-robot (+)
|    |        |--current-sidebar (+)
|    |        |--session-on-behalf-of (+)
|    |        |--client-recording (+)
|    |        |--separator (+)
|-- sidebars-by-ref (*)
|
|-- sidebars-by-val (*)
|
|-- conference-view (+)
|   |- entity-view (+)
|      |- entity-capabilities (+)
|      |- entity-policy (+)
     |      |- entity-settings (+)
|      |- entity-state (+)
|
```

2.2.2.3 conference-description Extensions

The following elements are extensions to the **conference-description** element.

disclaimer: A descriptive string that can be used as a general purpose disclaimer in the conference notification document.

disclaimer-title<1>: A string that can be used as a title for a general purpose disclaimer in the conference notification document.

lobby-capable: Specifies whether the conference supports changing the admission policy after activation by issuing a **modifyConferenceLock** command, as defined in section 3.5. The value of this element is affected by the **server-mode** element given during conference provisioning, as specified in [MS-CONFPRO] section 2.2.2.1. If the **server-mode** for the conference is "13", this value MUST be "false". Otherwise, it MUST be "true".

anonymous-type-allowed<2>: Specifies whether the conference supports the anonymous admission policy. Note that if a conference supports changing the admission policy after activation, it might not support the anonymous admission policy.

autopromote<3>: The current automatic promotion policy applied to the conference that identifies who should be promoted to presenter upon being admitted to the conference. The value for this element is defined by an unsigned integer, as defined in [MS-CONFPRO] section 2.2.2.1. Each bit represents a class of users that are automatically promoted. The currently defined values are as follows:

- "None": 0x00000000 (as default)
- "Everyone": 0x80000000 (bit 31)
- "Company" (Authenticated users): 0x00008000 (bit 15)

New values might be added in future releases.

autopromote-allowed<4>: The allowed automatic promotion policies that could be applied to the conference by a presenter through issuing a modifyConferenceLock command, as defined in section 3.5. The value of this element is defined the same as the autopromote element.

admission-policy: The conference admission policy. Three admission policies are defined by this protocol as follows:

- **closedAuthenticated:** Admission to the conference is restricted to the invitees specified by this organizer. Invitee list creation and modification is specified in [MS-CONFPRO].
- **openAuthenticated:** Admission to the conference is permitted for any authenticated user. Authentication mechanisms are specified in [MS-SIPAE]. This is the default admission policy. If the conference was provisioned with the **server-mode** property set to "13", as described in [MS-CONFPRO] section 2.2.2.1, users authenticated through federation are classified as authenticated in this context. For other values of **server-mode**, such users are considered not authenticated<5>.
- **anonymous:** Admission to the conference is permitted for any user.

notification-data: Conference level notification data that can be specified by the organizer. This is made available to all participants through conference notification. The semantics of this attribute are further defined in [MS-CONFPRO].

pstn-access: The **public switched telephone network (PSTN)** access information for the conference.

pstn-lobby-bypass<6>: Specifies whether PSTN users can bypass the conference lobby.

pstn-lobby-bypass-allowed<7>: Specifies whether the conference could be modified to allow users joining the conference though the PSTN to bypass the conference lobby. A presenter can change the current lobby bypass setting by issuing a **conferenceModifyLock** command, as defined in section 3.5.

recording-allowed<8>: Specifies whether internal clients can record the conference. This is set by the administrator.

external-recording-allowed<9>: Specifies whether external clients can record the conference. This is set by the administrator. This element is applicable only when the **recording-allowed** element is "TRUE".

2.2.2.4 Extensions to conf-uris Element Semantics

Implementations conforming to this protocol MUST generate and consume the **conf-uris** element based on the following extension semantics, in addition to those defined in [RFC4575] section 5.3.1:

- An **entry** element MUST be listed for each MCU in the conference.
- An **entry** element MUST be listed for each alternative conference join URLs.
- This specification specifies the following for the **purpose** subelement:
  - **audio-video:** An Audio-Video MCU.
  - **chat:** An Instant Messaging MCU.
  - **meeting:** A legacy Web Conferencing / Application Sharing MCU.
  - **data-conf:** A Web Conferencing MCU implementing the protocols specified in [MS-PSOM]<10>.
  - **phone-conf:** A Phone Conferencing MCU.
  - **applicationsharing:** An Application Sharing MCU<11>.
  - **web-internal:** A URL to join the conference via a web browser interface from inside the enterprise<12>.
  - **web-external:** A URL to join the conference via a web browser interface from outside the enterprise<13>.
- The **uri** subelement lists a SIP URI corresponding to an **audio-video**, **chat**, **meeting**, **data-conf**, **phone-conf**, **applicationsharing purpose** element.
- The **encrypted-uri** subelement lists the encrypted URL corresponding to the **web-internal** or **web-external purpose** element<14>.

Extensions to this document can specify one or more signaling protocols to enable clients to join and access these MCUs.

2.2.2.5 Extensions to roles Element Semantics

The cardinality of the **roles** element is constrained to zero or one by this specification. Thus, entities MUST NOT list more than one **role** inside the **roles** element when used as a sub-element of the **user** element or the **endpoint** element.

2.2.2.6 endpoint Element Extensions

The **endpoint** extensions are used to communicate various focus and MCU-specific extension data for each connected **endpoint**.

The following attributes are defined for each **endpoint** element:

- **session-type:** The entity to which the **endpoint** is connected. This attribute MUST be set to the **MCU-Type** if the **endpoint** is connected to an MCU. If the **endpoint** is connected to the focus itself, it is set to "focus".
- **epid:** This attribute is an **endpoint identifier (EPID)**, if available, as defined in [MS-SIPRE].
- **sip-instance:** This attribute is endpoint SIP instance identifier, if available, as defined in [MS-SIPRE].
- **endpoint-uri:** A URI that can be used to target messages to the **endpoint**, if applicable. For example, this attribute can be a **Globally Routable User Agent URI (GRUU)** identifying the **endpoint**.
- **refer-to-uri:** A URI that can be used as the **request-URI** of any outgoing requests generated by the receiver when processing a C3P request. This attribute MUST be a SIP URI, and can include any SIP headers. Any SIP headers in the **refer-to-uri** attribute SHOULD be propagated to the outgoing request, suitably encoded, as specified in [RFC3261] section 19.1.5.
- **asserted-identity:** The asserted identity URI of an **endpoint**, as defined in [RFC3325]. This attribute MUST be a SIP URI.

The following child elements are defined for each **endpoint** element:

- **roles:** The roles for the **endpoint**, as defined in [RFC4575] section 5.6.3. This element is optional.
- **authMethod:** Whether the user is an enterprise user with a value of "enterprise", an anonymous user with a value of "anonymous", or a federated user with a value of "federated", as determined by the focus or the MCU. This element is optional.
- **accessMethod:** Whether the user is connected from within the enterprise with a value of "internal" or from the public Internet through the enterprise edge with a value "external". This element is optional.
- **clientInfo:** Various client-specific information. Extensions to this specification can define additional elements and attributes within the **clientInfo** extension and their processing semantics.
- **endpoint-capabilities:** The capabilities supported by this **endpoint**. Extensions to this specification can define the subelements and attributes within this extension and their processing semantics.
- **separator:** An element indicating a new extension start. All elements after one separator are in the same extension group.
- **session-on-behalf-of:** An element containing the URI on behalf of whom the user is joining the conference<15>. This element MUST be put after the first **separator** if it is specified.
- **client-recording:** An element to specify if the **endpoint** is in recording status. This element MUST be put after the second **separator** if it is specified.

2.2.2.7 conference-view Element Extensions

The **conference-view** extension is used to communicate settings, policies, capabilities, and state information for the focus and the MCUs within the conference.

The **conference-view** element is a container of **entity-view** elements, each describing either an MCU or the focus.

The following attributes are defined for the **conference-view** element:

- **state:** Indicates whether the document contains all of the **conference-view** information, with the value "full", or only the information that has changed after the previous document, with the value "partial". The value "deleted" SHOULD be used only if the conference is deleted. In this case, the **conference-info** state MUST be set to "deleted".

The following attributes are defined for each **entity-view** element:

- **entity:** The URI that identifies the entity being described in the document. The focus MUST always be addressed using the conference-URI. The MCUs MUST always be addressed using the appropriate **MCU-Conference-URI**.
- **state:** Indicates whether the document contains the entire **entity-view** information, with the value "full", or only the information that has changed after the previous document, with the value "partial". The value "deleted" SHOULD be used only if the entity is removed from the conference.

The following child elements are defined for each **entity-view** element:

- **entity-capabilities:** The capabilities supported by the entity. It MUST NOT be present in **conference control requests**. It SHOULD be present in conference notifications.
- **entity-policy:** The policies that need to be applied by the entity for the conference. It MAY be present in conference control requests sent to the entity. It MUST NOT be present in conference notifications.
- **entity-settings:** The settings that need to be applied by the entity for the conference. It MAY be present in conference control requests sent to the entity. It MUST NOT be present in conference notifications.
- **entity-state:** The current state of the conference within the entity. It MUST NOT be present in conference control requests sent to the entity. It MUST be present in "full" entity-view notifications. It SHOULD be present in other entity-view notifications.

Child elements of **entity-capabilities**, **entity-policy**, and **entity-settings** elements can be specified in extension specifications.

The following child elements are defined for each **entity-state** element:

- **displayText: display-text**, as defined in [RFC4575] section 5.6.1.
- **userCount: user-count**, as defined in [RFC4575] section 5.5.1.
- **active: active**, as defined in [RFC4575] section 5.5.2.
- **locked: locked**, as defined in [RFC4575] section 5.5.3.
- **data-mcu-state: data-mcu-state**, as defined in section 2.2.2.8.
- **permission-options: permission-options**, as defined in section 2.2.2.9.
- **permissions: permissions**, as defined in section 2.2.2.10.

Other elements and attributes can be defined in extension specifications.

2.2.2.8 data-mcu-state Element Extensions

The **data-mcu-state** extension is used to communicate state information for Web Conferencing MCUs within the conference. An MCU MAY provide these elements through C3P, and a client MAY use these values to avoid establishing a PSOM connection when there is no content to be displayed to the user. If these elements are not present, a client MUST NOT assume they have any particular default value. For additional details about Web Conferencing MCU-specific concepts, see [MS-PSOM].

The following child elements are defined for each **data-mcu-state** element:

- **hasContent:** An **xs:boolean** value that specifies whether the Web Conferencing MCU contains any content for the conference.
- **hasContentInMeeting:** An **xs:boolean** value that is "true" when both of the following conditions are met: 1) the Web Conferencing MCU contains content for the conference, and 2) the content is visible to participants other than the organizer.
- **hasPresentedContentInMeeting:** An **xs:boolean** value that is "true" when both of the following conditions are met: 1) the Web Conferencing MCU contains content for the conference, and 2) the content is in the "presented" state.

2.2.2.9 permission-options Element Extensions

The **permission-options** extension is used to communicate potentially mutable permission values that affect the availability of various Web Conferencing MCU functionality to specific groups of participants within the conference.

The **permission-options** element is a container of **permission-option** elements, each describing the value and mutability of a particular permission for the Web Conferencing MCU in the conference.

The following child elements are defined for each **permission-option** element:

- **name:** An **xs:string** element identifying the permission option.
- **value:** An **xs:string** element describing the current value of the permission option.
- **mutable:** An **xs:boolean** element specifying whether the permission option value can be modified by clients by sending a **modifyConference** request to the MCU.

The Web Conferencing MCU SHOULD recognize the following permission option names and values. For additional details about Web Conferencing MCU-specific concepts, see [MS-PSOM].

- **PptAnnotationsAllowedPresenter:** The value is "true" if the Web Conferencing MCU allows participants with the role of "Presenter" in the conference to create annotations on PowerPoint content. Otherwise, the value is "false". If unspecified, the value SHOULD be assumed to be "false".
- **PptAnnotationsAllowedAttendee:** The value is "true" if the Web Conferencing MCU allows participants with the role of "Attendee" in the conference to create annotations on PowerPoint content. Otherwise, the value is "false". If unspecified, the value SHOULD be assumed to be "false".
- **AsynchronousBrowsingAllowedPresenter:** The value is "true" if the Web Conferencing MCU allows participants with the role of "Presenter" in the conference to browse content independently from the active presenter. Otherwise, the value is "false". If unspecified, the value SHOULD be assumed to be "false".

- **AsynchronousBrowsingAllowedAttendee:** The value is "true" if the Web Conferencing MCU allows participants with the role of "Attendee" in the conference to browse content independently from the active presenter. Otherwise, the value is "false". If unspecified, the value SHOULD be assumed to be "false".

2.2.2.10 permissions Element Extensions

The **permissions** extension is used to communicate static and immutable permission values which affect the availability of various Web Conferencing MCU functionality to specific groups of participants within the conference.

The **permissions** element is a container of **permission-type** elements, each describing the value of a particular permission for the Web Conferencing MCU in the conference.

The following child-elements are defined for each **permission-type** element:

- **name:** An **xs:string** element identifying the permission option.
- **value:** An **xs:string** element describing the current value of the permission option.

The Web Conferencing MCU SHOULD publish the following permission names and values. For additional details about Web Conferencing MCU-specific concepts, see [MS-PSOM].

- **EnableDataCollaboration:** The value is "false" if the Web Conferencing MCU will deny any participant in the conference from using Web Conferencing functionality. Otherwise, the value is "true". If unspecified, the value SHOULD be assumed to be "false".
- **AllowAnnotations:** The value is "false" if the Web Conferencing MCU will deny any participant in the conference from creating annotations on any type of content. Otherwise, the value is "true". If unspecified, the value SHOULD be assumed to be "false".
- **AllowExternalUsersToSaveContent:** The value is "false" if the Web Conferencing MCU will deny any anonymous or federated participant in the conference from downloading the original binary file contents for content on the Web Conferencing MCU. Otherwise, the value is "true". If unspecified, the value SHOULD be assumed to be "false".
- **EnableFileTransfer:** The value is "false" if the Web Conferencing MCU will deny any participant in the conference from downloading the original binary file contents for content on the Web Conferencing MCU. Otherwise the value is "true". If unspecified, the value SHOULD be assumed to be "false".

2.2.2.11 application/cccp+xml Document Format

The application/cccp+xml document format specifies the document format for the Centralized Conference Control Protocol (C3P), as specified in this protocol. A well-formed C3P document MUST be valid XML, as specified in [XML10], conformant to the C3P schema defined in section 7, and MUST be encoded using **UTF-8**, as specified in [RFC3629]. The protocol includes requests and responses.

The cardinality of each extension element is specified in the XML schema using standard **minOccurs** and **maxOccurs** XSD conventions, unless explicitly specified otherwise in the following subsections. Similarly, the cardinality of each extension attribute is specified in the XML schema using the standard "required" attribute value. Similarly, the namespace of each extension attribute or element is specified in the XML schema using standard conventions and is omitted here for brevity.

Unless otherwise specified, implementations MUST ignore the elements and attributes that are not necessary for executing a particular command, as specified in section 2.2.2.2.

Unless otherwise specified, implementations MUST ignore extension elements and attributes that they cannot parse.

Requests and responses can further restrict this data model. Any such restrictions are specified by their message syntaxes, as needed.

2.2.3 C3P Request and Response Document Formats

This section specifies the command syntax for the C3P commands supported by the conferencing system. The supported C3P command set is a subset of the command set defined in the C3P schema. Each request or response document is carried inside a C3P request or response element, as specified in section 7.

The cardinality of each extension element is specified in the XML schema using standard **minOccurs** and **maxOccurs** XSD conventions, unless explicitly specified otherwise in the following subsections. Similarly, the cardinality of each extension attribute is specified in the XML schema using the standard "required" attribute value. Similarly, the namespace of each extension attribute or element is specified in the XML schema using standard conventions and is omitted here for brevity.

Unless otherwise specified, implementations MUST ignore the elements and attributes that are not necessary for executing a particular command, as specified in section 2.2.2.2.

Unless otherwise specified, implementations MUST ignore extension elements and attributes that they are not capable of parsing.

2.2.3.1 Requests

2.2.3.1.1 request Element

The **request** element is the root element of C3P requests. This element has the following attributes:

- **requestId**: The sender MUST generate a unique nonnegative integer, within its local scope, and set **requestId** to this value. The **requestId** attribute is used to match requests and responses. Even though **requestId** is defined as an **xs:string**, it MUST be composed of numeric characters for decimal digits. In other words, it MUST be composed of the set of characters "0" through "9".
- **C3PVersion**: Entities compliant with this specification MUST set this to "1".
- **from**: The sender's SIP URI. Typically, this is identical to the URI in the **From** header field of the SIP request that carries the request body.
- **to**: The target's SIP URI. Unless otherwise specified, this is identical to the conference-URI.

Exactly one **command** element MUST be included inside the **request** element. This is the case even though the schema supports specifying multiple commands for batching purposes. Commands are defined later, as needed, as part of the protocol operation. However, many common elements that occur inside most commands are defined subsequently.

2.2.3.1.2 conferenceKeys Element

The **conferenceKeys** element specifies a conference in the context of a conference control command. This element has the following attributes:

- **confEntity**: Supplies the conference-URI. It MUST be a SIP URI.
- **Conference-Id:** Supplies the conference identifier, as defined in [MS-CONFPRO]. This attribute is optional.

2.2.3.1.3 userKeys Element

The **userKeys** element specifies a user in the context of a conference control command. This element has the following attributes:

- **confEntity:** The conference-URI. It MUST be a SIP URI.
- **userEntity attribute**: The participant URI. It MUST be a SIP URI.

2.2.3.1.4 endpointKeys element

The **endpointKeys** element specifies the **endpoint** of a user in the context of a conference control command. This element has the following attributes:

- **confEntity:** The conference-URI. It MUST be a SIP URI.
- **userEntity**: The participant URI. It MUST be a SIP URI.
- **endpointEntity attribute**: The EPID, typed as **xs:string**. Two **endpointEntity** attributes are considered equal if they are equal using XML **xs:string** element comparison rules. It is recommended that implementations generate a **globally unique identifier (GUID)** and use it as the **endpointEntity** attribute.

2.2.3.1.5 mediaKeys Element

The **mediaKeys** element specifies the media session of an **endpoint** in the context of a conference control command. This element has the following attributes:

- **confEntity:** The conference-URI. It MUST be a SIP URI.
- **userEntity**: The participant URI. It MUST be a SIP URI.
- **endpointEntity**: The EPID, typed as **xs:string**. Two **endpointEntity** attributes are considered equal if they are equal using XML **xs:string** element comparison rules. It is recommended that implementations generate a **GUID** and use it as the **endpointEntity** attribute.
- **mediaId**: The media identifier that typically identifies a media session established between a client and an MCU. Extensions to this specification can specify the mechanism for establishing a media session with an MCU.

2.2.3.2 Responses

2.2.3.2.1 response Element

The **response** element is the root element of C3P responses. This element has the following attributes:

- **requestId**: Unique identifier for the response. This MUST be copied from the corresponding C3P request. Even though **requestId** is defined as an **xs:string**, it MUST be composed of the decimal digit characters in the set "0" through "9".

- **C3PVersion**: Entities compliant with this specification MUST set this to "1".
- **from**: The SIP URI of the sender. Implementations MUST populate this attribute from the **to** attribute of the corresponding request. Note that this is different from the standard SIP technique of generating responses that preserve the order of the **From** and **To** header fields.
- **to**: The SIP URI of the receiver. Implementations MUST populate this attribute from the **from** attribute of the corresponding request.
- **responder**: The SIP URI of the entity that actually generated this response, if multiple entities, such as the focus and an MCU, were involved in processing the request. This attribute is optional.
- **code**: Type of response. Valid values are "success", "pending", and "failure".
- **reason**: Specifies the failure reason. Values are defined in the schema.
- **displayString**: Specifies a failure reason phrase. This value can be used for client display purposes. It SHOULD NOT be used to make processing decisions.
- **timeout**: This attribute SHOULD NOT be set and SHOULD be ignored.
- **retryAfter**: Specifies a retry interval in seconds. This value can be used by the client to retry the command.
- **version attribute**: This attribute SHOULD be set and can be ignored if present.

Zero or one **command** element MUST be included inside the **response** element. This is the case even though the schema supports specifying multiple commands for batching purposes. Commands are defined later, as needed, as part of the protocol operation.

Implementations SHOULD be capable of handling C3P responses that have no command body inside the **response** envelope.

2.2.3.2.2 diagnostics-info Subelement

Zero or one instance of the **diagnostic-info** element SHOULD be present in the **response** element. The **diagnostic-info** element contains zero or more pairs of **key** and **value** elements that provide debugging and diagnostic information as follows.

The **diagnostics-info** subelement schema is as follows:

```
|
     | -- entry
|     | -- key
|     | -- value
```

The following keys are defined by this specification:

- ms-diagnostics
- ms-diagnostics-public

These keys have the same semantics as the corresponding headers with the same name in [MS-OCER].

The **value** element MUST be set to the header value constructed using the rules specified in [MS-OCER] for the header element of the same name.

An example of a **diagnostics-info** element is as follows:

```
<diagnostics-info>
  <entry>
    <key>ms-diagnostics</key>
    <value>1007;reason="Temporarily cannot route";source="sip.contoso.com";ErrorType="Connect
Attempt Failure"
;WinsockFailureDescription="The peer actively refused the connection
attempt";WinsockFailureCode="274D(WSAECONNREFUSED)";Peer="sip.fabrikam.com"
    </value>
  </entry>
</diagnostics-info>
```

Extensions to this specification can define other key-value pairs applicable to command responses.

2.2.3.3 addUser Request Document Format for Focus INVITE Requests

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules for specific elements apply.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI in the **To** header field of the INVITE request.

The following rules apply to the **user** element:

- Exactly one **user** element MUST be present inside the **addUser** body.
- The **entity** attribute of the **user** element MUST be a SIP URI and MUST equal the SIP URI in the **From** header field in the corresponding INVITE request.
- Only the **roles** and **endpoint** child elements are permitted inside the **user** element. Each of these elements MUST be listed exactly once.

roles: The **roles** element MUST be populated with the desired role, which can be either "presenter" or "attendee".

The following rules apply to the **endpoint** element:

- The **endpoint** element SHOULD<16> contain a **session-on-behalf-of** element that indicates the user on behalf of whom the user is joining the conference.
- The **endpoint** element MUST specify a valid **entity** attribute. Implementations SHOULD use a GUID for this purpose.

Following is an example of an **addUser** document.

```
<addUser>
  <conferenceKeys
    confEntity="sip:user@fabrikam.com;gruu;opaque=app:conf:focus:id:BE92"/>
  <user entity="sip:user@fabrikam.com">
    <roles>
      <entry>attendee</entry>
    </roles>
    <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}">
        <clientInfo
         xmlns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions" >
         <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator" />
```

```
            <lobby-capable
              xmlns="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions">
                true
            </lobby-capable>
          </clientInfo>
          <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator"/>
          <session-on-behalf-of>
            <entity>sip:carol@fabrikam.com</entity>
          </session-on-behalf-of>









      </endpoint>
    </user>
  </addUser>
```

2.2.3.4 addUser Response Document Format for Focus INVITE Responses

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conferenceKeys: This element MUST have the same values as those specified in the corresponding **addUser** request.

The following rules apply to the **user** element:

- Exactly one **user** element MUST be present inside the **addUser** body.
- The **entity** attribute of the **user** element MUST be the same as the **entity** attribute specified in the request.
- The **roles** element MUST be present and listed exactly once.
- The **endpoint** element SHOULD be present.

roles: The **roles** element MUST be populated with the granted role. This might be different from the requested role in some cases because of policy.

The following rules apply to the **endpoint** element:

- If the **endpoint** element is present, the **entity** attribute MUST have the same value as that specified in the corresponding request.
- The **endpoint** element SHOULD contain a **client-info** element with the **lobby-capable** element, which indicates that the client can enable its user to manage the conference **lobby**. For example, with these permissions the user can admit or deny participants and change the current admission policy.

The following example is an **addUser** document. Note that the conference ID is shown with four characters for readability, although it is usually a string with 16 to 32 alphanumeric characters, as defined in [MS-CONFPRO].

```
<addUser>
  <conferenceKeys
    confEntity="sip:user@fabrikam.com;gruu;opaque=app:conf:focus:id:BE92"/>
  <user entity="sip:user@fabrikam.com">
    <roles>
      <entry>attendee</entry>
    </roles>
    <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}">
        <clientInfo
         xmlns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions" >
         <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator" />
           <lobby-capable
             xmlns="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions">
               true
           </lobby-capable>
        </clientInfo>
        <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator"/>
        <session-on-behalf-of>
          <entity>sip:carol@fabrikam.com</entity>
        </session-on-behalf-of>









     </endpoint>
  </user>
</addUser>
```

2.2.3.5 modifyUserRoles Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

The following rules apply to the **userKeys** element:

- The conference-URI specified in the **confEntity** attribute of the **userKeys** element MUST match the SIP URI in the **To** header field of the INFO request.
- The **userEntity** attribute of the **userKeys** element MUST be a SIP URI.

user-roles: The **user-roles** element MUST be populated with the desired role, which can be either "presenter" or "attendee".

The following example is a **modifyUserRoles** request body:

```
<modifyUserRoles>
  <userKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    userEntity="sip:cathy@fabrikam.com"/>
  <user-roles>
    <entry>presenter</entry>
  </user-roles>
</modifyUserRoles>
```

2.2.3.6 modifyUserRoles Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conferenceKeys: The **confEntity** attribute MUST have the same value as that specified in the corresponding request.

The following rules apply to the **user** element:

- Exactly one **user** element SHOULD be present inside the **modifyUserRoles** body.
- The **entity** attribute of the **user** element MUST be the same as the **userEntity** attribute specified in the request.

The following rules apply to the **roles** element:

- The **roles** element MUST be present and listed exactly once.
- The **entity** element MUST be populated with the new role and exactly one **entity** element SHOULD be present.

The following example is a **modifyUserRoles** response body:

```
<modifyUserRoles>
  <conferenceKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user entity="sip:cathy@fabrikam.com">
    <roles>
      <entry>presenter</entry>
    </roles>
  </user>
</modifyUserRoles>
```

2.2.3.7 modifyConferenceLock Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI in the **To** header field of the INFO request.

The following rules apply to the **locked** element:

- The **locked** element MUST be included and it MUST specify the desired new lock state for the conference.
- A conference is considered locked whenever the locked value is set to "true" regardless of the value included in the **admission-policy** element.

The following rules apply to the **admission-policy element:**

- The **admission-policy** element is optional but, if included, the **autopromote** and **pstn-lobby-bypass** elements MUST also be included.
- The value of the **admission-policy** element MUST be set as defined in section 2.2.2.3.

The following rules apply to the **autopromote** element:

- The **autopromote** element is optional but, if included, the **admission-policy** and **pstn-lobby-bypass** elements MUST also be included.
- The value of the **autopromote** element MUST be set as defined in section 2.2.2.3.

The following rules apply to the **pstn-lobby-bypass** element:

- The **pstn-lobby-bypass** element is optional but, if included, the **admission-policy** and **autopromote** elements MUST also be included.
- The value of the **pstn-lobby-bypass** element MUST be set to either "true" or "false".

The following example is a **modifyConferenceLock** request body:

```
<modifyConferenceLock>
  <conferenceKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <locked>true</locked>
</modifyConferenceLock>

<modifyConferenceLock>
  <conferenceKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <locked>true</locked>
  <separator />
  <admission-policy>openAuthenticated</admission-policy>
  <autopromote>32768</autopromote>
<pstn-lobby-bypass>false</pstn-lobby-bypass>
</modifyConferenceLock>
```

2.2.3.8 modifyConferenceLock Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conference-info element: The **entity** attribute MUST be set to the corresponding **confEntity** attribute specified in the **conferenceKeys** element of the request.

locked element: The **locked** element MUST be present and MUST specify the new lock state of the conference.

The following rules apply to the **admission-policy element:**

- The **admission-policy** element is optional but MUST be included if the **modifyConferenceLock** request included an **admission-policy** element along with the **autopromote** and **pstn-lobby-bypass** elements.
- It MUST specify the new admission policy applied to the conference.

- The value of the **admission-policy** element MUST be set as defined in section 2.2.2.3.

The following rules apply to the **autopromote element:**

- The **autopromote** element is optional but MUST be included if the **modifyConferenceLock** request included an **autopromote** element along with the **admission-policy** and **pstn-lobby-bypass** elements.
- It MUST specify the new automatic promotion applied to the conference.
- The value of the **autopromote** element MUST be set as defined in section 2.2.2.3.

The following rules apply to the **pstn-lobby-bypass element:**

- The **pstn-lobby-bypass** element is optional but MUST be included if the modifyConferenceLock request included an **autopromote** element along with the admission-policy and **admission-policy** elements.
- It MUST specify the new automatic promotion applied to the conference.
- The value of the **pstn-lobby-bypass** element MUST be set to either "true" or "false".

The following example is a **modifyConferenceLock** response body:

```
<modifyConferenceLock>
  <conference-info
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <locked>true</locked>
</modifyConferenceLock>
<modifyConferenceLock>
  <conference-info
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <locked>true</locked>
  <separator />
  <admission-policy>openAuthenticated</admission-policy>
  <autopromote>32768</autopromote>
  <pstn-lobby-bypass>false</pstn-lobby-bypass>
</modifyConferenceLock>
```

2.2.3.9 deleteUser Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

mcuUri: The **mcuUri** attribute MUST be left empty.

The following rules apply to the **userKeys** element:

- The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI in the **To** header field of the INFO request.
- The **userEntity** attribute specifies the SIP URI of the user to be deleted.

endpointEntity: The **endpointEntity** element MUST be empty.

The following rules apply to the **clientReason** element:

- The **clientReason** element is optional.

- If the **clientReason** element is present, the value MUST be set to either "newPresenter", "participantEjected" or "connectedAtAnotherEndpoint".

The following example is a **deleteUser** request body:

```
<deleteUser>
  <userKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    userEntity="sip:cathy@fabrikam.com"/>
</deleteUser>
```

2.2.3.10 deleteUser Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conferenceKeys: The **confEntity** attribute MUST be set to the same value as the one specified in the corresponding request.

The following rules apply to the **user** element:

- The **user** element MUST be included in success responses.
- The **entity** attribute of the **user** element MUST be the same as the **userEntity** attribute supplied in the corresponding request.

The following example is a **deleteUser** response body:

```
<deleteUser>
  <conferenceKeys
    confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C" />
  <user entity="sip:cathy@fabrikam.com"/>
</deleteUser>
```

2.2.3.11 deleteConference Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rule applies.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI in the **To** header field of the INFO request.

The following example is a **deleteConference** request body:

```
<deleteConference>
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
</deleteConference>
```

2.2.3.12 deleteConference Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rule applies.

conference-info: The **entity** attribute MUST be set to the **confEntity** attribute specified in the corresponding request.

The following example is a **deleteConference** response body:

```
<deleteConference>
  <conference-info entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
</deleteConference>
```

2.2.3.13 addUser Dial-out Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI in the **To** header field of the INVITE request.

The following rules apply to the **user** element:

- Exactly one **user** element MUST be present inside the **addUser** body.
- The **entity** attribute of the **user** element MUST be a SIP URI.

The following rules apply to the **roles** element:

- Zero or one **roles** element MUST be present inside the **user** element.
- The **entry** element MUST be populated with the desired role.

The following rules apply to the **endpoint** element:

- Zero or one **endpoint** element MUST be present inside the **user** element.
- The **endpoint** element MUST specify a valid **entity** attribute. Implementations SHOULD use a GUID for this purpose.
- The **endpoint-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **refer-to-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **joining-method** element MUST be populated with the value "dialed-out".
- The **accessMethod** and **authMethod** elements MUST be left empty when sent from the client.

mcuUri: The **mcuUri** attribute MUST be populated with the MCU-Conference-URI of the MCU that needs to execute the **addUser** dial-out command.

Extensions to this specification can specify the semantics of other elements and attributes.

The following example is an **addUser** request document:

```
<addUser mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user entity="sip:cathy@fabrikam.com">
    <display-text>Cathy Baker</display-text>
    <roles>
```

```
      <entry>presenter</entry>
    </roles>
    <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
      <joining-method>dialed-out</joining-method>
      <!-- other extension elements can follow -->
    </endpoint>
  </user>
</addUser>
```

2.2.3.14 addUser Dial-out Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conferenceKeys: The **confEntity** attribute of this element MUST be set to the **confEntity** value specified in the corresponding **addUser** request.

The following rules apply to the **user** element:

- Exactly one **user** element MUST be present inside the **addUser** body. The **user** element MUST have the same **entity** attribute as the one specified in the **addUser** request.
- The **roles** element SHOULD be populated from the corresponding request.

The following rules apply to the **endpoint** element:

- Zero or one **endpoint** element SHOULD be present inside the **user** element.
- The **endpoint** element MUST specify a valid **entity** attribute. It MUST be the same as the one present in the corresponding request.
- The **endpoint-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **joining-method** element MUST be populated with the value "dialed-out".

Extensions to this specification can specify the semantics of other elements and attributes.

The following example is an **addUser** response document:

```
<addUser >
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user entity="sip:cathy@fabrikam.com">
    <display-text>Cathy Baker</display-text>
    <roles>
      <entry>presenter</entry>
    </roles>
    <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
      <joining-method>dialed-out</joining-method>
      <!-- other extension elements can follow -->
    </endpoint>
  </user>
</addUser>
```

2.2.3.15 addUser Dial-in Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI of the **To** header field of the INVITE request.

The following rules apply to the **user** element:

- One **user** element instance MUST be present inside the **addUser** body.
- The **entity** attribute of the **user** element MUST be a SIP URI.
- The **entity** attribute MUST be set to the C3P **from** attribute.

The following rules apply to the **roles** element:

- Zero or one **roles** element SHOULD be present inside the **user** element.
- The **roles** element MUST be populated with the desired role, which is either "presenter" or "attendee".
- If the **roles** element is absent, the default is "attendee".

The following rules apply to the **endpoint** element:

- Zero or one **endpoint** element SHOULD be present inside the **user** element.
- The **endpoint** element MUST specify a valid **entity** attribute. Implementations SHOULD use a GUID for this purpose.
- The **endpoint-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **refer-to-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **joining-method** element MUST be populated with the value "dialed-in".
- The **accessMethod** and **authMethod** elements MUST be left empty.

The following rules apply to the **mcuUri** attribute:

- The **mcuUri** attribute MUST be populated with the MCU-Conference-URI of the MCU that needs to execute the **addUser** dial-in command.
- Extensions to this specification can specify the semantics of other elements and attributes.

The following example is an **addUser** request document:

```
<addUser mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user entity="sip:cathy@fabrikam.com">
    <display-text>Cathy Baker</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
    <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
      <joining-method>dialed-in</joining-method>
```

```
      <!-- other extension elements can follow -->
    </endpoint>
  </user>
</addUser>
```

2.2.3.16 addUser Dial-in Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

conferenceKeys: This element MUST have the same values as those specified in the corresponding **addUser** request.

The following rules apply to the **user** element:

- Exactly one **user** element MUST be present inside the **addUser** body.
- The **user** element MUST have the same **entity** attribute specified in the **addUser** request.
- The **roles** element SHOULD be populated from the corresponding request.

The following rules apply to the **endpoint** element:

- Zero or one **endpoint** element SHOULD be present inside the **user** element.
- The **endpoint** element MUST specify a valid **entity** attribute. It MUST be the same as the one present in the corresponding request.
- The **endpoint-uri** attribute can be specified. If specified, it MUST be a "sip:" URI or a "tel:" URI.
- The **joining-method** element MUST be populated with the value "dialed-in".

The following rules apply to the **connection-info** element:

- Zero or more **entry** subelements SHOULD be present inside the **connection-info** element.
- Each **entry** element specifies a key-value property pair.

The following well-known keys are defined by this protocol. Their usage is discussed in section 3. They are case-insensitive.

- **Mcu-Server-Uri:** Specifies a SIP URI that can be used to send requests to this MCU. This SIP URI SHOULD be constructed in such a way that it can be used as the remote-target URI of SIP requests, as defined in [RFC3261].
- **Mcu-Conference-Uri:** Specifies the MCU-Conference-URI defined earlier.

The following well-known keys are defined by this protocol specifically for the Web Conferencing MCU. They are case-sensitive. For additional details about Web Conferencing MCU-specific concepts, see [MS-PSOM].

- **sAuthId:** An **xs:string** value composed of decimal digits and the letters "A", "B", "C", "D", "E", and "F". The Web Conferencing MCU MUST send this value, and it MUST be uniquely generated so that it is not duplicated in any other **addUser** dial-in response.
- **pwrpc.modes:** An **xs:string** value of either "tls" or "fwdtls". The Web Conferencing MCU MUST send this value to the client.

- **pwrpc.port:** An **xs:string** value containing a decimal string indicating the port number that the client can use to establish a PSOM connection to the Web Conferencing MCU. The Web Conferencing MCU MUST send this key to the client.
- **pwrpc.authPattern:** An **xs:string** value that indicates the authentication mechanism supported by the Web Conferencing server. This MUST be set to the string "<sAuthId>".
- **numberOfProxies:** An **xs:string** value containing a decimal string indicating how many proxy servers are being communicated from the Web Conferencing MCU to the client. This value is used by the client to look for specific keys of the form "proxy[N].FQDN" and **proxy[N].Port**. The Web Conferencing MCU MUST send this key to the client if it also sent the **pwrpc.modes** key with a value of "fwdtls".
- **proxy[N].FQDN:** Any key with a name of the form "proxy[N].FQDN", where **N** is a normalized decimal string less than the value of the **numberOfProxies** key sent to the client, and greater than or equal to zero. Any key of this form has an **xs:string** value indicating the **fully qualified domain name (FQDN)** of a proxy server capable of establishing a PSOM connection to the Web Conferencing MCU on behalf of the client. The Web Conferencing MCU MUST send a key of this form for every value of **N** from zero up to, but not including, the value of the **numberOfProxies** key.
- **proxy[N].Port:** Any key with a name of the form "proxy[N].Port" where **N** is a normalized decimal string less than the value of the **numberOfProxies** key sent to the client, and greater than or equal to zero. Any key of this form has an **xs:string** value containing a decimal string indicating the port number of a proxy server capable of establishing a PSOM connection to the Web Conferencing MCU on behalf of the client. The Web Conferencing MCU MUST send a key of this form for every value of **N** from zero up to, but not including, the value of the **numberOfProxies** key.
- **pwrpc.pwsURI:** An **xs:string** value containing a URI for the Web Conferencing MCU that the client can use to establish a PSOM connection. The Web Conferencing MCU MUST send this key to the client if it also sent the **pwrpc.modes** key with a value of "tls".
- **alternativeName:** An **xs:string** value that can be used in place of the **X.509** certificate subject for TLS negotiation. If the Web Conferencing MCU has sent the **pwrpc.modes** key with a value of "fwdtls", the client MUST ignore this value. Otherwise, the Web Conferencing MCU MUST send this key, and the client MUST establish TLS negotiation using this value as the certificate subject for verification.

Extensions to this specification can specify the semantics of other elements and attributes.

The following example is an **addUser** response document:

```
<addUser mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user entity="sip:cathy@fabrikam.com">
  <display-text>Cathy Baker</display-text>
    <roles>
      <entry>presenter</entry>
    </roles>
    <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
      <joining-method>dialed-in</joining-method>
      <!-- other extension elements can follow -->
    </endpoint>
  </user>
  <connection-info>
```

```
      <entry>
        <key>Mcu-Server-Uri</key>
        <value>sip:mcu.domain.com:5061;transport=tls</value>
      </entry>
      <entry>
        <key>Mcu-Conference-Uri</key>
        <value> sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:focus:id:5D3747C</value>
      </entry>
    </connection-info>
  </addUser>
```

2.2.3.17 getConference Request Document

This section follows the product behavior described in endnote <17>.

In addition to the syntax rules given in section 2.2.1 for C3P requests, the request follows the same syntax as the **getConference** request sent to the Focus Factory, as specified in [MS-CONFPRO].

The difference between the **getConference** request sent to the Focus Factory and the one sent to the focus is that the one sent to the Focus Factory retrieves the static provisioning information of the conference, while the one sent to the focus retrieves the current roster state of an active conference.

2.2.3.18 getConference Response Document

This section follows the product behavior described in endnote <18>.

In addition to the syntax rules given in section 2.2.1 for a C3P response, the following additional rules apply.

The response contains the current roster state of the active conference in the conference roster document format, as described in section 2.2.4.

The difference between the roster state retrieved using the SUBSCRIBE and **getConference** is that the **getConference** response contains information which is potentially encrypted using the key provided in the **getConference** request.

The response to the **getConference** request sent to the focus might contain sensitive information such as the conference key and the encrypted conferencing join web URL. The sensitive information is encrypted using the public key provided by the client. If the client does not provide a public key, the response does not contain any sensitive information.

2.2.3.19 setLobbyAccess Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests<19>, the following additional rules apply.

conferenceKeys: The conference-URI specified in the **confEntity** attribute of the **conferenceKeys** element MUST match the SIP URI of the **To** header field of the INVITE request.

The following rules apply to the **userEntity** element:

- One or more **userEntity** element instances MUST be present inside the **setLobbyAccess** body.
- The value of a **userEntity** element MUST be set to a SIP URI.

The following rules apply to the **access** element:

- One **access** element instance MUST be present inside the **setLobbyAccess** body.
- The value of the **access** element MUST be set to "granted" or "denied".

2.2.3.20 setLobbyAccess Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses<20>, the following additional rules apply.

conferenceKeys: This element MUST have the same values as those specified in the corresponding **setLobbyAccess** request.**status** element:

- One status element MUST be included for each user admitted or denied form the conference in the setLobbyAccess request.
- A **status** element MUST have one **reason** attribute set to "success", "conferenceFull", "userDoesntExist" or "alreadyGranted".
- A **status** element MUST have one **userEntity** element.

2.2.3.21 modifyEndpoint Request Document

In addition to the syntax rules given in section 2.2.1 for C3P requests, the following additional rules apply.

The following rules apply to the **endpointKeys** element:

- One **endpointKeys** element instance MUST be present inside the **modifyEndpoint** body.
- The conference-URI specified in the **confEntity** attribute of the **endpointKeys** element MUST match the SIP URI of the **To** header field of the INVITE request.

The following rules apply to the **endpoint** element:

- One **endpoint** element instance MUST be present inside the **modifyEndpoint** body.
- The value of the **entity** attribute MUST have the same value as that specified in the **endpointEntity** attribute in **endpointKeys** in the same request.

The following example is a **modifyEndpoint** request document:

```
<modifyEndpoint>
    <endpointKeys confEntity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH"
userEntity="sip:bob@fabrikam.com" endpointEntity="{F95CF5F8-2A22-4C1F-B65E-2014CF07BD11}"/>
    <ci:endpoint xmlns:ci="urn:ietf:params:xml:ns:conference-info" entity="{F95CF5F8-2A22-
4C1F-B65E-2014CF07BD11}">
      <!-- other extension elements can follow -->
    </ci:endpoint>
  </modifyEndpoint>
```

2.2.3.22 modifyEndpoint Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

The **modifyEndpoint** command modifies the current **endpoint** only. There is no requirement to identify which user has been modified in the response.

The following example is a **modifyEndpoint** response document:

```
<modifyEndpoint/>
No child element is needed for the modifyEndpoint element in response.
```

2.2.3.23 modifyConferenceAnnouncements Request Document

2.2.3.24 modifyConferenceAnnouncements Response Document

2.2.3.25 modifyConference Request Document

In addition to the syntax rules given in section section 2.2.1 for C3P requests, the following additional rules apply.

- The **mcuUri** attribute MUST be non-empty.
- The **conference-info** element MUST contain only one **conference-view** element, and no other children.
- The **conference-view** element MUST contain only one **entity-view** element, and no other children.
- The **entity** attribute of the **entity-view** element MUST match the value of the **modifyConference**.**mcuUri** attribute.

2.2.3.26 modifyConference Response Document

In addition to the syntax rules given in section 2.2.1 for C3P responses, the following additional rules apply.

- The **conference-info**.**entity** attribute MUST be set to the **confEntity** attribute specified in the corresponding request.
- The conference-info element MUST have state set to partial.
- The conference-info element MUST contain no other children.

2.2.4 Conference Roster Document Format

The constructed conference roster MUST be a valid XML document conforming to the conference schema defined in section 7.

The format for constructing the relevant subelements of the conference roster is given in the following subsections.

Unless specified otherwise, if a roster element has a **state** attribute associated with it, rules for constructing it follow [RFC4575] section 4.4.

2.2.4.1 conference-description Element Syntax

The **conference-description** element MUST be constructed with the **conf-uris** element populated using the rules specified in section 2.2.2.4.

2.2.4.2 Participant user Element Syntax

The focus MUST generate a **user** element for each participant in the conference using the following rules:

- The **user** element MUST be valid according to the application/conference-info+xml syntax rules.
- The focus MUST populate the **roles** subelement with the granted role of the user.
- The focus SHOULD populate the **display-text** subelement with the display name of the user as obtained from the corresponding **addUser** request or using some other directory lookup.
- The focus MUST add an **endpoint** element for each focus-connected **endpoint (5)** using the focus **endpoint** element syntax rules defined in section 2.2.4.2.1.
- The focus MUST add an **endpoint** element for each MCU-connected endpoint (5) using the MCU **endpoint** element syntax rules defined in section 2.2.4.2.2.
- The focus MAY preserve all other elements, except the **endpoint** elements, and attributes specified in the **user** element of the **addUser** request and add it to the **user** element it constructs for notification purposes.

An example of a **user** element is as follows:

```
<user entity="sip:alice@fabrikam.com" state="full">
  <display-text>Alice Gates</display-text>
  <roles>
    <entry>presenter</entry>
  </roles>
  <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}" session-type="focus" endpoint-
uri="sip:alice@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu">
    <status>connected</status>
  </endpoint>
</user>
```

2.2.4.2.1 Focus endpoint Element Syntax

The focus MUST generate an **endpoint** element for each focus-connected endpoint (5) of a conference participant using the following rules:

- The **entity** attribute of the **endpoint** element MUST have the same value that was specified in the corresponding **addUser** request used by the client at the time it connected to the focus.
- The **session-type** attribute of the **endpoint** element MUST be set to the value "focus".
- The focus SHOULD populate the **endpoint-uri** attribute with an appropriate **endpoint URI**. Normally, for GRUU-based clients, this is the **endpoint GRUU** specified in the **Contact** header field of the corresponding INVITE request. If an **endpoint GRUU** is available from the corresponding INVITE request, it SHOULD be preferred over any **endpoint-uri** value specified in the **addUser** request itself.
- The **status** subelement MUST be set to either the value "connected" or the value "on-hold". The value "connected" indicates a user who has been admitted to the conference. The value "on-hold" indicates a user placed in the lobby.
- The focus MAY preserve all other elements and attributes specified in the **endpoint** element of the **addUser** request and add it to the **endpoint** element it constructs for notification purposes.

An example of a focus **endpoint** element is as follows:

```
<endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}" session-type="focus" endpoint-
uri="sip:alice@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu">
  <status>connected</status>
</endpoint>
```

2.2.4.2.2 MCU endpoint Element Syntax

The focus MUST generate an **endpoint** element for each MCU-connected endpoint (5) of a conference participant using the following rules:

- The focus MUST preserve all elements and attributes of the **endpoint** element published by the MCU and use it to construct the **endpoint** element.
- The **session-type** attribute of the **endpoint** element MUST be set according to the semantics defined for the **session-type** attribute defined in section 2.2.2.6, thereby overriding any **session-type** attribute set by the MCU.
- The **status** element MUST be populated with an appropriate value as defined in the xml schema. Only the "connected" value SHOULD be used in notifications.

MCU-specific extensions to this specification can specify additional syntax rules for generating MCU **endpoint** elements.

2.2.4.3 conference-view Element Syntax

In full conference notifications, the focus MUST generate a **conference-view** element using the following rules:

- The focus MUST add an **entity-view** element for itself using the focus **entity-view** element syntax rules that follow.
- The focus MUST add any **entity-view** elements published by MCUs in the conference to the roster document.

2.2.4.3.1 Focus entity-view Element Syntax

The focus MUST construct and include an **entity-view** element for itself in the conference roster. The constructed **entity-view** element MUST conform to the following additional rules:

- The **entity-state** element MUST be populated in "full" **entity-view** notifications.
- The **entity-state** element MUST list the current lock state of the conference.

Following is an example of an **entity-view** element for the focus.

```
<entity-view state="full" entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:BE92">
  <entity-state>
    <locked>false</locked>
  </entity-state>
</entity-view>
```

2.2.5 MCU Conference Roster Document Format

The conference roster document published by an MCU to the focus MUST conform to the following rules, in addition to being a valid application/conference-info+xml document, as defined in section 8.

The following rules apply to the **endpoint** element:

- Zero or one **endpoint** element MUST be listed inside a **user** element.
- The **state** attribute of the **endpoint** element MUST be set to "full" or "deleted".

The following rules apply to the **conference-view** element:

- Exactly one **entity-view** entry SHOULD be present inside the **conference-view** element. This entry MUST have the entity URI set to the MCU-Conference-URI.
- The size of the XML fragment for each child element of **entity-view** MUST NOT exceed 2048 bytes.

2.2.6 HTTP Request and Response

This section follows the product behavior described in endnote <21>.

2.2.6.1 HTTP Request

The HTTP request from the client side browser will hit the Join Manager, which would first detect the supported browser versions. If the type of Browser and the Version is supported then the Join Manager will respond back with a HTML document containing a Java-Script that needs to run on the client side.

2.2.6.2 HTTP Response

The Join Manager responds back to the client with a HTML document that contains the Java-Script described in section 2.2.6.4. The Join Manager generates an XML document body that MUST conform to the response format defined in section 2.2.6.3 and places this as one of the parameters in the Java-Script.

2.2.6.3 ocsmeet Document Format

The ocsmeet document format specifies the document format of the XML document body sent back in the HTTP response that Join Manager sends to the browser, and this is embedded as a String in the Java-Script present in the document. A well-formed XML document String MUST be valid XML, as specified in [XML10], conformant to the schema defined in section 6.

The cardinality of each element is specified in the XML schema using standard **minOccurs** and **maxOccurs** XSD conventions, unless explicitly specified otherwise.

Unless otherwise specified, implementations MUST ignore elements and attributes that they cannot parse.

conf-uri: This element MUST be set to the parsed conference URI (conference-URI) or an empty string if the conferencing join web URL could not be parsed.

server-time: This element MUST be set to the time in milliseconds taken by Join Manager to process the request.

original-incoming-url: This element MUST be set to the original URL that the user requested.

conf-key: This element MUST be set to the conference key of the conference. The following example is a content body:

```
<conf-info xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc">
  <conf-uri>sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:MJMAY7RF</conf-uri>
  <server-time>1.0001</server-time>
  <original-incoming-url>https://www.fabrikam.com/meet/bob/MJMAY7RF</original-incoming-url>
  <conf-key>MJMAY7RF</conf-key>
</conf-info>
```

2.2.6.4 Simple Join Java-Script

The following examples are the HTML document and the Java-Script code that the Join Manager sends back to the client in response to the HTTP request sent on the simple join URL.

The following example is the main HTML document:

```
<html xmlns="http://www.w3.org/1999/xhtml" class="reachJoinHtml">
<head>
    <title>Microsoft Lync 2010</title>
    <script type="text/javascript">
        var reachURL =
"https://fabrikam.com/Reach/Client/WebPages/ReachJoin.aspx?xml=PD94bWwgdmVyc2lvbj0iMS4wIiBlbm
NvZGluZz0idXRmLTgiPz48Y29uZi1pbmZvIHhtbG5zOnhzaT0iaHR0cDovL3d3dy53My5vcmcvMjAwMS9YTUxTY2hlbWE
taW5zdGFuY2UiIHhtbG5zOnhzZD0iaHR0cDovL3d3dy53My5vcmcvMjAwMS9YTUxTY2hlbWEiIHhtbG5zPSJodHRwOi8v
c2NoZW1hcy5taWNyb3NvZnQuY29tL3J0Yy8yMDA5LzA1L3NpbXBsZWpvaW5jb25mZG9jIj48Y29uZi1cmk-
c2lwOnN0ZXZlY2hAbWljcm9zb2Z0LmNvbTtncnV1O29wYXF1ZT1hcHA6Y29uZjpmb2N1czppZDpJQlVCU0s3VzwvY29uZ
i11cmk-PHNlcnZlci10aW1lPjE8L3NlcnZlci10aW1lPjxvcmlnaW5hbC1pbmNvbWluZy11cmw-
aHR0cHM6Ly9sc2xtODQubWVldC5taWNyb3NvZnQuY29tL2llZXQvc3RldmVjaC9JQlVCU0s3Vzwvb3JpZ2luYWwtaW5jb
21pbmctdXJsPjxjb25mLWtleT5JQlVCU0s3VzwvY29uZi1rZXk-PC9jb25mLWluZm8-";

/* Escaped ocsmeet XML document Body */
        var escapedXML = "&lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-
8&quot;?&gt;&lt;conf-info xmlns:xsi=&quot;http://www.w3.org/2001/XMLSchema-instance&quot;
xmlns:xsd=&quot;http://www.w3.org/2001/XMLSchema&quot;
xmlns=&quot;http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc&quot;&gt;&lt;conf-
uri&gt;sip:bob@fabricam.com;gruu;opaque=app:conf:focus:id:IBUBSK7W&lt;/conf-
uri&gt;&lt;server-time&gt;1&lt;/server-time&gt;&lt;original-incoming-url&gt;
https://fabrikam.com/meet/bob/IBUBSK7W &lt;/original-incoming-url&gt;&lt;conf-
key&gt;IBUBSK7W&lt;/conf-key&gt;&lt;/conf-info&gt;";

       var showJoinUsingLegacyClientLink = "True";
        var urlCrackingError = "False";
        var currentLanguage = "en-US";
    </script>

/* Including Java-Script code */
/*
  Utilities.js -
  Contains other methods such methods that verify if the browser version is supported*/
    <script type="text/javascript" src="/meet/JavaScript/Utilities.js"></script>

/*
  PluginLoader.js -
```

```
  Contains classes and methods that contain configuration and loading of Active-X control or
Firefox Plug-in
  */
   <script type="text/javascript" src="/meet/JavaScript/PluginLoader.js"></script>

/*
  Launch.js -
  Contains the OnLoad() method and the related code that invokes utility functions, does
supported browser checks, loads available Active-X Control or Firefox plug-ins, shortlists
the candidate and invokes Launch API on it. It would also contain fall back code to launch
the web offering, provide links to alternate clients, and provide a help link. It will also
Close the browser window in case of a successful launch.
  */
    <script type="text/javascript" src="/meet/JavaScript/Launch.js"></script>

    <link rel="Stylesheet" type="text/css" href="/meet/Resources/ReachClient.css" />
</head>
<body onload="mainWindow.OnLoad();" class="reachJoinBody">

/* HTML BODY goes here, typically there is an IFrame here and Java-Script populates the
content */

</body>
</html>
```

The following example is PluginLoader.js:

```
var InstalledClient = new Object( );
InstalledClient.OC = 0;
InstalledClient.Samara = 1;
InstalledClient.AOC = 2;

//
// Name = Registered name of the Plugin
//
PluginConfigOC =
{
    IE:
    {
        Version_Name:         "CommunicatorMeetingJoinAx.JoinManager"
    },

    FF:
    {
        Version_CLSID:        "application/vnd.microsoft.communicator.ocsmeeting"
    }
}

PluginConfigSamara =
{
    IE:
    {
        Version_Name:         "AttendantConsoleMeetingJoinAx.JoinManager"
    },

    FF:
    {
        Version_CLSID:        "application/vnd.microsoft.attendantconsole.ocsmeeting"
```

```
    }
}

PluginConfigAOC =
{
    IE:
    {
        Version_Name:         "CommunicatorAttendeeMeetingJoinAx.JoinManager"
    },

    FF:
    {
        Version_CLSID:        "application/vnd.microsoft.communicatorattendee.ocsmeeting"
    }
}

// Format the string
function StringFormat()
{
    var argumentList = StringFormat.arguments;
    var argsLen = argumentList.length;

    if(!argsLen)
    {
        return "";
    }
    else
    {
        var newString = argumentList[0];
        var tmp = argsLen - 1;

        for(var i = 0; i < tmp; ++i)
        {
            // '$' is a sepcial character in regular expression
            // if there is '$_' in string, it will break this function in IE
            // Replace it with '$$'
            newString = newString.replace(new RegExp("%" + i, "g"), (argumentList[i + 1] +
"").replace(/\$/g, "$$$$"));
        }

        return newString;
    }
}

function CreateNodeOutside(nodeType, nodeId)
{
    var node = document.createElement(nodeType || "DIV");
    document.body.appendChild(node);

    node.style.position = "absolute";
    node.style.width = "1px";
    node.style.height = "1px";
    node.style.left = "-100px";
    node.style.top = "0px";
    node.style.overflow = "hidden";
    node.style.visibility = "hidden";

    if (nodeId != null)
        node.id = nodeId;
```

```
    return node;
}

function GetBrowserTag()
{
    var browserTag = "";
    if (isIE())
    {
        browserTag = "IE";
    }
    else
    {
        // Treat all non-IE browsers the same as Firefox
        // the reason being, Opera/Chrome/Safari all look
        // for plugins in the Mozilla folder and load them
        // even if they were not installed for this browser.
        browserTag = "FF";
    }

    return browserTag;
}

function GetConfigForClient(installedClient, configTag)
{
    var config;

    switch (installedClient)
    {
        case InstalledClient.OC:
            config = PluginConfigOC[configTag];
            break;
        case InstalledClient.Samara:
            config = PluginConfigSamara[configTag];
            break;
        case InstalledClient.AOC:
            config = PluginConfigAOC[configTag];
            break;
        default:
            break;
    }

    return config;
}

//
// A general component used to load any plugin into browser (IE & Firefox)
//
function PluginLoader()
{
    this._isIE = null;
    this._name = null;
    this._clsname = null;
    this._clsid = null;
    this._initProps = null;

    // the loaded plugin object
    this._pluginInstance = new Object();
}
```

```
PluginLoader.IdPrefix = "_ucclient_plugin_";
PluginLoader.TagHtmlTemplateIE = "<object classid='%0'%1></object>";
PluginLoader.TagHtmlTemplateFF = "<embed type='%0'%1></embed>";

//
// Initialize it with plugin information, such as name and id.
//
PluginLoader.prototype.Initialize = function(name, clsname, clsid, initProps)
{
    if (this._pluginInstance.object)
    {
        return;
    }

    this._isIE = (document.all != null);
    this._name = name;
    this._clsname = clsname;
    this._clsid = clsid;
    this._initProps = initProps;
}

//
// Load plugin
//
PluginLoader.prototype.LoadPlugin = function()
{
    if (this._pluginInstance.object)
    {
        return this._pluginInstance;
    }

    this._CreatePlugin();
    return this._pluginInstance;
}

//
// UnLoad plugin
//
PluginLoader.prototype.UnloadPlugin = function()
{
    if (!this._pluginInstance.object)
    {
        return;
    }

    //if it created a dom object, remove it.
    if ((this._clsname == null) && (this._clsid != null))
    {
        try
        {
            var createdContainerNode = this._pluginInstance.object.parentNode;
            Assert(createdContainerNode != null && createdContainerNode.parentNode ==
document.body, "Plugin parent node is not correct");
            document.body.removeChild(createdContainerNode);
        }
        catch( ex )
        {
            // Error in removing DOM object.
```

```
            return;
        }
    }

    this._pluginInstance.object = null;
    return;
}

PluginLoader.prototype._IsPluginAvailable = function()
{
    var isAvailable = false;

    // Check for the availability of the Plugin
    // before actually trying to load the Plugin
    // in case of non-IE browsers: this prevents
    // the "gold bar" indicating "Additional plugins
    // are required to display all the media on the
    // page" that comes in Firefox and other browsers.
    if (!this._isIE)
    {
        var mimetype = navigator.mimeTypes[this._clsid];

        if (mimetype)
        {
            var enabled = mimetype.enabledPlugin;

            if (enabled != null)
            {
                isAvailable = true;
            }
        }
    }

    return isAvailable;
}

//
// Create the plugin object
//
PluginLoader.prototype._CreatePlugin = function()
{
    if (this._clsname)
    {
        try
        {
            if (this._isIE)
            {
                this._pluginInstance.object = new ActiveXObject(this._clsname);
            }
            else
            {
                this._pluginInstance.object = null;
            }
        }
        catch( ex )
        {
            // Error creating ActiveX object.
        }
    }
```

```
    else if (this._clsid)
    {
        if (!this._IsPluginAvailable())
        {
            return;
        }

        var propStr = "";
        if (this._initProps)
        {
            var props = this._initProps;
            for (var key in props)
            {
                propStr += " " + key + "='" + props[key] + "'";
            }
        }

        var tagHtml = StringFormat(this._isIE ? PluginLoader.TagHtmlTemplateIE :
PluginLoader.TagHtmlTemplateFF, this._clsid, propStr);

        try
        {
            var containerNode = CreateNodeOutside("DIV");
            containerNode.innerHTML = tagHtml;
            var node = containerNode.firstChild;
            node.id = PluginLoader.IdPrefix + this._name;
            this._pluginInstance.object = node;
        }
        catch( ex )
        {
            // Error creating DOM object.
        }
    }
}
```

The following example is Launch.js:

```
// Constants
var MINIMUM_CLIENT_VERSION_FOR_W14 = '4.0';
var REDIRECT_TO_REACH_SL_OVERRIDE = 'sl=';

// Plugin Loaders for each client
var pluginLoaderOC = null;
var pluginLoaderSamara = null;
var pluginLoaderAOC = null;

// Plugin Objects for each client
var pluginObjectOC = null;
var pluginObjectSamara = null;
var pluginObjectAOC = null;

// Version info for each client
var majorVersionOC = null;
var majorVersionSamara = null;
var majorVersionAOC = null;

var ResourceURL = "/meet/JavaScriptResourceHandler.ashx?language=";
```

```
var loading = "true";

//
// Initialize resource strings before trying to use them.
//
var txt_logoLabel = "";
var txt_languageSettingsLabel = "";
var msgBadMeetingURL = "";
var txt_launchRichClientHeaderLabel = "";
var txt_launchRichClientTextLabel = "";
var txt_unableToJoinLabel = "";
var txt_onlineHelpLink = "";
var txt_copyRightTextLabel = "";
var textDirection="";
var txt_joinUsingReachLink = "";
var txt_troubleshootingLabel = "";
// Add any new resource strings here.

function MainForm()
{
    this.connection = new ConnectionObject();
}

MainForm.prototype.GetMajorVersion = function(fullVersion)
{
    if (fullVersion == null)
    {
        return null;
    }

    var majorVersion = "";

    // Full version number string format: "4.0.1234.5678"
    // Major version number string format: "4.0"
    // Ignore any non-numeric characters preceding the version number start
    if (/(\D*)(\d\.\d)(\.\d{4}\.\d+)/.test(fullVersion)) {
        majorVersion = RegExp.$2;
    }
    return majorVersion;
}

//
// Converts
// '&lt;' -> '<'
// '&gt;' -> '>'
// '&amp;' -> '&'
// '&apos;' -> '''
// '&quot;' -> '"'
// for text data in XML format
//
MainForm.prototype.UnEscapeXML = function(str)
{
    if(str == null)
    {
        return "";
    }

    str = str.replace(/&amp;/ig, "&");
    str = str.replace(/&lt;/ig, "<");
```

```
    str = str.replace(/&gt;/ig, ">");
    str = str.replace(/\&apos;/ig, "'");
    str = str.replace(/\&quot;/ig, "\"");

    return str;
}

MainForm.prototype.OnLoad = function()
{
    this.UpdateSelectedLanguage(currentLanguage);

    // Fetch the current language resources.
    this.GetUpdatedResources(currentLanguage);
}

MainForm.prototype.OnLoadComplete = function()
{
    loading = "false";

    //
    // Make sure URL was cracked
    //
    var error = urlCrackingError.toLowerCase();
    if (error == "true") {
        this.ShowError(msgBadMeetingURL);
        return;
    }

    //
    // Parse URL parameters
    //
    var urlParam = getUrlParameters();
    if (urlParam.search(REDIRECT_TO_REACH_SL_OVERRIDE) != -1)
    {
        // User explicitly wants to use reach to join the meeting,
        // redirect to Reach.
        this.RedirectToReach();
        return;
    }

    //
    // Initialize version info of all clients
    //
    this.InitializeVersionInformationOfAllClients();

    //
    // Now that the clients and their version
    // info are installed, figure out which client
    // to launch and try to launch it.
    //
    var launched = "false";

    try
    {
        if ((majorVersionOC != null) && (majorVersionOC == MINIMUM_CLIENT_VERSION_FOR_W14))
        {
            // Launch using OC plugin
            // Note: OC Plugin launches whichever client ran last (OC/Samara)
```

```
            // Unescape the XML before passing it on to the client
            pluginObjectOC.object.LaunchUCClient(this.UnEscapeXML(escapedXML));
            launched = "true";
            this.DisplayInstalledClientLaunchedPage("oc");
        }
        else if ((majorVersionSamara != null) && (majorVersionSamara ==
MINIMUM_CLIENT_VERSION_FOR_W14))
        {
            // Launch using Samara plugin
            // Note: Samara Plugin launches whichever client ran last (OC/Samara)

            // Unescape the XML before passing it on to the client
            pluginObjectSamara.object.LaunchUCClient(this.UnEscapeXML(escapedXML));
            launched = "true";
            this.DisplayInstalledClientLaunchedPage("oc");
        }
        else if ((majorVersionAOC != null) && (majorVersionAOC ==
MINIMUM_CLIENT_VERSION_FOR_W14))
        {
            // Launch using AOC plugin
            // Note: AOC Plugin launches only AOC

            // Unescape the XML before passing it on to the client
            pluginObjectAOC.object.LaunchUCClient(this.UnEscapeXML(escapedXML));
            launched = "true";
            this.DisplayInstalledClientLaunchedPage("aoc");
        }
    } catch (ex) {
        // Failed to launch client
        launched = "false";
    }

    //
    // Unload all Plugins
    //
    this.UnloadAllPlugins();

    if (launched == "false") {
        // Either it failed to detect an installed client OR
        // the installed client is not the right version OR
        // launching the installed client failed.
        // In any case, redirect to Reach.
        this.RedirectToReach();
    }
    else {
        // Managed to successfully launch the client to join
        // the meeting, try to close the browser window on the
        // browsers that support closing it silently.
        this.ClosePageOnSupportedBrowsers();
    }
}

MainForm.prototype.InitializeVersionInformationOfAllClients = function()
{
    try
    {
        navigator.plugins.refresh();
    } catch (ex) {
        // no need to do anything here
```

```
    }

    // First determine the browser tag
    // needed to look up the config for
    // the clients
    var configTag = GetBrowserTag();

    this.InitializeVersionInformationForOC(configTag);
    this.InitializeVersionInformationForSamara(configTag);
    this.InitializeVersionInformationForAOC(configTag);
}

MainForm.prototype.InitializeVersionInformationForOC = function(configTag)
{
    var pluginConfig = GetConfigForClient(InstalledClient.OC, configTag);
    if (!pluginConfig)
    {
        return;
    }

    pluginLoaderOC = new PluginLoader();
    pluginLoaderOC.Initialize(
                "OC",
                pluginConfig.Version_Name,
                pluginConfig.Version_CLSID,
                null);

    pluginObjectOC = pluginLoaderOC.LoadPlugin();
    if (!pluginObjectOC.object)
    {
        return;
    }

    try
    {
        var fullVersionOC = pluginObjectOC.object.GetVersionString();
        majorVersionOC = this.GetMajorVersion(fullVersionOC);
    } catch (ex) {
        // Unable to get version details, continue anyway
    }
}

MainForm.prototype.InitializeVersionInformationForSamara = function(configTag)
{
    var pluginConfig = GetConfigForClient(InstalledClient.Samara, configTag);
    if (!pluginConfig)
    {
        return;
    }

    pluginLoaderSamara = new PluginLoader();
    pluginLoaderSamara.Initialize(
                "Samara",
                pluginConfig.Version_Name,
                pluginConfig.Version_CLSID,
                null);

    pluginObjectSamara = pluginLoaderSamara.LoadPlugin();
    if (!pluginObjectSamara.object)
```

```
    {
        return;
    }

    try
    {
        var fullVersionSamara = pluginObjectSamara.object.GetVersionString();
        majorVersionSamara = this.GetMajorVersion(fullVersionSamara);
    } catch (ex) {
        // Unable to get version details, continue anyway
    }
}

MainForm.prototype.InitializeVersionInformationForAOC = function(configTag)
{
    var pluginConfig = GetConfigForClient(InstalledClient.AOC, configTag);
    if (!pluginConfig)
    {
        return;
    }

    pluginLoaderAOC = new PluginLoader();
    pluginLoaderAOC.Initialize(
                "AOC",
                pluginConfig.Version_Name,
                pluginConfig.Version_CLSID,
                null);

    pluginObjectAOC = pluginLoaderAOC.LoadPlugin();
    if (!pluginObjectAOC.object)
    {
        return;
    }

    try
    {
        var fullVersionAOC = pluginObjectAOC.object.GetVersionString();
        majorVersionAOC = this.GetMajorVersion(fullVersionAOC);
    } catch (ex) {
        // Unable to get version details, continue anyway
    }
}

MainForm.prototype.UnloadAllPlugins = function()
{
    // Unload all Plugins so that references to the DLLs are released

    // Unload OC Plugin
    if (pluginLoaderOC != null)
    {
        pluginLoaderOC.UnloadPlugin();
    }

    // Unload Samara Plugin
    if (pluginLoaderSamara != null)
    {
        pluginLoaderSamara.UnloadPlugin();
    }
```

```
    // Unload AOC Plugin
    if (pluginLoaderAOC != null)
    {
        pluginLoaderAOC.UnloadPlugin();
    }
}

MainForm.prototype.ShowError = function(errorMessage)
{
    document.getElementById("joinLauncherErrorDiv").style.display = "block";

    document.getElementById("errorTextLabel").innerHTML = errorMessage;
}

MainForm.prototype.DisplayInstalledClientLaunchedPage = function(launchedClient)
{
    // Show the Rich client launched text, and the rest.
    document.getElementById("launchRichClientDiv").style.display = "block";

    // Show the Contact Support text with Reach option as well as the link to
    // launch Reach.
    document.getElementById("contactSupportLabelWithReachOption").style.display = "block";
    document.getElementById("launchReachLink").style.display = "block";

    // Hide the Contact Support text without Reach option.
    document.getElementById("contactSupportLabelWithoutReachOption").style.display = "none";

    // Tell Reach which client was launched so the Reach Landing
    // Page can be more intelligent about the links it displays
    // to the user.
    var fullReachURL = reachURL + "&launched=" + launchedClient;

    // Update the link for launching Reach.
    document.getElementById("launchReachLink").href = fullReachURL;

    // Hide the iFrame used to launch Reach and expand it to 100%
    // so it occupies the entire area of the page.
    document.getElementById("launchReachDiv").style.display = "none";
    document.getElementById("launchReachFrame").style.width="0px";
    document.getElementById("launchReachFrame").style.height="0px";
    document.getElementById("launchReachFrame").style.display="";
}

MainForm.prototype.RedirectToReach = function()
{
    // Launch Reach from an iFrame, so that the
    // URL does not change in the Browser window.

    // Hide the main table that has all the page content.
    document.getElementById("mainTable").style.display = "none";

    // Un-hide the iFrame used to launch Reach.
    document.getElementById("launchReachDiv").style.display = "block";
    document.getElementById("launchReachFrame").style.width="100%";
    document.getElementById("launchReachFrame").style.height="100%";
    document.getElementById("launchReachFrame").style.display="block";

    // Hide the scrollbar for the main window, because the iFrame
    // will have its own scrollbar and that is the only one that
```

```
    // is relevant.
    window.document.body.style.overflow = "hidden";

    // Launch Reach by updating the src of the iFrame.
    document.getElementById("launchReachFrame").src = reachURL;
}

MainForm.prototype.ClosePageOnSupportedBrowsers = function()
{
    // Inspect browser version and take appropriate close action
    if ( isIE8() || isIE7() )
    {
        // set opener as self by invoking open - for IE7 and IE8 browsers
        window.open("", "_self");

        // close the browser window
        window.close();
    }
    else if ( isIE6() )
    {
        // set the opener as self by directly setting the value of
        // self.opener - for IE6 browser only
        self.opener = this;

        // close the browser window
        window.close();
    }
    // Do not close the window on other browsers now.
}

MainForm.prototype.LanguageSelectionChanged = function()
{
    var languageSelector = document.getElementById("languageSelectCmb");

    if (languageSelector.selectedIndex != -1)
    {
        var newLanguage = languageSelector.options[languageSelector.selectedIndex].value;
        if (newLanguage.toLowerCase() != currentLanguage.toLowerCase())
        {
            this.UpdateSelectedLanguage(newLanguage);
            this.GetUpdatedResources(newLanguage);
            currentLanguage = newLanguage;
        }
    }
}

MainForm.prototype.UpdateSelectedLanguage = function(language)
{
    var languageSelector = document.getElementById("languageSelectCmb");

    var index = -1;
    for (i=0; i < languageSelector.options.length; i++)
    {
        if (languageSelector.options[i].value.toLowerCase() == language.toLowerCase())
        {
    index = i;
    break;
}
    }
```

```
    languageSelector.selectedIndex = index;
}

MainForm.prototype.GetUpdatedResources = function(language)
{
    var url = ResourceURL + language;

    try {
        window.setTimeout(TimerHandler(this.connection, this.connection.SendHttpRequest,
"GET", url), 0);

        // Disable the language selector (so the user cannot keep changing
        // the language) until the response is received with language resources
        // for the currently selected language.
        var languageSelector = document.getElementById("languageSelectCmb");
        languageSelector.disabled = true;
    } catch (e) {
        // Ignore language related failures
    }
}

MainForm.prototype.OnUpdatedResourcesCallback = function()
{
    var XMLHTTPREQUEST_COMPLETE = 4;
    var XMLHTTPREQUEST_OK = 200;

    var currentState = null;
    var httpCode = null;

    try {
        currentState = this.connection._httpRequest.readyState;
    } catch (e) {
        // Ignore language related failures
        return;
    }

    try {
        // For Safari 10.1.3 the end status is 0
        if (currentState == 0 || currentState == XMLHTTPREQUEST_COMPLETE) {

            // Enable the language selector again.
            var languageSelector = document.getElementById("languageSelectCmb");
            languageSelector.disabled = false;

            var httpCode = this.connection._httpRequest.status;

            if (httpCode == XMLHTTPREQUEST_OK) {
                // Get the text that the Server sent back
                // for the request made.
                var text = this.connection._httpRequest.responseText;
                eval(text);

                if (loading == "true") {
                    this.OnLoadComplete();
                } else {
                    this.UpdateUI();
                }
            } else if (currentState != 0) {
```

```
                //Safari cannot get httpcode if network is down.

                // Ignore language related failures
            }
        }
    } catch (e) {
        // Ignore language related failures
    }
}

MainForm.prototype.UpdateUI = function()
{
    document.title = txt_logoLabel;
    document.getElementById("languageSettingsLabel").innerHTML = txt_languageSettingsLabel;
    document.getElementById("launchRichClientHeaderLabel").innerHTML =
txt_launchRichClientHeaderLabel;
    document.getElementById("launchRichClientTextLabel").innerHTML =
txt_launchRichClientTextLabel;
    document.getElementById("troubleShootingLabel").innerHTML = txt_troubleshootingLabel;
    document.getElementById("contactSupportLabelWithReachOption").innerHTML =
txt_unableToJoinLabel;
    document.getElementById("launchReachLink").innerHTML = txt_joinUsingReachLink;
    document.getElementById("onlineHelpLink").innerHTML = txt_onlineHelpLink;
    document.getElementById("copyRightTextLabel").innerHTML = txt_copyRightTextLabel;

    if (textDirection)
    {
        window.document.dir = textDirection;
    }
}

//
// Connection object - BEGIN
//
function ConnectionObject()
{
    // Properties of the object
    this._httpRequest = this._CreateXMLHttpRequestObject( );
}

ConnectionObject.prototype._CreateXMLHttpRequestObject = function()
{
    var httpRequest = null;

    try
    {
        if( window.XMLHttpRequest ) // for none IE browsers
        {
            httpRequest = new XMLHttpRequest();
        }
        else if( window.ActiveXObject ) // for IE
        {
            var MSXML_XMLHTTP_PROGIDS = new Array(
                        'Microsoft.XMLHTTP',
                        'MSXML2.XMLHTTP.5.0',
                        'MSXML2.XMLHTTP.4.0',
                        'MSXML2.XMLHTTP.3.0',
                        'MSXML2.XMLHTTP'
                        );
```

```
            for (var i=0; i < MSXML_XMLHTTP_PROGIDS.length; i++)
            {
                httpRequest = new ActiveXObject(MSXML_XMLHTTP_PROGIDS[i]);

                if( httpRequest != null )
                {
                    break;
                }
            }
        }
    } catch (e) {
        // Ignore any exception because
        //it is just initializing.
    }

    return httpRequest;
}

ConnectionObject.prototype.SendHttpRequest = function(type, url) {
    try {
        // Set up a new request to the Server.
        // Request is async.
        this._httpRequest.open(type, url, true);

        this._httpRequest.onreadystatechange = Delegate(mainWindow,
mainWindow.OnUpdatedResourcesCallback);

        // Send the request to the Server.
        // No data to send, so pass null.
        this._httpRequest.send(null);
    } catch (e) {
        // Ignore language related failures
    }
}
//
// Connection object - END
//
```

3 Protocol Details

3.1 Conference Activation and Deactivation

Activation refers to the act of instantiating a conference. Deactivation refers to the act of closing a particular instance of the conference. The focus controls activation and deactivation of conferences based on participants joining and leaving the conference.

The trigger for the activation is up to the focus implementation, but typically happens when the first user joins the conference. The trigger for deactivation is up to the focus implementation, but typically happens after the last user leaves the conference.

3.1.1 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 what is described in this document.

Recommendations for the implementer:

- The focus SHOULD maintain a table of participants connected to it per conference.
- The focus SHOULD maintain a table of participants connected to each MCU per conference.
- The focus SHOULD maintain a table of **entity-view** elements for each MCU that contains all of the **entity-view** subelements per conference.
- The focus SHOULD maintain a table, MCU-Conference-URI table that is keyed by **MCU-Type**, and stores the MCU-Conference-URI for each MCU per conference.

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

3.1.2 Timers

None.

3.1.3 Initialization

None.

3.1.4 Higher-Layer Triggered Events

This section specifies the higher-layer triggered events for conference activation and deactivation.

3.1.4.1 Activating a Conference

During conference activation, the focus performs the initial initialization for each conference.

The focus first enumerates all of the MCU types requested by the organizer for the conference and then bootstraps an MCU for each requested **MCU-Type**. The actual bootstrap protocol is outside the

scope of this specification, but typically this includes sending policy and initial setting information specified by the organizer to the MCU.

The conference MUST be activated, even if one or more MCUs fail to bootstrap. The focus can retry failed MCU bootstraps during the lifetime of the conference or use any resource management algorithm to manage MCUs.

3.1.4.1.1 Obtaining MCU-Conference-URIs

For each **MCU-Type**, the focus obtains a SIP URI, the MCU-Conference-URI, and populates the MCU-Conference-URI table with it. The actual mechanism to obtain this URI is implementation-dependent, so is not specified here.

The MCU-Conference-URI MUST satisfy the following two requirements:

- INVITE requests targeted to this MCU-Conference-URI MUST be routable to the MCU.
- The MCU MUST be able to identify the conference in its local state from the MCU-Conference-URI.

3.1.4.2 Deactivating a Conference

Deactivation removes an instance of the conference at the focus, along with all associated instance-specific information. This can happen manually or automatically. The first occurrence of any manual or automatic scenario deactivates the conference.

The actual trigger for deactivation is outside the scope of this specification. Following are some examples of triggers.

- The presenter sends a **deleteConference** command to the focus.
- The organizer sends a **deleteConference** command to the Focus Factory.
- The organizer leaves the company.
- The conference is inactive.

In all of these cases, the deactivation protocol MUST follow the procedures specified for the **deleteConference** command defined in section 4.4.4.

3.1.5 Message Processing Events and Sequencing Rules

None.

3.1.6 Timer Events

None.

3.1.7 Other Local Events

None.

3.2 Joining and Leaving a Conference

Simple Join

A client registers an Active-X control and places the Firefox plug-in in the Firefox plug-ins folder on the client computer during the setup or installation process of the respective applications. The Active-X controls are associated with a specific prog-ID and the Firefox plug-ins with a specific **MIME** type. When user clicks on the link in the client computer, the browser used sends a HTTP request to the Join Manager. The ".ocsmeet" is an extension type used for the document created by the Active-X Control or Firefox plug-in, and we will refer to this document as the ocsmeet document and the contents of this document as the ocsmeet XML string. The Join Manager at this point first computes the ocsmeet XML string, embeds it in the Java-Script and returns it back as a HTML document in the HTTP response. In Internet Explorer, the Java-Script instantiates the Active-X control to detect the presence of the client and queries it for the Version information. In Firefox, the Java-Script detects the presence of the client by loading the appropriate Firefox Plug-in and queries it for version information. The Active-X Control or Plug-in then creates an XML document with a ".ocsmeet" extension in the temporary folder on the client computer. The content body of this document SHOULD<22> conform to the format defined in section 6. Then the Active-X control or Firefox Plug-in eventually tries to open this file using ShellExecute() and the appropriate client installed on the computer is launched. The call flow for Simple Join is shown in the following figure.

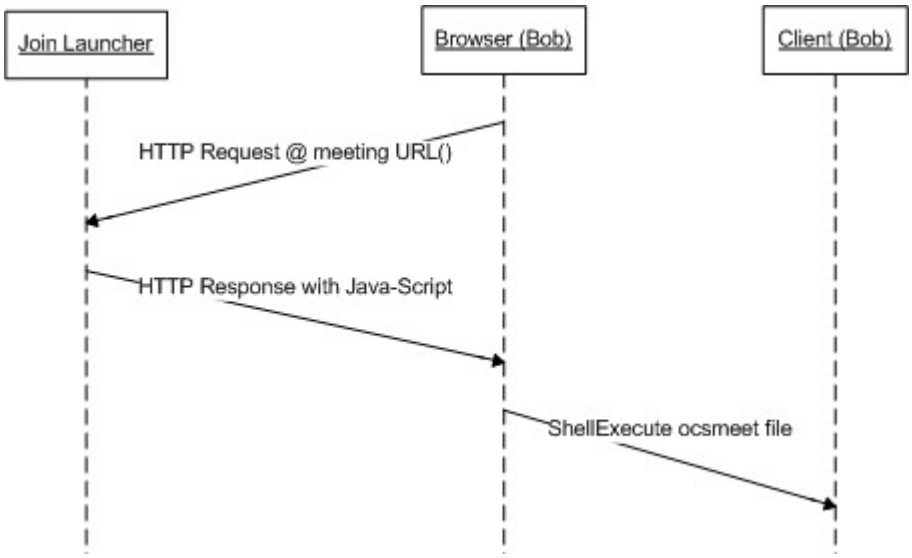


Figure 3: Simple Join

Joining and Leaving a Conference

The conceptual model of joining and leaving a conference is similar to that described in [RFC4353] section 4.1.

A participant joins a conference by establishing a signaling dialog with the focus. This is done by sending an INVITE request to the focus with an **addUser** body, as shown in the following figure.

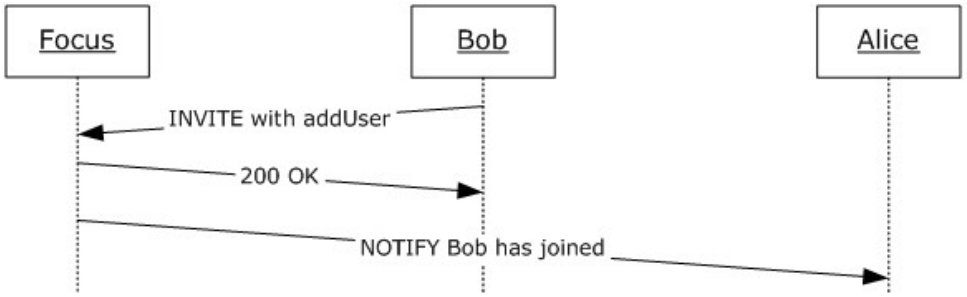


Figure 4: Joining a conference

A participant leaves the conference by terminating the signaling dialog with the focus, as shown in the following figure.

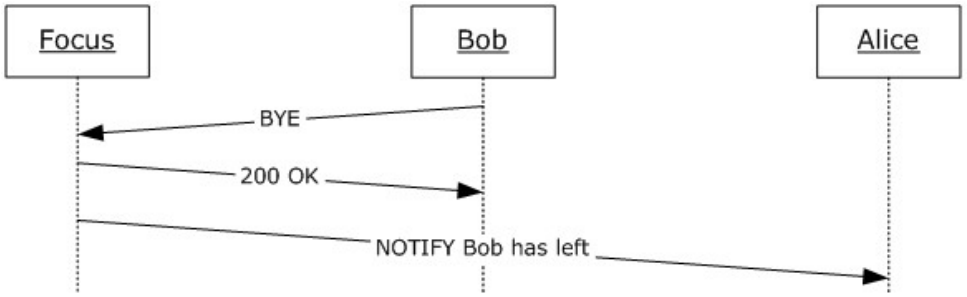


Figure 5: Joining a conference

3.2.1 Abstract Data Model

None.

3.2.2 Timers

There are no additional timers required beyond what is specified in [RFC3261], [RFC4028], and [MS-CONMGMT].

3.2.3 Initialization

None.

3.2.4 Higher-Layer Triggered Events

3.2.4.1 Client Role

A participant joins a conference by establishing a signaling dialog with the focus. This is done by sending an INVITE request to the focus. A participant can subsequently send re-INVITE requests to the focus with the same message body that it used for the original INVITE.

If the client decides to cancel a join request while the INVITE has not received its **final response**, it can send a CANCEL request to the focus.

A participant leaves a conference by terminating the signaling dialog with the focus. The participant uses a SIP BYE request for this purpose.

The SIP UPDATE request is used with a session timer, as specified in [RFC4028], to refresh the dialog state.

Except as specified in the following sections, the message processing rules follow [RFC3261] and [RFC4028].

A client needs to rely on the focus **endpoint** element state to decide whether it has been placed in the conference lobby or not, as specified in section 3.3.

3.2.4.1.1 Constructing the SIP INVITE Request

The INVITE request SHOULD be constructed using the message syntax rules specified in section 2.2.1. It MUST be populated with a valid **addUser** request body, in accordance with the **addUser** request syntax defined in section 2.2.3.3.

If the user is joining on behalf of another user, the client MUST add a **p-session-on-behalf-of** SIP header as defined in [MS-SIPAE]. The client SHOULD<23> also add a **session-on-behalf-of** element to the **addUser** request body, as specified in section 2.2.3.3, with the value of the **p-session-on-behalf-of** SIP header.

A client that enables its users to manage the lobby sets the **lobby-capable** element within the **addUser** body to "true".

```
<addUser>
    <conferenceKeys
      confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:MH22CRI4" />
    <user
      entity="sip:alice@fabrikam.com" xmlns="urn:ietf:params:xml:ns:conference-info">
      <display-text>Alice</display-text>
      <roles>
        <entry>attendee</entry>
      </roles>
      <endpoint
        entity="{ac53488e-4c53-4b6f-a6e1-1b862a16e378}"
        msci:endpoint-
uri="sip:alice@fabrikam.com;opaque=user:epid:AbFhptOVHFeNUhhq_bZ_QQAA;gruu">
        <joining-method>dialed-in</joining-method>
        <msci:clientInfo>
          <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator" />
          <lobby-capable
xmlns="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions">true</lobby-capable>
        </msci:clientInfo>
      </endpoint>
    </user>
</addUser>
```

3.2.4.1.2 Joining conference as anonymous user

The mechanism for joining as an anonymous user is described in [MS-SIPAE] section 3.2.4.3. Once an anonymous user has joined a conference, conference control is the same as for authenticated users, as specified in this protocol.

3.2.4.2 Focus Role

When the focus receives an INVITE request targeted to a conference, it SHOULD authorize the request and admit the user into the conference. If the user is not authorized to join the conference, the focus MUST respond with a **403 Forbidden** response. If the user is admitted, the focus SHOULD respond with a 200 OK response to the INVITE and then establish a dialog.

If a **session-on-behalf-of** element is present in the **addUser** request body, the focus MUST validate that a **p-session-on-behalf-of** SIP header is present as well, and that their values are equal.

If the validation fails, the focus SHOULD<24> respond with a 403 Forbidden response.

If the focus receives a CANCEL request when the INVITE has not received its final response, it SHOULD treat the CANCEL request as a participant leave event.

If the focus receives a BYE request for an existing dialog, it SHOULD treat the request as a participant leave event.

If the focus decides to end the conference, it SHOULD eject all of the participants in the conference. In such a scenario, the focus SHOULD send a BYE to each participant, thereby terminating the signaling dialog.

If the focus receives a SIP UPDATE request, it SHOULD extend the dialog lifetime using the procedures described in [RFC4028]. It can also accept re-INVITE requests and extend the dialog lifetime by some predetermined value. It is recommended that a value of 600 seconds be used for this purpose.

When a participant is accepted, the focus MUST send a notification to other participants who have subscribed to receive conference notifications. This functionality is similar to the conference notification service defined in [RFC4353] section 4.4, and is specified in section 3.3.

Except as specified in the following sections, the message processing rules follow the specifications in [RFC3261] and [RFC4028].

3.2.4.2.1 Processing the addUser request

The focus SHOULD first parse the **addUser** request body and apply the basic syntax validation rules given in section 2.2.3.3.

Detailed signaling dialog establishment examples are given in section 4.

3.2.4.2.2 Processing INVITE from anonymous client

When the focus receives a conference join INVITE from an anonymous user, it MUST process it according to rules described in [MS-SIPAE] section 3.3.5.4. For an example, see [MS-SIPAE] section 4.5.

3.2.5 Message Processing Events and Sequencing Rules

Except as specified in the following subsections, the rules for message processing are as specified in [RFC3261], [RFC4028], and [MS-CONMGMT].

3.2.5.1 Client Role

The UAC SHOULD parse the body of the 200 OK response, extract the role returned by the focus, and use it as the role for the rest of the conference.

3.2.5.2 Focus Role

This section specifies the message processing events and sequencing rules for the focus role in joining a conference and leaving a conference.

3.2.5.2.1 Constructing the SIP INVITE Response

If the focus decides to accept the participant into the conference, it MUST generate and send a 200 OK INVITE response constructed using the message syntax rules specified in section 2.2.1.

The response MUST be populated with a valid **addUser** response body in accordance with the general body format rules given in section 2.2.3.4.

3.2.5.2.2 Multiple Endpoints Connecting to the Focus

The focus SHOULD allow multiple endpoints (5) of the same participant to connect to the conference.

If the focus allows multiple endpoints (5) of the same participant to connect to the conference, all such connected **endpoints** MUST be listed inside the **user** element of the conference roster for that user. The **user** element is defined in section 2.2.4.2.

The focus MUST manage each endpoint (5) independent of the others. When one endpoint (5) terminates the dialog with the focus, it MUST NOT affect other endpoints (5) connected to the focus.

3.2.5.2.3 Notifying Watchers When a Participant Joins

When a participant joins, all other participants in a conference MUST be notified of the participant connected event using the procedures described in section 3.3.

The generated document MUST conform to the syntax rules given in section 2.2.4.

3.2.5.2.4 SIP Error Response Codes

The following table specifies the extension **SIP response** codes that are defined by this specification.

SIP response code	Reason
400	Malformed request with one or more invalid headers or invalid content-body.
403	Forbidden: User is not authorized to join the conference, as defined by conference policy.
404	Failure: Conference Not Found.
603	Failure: Meeting size has been exceeded and no more participants are allowed.

3.2.6 Timer Events

None.

3.2.7 Other Local Events

None.

3.3 Conference Subscriptions and Notifications

Participants in a conference can subscribe to the focus to receive conference state change notifications using the procedures specified in [RFC4575]. Basic call flow for the subscription is shown in the following figure.

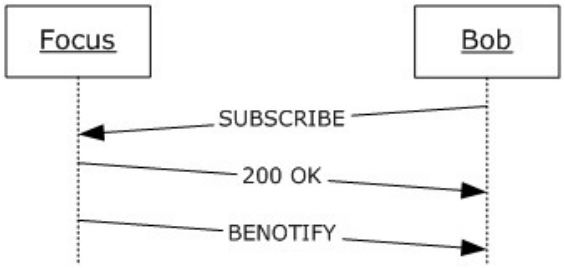


Figure 6: Subscribing to a conference

The 200 OK contains the client focus **endpoint** details that determine if the client was directly admitted into the conference or placed in the conference lobby. A "connected" **endpoint** state determines that the client is connected to the conference.

Following is an example of a connected state.

```
<user entity="sip:bob@fabrikam.com" state="full">
<display-text>Bob</display-text>
<roles>
<entry>attendee</entry>
</roles>
<endpoint entity="{659dae6c-82aa-46c8-8b37-da684a1a0d06}"
 session-type="focus" epid="732A323248" endpoint-
uri="sip:bob@fabrikam.com;opaque=user:epid:vAaOGYq2H12kV5u7RWPdEQAA;gruu">
<status>connected</status>
</endpoint>
</user>
```

Following is an example of an "on-hold" state that indicates that a client has been placed in the lobby.

```
<user entity="sip:bob@fabrikam.com" state="full">
<display-text>Bob</display-text>
<roles>
<entry>attendee</entry>
</roles>
<endpoint entity="{659dae6c-82aa-46c8-8b37-da684a1a0d06}"
 session-type="focus" epid="732A323248" endpoint-
uri="sip:bob@fabrikam.com;opaque=user:epid:vAaOGYq2H12kV5u7RWPdEQAA;gruu">
```

```
<status>on-hold</status>
</endpoint>
</user>
```

The basic call flow for subscription termination is shown in the following figure. In this figure, the Client "Bob" sends a SUBSCRIBE request with an **Expires: 0** header, as specified in [RFC3265], to terminate the subscription dialog.

Figure 7: Terminating a subscription

3.3.1 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 what is described in this document.

3.3.1.1 Client Role

Recommendations for implementer are as follows:

- The client SHOULD maintain an internal table that is keyed by the various elements and attributes of the conference data model and stores the corresponding value obtained from the conference notifications.
- The client SHOULD maintain a static list of **MCU-Types** that it recognizes and supports.
- The client SHOULD maintain a table, called the MCU-Conference-URI table, that is keyed by the **MCU-Type** and stores the MCU-Conference-URI for each MCU.

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

3.3.2 Timers

3.3.2.1 Client Role

When a subscription dialog is established, the client can start an internal timer set to a value of 32 seconds for receiving the first full notification.

Other timers are specified in [RFC3265].

3.3.3 Initialization

3.3.3.1 Client Role

Before establishing a subscription with the focus, the client MUST have joined the conference by establishing a valid signaling dialog with the focus, as specified in section 3.2. This is an extension to the specifications in [RFC4575].

The client can indicate support for the **ms-benotify** or **ms-piggyback-first-notify** extensions specified in [MS-SIP] section 3.4. If it indicates support for them, it SHOULD be prepared to accept responses and notifications conforming to those extensions, as specified in [MS-SIP] section 3.4.5.1.

Detailed subscription establishment examples are given in section 4.

3.3.4 Higher-Layer Triggered Events

3.3.4.1 Client Role

The message processing rules follow the specifications in [RFC3265] and [RFC4575].

After a subscription dialog is established between the subscriber and the focus, the standard rules specified in [RFC3265] MUST be followed for dialog extension, dialog termination, and notify generation.

A client placed in the lobby receives limited data in the notification. When admitted into the conference, it receives a full notification.

A client in the lobby does not receive any data about other lobby or admitted clients, the current conference state while in the lobby or the MCUs that have been activated for the conference.

A client in the lobby SHOULD check the **lobby-capable** element in the limited notification it receives. If the **lobby-capable** element is present and set to "true", the client MAY wait to be admitted to the conference.<25> If the **lobby-capable** element is absent or is set to "false", the client SHOULD immediately terminate its focus and subscription dialogs with the conference.

3.3.4.2 Focus Role

The focus behaves as a conference notification service, as specified in [RFC4353] section 4.4, and implements the conference event package specified in [RFC4575].

The focus SHOULD also behave as a subscriber to each MCU in the conference. If it does so, it SHOULD maintain a local copy of the MCU conference state received from the MCU. It SHOULD compose the conference document by aggregating the conference state maintained locally with the conference state received from all MCUs using the roster aggregation algorithm defined in section 3.3.4.2.1. This algorithm is given as an example of how implementations can aggregate the conference roster. Implementations can choose any mechanism as long as the external behavior is conformant.

When a participant is admitted into the conference, the focus MUST send that participant a notification setting that sets the state of the **endpoint** of the participant to "connected".

The focus MUST send a full notification to a lobby participant that has been admitted into the conference.

The focus SHOULD NOT send information about other participants in a conference to a participant in the lobby.

The **conference-info** document generated by the focus and sent to watchers MUST conform to the conference roster document format, as specified in section 2.2.4.

The **conference-info** document MUST NOT contain any information that needs to be encrypted when sent back to the client. For retrieving sensitive information in encrypted form, the client MUST use the **getConference** control command, as specified in section 3.11.

The focus SHOULD reject a subscription from a participant that does not have an existing signaling dialog. The lifetime of the subscription dialog SHOULD be scoped to the signaling dialog. Thus, if the signaling dialog terminates for some reason, the focus SHOULD automatically terminate the subscription dialog if one exists, even if the subscriber does not explicitly request its termination. Standard subscription termination procedures specified in [RFC3265] MUST be followed.

After a subscription dialog is established between the subscriber and the focus, the standard rules specified in [RFC3265] and [MS-SIP] MUST be followed for dialog extension, dialog termination, and notify generation.

Detailed subscription establishment examples are given in section 4.

3.3.4.2.1 Roster Aggregation Algorithm

Using [RFC4575] terminology, the focus acts as the subscriber for the conference state events published by the MCUs, independent of whether the MCUs are collocated or located separately. The focus thus receives independent state events from each MCU participating in a conference. The focus also maintains local conference state such as conference meta-data and participant state. Because the conference roster is logically distributed across multiple MCUs and the focus, it becomes necessary to aggregate all the fragments to produce a consistent view of the conference roster.

This section provides a description of the basic aggregation logic that has to be supported by all focus implementations that expose multiple MCU support to the client. The aggregation algorithm described later in this section requires the following focus behavior:

- The focus supports multiple MCUs in a conference, which are capable of reporting conference state independently.
- Only one endpoint (5) is supported per MCU per user. Thus, an MCU SHOULD NOT allow more than one endpoint (5) for each participant in the conference.
- The focus SHOULD accept conference state changes reported in **conference-view** and **users** elements only. These elements are inside the **conference-info** document. It SHOULD ignore everything else.

Extensions can specify alternate or extension algorithms to suit other types of focus or MCU implementations, as long as the client interface remains identical to this specification.

Aggregation Algorithm

The focus SHOULD validate the MCU published document using the MCU conference roster syntax rules specified in section 2.2.5.

The focus SHOULD apply the procedures described in [RFC4575] section 3.7 for processing the received notification with the following extensions:

- The **version** number MUST be maintained per MCU.

- A copy of the conference state for each MCU MUST be maintained locally, independent of the other MCUs, and hence the processing for a received notification affects only the local state of that MCU.
- Except for the **conference-view** and **users** elements and their subelements, all other elements MUST be ignored for processing purposes. The subelements of the **conference-view** and **users** elements MUST be updated in the local state using the algorithm specified in [RFC4575] section 3.7.
- The focus MUST accept a participant listed in the MCU notification, even if the participant is not connected to the focus.

If any local MCU state was changed by executing the preceding algorithm, the focus MUST construct and send out an appropriate conference event notification to all watchers. The generated document MUST conform to the conference roster syntax rules specified in section 2.2.4.

3.3.4.3 MCU Role

When an MCU is bootstrapped for a conference, it MUST establish a notification channel with the focus. The actual protocol for establishing the notification channel is outside the scope of this specification, but it can be based on [RFC3265]. However, the notification document MUST conform to the application/conference-info+xml document format.

3.3.4.3.1 MCU Notifications

The MCU behaves as a notifier, as specified in [RFC4575]. It MUST report conference event notifications to the focus.

This specification defines the following extensions to [RFC4575] section 3.6:

- On being bootstrapped for a conference, an MCU MUST publish a full notification populating the **conference-view** element with all relevant elements. The **entity-state** subelement MUST be included. Other subelements are optional.
- An MCU SHOULD generate partial **conference-info** notifications whenever the state of any **conference-view** subelement changes.
- An MCU SHOULD generate partial **user** notifications whenever the state of any connected user changes.

3.3.5 Message Processing Events and Sequencing Rules

Except as specified in the following sub-sections, the processing rules follow [RFC4575] and [RFC3265].

3.3.5.1 Client Role

This section specifies the message processing events and sequencing rules for the client role that are related to conference subscriptions and notifications.

3.3.5.1.1 Processing the First Full Notification

On receipt of the first full notification, the client MUST terminate the 32-second timer, as specified in section 3.3.2.1.

The client MUST process the first full notification received using the procedures specified in [RFC4575] section 3.7.

The client SHOULD extract all of the **entry** elements listed in the **conf-uris** element of the conference document and use them to construct the MCU-Conference-URI table using the **conf-uris** semantics defined in section 2.2.2.4.

The client MUST extract its own focus **endpoint** element state to determine whether it has been admitted immediately into the conference or placed in the lobby.

A client placed in the lobby SHOULD wait for a full notification from the focus with the **endpoint** element state set to "connected" to determine that it has been connected to the conference.

A client placed in the lobby SHOULD implement a timeout to disconnect from the conference if not admitted in a specific time chosen by the client.

3.3.5.2 Focus Role

This section specifies the message processing events and sequencing rules for the focus role that are related to conference subscriptions and notifications.

3.3.5.2.1 Generating a Full Notification Document

The focus MUST immediately send a full notification document to the client when the client establishes a subscription with the focus. If the **ms-piggyback-first-notify** extension is supported, this document SHOULD be returned in the 200 OK response body itself.

The generated full notification document MUST conform to the conference roster syntax rules in section 2.2.4. In addition, the document generated by the focus MUST include the following child elements of the **conference-info** element:

conf-uris: Lists all the MCUs provisioned for the conference and their corresponding MCU-Conference-URI, as specified in section 2.

conference-view: SHOULD initially list the **entity-view** of the focus itself. Subsequent full notifications SHOULD list the full **conference-view**, which includes the **entity-view** elements for the focus and the MCUs.

users: SHOULD list all the participants in the conference, including those who are connected only to MCUs.

3.3.5.2.2 SIP Error Response Codes

None.

3.3.6 Timer Events

3.3.6.1 Client Role

If the 32-second timer defined in section 3.3.2 fires, the client SHOULD fail the conference subscription because it has not received any notifications and perform appropriate user notification action.

3.3.7 Other Local Events

None.

3.4 Common Conference Control

After a participant joins a conference, it can perform various conference control operations. In general, conference control requests can be classified into three categories:

- **Focus Commands**. These are commands that terminate on the focus and do not involve MCU interaction. These commands change some conference state and result in notification to watchers. No commands are currently defined for this category.
- **MCU Commands**. These are commands that are authorized by the focus but are simply forwarded to the MCU. Thus, no focus state is modified unless the MCU generates a notification indicating the change of state. In this case, the client MUST indicate the MCU to process the command. Such commands can be specified by extension specifications.
- **General Conference Commands**. These are commands that are processed by the focus, as well as by all MCUs in the conference. For such commands, the focus and all of the MCUs can generate conference-state change notifications, which are then sent to all participants. The **modifyConferenceLock** command is an example of this category.

Regardless of the **command-type**, all conference control commands share a common protocol sequence that is described later in this section. The protocol sequence is divided into client, focus, and MCU roles. The following figure shows a basic conference control call flow.

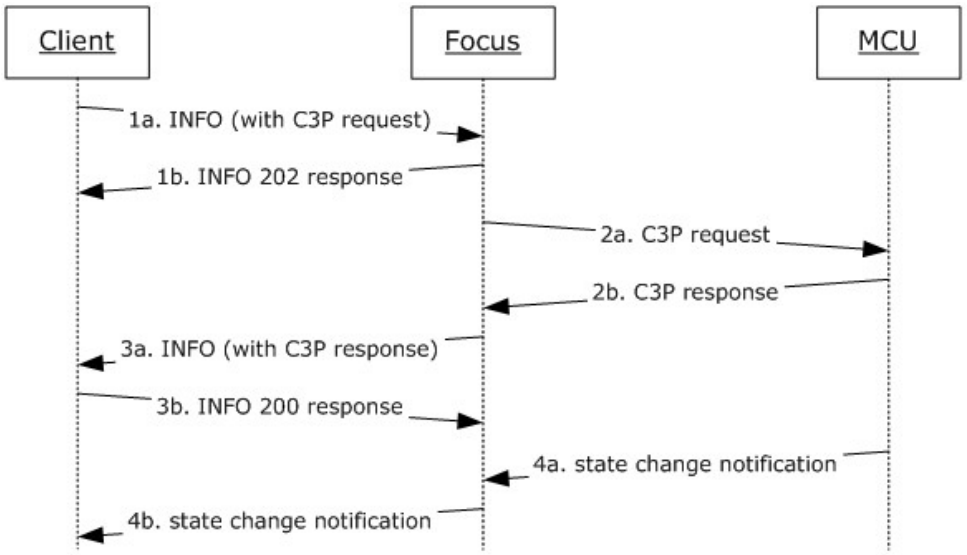


Figure 8: Basic conference control call flow

In step 1a, the client sends a conference control request to the focus. This is done by sending a SIP INFO request with a C3P-request body. The focus accepts the SIP INFO and responds to it with a SIP INFO **202 Accepted** response indicating that the command is being processed.

If the command semantics involved processing by MCUs, the focus sends the request to the MCUs and waits for their response. This is shown as steps 2a and 2b.

When command execution completes, the focus generates a C3P final response and sends it to the client over a SIP INFO response. The client responds with a SIP 200 OK response indicating that it has received the response. This is shown as steps 3a and 3b.

If the command execution results in a change of conference state, the MCUs and focus generate state change notifications to all watchers. This is shown as steps 4a and 4b.

Processing details are described in the following subsections. Note that some commands require intermediate processing that is specified in the appropriate subsection. Extensions to this specification can also specify extra processing behavior.

The protocol used between the focus and the MCUs for exchanging conference control requests, responses, and notifications is outside the scope of this specification. This protocol deals only with the client-to-focus interface.

Detailed command call flow examples are given in section 4.

3.4.1 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.

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

3.4.1.1 Client Role

Recommendations for implementers:

- The client SHOULD maintain a table, such as a table of MCU-Conference-URI values, which is keyed by **MCU-Type**, and stores the MCU-Conference-URI for each MCU.
- The client SHOULD maintain a table of pending requests. This table SHOULD be keyed by the C3P Request-Id and store all of the C3P requests that have been sent to the server but have not received final responses and have not timed out.

3.4.1.2 Focus Role

The term focus is used synonymously with server in the following section.

The focus SHOULD use an enumeration variable, **participantRole**, to track the granted role for each participant. The valid values for this enumeration are "presenter" and "attendee".

It is recommended that the focus maintain a table of C3P command types that maps all supported C3P commands to an enumeration that has two values:

- "CommandTypeConference"
- "CommandTypeUser"

These values indicate whether a command operates on the conference as a whole or on a particular user.

The focus SHOULD use a Boolean variable, **isFirstPartyRequest**, to track whether a request is a **first-party request** or **third-party request**.

3.4.2 Timers

3.4.2.1 Client Role

A timer is associated with every conference control command sent to the focus. The initial value of this timer SHOULD be set to 32 seconds. Every time a C3P "pending" response is received for that command, the timer SHOULD be extended by three minutes, or 180 seconds. If no further responses are received within the timeout interval, the command MUST be considered timed-out.

3.4.3 Initialization

3.4.3.1 Client Role

The client MUST have established a valid signaling dialog with the focus, as illustrated in section 4.2.1.

It MUST have constructed the MCU-Conference-URI table using the procedures described in section 3.3.5.1.1.

3.4.4 Higher-Layer Triggered Events

3.4.4.1 Client Role

Except as specified in the following sections, the rules for message processing are as specified in [RFC3261] and [RFC2976].

3.4.4.1.1 Sending a Conference Control Request

The client constructs a SIP INFO request within the signaling dialog established with the focus. The constructed SIP INFO request MUST conform to the conference control message syntax rules in section 2.2.1.5. The request MUST contain a valid C3P request, as specified in section 2.2.3, populating all of the relevant attributes and elements needed for processing the command. The client sends this request to the focus. The client then starts the transaction timer, as specified in section 3.4.2.

3.4.4.2 Focus Role

The following sections specify the higher-layer triggered events for the focus role related to the common conference control.

3.4.4.2.1 Receiving a Conference Control Request

On receipt of a SIP INFO request containing the conference control command, the focus MUST validate that the SIP INFO request belongs to a known signaling dialog that is not currently in the lobby.

It SHOULD then parse and validate the C3P request body using the C3P request syntax rules specified in section 2.2.3.

If parsing or validation fails, the focus SHOULD reject the request.

If parsing and validation succeed, the focus SHOULD immediately generate a SIP 202 Accepted response indicating that the request has been accepted for further processing.

It SHOULD then begin processing the command, as specified in the following sections.

3.4.4.2.2 Authorizing a Conference Control Request

This section describes a conceptual authorization model that an implementation performs 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.

The focus SHOULD initialize the **participantRole** variable to the role of the sending participant.

The focus SHOULD then determine the command contained in the request.

It SHOULD then find the **CommandType** by looking it up in the C3P command type table, as described in section 3.4.1.2.

If the **CommandType** is "UserLevel", the focus SHOULD extract the **userEntity** URI from the command body.

The focus SHOULD then check whether the URI specified in **userEntity** matches the URI specified in the **from** attribute of the C3P request using standard URI matching rules, as specified in [RFC3261] section 19.1.4. The focus SHOULD set the variable **isFirstPartyRequest** to "true" if they match, and to "false" if they do not match.

The focus SHOULD reject a request with an "unauthorized" C3P response if the **CommandType** is "ConferenceLevel" and the **participantRole** is "attendee".

The focus SHOULD reject a request with an "unauthorized" C3P response if the **CommandType** is "UserLevel", if **isFirstPartyRequest** is set to "false", and the **participantRole** is "attendee".

This concludes the basic authorization logic. Extensions to this specification can specify additional authorization procedures to be carried out by the focus.

3.4.4.2.3 Processing a Command

The focus SHOULD then process the command using the processing rules specified for the command.

Detailed command-level processing rules are specified in subsequent sections.

3.4.4.2.3.1 SIP Response Codes

The following table specifies additional failure response codes that are defined for the SIP INFO request.

SIP response code	Reason
503	Service unavailable. One or more services required to execute the command are currently unavailable. This is a retriable failure.
603	Non-retriable command-specific failure. This can also happen if the command specified an **MCU-Type** that is not enabled for the conference.

3.4.4.2.3.2 Common C3P Failure Response Codes

The following table specifies generic processing failure response codes that are applicable to all commands.

C3P response code	Reason
unauthorized	Forbidden. User does not have the privileges to perform the specified conference control command.
Timeout	Timeout encountered when processing a command. This is a retriable failure.
requestMalformed	The specified body is not valid according to the syntax rules specified for the command.
notSupported	The specified command is not implemented.
serverBusy	Service unavailable. One or more services required to execute the command are currently unavailable or too busy to accept new requests. This is a retriable failure.
otherFailure	Unspecified failure. Command-specific failure codes can be defined for this case.

The following table specifies some specific processing failure response codes applicable to most commands. For exact applicability, refer to the command schema.

C3P response code	Reason
conferenceDoesntExist	Specified conference does not exist.
userDoesntExist	Specified Participant does not exist.
endpointDoesntExist	Specified target endpoint does not exist.

3.4.5 Message Processing Events and Sequencing Rules

3.4.5.1 Client Role

Unless specified otherwise, the message processing rules follow the procedures described in [RFC2976].

3.4.5.1.1 Receiving SIP 202 Response to the INFO Request

A client can ignore the 202 Accepted response from the server. The client SHOULD then wait for another INFO response from the server containing the command response.

3.4.5.1.2 Receiving SIP 200 Response to the INFO Request

If the client receives a 200 OK response to its INFO request, it SHOULD then extract the C3P **response** body contained in the 200 OK response and process it as specified in section 3.4.5.1.4.

3.4.5.1.3 Receiving an INFO Request From the Focus

If the client receives an INFO request from the focus, it SHOULD respond immediately with a 200 OK to the INFO request.

It SHOULD then extract the C3P response body contained in the request and process it as specified in section 3.4.5.1.4.

3.4.5.1.4 Processing a Command Response

The client SHOULD parse and validate the document using the syntax rules specified in section 2.2.3.

It SHOULD then find the corresponding request from the PendingRequestTable, using **requestId** as the key. If a matching request is not found, it is likely that the request has timed-out or has been cancelled by the application, hence the client SHOULD silently drop the response.

If the response is a C3P 'pending' response (as indicated by the **code** attribute), the client SHOULD extend its transaction timer, as described in section 3.4.2, and can then ignore the response.

If the response is a C3P 'success' or 'failure' response, the client SHOULD remove the request from the PendingRequestTable. It SHOULD also stop any pending timers. It SHOULD then process the response in the context of the request.

The actual processing action for a C3P response is outside the scope of this specification, but it can include alerting the participant that the command succeeded or failed.

3.4.5.1.5 Processing Incoming Notifications

If the client receives conference state change notifications on the subscription channel, it SHOULD process them and modify its conference state, as described in section 3.4. This SHOULD be done independent of the command processing described in this section.

3.4.5.2 Focus Role

The following sections specify the message processing and sequencing rules for the focus role related to the common conference control.

3.4.5.2.1 Forwarding Command to an MCU

If the command needs to be processed by a specific MCU, the focus SHOULD appropriately forward the command to the MCU.

The focus SHOULD then wait for responses from the MCU and forward all responses individually to the client. The protocol for creating and sending commands to the MCUs and processing the responses received from them is outside the scope of this specification.

The command processing is considered complete when the focus has sent a final response to the client.

3.4.5.2.2 Forking Command to All MCUs

If the command needs to be processed by all of the MCUs participating in the conference, such as the **modifyConferenceLock** C3P request, the focus SHOULD appropriately forward the command to the MCU.

The focus SHOULD also generate a final C3P response back to the client, giving the result of executing the command at the focus.

The protocol for creating and sending commands to the MCUs and processing the responses received from them is outside the scope of this specification.

This specification does not define any particular mechanism for processing the responses received from the MCUs for the forking scenario.

3.4.5.2.3 Completing Command Processing

When the focus completes processing for the command, it SHOULD generate a SIP INFO response and send it back to the client. The INFO response SHOULD have a valid C3P response body containing the result of executing the command.

If the local conference state was modified, the focus SHOULD generate a conference state change notification to all watchers.

3.4.6 Timer Events

3.4.6.1 Client Role

If no response has been received for an outgoing request, and the timer fires, the client SHOULD treat this timeout event as if the command has failed and perform the processing steps outlined earlier.

3.4.6.2 Focus Role

If the timer fires and the request is still in processing, the focus SHOULD generate and send a "pending" response to the client to indicate that the request is being processed. It SHOULD then reinitialize the timer to 28 seconds.

3.4.7 Other Local Events

None.

3.5 Conference Control – modifyConferenceLock Command

The **modifyConferenceLock** command controls the current lock state of the conference. After the conference is locked, no new users are allowed to join. This is a conference level command.

Before issuing this command, a client SHOULD check the permissible settings by reading the extensions specified in section 2.2.2.3.

The lock state is maintained by the focus and the MCUs**.** They report the lock state independent of each other. The global conference lock state is the lock state reported by the focus. A client SHOULD track the lock state of the MCUs independent of the global conference lock state.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.5.1 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.

The abstract data model specified in section 3.4 is extended as specified in the following subsections.

3.5.1.1 Focus Role

The term focus is used synonymously with server in the following section.

The focus SHOULD use a Boolean variable, **currentConferenceLockState,** to track the current lock state of the conference.

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

3.5.2 Timers

Timers are specified in section 3.4.

3.5.3 Initialization

Initialization steps are specified in section 3.4.

3.5.4 Higher-Layer Triggered Events

None.

3.5.5 Message Processing Events and Sequencing Rules

The rules for message processing and sequencing are specified in section 3.4.

3.5.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.5.7 Other Local Events

None.

3.6 Conference Control – modifyUserRoles Command

The **modifyUserRoles** command is used to modify the role of a participant. This is a user level command.

The participant role is maintained by both the focus and the MCUs, and each entity reports the user-role in the conference roster independent of the other entity. The global user role for the conference is the role reported by the focus.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.6.1 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.

The abstract data model specified in section 3.4 is extended as specified in the following subsections.

3.6.1.1 Focus Role

The term focus is used synonymously with server in the following section.

The focus SHOULD use the enumeration **participantRole** to track the current role of the participant in the conference. This enumeration maps to the **user-role** element defined in the application/conference-info+xml schema.

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

3.6.2 Timers

Timers are specified in section 3.4.

3.6.3 Initialization

Initialization steps are specified in section 3.4.

3.6.4 Higher-Layer Triggered Events

None.

3.6.5 Message Processing Events and Sequencing Rules

The rules for message processing and sequencing are as specified in section 3.4.

3.6.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.6.7 Other Local Events

None.

3.7 Conference Control – deleteUser Command

The **deleteUser** command is used to eject a participant. This is a user level command.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.7.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.7.2 Timers

Timers are specified in section 3.4.

3.7.3 Initialization

Initialization steps are specified in section 3.4.

3.7.4 Higher-Layer Triggered Events

None.

3.7.5 Message Processing Events and Sequencing Rules

3.7.5.1 Focus Role

Unless otherwise specified, the rules for message processing and sequencing are as specified in section 3.4.

The focus SHOULD reject a request if the **endpointEntity** element is specified.

The focus SHOULD reject a request if the **mcuUri** attribute is specified.

When the focus ejects a participant, it MUST perform the following steps:

- Terminate all of the subscription dialogs for that user, for all endpoints (5). This involves sending a **NOTIFY** with an **Expires** header field whose value is "0", as specified in [RFC3265]. If the participant was ejected from the conference, the focus MUST add a **subscription-state** header with a **Reason** header field with its **text** parameter set to "ParticipantRemoved**"**. If the participant was ejected from the lobby, the focus MUST add a **subscription-state** header with a **Reason** header field with its **text** parameter set to "Participant Denied".
- Terminate all signaling dialogs for that user, for all endpoints (5). This involves sending a BYE to the participant. If the participant was ejected from the conference, the focus MUST add a **Reason** header field with its **text** parameter set to "Participant Removed". If the participant was ejected from the lobby, the focus MUST add a **Reason** header field with its **text** parameter set to "Participant Denied"**.**
- It is recommended that the subscription dialog termination be done before the signaling dialog termination.
- The focus MUST notify all remaining participants in the conference with the updated state.
- The focus MUST copy and send the command to all the MCUs in the conference so that they can also eject the user.

3.7.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.7.7 Other Local Events

None.

3.8 Conference Control – deleteConference Command

The **deleteConference** command is used to end a conference, eject all participants, and deactivate all MCUs from that conference. This is a conference level command.

Note that the **deleteConference** command sent to the focus from the client MUST NOT modify the provisioning state of the conference. The Focus Factory SHOULD be used to delete the conference from the system to completely deprovision the conference, as specified in [MS-CONFPRO].

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.8.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.8.2 Timers

Timers are specified in section 3.4.

3.8.3 Initialization

Initialization steps are specified in section 3.4.

3.8.4 Higher-Layer Triggered Events

None.

3.8.5 Message Processing Events and Sequencing Rules

None.

3.8.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.8.7 Other Local Events

None.

3.9 Conference Control – addUser Dial-out Command

The **addUser** dial-out command is used to connect a participant to an MCU. This section specifies the basic **addUser** dial-out behavior that all client, focus, and MCU implementations SHOULD adhere to. This command is typically used only when the MCU supports SIP as the protocol for user session establishment.

This is a user level command. This command is typically sent to the MCU through the focus. The **addUser** dial-out protocol sequence is shown in the following figure.

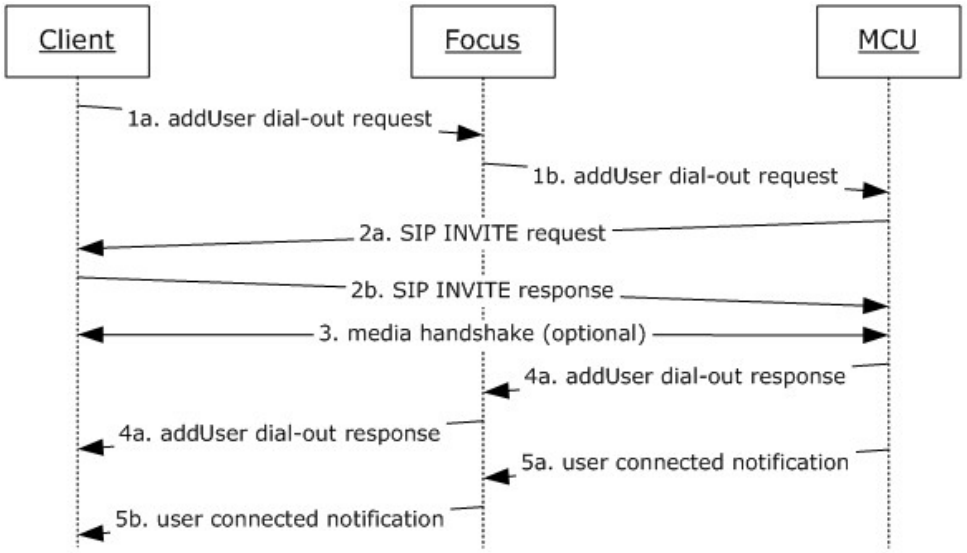


Figure 9: addUser Dial-out call flow

The client sends an **addUser** dial-out request, over a SIP INFO, to the focus. The focus authorizes

it and then forwards it to the MCU. This is shown in steps 1a and 1b.

The MCU processes the **addUser** dial-out request. It can generate an **addUser** dial-out pending C3P responses, which is not shown in the preceding figure.

The MCU then sends an INVITE request to the user endpoint (5) specified in the request. This is shown as steps 2a and 2b. Note that the INVITE request can be sent to some other endpoint (5) of the user or to a different user. For simplicity, these are not shown here. This specification does not specify the actual contents of the SIP request body. Extensions to this specification can define the actual contents, such as the **Session Description Protocol (SDP)** format, as described in [RFC3264].

At this point, the client and the MCU can negotiate media. Extensions to this specification can define how the media negotiation is done. This is shown as step 3.

After the user is connected to the MCU, the MCU generates an **addUser** dial-out C3P response indicating that the command has succeeded. This is shown in steps 4a and 4b.

Finally, the MCU notifies the focus that the user has connected, as shown in step 5a, which in turn is propagated to all conference watchers, as shown in step 5b.

MCU implementers can specify extensions to the **addUser** dial-out protocol.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.9.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.9.2 Timers

Timers are specified in section 3.4.

3.9.3 Initialization

Initialization steps are specified in section 3.4.

3.9.4 Higher-Layer Triggered Events

3.9.4.1 MCU Role

Unless otherwise specified in the following subsection, the rules are as specified in section 3.4.

3.9.4.1.1 Constructing an Outgoing SIP INVITE Request

Unless otherwise specified in the following section, the rules for constructing and sending an INVITE request are as specified in [RFC3261].

When the MCU receives an **addUser** dial-out request, it SHOULD initiate a signaling/media handshake with that user. The outgoing INVITE SHOULD be constructed using the following rules:

- The URI in the SIP **From** header field MUST be set to the MCU-Conference-URI.
- The URI in the SIP **To** header field SHOULD be set to the SIP URI specified in the **entity** attribute of the **user** element of the dial-out request.
- The SIP **Request-URI** header field SHOULD be set as follows:
  - If the **refer-to-uri** attribute is supplied in the **addUser** dial-out request, it SHOULD be used to populate the SIP **Request-URI** header field.
  - If the **endpoint-uri** attribute is supplied in the **addUser** dial-out request, it SHOULD be used to populate the SIP **Request-URI** header field, provided that **refer-to-uri** is empty.
  - Otherwise, the SIP **Request-Uri** header field SHOULD be set to the URI in the **To** header field.
- If the **refer-to-uri** attribute is specified, the MCU SHOULD parse this attribute, treating it as a SIP URI. It SHOULD extract all of the headers present and add them to the outgoing request, as defined in [RFC3261] section 19.

Extensions can define additional rules for session establishment between the MCU and the client.

3.9.5 Message Processing Events and Sequencing Rules

3.9.5.1 MCU Role

Unless otherwise specified, the rules for message processing and sequencing are as specified in section 3.4.

The MCU MUST generate a user connected notification if the dial-out requests and the user successfully connect to the MCU. An example is given in section 4.

3.9.6 Timer Events

The rules for timer event processing are specified in section 3.4

3.9.7 Other Local Events

None.

3.10 Conference Control – addUser Dial-in Command

The **addUser** dial-in command is used to connect a participant to an MCU. This section specifies the basic **addUser** dial-in behavior to which all client, focus, and MCU implementations SHOULD adhere.

The **addUser** dial-in protocol sequence is shown in the following figure.

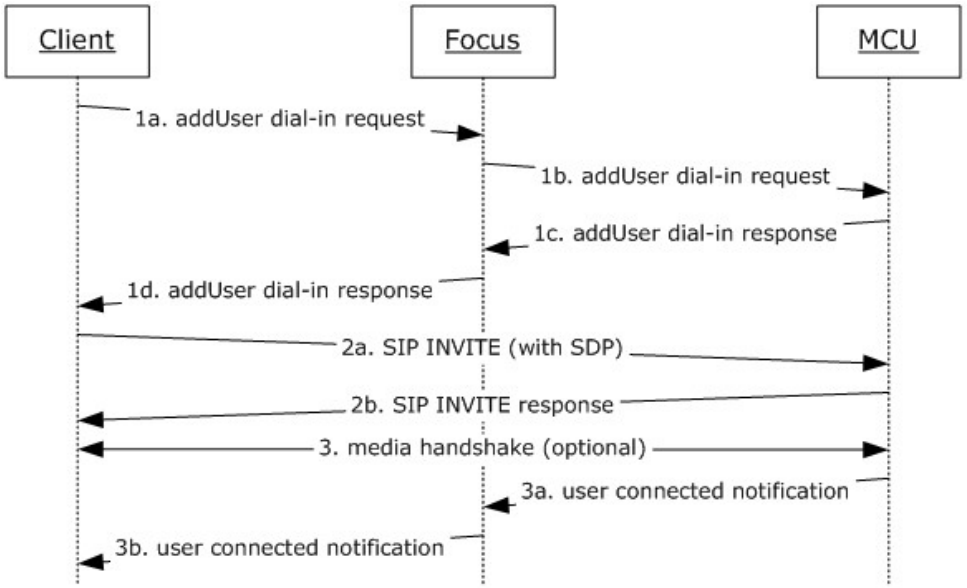


Figure 10: addUser Dial-in call flow

In the preceding figure, the client sends an **addUser** dial-in request, over SIP INFO, to the focus. The focus authorizes it and then forwards it to the MCU. This is shown in steps 1a and 1b.

The MCU processes the **addUser** dial-in request and then responds with an **addUser** dial-in response to the focus, which is forwarded by the focus to the client, as shown in steps 1c and 1d.

Subsequent to processing the **addUser** dial-in response, the client establishes a session with the MCU. The figure titled addUser Dial-in call flow assumes a SIP-based MCU. Thus, in step 2a, the client sends an INVITE request to the MCU. This specification does not specify the actual contents of the SIP request body. Extensions to this specification can define the actual contents, such as the SDP format. The MCU accepts the INVITE request, processes the request, and then responds to it in step 2b.

At this point, the client and the MCU might negotiate media. Extensions to this specification can define how media negotiation is done.

At the end of media negotiation, the MCU notifies the focus that the user has connected, as shown in step 3a, which in turn is propagated to all conference watchers, as shown in step 3b.

The dial-in request can be sent by the client or by the focus itself to the MCU. It specifies the user who is attempting to join the MCU. The dial-in response indicates whether the MCU is prepared to accept the user's join request and also conveys extra information necessary to attempt the join. Typically, the dial-in request is followed by a client-to-MCU- specific signaling handshake such as an INVITE. This is a user level command.

The MCU MAY fail the **addUser** dial-in request if it has not indicated successful activation, for example by publishing its **entity-view** element in the conference document. MCU implementers can specify extensions to the **addUser** dial-in protocol.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.10.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.10.2 Timers

Timers are specified in section 3.4.

3.10.3 Initialization

Initialization steps are specified in section 3.4.

3.10.4 Higher-Layer Triggered Events

3.10.4.1 Client Role

Unless otherwise specified in the following subsections, the rules for constructing the outgoing **addUser** dial-in request are as specified in section 3.4.

3.10.4.1.1 Constructing an Outgoing addUser Dial-in Request

The **entity** attribute of the **user** element of the **addUser** dial-in request MUST be the same as the SIP URI in the **From** header field. In other words, the dial-in request MUST be for the initiator and cannot be done on behalf of some other user.

3.10.4.2 MCU Role

MCU extensions to this document can specify the actual processing behavior of the **addUser** dial-in request.

3.10.5 Message Processing Events and Sequencing Rules

3.10.5.1 Client Role

Unless otherwise specified in the following subsections, the rules for message processing and sequencing are as specified in section 3.4.

3.10.5.1.1 Processing an addUser Dial-in Response

When an **addUser** dial-in response is received, the client SHOULD parse and extract the **connection-info** element to be used for constructing the outgoing INVITE request, if applicable, as defined in the next subsection.

3.10.5.1.2 Constructing an Outgoing SIP INVITE Request

Unless otherwise specified in this section, the rules for constructing and sending an INVITE request are as specified in [RFC3261].

The outgoing INVITE SHOULD be constructed using the following rules:

- The SIP URI in the **From** header field MUST be set to the client's SIP URI. This MUST also be the same as the SIP URI specified in the **entity** attribute of the **user** element of the dial-out request.
- The SIP URI in the **To** header field SHOULD be set to the MCU-Conference-URI extracted from the **addUser** dial-in response specified in section 2.2.3.16.
- The SIP **Request-URI** header field SHOULD be set to the SIP URI in the **To** header field.

Extensions can define additional rules for session establishment between the MCU and the client.

3.10.5.2 MCU Role

Unless otherwise specified in the following subsection, the rules for message processing and sequencing are as specified in section 3.4.

3.10.5.2.1 Constructing an addUser Dial-in Response

An MCU SHOULD generate an **addUser** dial-in response after it has finished processing the **addUser** request.

The **connection-info** element of the **addUser** dial-in response SHOULD be populated with key-value pairs that are then used by the client to initiate a connection. If the MCU supports user session establishment through SIP, it is recommended that it adds the following **keys** to the **connection-info** element, as specified in section 2.2.3.16.

- Mcu-Server-Uri
- MCU-Conference-URI

The MCU MUST generate a user state change notification on successful processing of a dial-in request.

Dial-in examples are given in section 4.

3.10.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.10.7 Other Local Events

None.

3.11 Conference Control – getConference Command

This section follows the product behavior described in endnote <26>.

The **getConference** command is used to retrieve the current active roster state. This is a conference level command.

The difference between the **getConference** request sent to the Focus Factory, specified in [MS-CONFPRO], and the one sent to the focus is that the one sent to the Focus Factory retrieves the static provisioning information of the conference, while the one sent to the focus retrieves the current roster state of an active conference.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.11.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.11.2 Timers

Timers are specified in section 3.4.

3.11.3 Initialization

Initialization steps are specified in section 3.4.

3.11.4 Higher-Layer Triggered Events

3.11.4.1 Focus Role

The term focus is used synonymously with server in the following section.

The focus SHOULD first apply the processing rules specified in section 3.4. After that, the focus SHOULD generate and send the full notification document, as specified in section 3.3.5.2.1.

If an **encryption-key** is specified, and a **conference-key** exists for the conference, encrypt the **conference-key** using the public key from the specified x509-certificate and include it in the **getConference** response.

If an **encryption-key** is specified, and a web join URL exists for the conference, encrypt the URL using the public key from the specified x509-certificate and include it in the **conf-uris** element as part of the **getConference** response. X.509 is specified in [RFC3280].

3.11.5 Message Processing Events and Sequencing Rules

None.

3.11.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.11.7 Other Local Events

None.

3.12 Conference Control – modifyEndpoint Command

The **modifyEndpoint** command is used to modify the **endpoint** extensions of a **participant (2)**. This is a user level command.

As an example, the recording notification feature can use this command for communication between client and server. The client is responsible to report recording state change, and the server is responsible to receive and notify the recording state change.

The **endpoint** entities are maintained by the focus, and each entity reports the extensions of an **endpoint** of a participant (2) that is independent of the other entity.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows with the recording notification feature as an example are given in section 4.

3.12.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.12.2 Timers

Timers are specified in section 3.4.

3.12.3 Initialization

Initialization steps are specified in section 3.4.

3.12.4 Higher-Layer Triggered Events

None.

3.12.5 Message Processing Events and Sequencing Rules

None.

3.12.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.12.7 Other Local Events

None.

3.13 Conference Control - setLobbyAccess Command

This section follows the product behavior described in endnote <27>.

The setLobbyAccess command is used to admit or deny users from the conference lobby. This is a user level command.

Unless specified otherwise, the protocol details specified in section 3.4 SHOULD be used for command processing.

Detailed command call flows are given in section 4.

3.13.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.13.2 Timers

Timers are specified in section 3.4.

3.13.3 Initialization

Initialization steps are specified in section 3.4.

3.13.4 Higher-Layer Triggered Events

3.13.4.1 Focus Role

Unless otherwise specified, the rules for message processing and sequencing are as specified in section 3.4

The focus should process each user specified in the setLobbyAccess request separately. The request is considered a success so long as the requestor was authorized and there was not an internal failure processing the request, even if no users were actually admitted or denied access because of the command.

If the value of the **access** element of the request is "granted", the command is meant to admit users from the lobby. Each **status.reason** attribute in the response is determined as follows:

- "conferenceFull": The focus SHOULD enforce any implementation-defined limits on the number of participants admitted to a conference and assign this value to any participants who are not admitted because of size limitations. The participant is neither admitted nor denied access to the conference. Such participants remain connected to the conference in the lobby and may be admitted later if space becomes available. If some, but not all, of the specified users will fit in the conference, the implementation is free to choose which users to admit and which to leave in the lobby.
- "userDoesntExist": There was no participant connected to the conference, in lobby or not, with the given URI. This can happen if the participant left or was denied access by another command before the command was processed.
- "alreadyGranted": The participant was already admitted to the conference by a previous command.
- "success": The participant was admitted from the lobby and granted a place in the conference.

If the value of the **access** element of the request is "denied", the command is meant to eject users from the lobby. Each **status.reason** attribute in the response is determined as follows:

- "userDoesntExist": As for the "granted" case.

- "alreadyGranted": The participant was already admitted to the conference by a previous command. setLobbyAccess cannot be used to eject a participant already admitted to the conference.
- "success": The participant was ejected from the lobby.

3.13.5 Message Processing Events and Sequencing Rules

3.13.5.1 Focus Role

Each participant admitted via this command MUST receive a full conference notify constructed as specified in section 3.3.5.2.1.

Each participant ejected via this command MUST have all signaling and subscription dialogs with the conference terminated, as specified in section 3.7.5.1.

Other participants in the conference that are watching the roster and not in the lobby after the completion of this command MUST receive a notification indicating the change of lobby status for users whose status is changed by this command. This notification MAY use partial notification semantics.

3.13.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.13.7 Other Local Events

None.

3.14 Conference Control - modifyConference Command

This section follows the product behavior described in endnote <28>.

The **modifyConference** command is used to change the state of MCUs in the conference. This is a conference level command.

3.14.1 Abstract Data Model

The abstract data model for processing conference control commands is specified in section 3.4.

3.14.2 Timers

Timers are specified in section 3.4.

3.14.3 Initialization

The abstract data model for processing conference control commands is specified in section 3.4.

3.14.4 Higher-Layer Triggered Events

3.14.4.1 MCU Role

MCU extensions to this document can specify the actual processing behavior of the **modifyConference** request.

3.14.5 Message Processing Events and Sequencing Rules

None.

3.14.6 Timer Events

The rules for timer event processing are specified in section 3.4.

3.14.7 Other Local Events

None.

4 Protocol Examples

4.1 Simple Join

A standard call flow sequence is shown in the following figure. The sequence shown is based on the protocol sequence described in section 3.2.

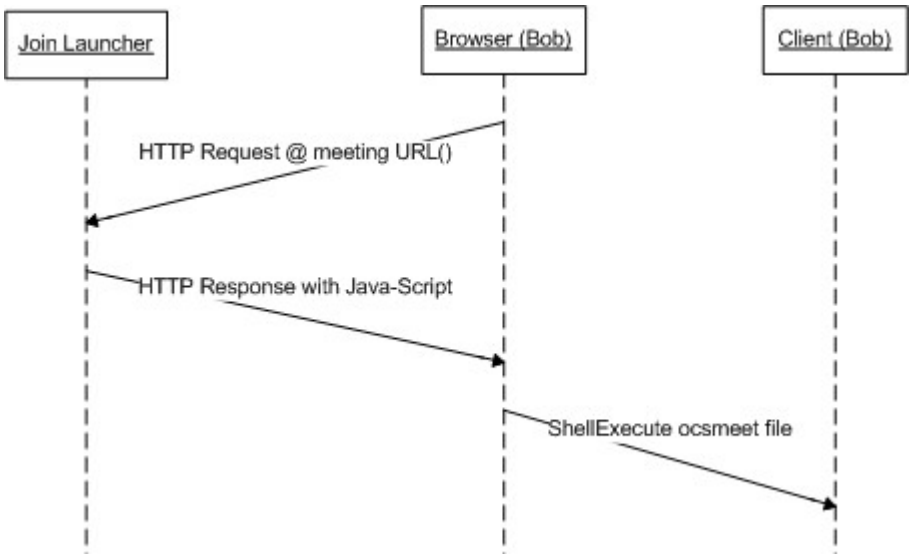


Figure 11: Standard call flow sequence

When user "Bob" initiates a Simple Join by clicking on the conferencing join web URL, Bob's browser, which is Internet Explorer in this case, sends an HTTP request to the Join Manager as shown in the following example:

```
GET /meet/bob/MJMVY7RF HTTP/1.1
Accept: image/gif, image/jpeg, image/pjpeg, application/x-ms-application, application/vnd.ms-
xpsdocument, application/xaml+xml, application/x-ms-xbap, application/vnd.ms-excel,
application/vnd.ms-powerpoint, application/msword, application/x-shockwave-flash
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.0; Trident/4.0; GTB6;; CWADS32;
SLCC1; .NET CLR 2.0.50727; Tablet PC 2.0; .NET CLR 1.1.4322; InfoPath.2; .NET CLR 3.5.21022;
MS-RTC LM 8; .NET CLR 3.5.30729; .NET CLR 3.0.30618)
Accept-Encoding: gzip, deflate
Host: www.fabrikam.com
Connection: Keep-Alive
Cookie: MC1=GUID=a0e6a97cda7641a496b14ca0665f35a7&HASH=a0e6&LV=20099&V=3;
WT_FPC=id=131.107.0.106-1797821888.30059372:lv=1265936990571:ss=1265936990571
```

The Join Manager parses the conferencing join web URL, creates the ocsmeet XML document string, and sends an HTTP response with embedded Java-Script to Bob's browser, as shown in the following example:

```
HTTP/1.1 200 OK
Cache-Control: private
Content-Type: text/html; charset=utf-8
```

```
Server: Microsoft-IIS/7.0
X-AspNet-Version: 2.0.50727
X-Powered-By: ASP.NET
Date: Thu, 19 Aug 2010 20:58:47 GMT
Content-Length: 12272

<!--NOTE: If DOCTYPE element is present, it causes the iFrame to be displayed in a small-->
<!--portion of the browser window instead of occupying the full browser window.-->
<html xmlns="http://www.w3.org/1999/xhtml" class="reachJoinHtml">
<head>
    <title>Microsoft Lync 2010</title>
    <script type="text/javascript">
        var reachURL =
"https://poolaweb.fabrikam.com/Reach/Client/WebPages/ReachJoin.aspx?xml=PD94bWwgdmVyc2lvbj0iM
S4wIiBlbmNvZGluZz0idXRmLTgiPz48Y29uZi1pbmZvIHhtbG5zOnhzaT0iaHR0cDovL3d3dy53My5vcmcvMjAwMS9YTU
xTY2hlbWEtaW5zdGFuY2UiIHhtbG5zOnhzZD0iaHR0cDovL3d3dy53My5vcmcvMjAwMS9YTUxTY2hlbWEiIHhtbG5zPSJ
odHRwOi8vc2NoZW1hcy5taWNyb3NvZnQuY29tL3J0Yy8yMDA5LzA1L3NpbXBsZWpvaW5jb25mZG9jIj48Y29uZi11cmk-
c2lwOmpvZ2lsYmVyMUBjbGllbnRlZS5ydG1wLnNlbGZob3N0LmNvcnAubWljcm9zb2Z0LmNvbTtncnV1O29wYXF1ZT1hc
HA6Y29uZjpmb2N1czppZDpIOTdHMTBXODwvY29uZi11cmk-PHNlcnZlci10aW1lPjcuODE1Mjwvc2VydmVyLXRpbWU-
PG9yaWdpbmFsLWluY29taW5nLXVybD5odHRwczovL21lZXQuY2xpZW50ZWUucnRtC5zZWxmaG9zdC5jb3JwLm1pY3Jvc
29mdC5jb20vam9naWxiZXIxL0g5N0cxMFc4PC9vcmlnaW5hbC1pbmNvbWluZy11cmw-
PGNvbmYta2V5Pkg5N0cxMFc4PC9jb25mLWtleT48L2NvbmYtaW5mbz4!";

        var escapedXML = "&lt;?xml version=&quot;1.0&quot; encoding=&quot;utf-
8&quot;?&gt;&lt;conf-info xmlns:xsi=&quot;http://www.w3.org/2001/XMLSchema-instance&quot;
xmlns:xsd=&quot;http://www.w3.org/2001/XMLSchema&quot;
xmlns=&quot;http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc&quot;&gt;&lt;conf-
uri&gt;sip:jogilber1@clientee.rtmp.selfhost.corp.microsoft.com;gruu;opaque=app:conf:focus:id:
H97G10W8&lt;/conf-uri&gt;&lt;server-time&gt;7.8152&lt;/server-time&gt;&lt;original-incoming-
url&gt;https://meet.fabrikam.com/Bob/H97G10W8&lt;/original-incoming-url&gt;&lt;conf-
key&gt;H97G10W8&lt;/conf-key&gt;&lt;/conf-info&gt;";
        var showJoinUsingLegacyClientLink = "True";
        var urlCrackingError = "False";
        var currentLanguage = "en-US";
    </script>
    <script type="text/javascript" src="/meet/JavaScript/Utilities.js"></script>
    <script type="text/javascript" src="/meet/JavaScript/PluginLoader.js"></script>
    <script type="text/javascript" src="/meet/JavaScript/Launch.js"></script>
    <link rel="Stylesheet" type="text/css" href="/meet/Resources/ReachClient.css" />
</head>
<body onload="mainWindow.OnLoad();" class="reachJoinBody">
/* HTML BODY */
</body>
</html>
```

Bob's browser receives the response, the Java-Script is executed on the computer starting from the **OnLoad()** method in the Launch.js, it does the necessary checks, loads the appropriate Active-X Controls or Firefox Plug-ins and eventually calls into a method into the Active-X Control or Firefox Plug-in. The API then writes the content of the ocsmeet XML document to a file in the temporary location on the computer, defined by the **GetTempFolder()** windows API, which is accessible to that user, and eventually calls **ShellExecute()** on that file path to start the client on Bob's computer.

4.2 Joining and Leaving a Conference

A standard call-flow sequence is shown in the following figure. This sequence is based on the protocol sequence described in section 3.2.

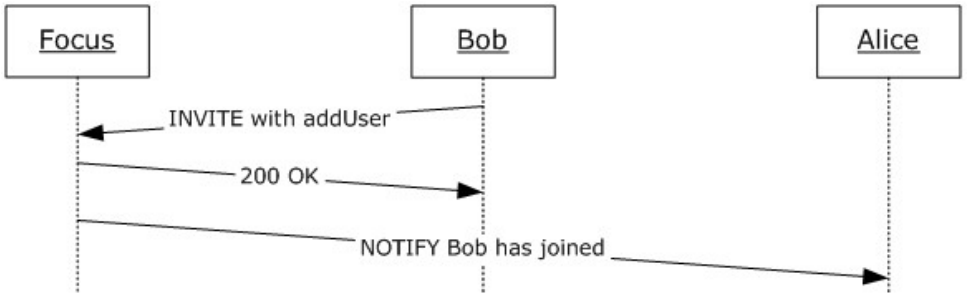


Figure 12: Joining a conference

4.2.1 Joining a Conference

Conference-aware client "Bob" joins a conference by establishing a SIP signaling dialog with the focus. It first sends an INVITE request as follows.

```
INVITE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 1 INVITE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    from="sip:bob@fabrikam.com" requestId="1"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <addUser>
    <conferenceKeys confEntity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C
"/>
    <user entity="sip:bob@fabrikam.com">
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{339F927D-6AD4-4090-9104-8414B99EE045}" />
    </user>
  </addUser>
</request>
```

When the **addUser** is accepted, the focus responds with the granted role in the 200 OK response.

```
SIP/2.0 200 Invite dialog created
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
```

```
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 1 INVITE
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 1095
Content-Type: application/cccp+xml
Allow: INVITE, BYE, ACK, CANCEL, INFO, UPDATE
Session-Expires: 7200;refresher=uac
Require: timer
Supported: timer

<response requestId="1" C3PVersion="1"
    from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    to="sip:bob@fabrikam.com" code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <addUser>
    <conferenceKeys confEntity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C
"/>
    <user entity="sip:bob@fabrikam.com">
      <roles>
        <entry>presenter</entry>
      </roles>
    </user>
  </addUser>
</response>
```

It then notifies the existing conference participants that the user has joined the conference. In the following example, user "alice@fabrikam.com" is assumed to have already joined and subscribed to the conference. An actual subscription example is shown in section 4.3.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 12 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600
<... Content-Length snipped ...>

<conference-info
    entity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    state="partial" version="2"
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="full">
```

```
      <display-text>Bob Freer</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{339F927D-6AD4-4090-9104-8414B99EE045}"
        msci:session-type="focus"
        msci:endpoint-
uri="sip:bob@fabrikam.com;opaque=user:epid:AD0UTS5DclOh9zyK1XWK2AAA;gruu">
      <status>connected</status>
    </endpoint>
    </user>
  </users>
</conference-info>
```

In this same example, if Bob is joining the conference on behalf of Carol, the INVITE and the BENOTIFY are as follows.

```
INVITE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 1 INVITE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
p-session-on-behalf-of: sip:carol@fabrikam.com
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    from="sip:bob@fabrikam.com" requestId="1"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <addUser>
    <conferenceKeys confEntity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C
"/>
    <user entity="sip:bob@fabrikam.com">
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}">
       <session-on-behalf-of>
       <entity> sip:carol@fabrikam.com </entity>       </session-on-behalf-of>
       </endpoint>
    </user>
  </addUser>
</request>


BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
```

```
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 12 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600
<... Content-Length snipped ...>

<conference-info
    entity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    state="partial" version="2"
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="full">
      <display-text>Bob Freer</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{339F927D-6AD4-4090-9104-8414B99EE045}"
        msci:session-type="focus"
        msci:endpoint-
uri="sip:bob@fabrikam.com;opaque=user:epid:AD0UTS5DclOh9zyK1XWK2AAA;gruu">
      <status>connected</status>
       <session-on-behalf-of>
       <entity> sip:carol@fabrikam.com </entity>       </session-on-behalf-of>
    </endpoint>
    </user>
  </users>
</conference-info>
```

4.2.2 Updating the Dialog

A standard call-flow sequence for updating a signaling dialog with the focus is shown in the following figure, and is based on a negotiated Session-Timer. The sequence shown here is based on the protocol sequence described in section 3.2.

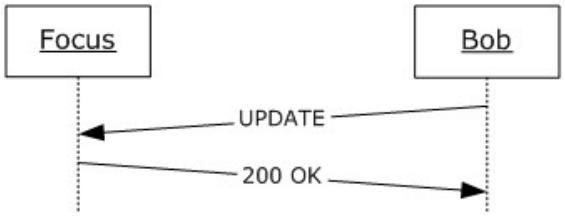


Figure 13: Updating a signaling dialog

The client sends an UPDATE request to the focus to refresh the existing signaling dialog.

```
UPDATE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
```

```
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 0
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 5 UPDATE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
```

After processing the UPDATE, the focus responds with a 200 OK.

```
SIP/2.0 200 OK
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 5 UPDATE
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
Session-Expires: 7200;refresher=uac
Require: timer
Supported: timer
```

4.2.3 Leaving a Conference

A standard call-flow sequence for leaving a conference is shown in the following figure. The client sends a BYE request to leave the conference. The focus notifies other participants in the conference that the participant has left. The sequence shown here is based on the protocol sequence described in section 3.2.

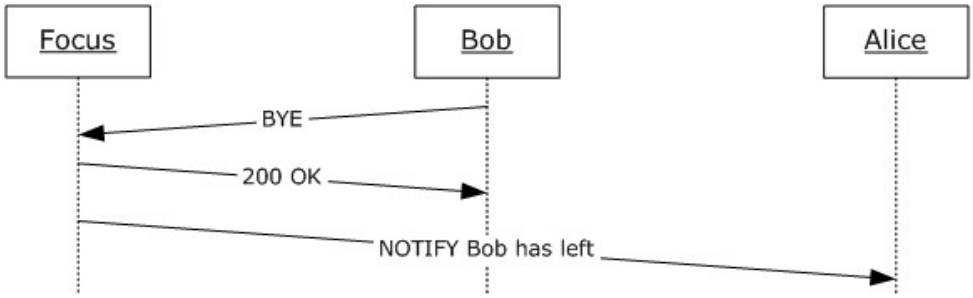


Figure 14: Leaving a conference

```
BYE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 0
```

```
Via: SIP/2.0/TLS 10.1.2.50:4237
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 10 BYE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
```

The focus processes the participant leave request and responds with a 200 OK.

```
SIP/2.0 200 OK
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 10 BYE
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
```

The focus then sends a notification to the watchers in the conference indicating that Bob has left the conference.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 12 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600
<... Content-Length header snipped ... >

<conference-info entity="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
state="partial" version="2"
xmlns="urn:ietf:params:xml:ns:conference-info"
xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="deleted"/>
  </users>
</conference-info>
```

4.3 Subscribing to a Conference

A standard call-flow sequence for subscribing to conference notifications is shown in the figure titled Establishing a subscription in section 4.3.1. The sequence shown here is based on the protocol sequence described in section 3.3.

4.3.1 Establishing a Subscription

The client sends a SIP SUBSCRIBE request to the focus and establishes a subscription dialog with the focus. The focus generates and sends notifications to the client whenever the conference state changes.

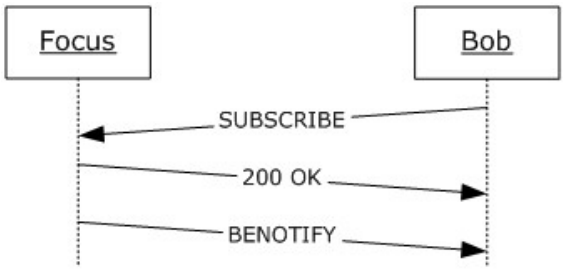


Figure 15: Establishing a subscription

```
SUBSCRIBE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: conference
Accept: application/conference-info+xml
Supported: com.microsoft.autoextend
Supported: ms-benotify
Proxy-Require: ms-benotify
Content-Length: 0
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
```

The focus processes the subscription and responds with a 200 OK.SIP/2.0 200 OK

```
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 1 SUBSCRIBE
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
```

```
subscription-state: active;expires=3618
```

The focus then sends the first full notification to this participant, listing the complete conference state. BENOTIFY is used in this example because the client indicated support for the BENOTIFY request. Otherwise, a SIP NOTIFY request would be sent by the focus to the client.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 2 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
  xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
  xmlns:msim="http://schemas.fabrikam.com/rtc/2005/08/imconfinfoextensions"
  xmlns:ci="urn:ietf:params:xml:ns:conference-info"
  xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
  entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
  state="full" version="4">
  <conference-description>
    <conf-uris>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C</uri>
        <display-text>chat</display-text>
        <purpose>chat</purpose>
      </entry>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:meeting:id:5D3747C</uri>
        <display-text>meeting</display-text>
        <purpose>meeting</purpose>
      </entry>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:5D3747C</uri>
        <display-text>audio-video</display-text>
        <purpose>audio-video</purpose>
      </entry>
<entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf: applicationsharing:id:5D3747C</uri>
        <display-text>applicationsharing</display-text>
        <purpose>applicationsharing</purpose>
</entry>

    </conf-uris>
  </conference-description>
  <users state="full">
    <user entity="sip:alice@fabrikam.com" state="full">
      <display-text>Alice Gates</display-text>
        <roles>
```

```
          <entry>presenter</entry>
        </roles>
        <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}" msci:session-type="focus"
msci:endpoint-uri="sip:alice@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;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:5D3747C">
      <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:5D3747C">
      <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: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>
      </msci:entity-state>
    </msci:entity-view>
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
      <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:5D3747C">
```

```
      <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:entity-view ci:state="full" entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:
applicationsharing:id:5D3747C ">
<msci:entity-capabilities>
<cis:separator/>
<msas:capabilities>
<msas:control-permission>ActiveDirectoryUsers</msas:control-permission>
</msas:capabilities>
</msci:entity-capabilities>
<msci:entity-state>
<msci:locked>false</msci:locked>
<msci:media>
<entry label="applicationsharing">
<type>applicationsharing</type>
</entry>
</msci:media>
</msci:entity-state>
</msci:entity-view>

  </msci:conference-view>
</conference-info>
```

4.3.2 Terminating the Subscription

A participant terminates the subscription with the focus by sending a SUBSCRIBE request with an **Expires** header field with the value "0", as described in [RFC3265]. A sample follows. The call flow is shown in the following figure.

Figure 16: Terminating the subscription

```
SUBSCRIBE sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 1 SUBSCRIBE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Event: conference
```

```
Accept: application/conference-info+xml
Supported: com.microsoft.autoextend
Supported: ms-benotify
Proxy-Require: ms-benotify
Content-Length: 0
Expires: 0
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
```

The focus then sends a 200 OK response.

```
SIP/2.0 200 OK
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 1 SUBSCRIBE
Expires: 0
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
```

4.4 Basic Conference Control

4.4.1 modifyConferenceLock

The current lock state of the conference is indicated in the conference roster in the **conference-view** container, as follows. In this example, there are three entities:

1. The focus
2. The **Audio/Video Multipoint Control Unit (AVMCU)**
3. The **IM MCU**.

Only the relevant code is shown. In this example, the AVMCU does not implement conference-locking, and hence does not report lock state in its **entity-view**. The focus and the IM MCU implement support for conference-locking and they report the current lock state in their respective **entity-view** sections.

```
<msci:conference-view ci:state="full">
  <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C">
    <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:5D3747C">
    <msci:entity-state>
      <!-- snipped -->
    </msci:entity-state>
```

```
  </msci:entity-view>
  <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
    <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:applicationsharing:id:5D3747C">
    <msci:entity-state>
      <msci:locked>false</msci:locked>
    </msci:entity-state>
  </msci:entity-view>
</msci:conference-view>
```

A presenter participant then sends a **modifyConferenceLock** command to lock the conference. The full call flow for that command is shown in the following figure. Note that the focus-to-MCU call flow is not shown in the following figure because it is outside the scope of this specification.

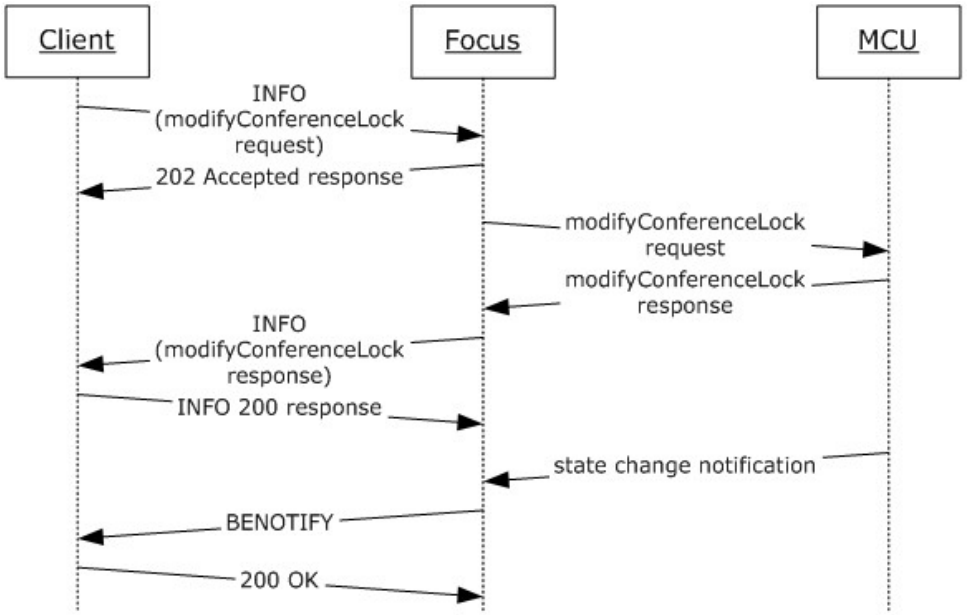


Figure 17: modifyConferenceLock call flow

The client sends a **modifyConferenceLock** request to the focus. In this example, Bob is a presenter who has the privileges to lock the conference.

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 10 INFO
```

```
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
    from="sip:bob@fabrikam.com" requestId="12"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <modifyConferenceLock>
    <conferenceKeys
      confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
    <locked>true</locked>
<separator xmlns="urn:ietf:params:xml:ns:conference-info-separator" />
<admission-policy>openAuthenticated</admission-policy>
<autopromote>2147483648</autopromote>
   <pstn-lobby-bypass>false</pstn-lobby-bypass>
  </modifyConferenceLock>
</request>
```

The focus validates the request and then responds with a 202 Accepted, indicating that the command is being processed.

```
SIP/2.0 202 Accepted
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 10 INFO
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
```

The focus applies the requested lock state and then sends a response, as follows.

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 50 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
<response requestId="12" C3PVersion="1"

from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
```

```
    to="sip:bob@fabrikam.com"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <modifyConferenceLock>
    <conference-info entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
    <locked>true</locked>
    <separator xmlns="urn:ietf:params:xml:ns:conference-info-separator" />
    <admission-policy>openAuthenticated</admission-policy>
    <autopromote>2147483648</autopromote>
    <pstn-lobby-bypass>false</pstn-lobby-bypass>
  </modifyConferenceLock>
</response>
```

The client responds with a 200 OK to indicate that the INFO response was received.

```
SIP/2.0 200 OK
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 0
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 50 INFO
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"
```

All watchers receive a notification from the focus, indicating that the conference state has changed.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 13 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    state="partial" version="13">
  <msci:conference-view ci:state="full">
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C">
      <msci:entity-state>
        <msci:locked>true</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:5D3747C">
      <!-- snipped -->
    </msci:entity-view>
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
      <msci:entity-state">
        <msci:locked>false</msci:locked>
        <!-- snipped -->
      </msci:entity-state>
    </msci:entity-view>
  </msci:conference-view>
</conference-info>
```

The client receives a second notification from the focus, indicating that the IM MCU conference lock state has changed.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 14 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"

entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    state="partial" version="14">
  <msci:conference-view ci:state="full">
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C">
      <msci:entity-state>
        <msci:locked>true</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:5D3747C">
      <!-- snipped -->
    </msci:entity-view>
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
      <msci:entity-state">
        <msci:locked>true</msci:locked>
          <!-- snipped -->
      </msci:entity-state>
    </msci:entity-view>
  </msci:conference-view>
```

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

A **modifyConferenceLock** "otherFailure" response follows for illustrative purposes.

```
< modifyConferenceLock reason="otherFailure">
    <mscp:diagnostics-info>
      <mscp:entry>
        <mscp:key>ms-diagnostics-public</mscp:key>
        <mscp:value>3126;reason="Unauthorized - The user does not have the privilege for the
requested operation";source="ocs.fabrikam.com"
        </mscp:value>
      </mscp:entry>
    </mscp:diagnostics-info>
 <ci:conference-info entity=" sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
/>
    <locked>false</locked>
  </modifyConferenceLock>
```

4.4.2 modifyUserRoles

The current role of a participant is reported in the conference roster in the **user** container, as follows.

```
<user entity="sip:cathy@fabrikam.com" state="full">
  <display-text>Cathy Baker</display-text>
  <roles>
    <entry>attendee</entry>
  </roles>
  <!-- snipped -->
</user>
```

A presenter participant then sends a **modifyUserRoles** command to promote another participant. The full call flow for that command is shown in the following figure. Note that the focus-to-MCU call flow is not shown in the following figure because it is outside the scope of this specification.

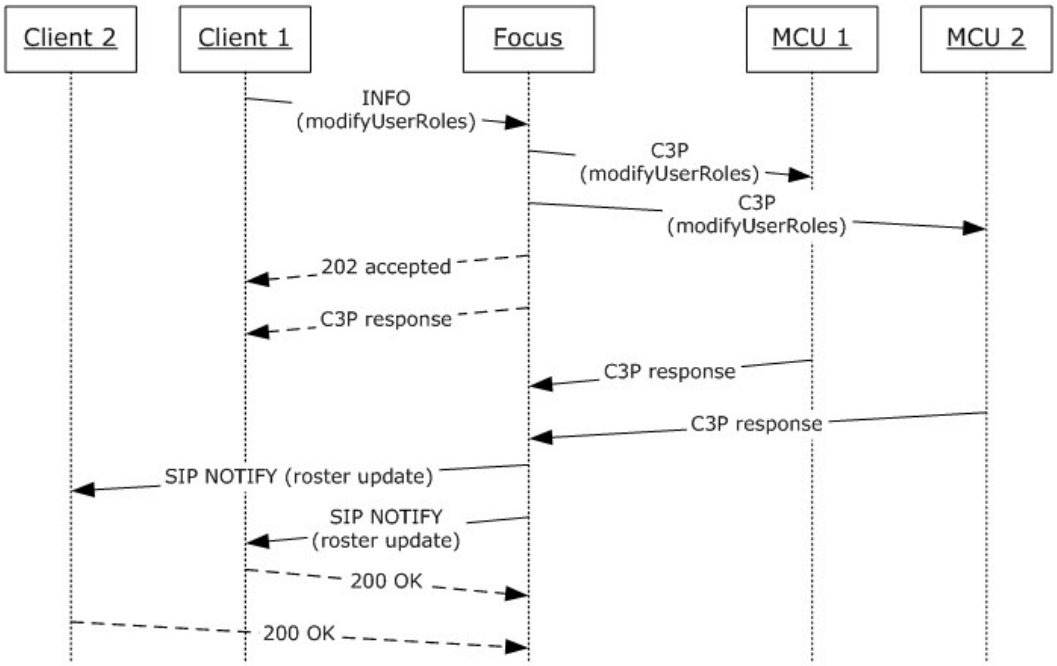


Figure 18: modifyUserRoles call flow

The client sends a **modifyUserRoles** request to the focus. In this example, Bob is a presenter who can promote other conference participants.

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 11 INFO
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    from="sip:bob@fabrikam.com" requestId="13"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions">
  <modifyUserRoles>
    <userKeys
     confEntity=
"sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
      userEntity="sip:cathy@fabrikam.com"/>
```

```
    <user-roles xmlns="urn:ietf:params:xml:ns:conference-info">
      <entry>presenter</entry>
    </user-roles>
  </modifyUserRoles>
</request>
```

The focus validates the request and then responds with a 202 Accepted, indicating that the command is being processed.

```
SIP/2.0 202 Accepted
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 11 INFO
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0
```

The focus applies the requested role and then sends a response, as follows.

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 51 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"

<response requestId="13" C3PVersion="1"
from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
to="sip:bob@fabrikam.com"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions">
  <modifyUserRoles>
    <userKeys confEntity=
"sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
     userEntity="sip:cathy@fabrikam.com"/>
    <roles xmlns="urn:ietf:params:xml:ns:conference-info">
      <entry>presenter</entry>
    </roles>
  </modifyUserRoles>
</response>
```

The client responds with a 200 OK to the INFO request, which is not shown in the following example.

All watchers receive a notification from the focus, indicating that the user role has changed.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 15 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D34163772389 67F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info
     xmlns="urn:ietf:params:xml:ns:conference-info"
     xmlns:ci="urn:ietf:params:xml:ns:conference-info"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
     state="partial" version="16">
  <users state="partial">
    <user entity="sip:cathy@fabrikam.com" state="full">
      <display-text>Cathy Baker</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
        <!-- snipped -->
    </user>
  </users>
</conference-info>
```

4.4.3 deleteUser

The **deleteUser** command is sent by the client to the focus to delete the same or another participant from the conference.

In the following example, Client3 ejects Client2 from the conference. The following figure shows the call flow for this operation. Client3 is assumed to be a presenter who has the privileges to eject other conference participants. Client2 is assumed to be connected to the focus and MCU1. When the focus processes the **deleteUser** command, it terminates the dialogs with Client2 and then sends a notification to the watchers indicating that Client2 has left the focus. It also sends the **deleteUser** command to all of the MCUs in the conference.

The termination protocol between MCU1 and Client2 is not discussed because it is outside the scope of this specification. However, when Client2 leaves MCU1, MCU1 notifies the focus of that event. The focus then sends another notification to all watchers indicating that Client2 has left MCU1, which in effect means that Client2 is no longer connected to the conference, because it has already left the focus. In the following figure, the notifications from MCU2 to the focus are not shown for simplicity.

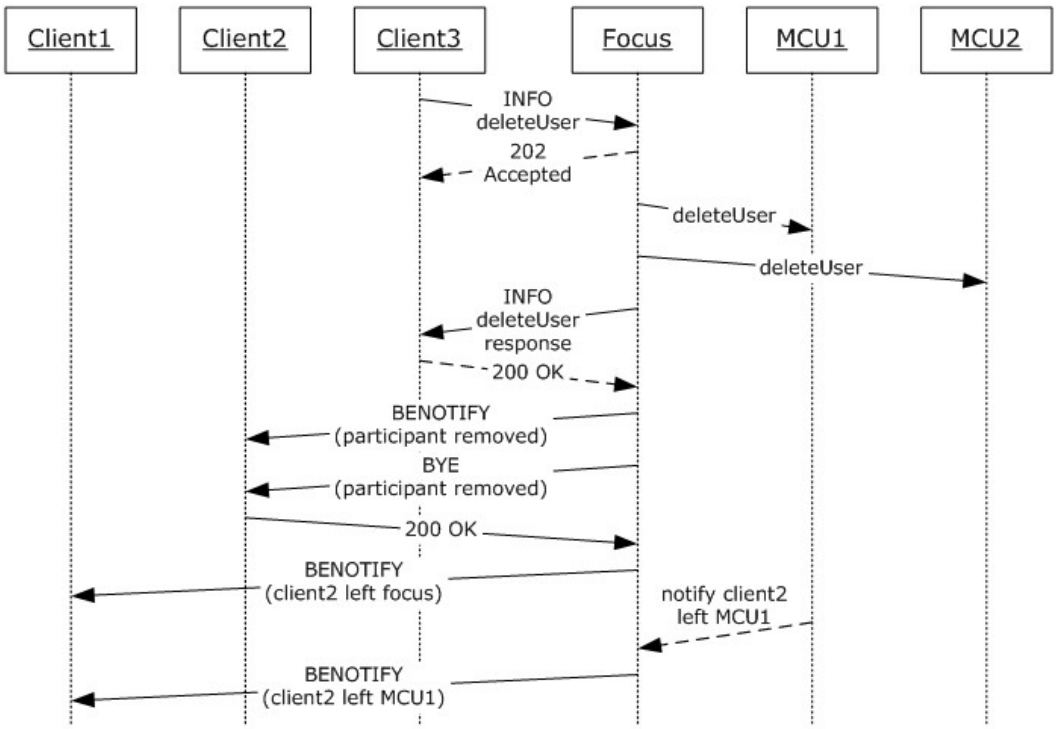


Figure 19: Ejecting a Participant from the conference

The client sends a **deleteUser** request to the focus.

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 11 INFO
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    from="sip:bob@fabrikam.com" requestId="13"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info">
  <deleteUser>
    <userKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
userEntity entity="sip:cathy@fabrikam.com"/>
  </deleteUser>
```

```
</request>
```

The focus responds with a 202 Accepted, which is not shown.

The focus terminates the subscription dialog with Cathy, as shown in the following example:

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:cathy@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 0
Event: conference
Call-ID: 7d6a36a36784cf58e7e7ab1a51deca2
CSeq:100 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: terminated;expires=0;reason=ParticipantRemoved
```

The focus terminates the signaling dialog with Cathy, as shown in the following example:

```
BYE sip:131.107.0.72:30505;transport=tls;ms-opaque=83dce3c514;ms-received-
cid=FCA7300;gridSIP/2.0
To: <sip:cathy@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 0
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 12 BYE
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
ms-diagnostics-public: 3118;reason="Participant Removed"
Reason: SIP;cause=481;text="Participant Removed"
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
```

The focus generates a notification to all watchers indicating that Cathy has left the focus. At this point, Cathy is still connected to the IM MCU.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 17 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
```

```
subscription-state: active;expires=3600

<conference-info
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    state="partial" version="17">
  <users state="partial">
    <user entity="sip:cathy@fabrikam.com" state="full">
      <display-text>Cathy Baker</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{9121CD6C-AFCC-4115-A5C4-089E4D17CCCE}" msci:session-type="chat"
msci:endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:uWgzMWHhRViEC9LK-xlgFAAA;gruu">
        <status>connected</status>
        <joining-method>dialed-in</joining-method>
         <!--snipped -->
      </endpoint>
    </user>
  </users>
</conference-info>
```

The focus sends an INFO request back to the user with a **deleteUser** C3P response, which is not shown.

The focus sends the **deleteUser** request to all of the MCUs in the conference, which is not shown.

The IM MCU ejects the user from the conference, which is not shown.

The IM MCU sends a notification to the focus, indicating that the user has left the IM MCU, which is not shown.

The focus sends a notification to all watchers indicating that Cathy has left the conference, because Cathy is no longer connected to either the focus or to any MCU.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 18 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
```

```
    state="partial" version="18">
  <users state="partial">
    <user entity="sip:cathy@fabrikam.com" state="deleted"/>
</users>
</conference-info>
```

A **deleteUser** "otherFailure" response is given here for illustrative purposes.

```
<deleteUser reason="otherFailure">
    <mscp:diagnostics-info>
      <mscp:entry>
        <mscp:key>ms-diagnostics-public</mscp:key>
        <mscp:value>3126;reason="Unauthorized - The user does not have the privilege for the
requested operation";source="ocs.fabrikam.com"
        </mscp:value>
      </mscp:entry>
    </mscp:diagnostics-info>
    <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
/>
    <ci:user entity="sip:cathy@fabrikam.com"/>
  </deleteUser>
```

4.4.4 deleteConference

The **deleteConference** command is sent by the client to the focus to deactivate the conference.

In the following figure, Client3 decides to end the conference and sends a **deleteConference** command to the focus. The focus terminates the signaling and subscription dialog with all clients. The focus also sends the **deleteConference** command to all MCUs in the conference.

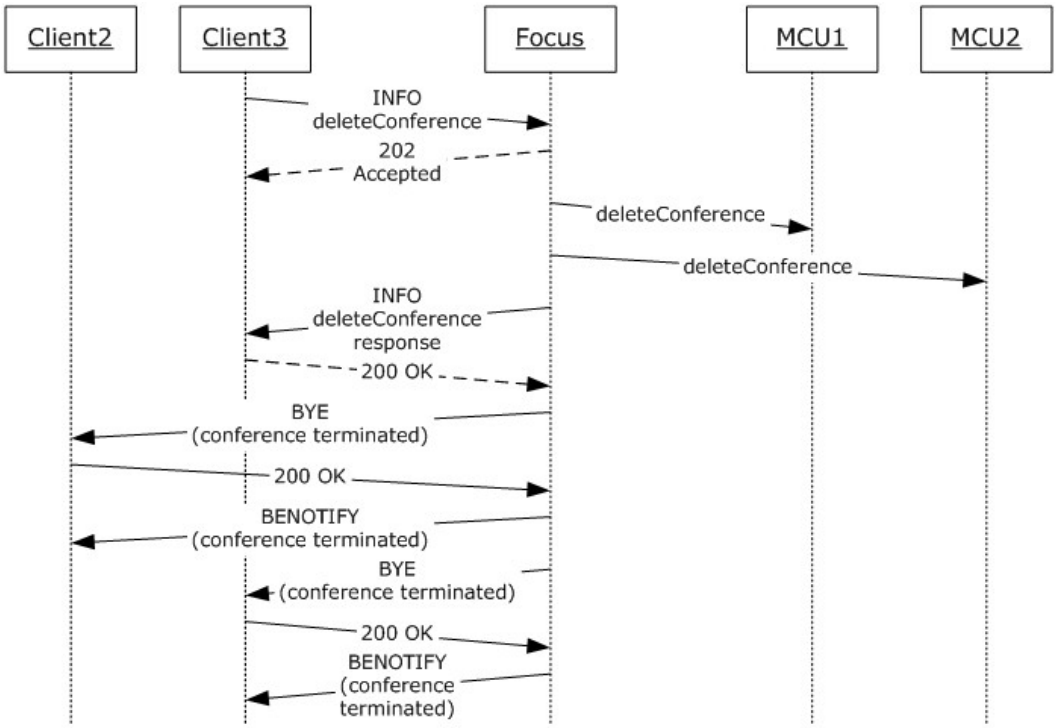


Figure 20: Terminating a conference

The termination messages are similar to the dialog termination messages shown in the **deleteUser** command, as specified in section 4.4.3.

4.4.5 addUser Dial-out

The **addUser** dial-out command is used to connect a participant to an MCU by making the MCU invite the participant.

The detailed call flow is shown in the following figure.

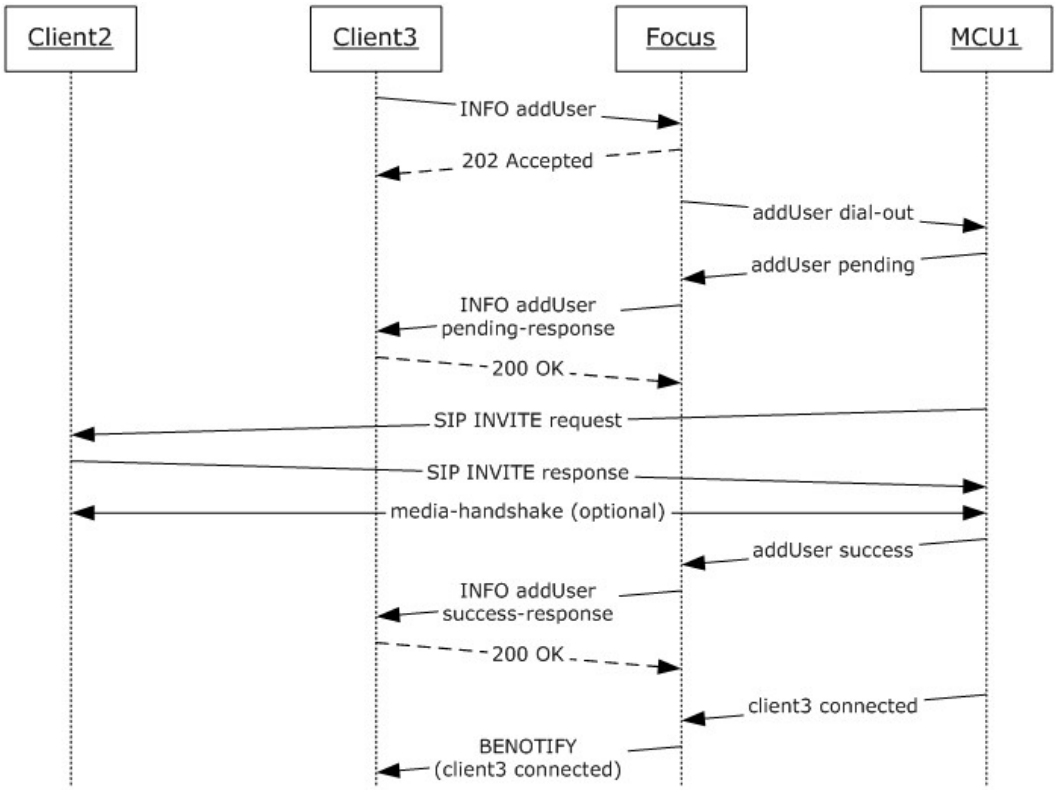


Figure 21: Connecting a Participant to an MCU

The client first sends an **addUser** dial-out request to the focus. In this example, MCU1 is an IM MCU with an MCU-Conference-URI with the value "sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C".

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 12 INFO
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    from="sip:bob@fabrikam.com"
    requestId="14"xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
```

```
    xmlns:mscp="http://schemas.fabrikam.com/rtc/2005/08/cccpextensions"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions">

  <addUser
mscp:mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
    <conferenceKeys
confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
    <user xmlns:ci="urn:ietf:params:xml:ns:conference-info" entity="sip:cathy@fabrikam.com">
      <display-text>Cathy Baker</display-text>
        <roles>
          <entry>presenter</entry>
        </roles>
        <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
msci:endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
          <joining-method>dialed-out</joining-method>
          <!-- other extension elements can follow -->
        </endpoint>
    </user>
  </addUser>
</request>
```

The focus authorizes this request and forwards it to the IM MCU. The actual protocol used between the focus and the IM MCU is not shown here.

The focus also responds with a 202 Accepted back to the client, which is not shown here.

In this example, the IM MCU sends a "pending" response to the focus. The actual protocol used between the IM MCU and the focus is not shown here.

The focus forwards the **addUser** "pending" response back to the client through the SIP INFO response, as shown in the following example:

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 52 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"

<response requestId="14" C3PVersion="1"
    from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    to="sip:bob@fabrikam.com"
    code="pending"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    responder="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C ">
  <!—valid addUser command body snipped -->
</response>
```

The client responds with a 200 OK response to this INFO request, which is not shown.

The MCU then initiates an INVITE request to the client, which is constructed using the rules specified in section 3.9.4.1.1.

```
INVITE sip:cathy@fabrikam.com SIP/2.0
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
From: <sip: Alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C
>;tag=44f0d128a3;epid=492a7ce35f
To: <sip:cathy@fabrikam.com>
Call-ID: 79f76d701f1844e496e2c1e37a26c213
CSeq: 1 INVITE
<!-- other headers snipped -->
Content-Type: application/sdp
Content-Length: 203

<!-- media specific content body snipped -->
```

The client accepts the INVITE and responds with an INVITE 200 OK response, which is not shown. The response looks similar to regular 200 OK responses to INVITE requests, as described in [RFC3261].

The MCU successfully completes the handshake and then responds with a success response to the focus. The focus forwards the **addUser** success response back to the client through the SIP INFO response, as shown in the following example:

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 53 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Content-Length: 0

<response requestId="14" C3PVersion="1"
    from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    to="sip:bob@fabrikam.com"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    responder="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C ">
  <!-- valid addUser command body snipped -->
</response>
```

The MCU then sends a notification to the focus indicating that the user has connected. This notification is also generally referred to as a user-connected notification. The focus sends a notification to all the watchers with the user-connected state change, as shown in the following example:

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 18 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    state="partial" version="18">
  <users state="partial">
    <user entity="sip:cathy@fabrikam.com" state="full">
      <!-- snipped -->
      <!—new connected endpoint with the IM MCU – the focus stamps the session-type attribute
-->
      <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
 msci:session-type="chat"
msci:endpoint-uri="sip:cathy@fabrikam.com;opaque=user:epid:123;gruu">
        <joining-method>dialed-out</joining-method>
        <!-- other extension elements can follow -->
      </endpoint>
    </user>
  </users>
</conference-info>
```

4.4.6 addUser Dial-in

The **addUser** dial-in command is used to connect a participant to an MCU. Typically, the first step is conference signaling, using an **addUser** dial-in, and then MCU-specific signaling, such as an INVITE handshake, and finally MCU-specific media signaling. The detailed call flow is shown in the following figure.

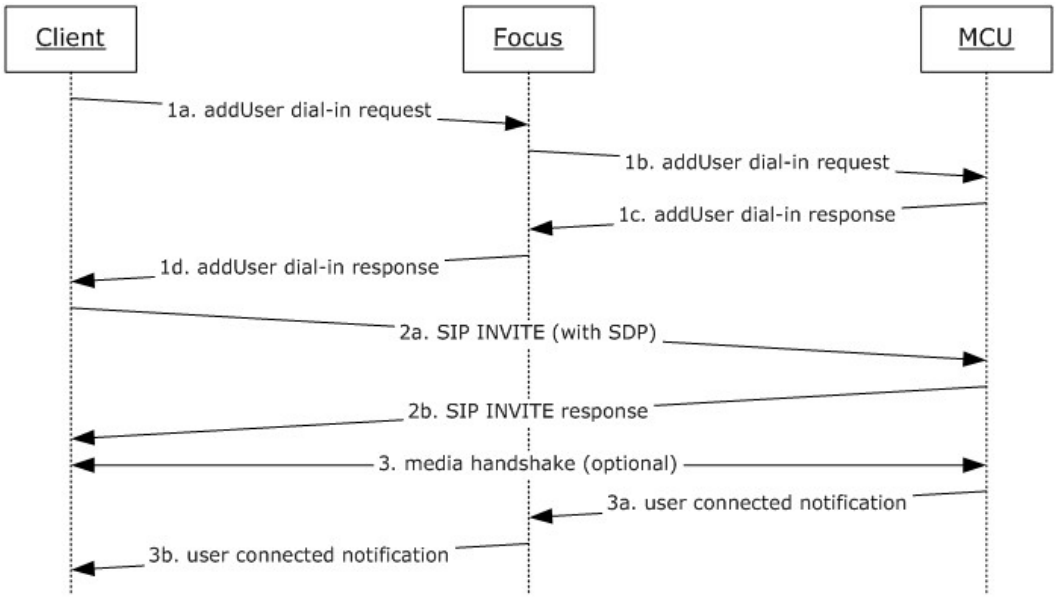


Figure 22: Connecting a Participant to an MCU

The client first sends an **addUser** dial-in request to the focus. In this example, the MCU is an IM MCU with an MCU-Conference-URI with the value "sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C".

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 12 INFO
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

<request C3PVersion="1"
    to="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    from="sip:bob@fabrikam.com" requestId="14"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:mscp="http://schemas.fabrikam.com/rtc/2005/08/cccpextensions"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions">

<addUser
mscp:mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
  <user xmlns:ci="urn:ietf:params:xml:ns:conference-info" entity="sip:bob@fabrikam.com">
    <display-text>Bob Baker</display-text>
```

```
    <roles>
      <entry>presenter</entry>
    </roles>
    <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:123;gruu">
    <joining-method>dialed-in</joining-method>
    <!-- other extension elements can follow -->
    </endpoint>
  </user>
</addUser>
</request>
```

The focus authorizes this request and forwards it to the IM MCU. The actual protocol used between the focus and the IM MCU is not shown here.

The focus also responds with a 202 Accepted back to the client, which is not shown here.

The MCU responds to the **addUser** dial-in response, which is not shown here. The focus forwards the **addUser** success response back to the client through the SIP INFO response, as shown in the following example:

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 53 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"

<response requestId="14" C3PVersion="1"
    from="sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    to="sip:bob@fabrikam.com"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    responder="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">

  <addUser>
    <conferenceKeys
confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"/>
    <user xmlns:ci="urn:ietf:params:xml:ns:conference-info" entity="sip:bob@fabrikam.com">
      <display-text>Bob Baker</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:123;gruu">
        <joining-method>dialed-in</joining-method>
        <!-- other extension elements can follow -->
      </endpoint>
    </user>
```

```
    <connection-info>
      <entry>
        <key>Mcu-Conference-Uri</key>
        <value>sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C</value>
      </entry>
      <entry>
        <key>Mcu-Server-Uri</key>
        <value>sip:chat.fabrikam.com:5061;transport=tls</value>
      </entry>
    </connection-info>
  </addUser>

</response>
```

The client then initiates an INVITE request to the MCU using the rules specified in section 3.9.4.1.1. Specifically, it extracts the MCU-Conference-URI value from the **addUser** response and uses that value to construct the SIP URIs in the **To** and **Request-URI** header fields of the outgoing request.

```
INVITE :alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C SIP/2.0
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
From: <sip: bob@fabrikam.com>;tag=44f0d128a3;epid=492a7ce35f
To: <sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C >
Call-ID: 79f76d701f1844e496e2c1e37a26c213
CSeq: 1 INVITE
<!-- other headers snipped -->
Content-Type: application/sdp
Content-Length: 203

<!-- media specific content body snipped -->
```

The MCU accepts the INVITE request and then responds with a 200 OK, which is not shown here.

The client and MCU then optionally perform media negotiation, if applicable, which is not shown here.

At the end of this handshake, the MCU sends a user-connected notification to the focus. The focus sends a notification to all the watchers with the user-connected state change, as shown in the following example:

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:Alice@fabrikam.com>;tag=ccb81c3509;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 18 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600
```

```
<conference-info
    xmlns="urn:ietf:params:xml:ns:conference-info"
    xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
    state="partial" version="18">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="full">
      <!-- snipped -->
      <!-- new connected endpoint with the IM MCU – the focus stamps the session-type
attribute -->
      <endpoint entity="{5CD3FC0A-05F7-4A17-A95B-430A28FC9EFA}"
 msci:session-type="chat"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:123;gruu">
        <joining-method>dialed-out</joining-method>
        <!-- other extension elements can follow -->
      </endpoint>
    </user>
  </users>
</conference-info>
```

4.4.7 getConference

The **getConference** command is sent by the client to the focus to retrieve the current active roster state of the conference.

Following is an example of a **getConference** request and a corresponding response. Note the response includes the encrypted conference key and web join URLs.

```
INFO sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C SIP/2.0
From: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
To: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 12 INFO
Contact: <sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu>
User-Agent: UCCP/2.0.6362.0 OC/2.0.6362.0 (Microsoft Office Communicator)
Supported: timer
ms-keep-alive: UAC;hop-hop=yes
Proxy-Authorization: NTLM qop="auth", realm="SIP Communications Service", opaque="99052D67",
crand="6475c82f", cnum="82", targetname="sip/ocs.fabrikam.com",
response="01000000e44de73c187afc887f8f5ef3"

<request C3PVersion="1"
         requestId="5"
         from="sip:alice@contoso.com"
         to="sip:alice@contoso.com;gruu;opaque=app:conf:focus:id:5D3747C "
         xmlns="urn:ietf:params:xml:ns:cccp"
         xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions">
  <getConference>
    <conferenceKeys confEntity="sip:alice@contoso.com;gruu;opaque=app:conf:focus:id:5D3747C"
/>
<msci:encryption-key>
  <msci:x509-certificate>123213789BC234D</msci:x509-certificate>
```

```
</msci:encryption-key>
  </getConference>
</request>


INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
To: <sip:bob@fabrikam.com>;tag=958d8a3fbc;epid=c5574cd6b6
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C>
Content-Type: application/cccp+xml
Content-Length: 736
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: ad0da39085864c768630674f17692101
CSeq: 53 INFO
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"

<response C3PVersion="1"
          requestId="5"
      from="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C "
          to="sip:bob@fabrikam.com"
          code="success"
          xmlns="urn:ietf:params:xml:ns:cccp"
          xmlns="urn:ietf:params:xml:ns:conference-info"
  xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
  xmlns:msim="http://schemas.fabrikam.com/rtc/2005/08/imconfinfoextensions"
  xmlns:ci="urn:ietf:params:xml:ns:conference-info"
 xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
>
  <getConference>
  <conference-info
    entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:5D3747C"
  state="full" version="4">
  <conference-description>
    <conf-uris>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C</uri>
        <display-text>chat</display-text>
        <purpose>chat</purpose>
      </entry>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:meeting:id:5D3747C</uri>
        <display-text>meeting</display-text>
        <purpose>meeting</purpose>
      </entry>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:5D3747C</uri>
        <display-text>audio-video</display-text>
        <purpose>audio-video</purpose>
      </entry>
      <entry>
        <uri>sip:alice@fabrikam.com;gruu;opaque=app:conf:applicationsharing:id:5D3747C</uri>
        <display-text>applicationsharing</display-text>
        <purpose>applicationsharing</purpose>
      </entry>

<ci:entry>
```

```
<ci:uri>http://schemas.microsoft.com/2008/05/confxmlenc#content</ci:uri>
<ci:purpose>web-internal</ci:purpose>
<cis:separator/>
<msci:encrypted-uri>
<msci:cms-data>MIIBxQYJKoZIhvcNAQcDoIIBtjCCAbICAQAxgd0wgdoCAQAwQzAvMS0wKwYDVQQD
HiQAVQBDAEMAUABfAEMATwBOAEYAXwBNAEcAUgBfAEMARQBSAFQCEJZl58V5VBmQ
TbGUrNBQKJIwDQYJKoZIhvcNAQEBBQAEgYAF3rI5FWWEDcrrxJGS93WC3HC4YlmQ
+D/Ti6cICFX0opLmIwwwoWIlKBAjob8/HAWxmHjns2LrOafkvdDB9s0TJF7Bhl0f
v9L5PYZmj4DhttJqYiN0SxHssSPsdWBi/JYNlno4erqeLEkRiJh5P8mz0XtooO0b
zgrMNVpf+woSlTCBzAYJKoZIhvcNAQcBMBQGCCqGSIb3DQMHBAgF+wlt0hvIRoCB
qLTZ/YMC/7B2pF+V6oBpwo66uJRh6MseJFHbdWT8ptJVx9wJ5jUjFP5lfP+vqIKd
eX8aRvXffwFZJD4Kynzfd/0gQjZMlR90GHhyNimbeehrWuPwli+G14n0s+tejJjS
mf8t9oUSEKHxoljcyaK/fptHFXpp4W0sXa5QRIxcaw6FLIJGdPclVArRw1Bkaxx+
n+f32O/1wBAq2TT2MfRlgW9qzc3sPekOrw==
</msci:cms-data>
</msci:encrypted-uri>
</ci:entry>
<ci:entry>
<ci:uri>http://schemas.microsoft.com/2008/05/confxmlenc#content</ci:uri>
<ci:purpose>web-external</ci:purpose>
<cis:separator/>
<msci:encrypted-uri>
<msci:cms-data>MIIBxQYJKoZIhvcNAQcDoIIBtjCCAbICAQAxgd0wgdoCAQAwQzAvMS0wKwYDVQQD
HiQAVQBDAEMAUABfAEMATwBOAEYAXwBNAEcAUgBfAEMARQBSAFQCEJZl58V5VBmQ
TbGUrNBQKJIwDQYJKoZIhvcNAQEBBQAEgYCaFGVjZW/VHnfp23SzJAuW/Q9OnkfB
+qbwTsH6RBkZ3kX9yzFIrZsU+2R7z+nd3JcvTkJw/mmKxTaKRZksEDwbQGt/kG/V
4Fwx/XRQP9mFI+m6YNy56jgnwTiG97ClNNe+/1tJtiy/h4cFn5lfvoz5eZXPC8MO
Ls75p7hXLPdemTCBzAYJKoZIhvcNAQcBMBQGCCqGSIb3DQMHBAjTd6Z5fKA2K4CB
qBhSEZ5JbWkQ0dLYq8pcK/tbYHfODpwUcDAglGOgrpVWjHmnzm6gwGMURlfQ4X5Y
4wYpamBqfR5XDe6MP+99gpxdAelXTgUKE2/unwAr1pn69BgeiceZlX7kCV748wU6
P7ouOzbKBbLqRGEDO09MXmEwuWSJMacVdo94WkdNKie8cyFt9Q5d0CRltvG87OT2
YOI1xotrcyNwxbJUATHDMjC3+TMaq5UAPg==
</msci:cms-data>
</msci:encrypted-uri>
</ci:entry>
    </conf-uris>
<msci:conference-key>
<msci:cms-data>MIIBGwYJKoZIhvcNAQcDoIIBDDCCAQgCAQAxgd0wgdoCAQAwQzAvMS0wKwYDVQQD
HiQAVQBDAEMAUABfAEMATwBOAEYAXwBNAEcAUgBfAEMARQBSAFQCEHnE9lxFzvyH
Sz3dZuY7xq4wDQYJKoZIhvcNAQEBBQAEgYATbyHXHUnm0kqBO5T7vbhkPTnx8eVJ
YAa0OBid9dh7MwOWhC27pffS83ZwLaZVHsSxAUUR5h0Weq+TU5W+7VlZaTLYaxHF
jh3qElBX9gSf+KPRKTde2tCnP2FcJJL2Ksn20P2tt0mXgBmnbrf20niJxmjY028O
qt4UURxU7M5eFzAjBgkqhkiG9w0BBwEwFAYIKoZIhvcNAwcECP++R9m7uH9QgAA=
</msci:cms-data>
</msci:conference-key>

<cis:separator/>
<msci:pstn-access>
<msci:id>37890</msci:id>
</msci:pstn-access>

</conference-description>
  <users state="full">
    <user entity="sip:alice@fabrikam.com" state="full">
      <display-text>Alice Gates</display-text>
        <roles>
          <entry>presenter</entry>
        </roles>
```

```
        <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}" msci:session-type="focus"
msci:endpoint-uri="sip:alice@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;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:5D3747C">
      <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:5D3747C">
      <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: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>
      </msci:entity-state>
    </msci:entity-view>
    <msci:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:chat:id:5D3747C">
      <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:5D3747C">
      <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:entity-view ci:state="full"
entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:applicationsharing:id:5D3747C ">
<msci:entity-capabilities>
<cis:separator/>
<msas:capabilities>
<msas:control-permission>ActiveDirectoryUsers</msas:control-permission>
</msas:capabilities>
</msci:entity-capabilities>
<msci:entity-state>
<msci:locked>false</msci:locked>
<msci:media>
<entry label="applicationsharing">
<type>applicationsharing</type>
</entry>
</msci:media>
</msci:entity-state>
</msci:entity-view>
  </msci:conference-view>
   </conference-info>
  </getConference>
</response>
```

4.4.8 setLobbyAccess

The **setLobbyAccess** command is sent by the client to the focus to admit users from the lobby into the conference or to eject users out of the lobby.

Following is an example of a **setLobbyAccess** request and a corresponding response.

The client sends a **setLobbyAccess** request to the focus to admit Bob into the conference. In this example, Bob is a presenter who has the privileges to lock the conference.

```
INFO sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU SIP/2.0
FROM: <sip:alice@fabrikam.com>;epid=C740CAAEB6;tag=3d98c7859
TO: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>;tag=F16D0080
CSEQ: 7 INFO
CALL-ID: 1aff74e7-cfc1-4618-93f1-8367bf6ff461
MAX-FORWARDS: 70
VIA: SIP/2.0/TLS 157.56.65.217:3221;branch=z9hG4bK7185d26c
AUTHORIZATION: Kerberos realm="SIP Communications
Service",targetname="sip/ocs.fabrika..com",response="040400ffffffffff0000000000000000eac97e53
b2595c3602614fea",crand="0a0ff851",cnum="11",opaque="F7BE5261",qop="auth"
CONTACT: <sip:alice@fabrikam.com;opaque=user:epid:-3O_9AUDMF-
qNGIrfd9GqwAA;gruu>;text;audio;video;image
CONTENT-LENGTH: 689
SUPPORTED: gruu-10
USER-AGENT: RTCC/4.0.0.0 AgentForCollaborationTests
CONTENT-TYPE: application/cccp+xml
<request xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

```
 requestId="5" C3PVersion="1"
 from="sip:alice@fabrikam.com"
 to="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU"
 xmlns="urn:ietf:params:xml:ns:cccp">
 <setLobbyAccess>
  <conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU"
/>
  <userEntity>sip:bob@fabrikam.com</userEntity>
  <access>granted</access>
 </setLobbyAccess>
</request>
```

The focus validates the request and then responds with a 202 Accepted, indicating that the command is being processed.

```
SIP/2.0 202 Accepted
FROM: <sip:alice@fabrikam.com>;epid=C740CAAEB6;tag=3d98c7859
TO:<sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>;tag=F16D0080
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 1aff74e7-cfc1-4618-93f1-8367bf6ff461
CSeq: 10 INFO
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3",
srand="D6CD41F7", snum="180", opaque="99052D67", qop="auth",
targetname="sip/ocs.fabrikam.com", realm="SIP Communications Service"
Content-Length: 0
```

The focus admits Bob and then sends a response, as shown in the following example:

```
INFO sip:10.54.78.109:4237;transport=tls;ms-opaque=6fb3a8330a;ms-received-cid=10A0D00;grid
SIP/2.0
FROM: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>;tag=F16D0080
TO: <sip:alice@fabrikam.com>;tag=3d98c7859;epid=C740CAAEB6
CSEQ: 1158 INFO
CALL-ID: 1aff74e7-cfc1-4618-93f1-8367bf6ff461
MAX-FORWARDS: 70
VIA: SIP/2.0/TLS 172.24.32.150:5061;branch=z9hG4bK03B71C21.61B8433131093661;branched=FALSE
CONTENT-LENGTH: 1591
SUPPORTED: ms-dialog-route-set-update
CONTENT-TYPE: application/cccp+xml
AUTHENTICATION-INFO: Kerberos qop="auth", opaque="F7BE5261", srand="E52E9F9F", snum="14",
rspauth="040401ffffffffff00000000000000041f3545d7770711ff3227c6e",
targetname="sip/ocs.fabrikam.com", realm="SIP Communications Service", version=4
<response xmlns="urn:ietf:params:xml:ns:cccp"
xmlns:msacp="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
xmlns:msas="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
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:msendpt="http://schemas.microsoft.com/rtc/2008/12/endpointinfoextensions"
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"
xmlns:msls="urn:ietf:params:xml:ns:msls" requestId="5" C3PVersion="1"
```

```
from="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU"
to="sip:alice@fabrikam.com" code="success">
<setLobbyAccess>
<conferenceKeys confEntity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU"/>
<status reason="success">
<userEntity>sip:bob@fabrikam.com</userEntity>
</status>
</setLobbyAccess>
</response>
```

The client responds with a 200 OK to indicate that the INFO response was received.

```
SIP/2.0 200 OK
TO: <sip:alice@fabrikam.com>;epid=C740CAAEB6;tag=3d98c7859
FROM:<sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>;tag=F16D0080
Via: SIP/2.0/TLS 10.1.2.50:4237
Max-Forwards: 70
Call-ID: 1aff74e7-cfc1-4618-93f1-8367bf6ff461
CSeq: 10 INFO
Contact: <sip:ocs.exchange.corp.fabrikam.com:5061;transport=tls>;isfocus
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3",
srand="D6CD41F7", snum="180", opaque="99052D67", qop="auth",
targetname="sip/ocs.fabrikam.com", realm="SIP Communications Service"
Content-Length: 0
```

All watchers receive a notification from the focus, indicating that Bob was admitted to the conference and that Cathy was ejected.

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
To: <sip:alice@fabrikam.com>;epid=C740CAAEB6;tag=3d98c7859
From: <sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>
Content-Length: 2373
Content-Type: application/conference-info+xml
Event: conference
Call-ID: 72d6a36a36784cf58e7e7ab1a51deca2
CSeq: 4 BENOTIFY
Authentication-Info: NTLM rspauth="01000000180D3416377238967F8F5EF3", srand="D6CD41F7",
snum="180", opaque="99052D67", qop="auth", targetname="sip/ocs.fabrikam.com", realm="SIP
Communications Service"
Max-Forwards: 70
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK86DA089F.780A7BCA;branched=FALSE
subscription-state: active;expires=3600

<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
  xmlns:msci="http://schemas.fabrikam.com/rtc/2005/08/confinfoextensions"
  xmlns:msim="http://schemas.fabrikam.com/rtc/2005/08/imconfinfoextensions"
  xmlns:ci="urn:ietf:params:xml:ns:conference-info"
  xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
  entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU"
  state="partial" version="8">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="full">
      <display-text>Bob</display-text>
        <roles>
          <entry>presenter</entry>
        </roles>
```

```
        <endpoint entity="{09AA504C-BA41-4458-8669-8F35470F6CA2}" msci:session-type="focus"
msci:endpoint-uri="sip:bob@fabrikam.com;opaque=user:epid:HT07tI-f3F-fdDyic8rblwAA;gruu">
          <status>connected</status>
        </endpoint>
    </user>
  </users>
</conference-info>
```

Assume that Cathy was also a lobby participant, but was denied access into the conference by Alice. Cathy receives a BYE, as shown in the following example:

```
BENOTIFY sip:10.1.2.50:2383;transport=tls;ms-opaque=02e9ae1f28;ms-received-cid=00031600;grid
SIP/2.0
FROM: < sip:Alice@fabrikam.com;gruu;opaque=app:conf:focus:id:SI1NZFAU>
>;tag=47150080
TO: <sip:cathy@fabrikam.com>;tag=cec49c86e1;epid=732A323248
CSEQ: 2 BENOTIFY
CALL-ID: 5a749d9794cb4639ab290033b9332a57
MAX-FORWARDS: 70
VIA: SIP/2.0/TLS 172.24.32.150:5061;branch=z9hG4bKDDA0E5C6.F0433D83D36FE063;branched=FALSE
CONTENT-LENGTH: 0
EVENT: conference
EXPIRES: 0
SUBSCRIPTION-STATE: terminated;expires=0;reason=ParticipantDenied
AUTHENTICATION-INFO: Kerberos qop="auth", opaque="D211BAF4", srand="0F7C511D", snum="7",
rspauth="040401ffffffffff00000000000000000008b2eea6f1b43bd4abe390a9",
targetname="sip/ocs.fabrikam.com", realm="SIP Communications Service", version=4
ms-diagnostics-public: 3119;reason="Participant Denied
```

4.4.9 modifyEndpoint

The **modifyEndpoint** command is sent by the client to the focus to modify the **endpoint** extensions between the **addUser** command and the **deleteUser** command. The **confEntity**, **user** entity and **endpoint** entity have already communicated with focus in the **addUser** command. The detailed call flow is shown in the following figure.

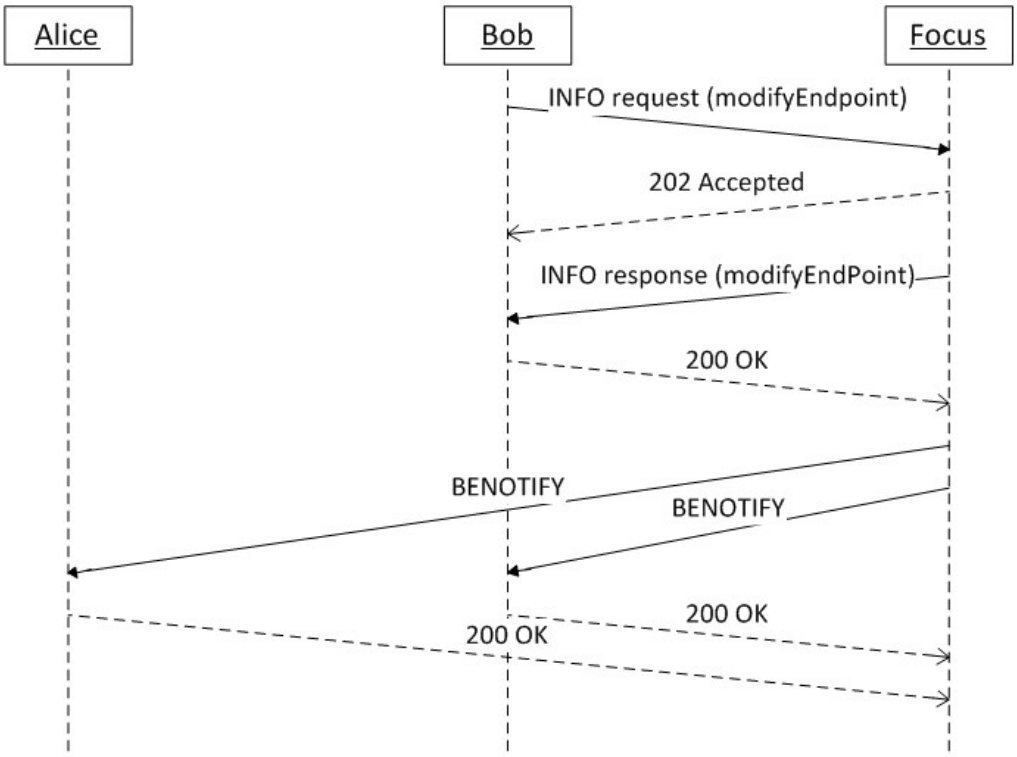


Figure 23: modifyEndpoint call flow

Following is an example of a **modifyEndpoint** request and a corresponding response, to indicate a user's **endpoint** is in recording status.

The client sends a **modifyEndpoint** request to the focus with the **client-recording** element, to notify the focus that Bob's **endpoint** is in recording status.

```
INFO sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH SIP/2.0
Via: SIP/2.0/TLS 10.1.2.173:10614
Max-Forwards: 70
From: <sip:bob@fabrikam.com>;tag=349cf9e3a2;epid=20d43f4ac2
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH>;tag=50230080
Call-ID: 5f73c101f4284d6dadd623cc50e914f3
CSeq: 4 INFO
User-Agent: UCCAPI/4.0.7424.2 OC/4.0.7424.2 (Microsoft Lync 2010)
Supported: ms-dialog-route-set-update
Supported: timer
Proxy-Authorization: TLS-DSK qop="auth", realm="SIP Communications Service",
opaque="1AB56531", targetname="test02.ocs.fabrikam.com", crand="6de69218", cnum="49",
response="8877a5e30c034549ea929572c924fc084dfba274"
Content-Type: application/cccp+xml
Content-Length: 912

<?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:97DJ3DYH" from="sip:bob@fabrikam.com"
requestId="298246656">
  <modifyEndpoint>
    <endpointKeys confEntity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH"
userEntity="sip:bob@fabrikam.com" endpointEntity="{F95CF5F8-2A22-4C1F-B65E-2014CF07BD11}"/>
    <ci:endpoint xmlns:ci="urn:ietf:params:xml:ns:conference-info" entity="{F95CF5F8-2A22-
4C1F-B65E-2014CF07BD11}">
      <cis:separator xmlns:cis="urn:ietf:params:xml:ns:conference-info-
separator"></cis:separator>
      <cis:separator xmlns:cis="urn:ietf:params:xml:ns:conference-info-
separator"></cis:separator>
      <msci:client-recording
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"/>
    </ci:endpoint>
  </modifyEndpoint>
</request>
```

The focus validates the request and then responds with a 202 Accepted, indicating that the command is being processed.

```
SIP/2.0 202 Accepted
Authentication-Info: TLS-DSK qop="auth", opaque="1AB56531", srand="F0F75721", snum="55",
rspauth="f8b2a652b7bf9610f4247ebb38e82c09a65b8f61", targetname="sip/ocs.fabrikam.com",
realm="SIP Communications Service", version=4
From: <sip:bob@fabrikam.com>;tag=349cf9e3a2;epid=20d43f4ac2
To: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH>;tag=50230080
Call-ID: 5f73c101f4284d6dadd623cc50e914f3
CSeq: 4 INFO
Via: SIP/2.0/TLS 10.1.2.173:10614;ms-received-port=10614;ms-received-cid=128FBF00
Content-Length: 0
```

The focus modifies Bob’s endpoint entity and then sends a response, as shown in the following example:

```
INFO sip:10.1.2.173:10614;transport=tls;ms-opaque=4ead93d4a9;ms-received-cid=1773A00;grid
SIP/2.0
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bKA834BA8E.74FC63AE43D39B53;branched=FALSE
Authentication-Info: TLS-DSK qop="auth", opaque="8D1F9B59", srand="470E95DB", snum="57",
rspauth="a27d72a29c83116751c5e2aa5cee890b891a9e3a", targetname="sip/ocs.fabrikam.com",
realm="SIP Communications Service", version=4
Max-Forwards: 70
Content-Length: 233
From: <sip: bob@fabrikam.com;gruu;opaque=app:conf:focus:id:296DSZL0>;tag=D41A0080
To: <sip:bob@fabrikam.com>;tag=f9a01ab5d9;epid=20d43f4ac2
Call-ID: 5a547b0eef2d42f79578d89b426426fc
CSeq: 86 INFO
Supported: ms-dialog-route-set-update
Content-Type: application/cccp+xml
<response xmlns="urn:ietf:params:xml:ns:cccp" requestId="289552528" C3PVersion="1"
from="sip:chuanz@microsoft.com;gruu;opaque=app:conf:focus:id:296DSZL0"
to="sip:chuanz@microsoft.com" code="success">
<modifyEndpoint/>
</response>
```

The client responds with a 200 OK response to this INFO request, which is not shown here.

All watchers receive a notification from the focus, indicating that Bob's **endpoint** is in recording status:

```
BENOTIFY sip:10.1.2.173:10614;transport=tls;ms-opaque=2c65e5a16c;ms-received-
cid=128FBF00;grid SIP/2.0
Via: SIP/2.0/TLS 10.54.67.185:5061;branch=z9hG4bK062C1DB6.017FEC5992C515EB;branched=FALSE
Authentication-Info: TLS-DSK qop="auth", opaque="1AB56531", srand="BEBB7238", snum="60",
rspauth="f45108ca82dcba0f9e5fcfe3e6eef06c51318dba", targetname="sip/ocs.fabrikam.com",
realm="SIP Communications Service", version=4
Max-Forwards: 70
To: <sip:bob@fabrikam.com>;tag=ea00db0f87;epid=20d43f4ac2
Content-Length: 1890
From: <sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH>;tag=366B0080
Call-ID: afcef931b4094348901ade21738d49dc
CSeq: 5 BENOTIFY
Content-Type: application/conference-info+xml
Event: conference
subscription-state: active;expires=3600

<conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:msim="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
xmlns:msci2="http://schemas.microsoft.com/rtc/2008/12/confinfoextensions"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
entity="sip:bob@fabrikam.com;gruu;opaque=app:conf:focus:id:97DJ3DYH" state="partial"
version="8" static="false">
  <users state="partial">
    <user entity="sip:bob@fabrikam.com" state="full">
      <display-text>Bob</display-text>
      <roles>
        <entry>presenter</entry>
      </roles>
      <endpoint entity="{F95CF5F8-2A22-4C1F-B65E-2014CF07BD11}" msci:session-type="focus"
msci:epid="20d43f4ac2" msci:endpoint-
uri="sip:bob@fabrikam.com;opaque=user:epid:3iWcDF_VQlSGstMIXmepcQAA;gruu">
        <!-- snipped -->
        <cis:separator/>
        <cis:separator/>
        <msci:client-recording/>
      </endpoint>
      <!-- snipped -->
    </user>
  </users>
</conference-info>
```

Another use of a **modifyEndpoint** request and a corresponding response is to send the stop recording message. The request, response and the corresponding notification are the same as they are when sending the start recording message except that the number of instances of the **client-recording** element is zero in the request and notification.

As a completion of recording status notification, the **client-recording** element can be specified in the endpoint element in INVITE command, to notice the server and other clients that the new user has the recording status enabled. Clients are responsible to handle the policy related work because the server will not respond failure regardless the recording policy is allowed or not.

4.4.10 modifyConference

The modifyConference command is sent by a client to an MCU to change conference state.

Following is an example modifyConference request and the corresponding response.

```
INFO sip:alice@fabrikam;gruu;opaque=app:conf:focus:id:TDVDW046 SIP/2.0
Via: SIP/2.0/TLS ...
Max-Forwards: 70
From: <sip:alice@fabrikam.com>;tag=7a5dbfcb7d;epid=5c1bcd7656
To: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046>;tag=145E0080
Call-ID: a521cfd600db43d48e2747bd75e7bbae
CSeq: 3 INFO
Route: ...
User-Agent: UCCAPI/4.0.7468.0 OC/4.0.7468.0 (Microsoft Lync 2010)
Supported: ms-dialog-route-set-update
Supported: timer
Proxy-Authorization: ...
Content-Type: application/cccp+xml
Content-Length: 1466

<?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:TDVDW046"
   from="sip:alice@fabrikam.com"
   requestId="355465752">
 <modifyConference mscp:mcuUri="sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-
video:id:TDVDW046">
  <conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
     entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046"
     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:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:TDVDW046">
     <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>

SIP/2.0 202 Accepted
Authentication-Info: ...
From: <sip:alice@fabrikam.com>;tag=7a5dbfcb7d;epid=5c1bcd7656
To: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046>;tag=145E0080
Call-ID: a521cfd600db43d48e2747bd75e7bbae
CSeq: 3 INFO
Via: ...
Content-Length: 0
INFO sip:10.121.251.124:57818;transport=tls;ms-opaque=fa63f33d72;ms-received-
cid=290F0100;grid SIP/2.0
Via: ...
Authentication-Info: ...
Max-Forwards: 70
Content-Length: 551
From: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046>;tag=145E0080
To: <sip:alice@fabrikam.com>;tag=7a5dbfcb7d;epid=5c1bcd7656
Call-ID: a521cfd600db43d48e2747bd75e7bbae
CSeq: 1396 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="355465752"
   C3PVersion="1"
   from="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046"
   to="sip:alice@fabrikam.com"
   responder="sip:alice@fabrikam.com;gruu;opaque=app:conf:audio-video:id:TDVDW046"
   code="success">
 <modifyConference>
  <conference-info xmlns="urn:ietf:params:xml:ns:conference-info"
     entity="sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046"
     state="partial"/>
 </modifyConference>
</response>

SIP/2.0 200 OK
Via: ...
From: <sip:alice@fabrikam.com;gruu;opaque=app:conf:focus:id:TDVDW046>;tag=145E0080
To: <sip:alice@fabrikam.com>;tag=7a5dbfcb7d;epid=5c1bcd7656
Call-ID: a521cfd600db43d48e2747bd75e7bbae
CSeq: 1396 INFO
Contact: <sip:alice@fabrikam.com;opaque=user:epid:v19MMU0DdFWrARgZPvwZuQAA;gruu>
User-Agent: UCCAPI/4.0.7468.0 OC/4.0.7468.0 (Microsoft Lync 2010)
Proxy-Authorization: ...
Content-Length: 0
```

5 Security

The following sections specify security considerations for implementers of the Centralized Conference Control Protocol.

5.1 Security Considerations for Implementers

None.

5.2 Index of Security Parameters

None.

6 Appendix A: application/vnd.microsoft.ocsmeeting Schema Reference

6.1 simplejoinconfdoc Namespace

This namespace is identified by the following URN:

http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema
    targetNamespace="http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc"
    version="1.0"
    attributeFormDefault="unqualified" elementFormDefault="qualified"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:tns="http://schemas.microsoft.com/rtc/2009/05/simplejoinconfdoc">

    <xs:complexType name="conf-info-type">
        <xs:sequence>
            <!-- conference uri -->
            <xs:element name="conf-uri" type="xs:string" />

            <!-- time (in milli-seconds) taken by the server to process the request -->
            <xs:element name="server-time" type="xs:string" />

            <!-- original url coming in to the server -->
            <xs:element name="original-incoming-url" type="xs:string" />

            <!-- conference key -->
            <xs:element name="conf-key" type="xs:string" />
            <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
        <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>
    <xs:element name="conf-info" type="tns:conf-info-type" />
</xs:schema>
```

7 Appendix B: application/cccp+xml Schema Reference

7.1 cccp Namespace

This namespace is identified by the following URN:

urn:ietf:params:xml:ns:cccp

This schema depends on several extension schema-sets, which are defined subsequently.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:cccp" elementFormDefault="qualified"
attributeFormDefault="unqualified" xmlns:tns="urn:ietf:params:xml:ns:cccp"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:mscp="http://schemas.microsoft.com/rtc/2005/08/cccpextensions">

  <!--
       This import brings in the standard Conference Package definitions
  -->
  <xs:import namespace="urn:ietf:params:xml:ns:conference-info" schemaLocation="ms-
ci.xsd"></xs:import>

  <!--
       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>

  <!--
       This import brings in the Conference Package MS extensions
  -->
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
schemaLocation="ms-ci-ext.xsd"></xs:import>

  <!--
       This import brings in the CCCP MS extensions
  -->
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/cccpextensions"
schemaLocation="ms-cccp-ext.xsd"></xs:import>

  <!-- REQUEST ELEMENT -->
  <xs:element name="request" type="tns:request-type"></xs:element>

  <!-- RESPONSE ELEMENT -->
  <xs:element name="response" type="tns:response-type"></xs:element>

  <!-- REQUEST TYPE -->
  <xs:complexType name="request-type">
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element name="addConference" type="tns:add-conference-type"></xs:element>
        <xs:element name="addEndpointMedia" type="tns:add-endpoint-media-type"></xs:element>
        <xs:element name="addUser" type="tns:add-user-type"></xs:element>
        <xs:element name="deleteConference" type="tns:delete-conference-type"></xs:element>
```

```
        <xs:element name="deleteEndpointMedia" type="tns:delete-endpoint-media-
type"></xs:element>
        <xs:element name="deleteUser" type="tns:delete-user-type"></xs:element>
        <xs:element name="getAvailableMcuTypes" type="tns:get-available-mcu-types-
type"></xs:element>
        <xs:element name="getConference" type="tns:get-conference-type"></xs:element>
        <xs:element name="getConferences" type="tns:get-conferences-type"></xs:element>
        <xs:element name="getEncryptionKey" type="tns:get-encryption-key-type"></xs:element>
        <xs:element name="modifyConference" type="tns:add-conference-type"></xs:element>
        <xs:element name="modifyConferenceLock" type="tns:modify-conference-lock-
type"></xs:element>
        <xs:element name="modifyEndpoint" type="tns:modify-endpoint-type"></xs:element>
        <xs:element name="modifyEndpointMedia" type="tns:add-endpoint-media-
type"></xs:element>
        <xs:element name="modifyUser" type="tns:add-user-type"></xs:element>
        <xs:element name="modifyUsersMediaFilters" type="tns:modify-users-media-filters-
type"></xs:element>
        <xs:element name="modifyUserRoles" type="tns:modify-user-roles-type"></xs:element>
        <xs:element name="setLobbyAccess" type="tns:set-lobby-access-type"></xs:element>
        <xs:element name="getConferencingCapabilities" type="tns:get-conferencing-
capabilities-type"></xs:element>       <xs:any namespace="##other" processContents="lax"
minOccurs="0" maxOccurs="unbounded"></xs:any>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="requestId" type="xs:string" use="required"></xs:attribute>

    <!-- The CCCP version of the originator of the request -->
    <xs:attribute name="C3PVersion" type="xs:string" use="optional"></xs:attribute>

    <!-- The  URI of the CCCP client sending the CCCP request -->
    <xs:attribute name="from" type="xs:anyURI" use="required"></xs:attribute>

    <!-- The URI of the CCCP server the request is destined to -->
    <xs:attribute name="to" type="xs:anyURI" use="required"></xs:attribute>

    <!-- The URI of the final CCCP target server the request is destined to -->
    <xs:attribute name="target" type="xs:anyURI" use="optional"></xs:attribute>

    <!-- The trusted Identifier of the originator of the request -->
    <xs:attribute name="originator" type="xs:anyURI" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- RESPONSE TYPE -->
  <xs:complexType name="response-type">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
      <xs:choice>
        <xs:element name="addConference" type="tns:add-conference-response-
type"></xs:element>
        <xs:element name="addEndpointMedia" type="tns:add-endpoint-media-response-
type"></xs:element>
        <xs:element name="addUser" type="tns:add-user-response-type"></xs:element>
        <xs:element name="deleteConference" type="tns:delete-conference-response-
type"></xs:element>
        <xs:element name="deleteEndpointMedia" type="tns:delete-endpoint-media-response-
type"></xs:element>
        <xs:element name="deleteUser" type="tns:delete-user-response-type"></xs:element>
        <xs:element name="getAvailableMcuTypes" type="tns:get-available-mcu-types-response-
type"></xs:element>
```

```
        <xs:element name="getConference" type="tns:get-conference-response-
type"></xs:element>
        <xs:element name="getConferences" type="tns:get-conferences-response-
type"></xs:element>
        <xs:element name="getEncryptionKey" type="tns:get-encryption-key-response-
type"></xs:element>
        <xs:element name="modifyConference" type="tns:add-conference-response-
type"></xs:element>
        <xs:element name="modifyConferenceLock" type="tns:modify-conference-lock-response-
type"></xs:element>
        <xs:element name="modifyEndpoint" type="tns:modify-endpoint-response-
type"></xs:element>
        <xs:element name="modifyEndpointMedia" type="tns:add-endpoint-media-response-
type"></xs:element>
        <xs:element name="modifyUser" type="tns:add-user-response-type"></xs:element>
        <xs:element name="modifyUsersMediaFilters" type="tns:modify-users-media-filters-
response-type"></xs:element>
        <xs:element name="modifyUserRoles" type="tns:modify-user-roles-response-
type"></xs:element>
        <xs:element name="setLobbyAccess" type="tns:set-lobby-access-response-
type"></xs:element>
        <xs:element name="getConferencingCapabilities" type="tns:get-conferencing-
capabilities-response-type"></xs:element>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="requestId" type="xs:string" use="required"></xs:attribute>

    <!--
           The CCCP version of the originator of the recipient of the request
     -->
    <xs:attribute name="C3PVersion" type="xs:string" use="optional"></xs:attribute>

    <!--
            The  URI of the CCCP server sending the CCCP response
      -->
    <xs:attribute name="from" type="xs:anyURI" use="required"></xs:attribute>

    <!--
            The  URI of the CCCP client the response is destined to
      -->
    <xs:attribute name="to" type="xs:anyURI" use="required"></xs:attribute>

    <!--
            The trusted Identifier of the responder
      -->
    <xs:attribute name="responder" type="xs:anyURI" use="optional"></xs:attribute>
    <xs:attribute name="code" type="tns:response-code-type" use="required"></xs:attribute>
    <xs:attribute name="reason" type="tns:reason-code-type-ex" use="optional"></xs:attribute>
    <xs:attribute name="displayString" type="xs:string" use="optional"></xs:attribute>
    <xs:attribute name="timeOut" type="xs:nonNegativeInteger" use="optional"></xs:attribute>
    <xs:attribute name="retryAfter" type="xs:nonNegativeInteger"
use="optional"></xs:attribute>
    <xs:attribute name="version" type="xs:nonNegativeInteger" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- RESPONSE CODE TYPE -->
  <xs:simpleType name="response-code-type">
```

```
    <xs:restriction base="xs:string">
      <xs:enumeration value="success"></xs:enumeration>
      <xs:enumeration value="pending"></xs:enumeration>
      <xs:enumeration value="failure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- REASON CODE TYPE -->
  <xs:simpleType name="reason-code-type-ex">
    <xs:union memberTypes="tns:reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="serverBusy"></xs:enumeration>
      <xs:enumeration value="timeout"></xs:enumeration>
      <xs:enumeration value="unauthorized"></xs:enumeration>
      <xs:enumeration value="requestMalformed"></xs:enumeration>
      <xs:enumeration value="requestTooLarge"></xs:enumeration>
      <xs:enumeration value="requestCancelled"></xs:enumeration>
      <xs:enumeration value="notSupported"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- CONFERENCE KEYS TYPE -->
  <xs:complexType name="conference-keys-type">
    <xs:attribute name="confEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute ref="msci:conference-id"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- USER KEYS TYPE -->
  <xs:complexType name="user-keys-type">
    <xs:attribute name="confEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute name="userEntity" type="xs:anyURI"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ENDPOINT KEYS TYPE -->
  <xs:complexType name="endpoint-keys-type">
    <xs:attribute name="confEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute name="userEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute name="endpointEntity" type="xs:string"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MEDIA KEYS TYPE -->
  <xs:complexType name="media-keys-type">
    <xs:attribute name="confEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute name="userEntity" type="xs:anyURI"></xs:attribute>
    <xs:attribute name="endpointEntity" type="xs:string"></xs:attribute>
    <xs:attribute name="mediaId" type="xs:string"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD CONFERENCE TYPE -->
  <xs:complexType name="add-conference-type">
    <xs:sequence>
      <xs:element ref="ci:conference-info"></xs:element>
```

```
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD CONFERENCE RESPONSE TYPE -->
  <xs:complexType name="add-conference-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element ref="ci:conference-info" minOccurs="0"></xs:element>
      <xs:element name="notification" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:attribute name="reason" type="tns:add-conference-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD CONFERENCE REASON CODE TYPE -->
  <xs:simpleType name="add-conference-reason-code-type-ex">
    <xs:union memberTypes="tns:add-conference-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="add-conference-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="anonymousUsersNotAllowed"></xs:enumeration>
      <xs:enumeration value="conferenceExistsAlready"></xs:enumeration>
      <xs:enumeration value="entitySettingsTooLarge"></xs:enumeration>
      <xs:enumeration value="federatedUsersNotAllowed"></xs:enumeration>
      <xs:enumeration value="forbidden"></xs:enumeration>
      <xs:enumeration value="invalidAdmissionPolicy"></xs:enumeration>
      <xs:enumeration value="invalidEncryptionKeyUsed"></xs:enumeration>
      <xs:enumeration value="invalidConferenceId"></xs:enumeration>
      <xs:enumeration value="invalidExpiryTime"></xs:enumeration>
      <xs:enumeration value="invalidPasscode"></xs:enumeration>
      <xs:enumeration value="invalidRole"></xs:enumeration>
      <xs:enumeration value="invalidUserEntity"></xs:enumeration>
      <xs:enumeration value="maxMeetingSizeExceeded"></xs:enumeration>
      <xs:enumeration value="mcuTypeNotAvailable"></xs:enumeration>
      <xs:enumeration value="maxConferencesExceeded"></xs:enumeration>
      <xs:enumeration value="notificationDataTooLarge"></xs:enumeration>
      <xs:enumeration value="organizerRoamingDataTooLarge"></xs:enumeration>

      <!-- conferenceDoesntExist only applies to modifyConference responses -->
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>

      <!-- invalidVersion only applies to modifyConference responses -->
      <xs:enumeration value="invalidVersion"></xs:enumeration>

      <!-- activeMediaDeletionForbidden only applies to modifyConference responses -->
      <xs:enumeration value="activeMediaDeletionForbidden"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
```

```
  <!-- GET AVAILABLE MCU TYPES TYPE -->
  <xs:complexType name="get-available-mcu-types-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
  <xs:complexType name="mcu-types-type">
    <xs:sequence>
      <xs:element name="mcuType" type="xs:string" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET AVAILABLE MCU TYPES RESPONSE TYPE -->
  <xs:complexType name="get-available-mcu-types-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="mcu-types" type="tns:mcu-types-type"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:get-available-mcu-types-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET AVAILABLE MCU TYPES REASON CODE TYPE -->
  <xs:simpleType name="get-available-mcu-types-reason-code-type-ex">
    <xs:union memberTypes="tns:get-available-mcu-types-reason-code-type
xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="get-available-mcu-types-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- GET CONFERENCE TYPE -->
  <xs:complexType name="get-conference-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
      <xs:element name="notify" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element ref="msci:encryption-key" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
```

```
  <!-- GET CONFERENCE RESPONSE TYPE -->
  <xs:complexType name="get-conference-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element ref="ci:conference-info" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:get-conference-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET CONFERENCE REASON CODE TYPE -->
  <xs:simpleType name="get-conference-reason-code-type-ex">
    <xs:union memberTypes="tns:get-conference-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="get-conference-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="invalidEncryptionKey"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- GET CONFERENCES TYPE -->
  <xs:complexType name="get-conferences-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
  <xs:complexType name="conferences-type">
    <xs:sequence>
      <xs:element ref="ci:conference-info" minOccurs="0" maxOccurs="unbounded"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET CONFERENCES RESPONSE TYPE -->
  <xs:complexType name="get-conferences-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="conferences" type="tns:conferences-type"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:get-conferences-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
```

```
  </xs:complexType>

  <!-- GET CONFERENCES REASON CODE TYPE -->
  <xs:simpleType name="get-conferences-reason-code-type-ex">
    <xs:union memberTypes="tns:get-conferences-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="get-conferences-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- GET ENCRYPTION KEY TYPE -->
  <xs:complexType name="get-encryption-key-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET ENCRYPTION KEY RESPONSE TYPE -->
  <xs:complexType name="get-encryption-key-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element ref="msci:encryption-key"></xs:element>
      <xs:element ref="msci:opaque" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:get-encryption-key-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET ENCRYPTION KEY REASON CODE TYPE -->
  <xs:simpleType name="get-encryption-key-reason-code-type-ex">
    <xs:union memberTypes="tns:get-encryption-key-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="get-encryption-key-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- DELETE CONFERENCE TYPE -->
  <xs:complexType name="delete-conference-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE CONFERENCE RESPONSE TYPE -->
  <xs:complexType name="delete-conference-response-type">
```

```
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element ref="ci:conference-info" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:delete-conference-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE CONFERENCE REASON CODE TYPE -->
  <xs:simpleType name="delete-conference-reason-code-type-ex">
    <xs:union memberTypes="tns:delete-conference-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="delete-conference-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- MODIFY CONFERENCE REASON CODE TYPE -->
  <xs:simpleType name="modify-conference-reason-code-type-ex">
    <xs:union memberTypes="tns:modify-conference-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="modify-conference-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="mediaNotSupported"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- MODIFY CONFERENCE LOCK TYPE  -->
  <xs:complexType name="modify-conference-lock-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
      <xs:element name="locked" type="xs:boolean"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY CONFERENCE LOCK RESPONSE TYPE  -->
  <xs:complexType name="modify-conference-lock-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element ref="ci:conference-info" minOccurs="0"></xs:element>
      <xs:element name="locked" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
```

```
    <xs:attribute name="reason" type="tns:modify-conference-lock-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY CONFERENCE LOCK REASON CODE TYPE  -->
  <xs:simpleType name="modify-conference-lock-reason-code-type-ex">
    <xs:union memberTypes="tns:modify-conference-lock-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="modify-conference-lock-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- ADD USER TYPE -->
  <xs:complexType name="add-user-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
      <xs:element ref="ci:user"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="mscp:allow-session-replace" use="optional"
default="false"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD USER RESPONSE TYPE  -->
  <xs:complexType name="add-user-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"
minOccurs="0"></xs:element>
      <xs:element ref="ci:user" minOccurs="0"></xs:element>
      <xs:element ref="mscp:info" minOccurs="0"></xs:element>
      <xs:element ref="mscp:connection-info" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:add-user-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD USER REASON CODE TYPE -->
  <xs:simpleType name="add-user-reason-code-type-ex">
    <xs:union memberTypes="tns:add-user-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="add-user-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="userExistsAlready"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
```

```
      <xs:enumeration value="userBusy"></xs:enumeration>
      <xs:enumeration value="userNotAnswered"></xs:enumeration>
      <xs:enumeration value="userDeclined"></xs:enumeration>
      <xs:enumeration value="userUnknown"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- MODIFY USER ROLES TYPE -->
  <xs:complexType name="modify-user-roles-type">
    <xs:sequence>
      <xs:element name="userKeys" type="tns:user-keys-type"></xs:element>
      <xs:element ref="ci:user-roles"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY USER ROLES RESPONSE TYPE -->
  <xs:complexType name="modify-user-roles-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"
minOccurs="0"></xs:element>
      <xs:element ref="ci:user" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:modify-user-roles-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY USER ROLES REASON CODE TYPE -->
  <xs:simpleType name="modify-user-roles-reason-code-type-ex">
    <xs:union memberTypes="tns:modify-user-roles-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="modify-user-roles-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- DELETE USER TYPE -->
  <xs:complexType name="delete-user-type">
    <xs:sequence>
      <xs:element name="userKeys" type="tns:user-keys-type"></xs:element>
      <xs:element name="endpointEntity" type="xs:string" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
```

```
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:attribute name="client-reason" type="tns:delete-user-client-reason-code-type-ext"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
  <!-- DELETE USER CLIENT REASON TYPE -->
  <xs:simpleType name="delete-user-client-reason-code-type-ext">
    <xs:union memberTypes="tns:delete-user-client-reason-code-type xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="delete-user-client-reason-code-type"
ms:className="CC3PDeleteUserClientReasonCodeType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="newPresenter" />
      <xs:enumeration value="participantEjected" />
      <xs:enumeration value="connectedAtAnotherEndpoint" />
    </xs:restriction>
  </xs:simpleType>
  <!-- DELETE USER RESPONSE TYPE -->
  <xs:complexType name="delete-user-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"
minOccurs="0"></xs:element>
      <xs:element ref="ci:user" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:delete-user-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE USER REASON CODE TYPE -->
  <xs:simpleType name="delete-user-reason-code-type-ex">
    <xs:union memberTypes="tns:delete-user-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="delete-user-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <!-- SET LOBBY ACCESS TYPE -->
  <xs:complexType name="set-lobby-access-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
      <xs:element name="userEntity" type="xs:anyURI" maxOccurs="unbounded"></xs:element>
      <xs:element name="access" type="tns:user-access-type"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded" ></xs:any>
      </xs:sequence>
    </xs:sequence>
```

```
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- SET LOBBY ACCESS RESPONSE TYPE -->
  <xs:complexType name="set-lobby-access-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"
minOccurs="0"></xs:element>
      <xs:element name="status" type="tns:set-lobby-access-status-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:set-lobby-access-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- SET LOBBY ACCESS STATUS TYPE -->
  <xs:complexType name="set-lobby-access-status-type">
    <xs:sequence>
       <xs:element name="userEntity" type="xs:anyURI"></xs:element>
       <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:set-lobby-access-status-code-type-
ex"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"><xs:anyAttribute>
  </xs:complexType>

  <!-- SET LOBBY ACCESS STATUS CODE TYPE -->
  <xs:simpleType name="set-lobby-access-status-code-type-ex">
    <xs:union memberTypes="tns:set-lobby-access-status-code-type xs:string"></xs:union>
  </xs:simpleType>

  <xs:simpleType name="set-lobby-access-status-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceFull"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
      <xs:enumeration value="alreadyGranted"></xs:enumeration>
      <xs:enumeration value="success"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- SET LOBBY ACCESS REASON CODE TYPE -->
  <xs:simpleType name="set-lobby-access-reason-code-type-ex">
    <xs:union memberTypes="tns:set-lobby-access-reason-code-type xs:string"></xs:union>
  </xs:simpleType>

  <xs:simpleType name="set-lobby-access-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
```

```
      <xs:enumeration value="notAuthorized"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- USER ACCESS TYPE -->
  <xs:simpleType name="user-access-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="granted"></xs:enumeration>
      <xs:enumeration value="denied"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <!-- MODIFY ENDPOINT TYPE -->
  <xs:complexType name="modify-endpoint-type">
    <xs:sequence>
      <xs:element name="endpointKeys" type="tns:endpoint-keys-type"/>
      <xs:element ref="ci:endpoint"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"
/>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <!-- MODIFY ENDPOINT RESPONSE TYPE -->
  <xs:complexType name="modify-endpoint-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"/>
      <xs:element name="endpointKeys" type="tns:endpoint-keys-type" minOccurs="0"/>
      <xs:element ref="ci:endpoint" 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="reason" type="tns:modify-endpoint-reason-code-type-ex"
use="optional"/>
    <xs:attribute ref="mscp:mcuUri" use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax" />
  </xs:complexType>

  <xs:simpleType name="modify-endpoint-reason-code-type-ex">
    <xs:union memberTypes="tns:modify-endpoint-reason-code-type xs:string"/>
  </xs:simpleType>

  <xs:simpleType name="modify-endpoint-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist" />
      <xs:enumeration value="userDoesntExist" />
      <xs:enumeration value="endpointDoesntExist" />
      <xs:enumeration value="recordingNotAllowed"/>
      <xs:enumeration value="otherFailure" />
    </xs:restriction>
  </xs:simpleType>

  <!-- MODIFY USERS MEDIA FILTERS TYPE -->
  <xs:complexType name="modify-users-media-filters-type">
    <xs:sequence>
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>
```

```
      <xs:element name="mediaLabel" type="xs:string" maxOccurs="unbounded"></xs:element>
      <xs:element ref="msci:media-ingress-filter" minOccurs="0"></xs:element>
      <xs:element ref="msci:media-egress-filter" minOccurs="0"></xs:element>

      <!-- zero or more "excludeRole" elements specifies the roles to which this filter is
not applied (that is leave them untouched) -->
      <xs:element name="excludeRole" type="xs:string" minOccurs="0"
maxOccurs="unbounded"></xs:element>

      <!-- zero or more "excludeUser" elements specify users to which this filter is not
applied(that is leave them untouched) -->
      <xs:element name="excludeUser" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY USERS MEDIA FILTERS RESPONSE TYPE -->
  <xs:complexType name="modify-users-media-filters-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>

      <!-- Correlation token -->
      <xs:element name="conferenceKeys" type="tns:conference-keys-type"></xs:element>

      <!-- Failures -->
      <xs:element name="unsupportedRole" type="xs:string" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:element name="unknownUser" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:element name="unsupportedMediaLabel" type="xs:string" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:modify-users-media-filters-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- MODIFY USERS MEDIA FILTERS REASON CODE TYPE -->
  <xs:simpleType name="modify-users-media-filters-reason-code-type-ex">
    <xs:union memberTypes="tns:modify-users-media-filters-reason-code-type
xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="modify-users-media-filters-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
```

```
  <!-- ADD USER MEDIA TYPE -->
  <xs:complexType name="add-endpoint-media-type">
    <xs:sequence>
      <xs:element name="mediaKeys" type="tns:media-keys-type"></xs:element>
      <xs:element ref="ci:media"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- ADD USER MEDIA REASON CODE TYPE -->
  <xs:simpleType name="add-endpoint-media-reason-code-type-ex">
    <xs:union memberTypes="tns:add-endpoint-media-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="add-endpoint-media-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
      <xs:enumeration value="endpointDoesntExist"></xs:enumeration>
      <xs:enumeration value="mediaExistsAlready"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!-- ADD USER MEDIA RESPONSE TYPE -->
  <xs:complexType name="add-endpoint-media-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="endpointKeys" type="tns:endpoint-keys-type"
minOccurs="0"></xs:element>
      <xs:element ref="ci:media" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:add-endpoint-media-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE ENDPOINT MEDIA TYPE -->
  <xs:complexType name="delete-endpoint-media-type">
    <xs:sequence>
      <xs:element name="mediaKeys" type="tns:media-keys-type"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE ENDPOINT MEDIA RESPONSE TYPE -->
  <xs:complexType name="delete-endpoint-media-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
```

```
      <xs:element name="endpointKeys" type="tns:endpoint-keys-type"
minOccurs="0"></xs:element>
      <xs:element ref="ci:media" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:delete-endpoint-media-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute ref="mscp:mcuUri" use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- DELETE MEDIA REASON CODE TYPE -->
  <xs:simpleType name="delete-endpoint-media-reason-code-type-ex">
    <xs:union memberTypes="tns:delete-endpoint-media-reason-code-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="delete-endpoint-media-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="conferenceDoesntExist"></xs:enumeration>
      <xs:enumeration value="userDoesntExist"></xs:enumeration>
      <xs:enumeration value="endpointDoesntExist"></xs:enumeration>
      <xs:enumeration value="mediaDoesntExist"></xs:enumeration>
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

<!-- GET CONFERENCING CAPABILITIES TYPE -->
  <xs:complexType name="get-conferencing-capabilities-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!-- GET CONFERENCING CAPABILITIES RESPONSE TYPE -->
  <xs:complexType name="get-conferencing-capabilities-response-type">
    <xs:sequence>
      <xs:element ref="mscp:diagnostics-info" minOccurs="0"></xs:element>
      <xs:element name="mcu-types" type="tns:mcu-types-type"></xs:element>
      <xs:element name="anonymous-scheduling" type="xs:boolean" minOccurs="0" ></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="reason" type="tns:get-conferencing-capabilities-reason-code-type-ex"
use="optional"></xs:attribute>
    <xs:attribute name="capability-version" type="xs:nonNegativeInteger" default="0"
use="optional"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!-- GET CONFERENCING CAPABILITIES REASON CODE TYPE -->
  <xs:simpleType name="get-conferencing-capabilities-reason-code-type-ex">
    <xs:union memberTypes="tns:get-conferencing-capabilities-reason-code-type
xs:string"></xs:union>
```

```
  </xs:simpleType>

  <xs:simpleType name="get-conferencing-capabilities-reason-code-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="otherFailure"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>


</xs:schema>
```

7.2 cccpextensions Namespace

This namespace is identified by the following URN:

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

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/rtc/2005/08/cccpextensions"
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/cccpextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ci="urn:ietf:params:xml:ns:conference-
info">

  <!--
       This import brings in the Conference Package definitions + MS extensions
    -->
  <xs:import namespace="urn:ietf:params:xml:ns:conference-info" schemaLocation="ms-
ci.xsd"></xs:import>

  <!--
      General INFO TYPE

      is used as a generic container to include opaque DATA MCU PSOM settings and
permissions.

      NOTE: This element is now obsolete, use connection-info instead.

    -->
  <xs:element name="info" type="tns:info-type"></xs:element>
  <xs:complexType name="info-type">
    <xs:sequence>
      <xs:element name="contact" type="xs:anyURI" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      DIAGNOSTICS INFO TYPE
    -->
  <xs:element name="diagnostics-info" type="tns:diagnostics-info-type"></xs:element>
  <xs:complexType name="diagnostics-info-type">
    <xs:sequence>
```

```
      <xs:element name="entry" type="tns:key-value-pair-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>

  <!--
     General CONNECTION INFO TYPE
    -->
  <xs:element name="connection-info" type="tns:connection-info-type"></xs:element>
  <xs:complexType name="connection-info-type">
    <xs:sequence>
      <xs:element name="entry" type="tns:key-value-pair-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>

  <!--
          KEY VALUE PAIR TYPE
    -->
  <xs:complexType name="key-value-pair-type">
    <xs:sequence>
      <xs:element name="key" type="xs:string"></xs:element>
      <xs:element name="value" type="xs:string"></xs:element>
    </xs:sequence>
  </xs:complexType>

  <!--
    MCU ID ATTRIBUTE
    -->
  <xs:attribute name="mcuUri" type="tns:mcu-id-type"></xs:attribute>
  <xs:simpleType name="mcu-id-type">
    <xs:restriction base="xs:anyURI"></xs:restriction>
  </xs:simpleType>

  <!--
    DRAINING STATUS ELEMENT
    -->
  <xs:element name="drainStatus" type="tns:draining-status-type"></xs:element>

  <!--
     DRAINING STATUS TYPE
    -->
  <xs:simpleType name="draining-status-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="request"></xs:enumeration>
      <xs:enumeration value="acknowledge"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!--
       allow-session-replaces ATTRIBUTE
    -->
  <xs:attribute name="allow-session-replace" type="xs:boolean"></xs:attribute>
</xs:schema>
```

8 Appendix C: application/conference-info+xml Schema Reference

8.1 conference-info Namespace

This namespace is identified by the following URN:

urn:ietf:params:xml:ns:conference-info

The schema for this section is based on [RFC4575], with extensions specified in namespaces, which are defined subsequently.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:conference-info"
elementFormDefault="qualified" attributeFormDefault="unqualified"
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">

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

  <!--
    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"></xs:import>

  <!--
    ELEMENTs and Attributes for CCCP definitions
    -->
  <xs:attribute name="state" type="state-type"></xs:attribute>
  <xs:element name="media" type="media-type"></xs:element>
  <xs:element name="endpoint" type="endpoint-type"></xs:element>
  <xs:element name="user-roles" type="user-roles-type"></xs:element>
  <xs:element name="user" type="user-type"></xs:element>

  <!--
    CONFERENCE ELEMENT
    -->
  <xs:element name="conference-info" type="conference-type"></xs:element>

  <!--
      CONFERENCE TYPE
    -->
  <xs:complexType name="conference-type">
    <xs:sequence>
      <xs:element name="conference-description" type="conference-description-type"
minOccurs="0"></xs:element>
      <xs:element name="host-info" type="host-type" minOccurs="0"></xs:element>
      <xs:element name="conference-state" type="conference-state-type"
minOccurs="0"></xs:element>
      <xs:element name="users" type="users-type" minOccurs="0"></xs:element>
      <xs:element name="sidebars-by-ref" type="uris-type" minOccurs="0"></xs:element>
```

```
      <xs:element name="sidebars-by-val" type="sidebars-by-val-type"
minOccurs="0"></xs:element>
      <xs:element ref="msci:conference-view" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute ref="msci:conference-id"></xs:attribute>
    <xs:attribute name="entity" type="xs:anyURI" use="required"></xs:attribute>
    <xs:attribute name="state" type="state-type" use="optional"
default="full"></xs:attribute>
    <xs:attribute name="version" type="xs:unsignedInt" use="optional"></xs:attribute>
<xs:attribute name="static" type="xs:boolean" use="optional" />
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      STATE TYPE
    -->
  <xs:simpleType name="state-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="full"></xs:enumeration>
      <xs:enumeration value="partial"></xs:enumeration>
      <xs:enumeration value="deleted"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!--
      CONFERENCE DESCRIPTION TYPE
    -->
  <xs:complexType name="conference-description-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="subject" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="free-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="keywords" type="keywords-type" minOccurs="0"></xs:element>
      <xs:element name="conf-uris" type="uris-type" minOccurs="0"></xs:element>
      <xs:element name="service-uris" type="uris-type" minOccurs="0"></xs:element>
      <xs:element name="maximum-user-count" type="xs:unsignedInt" minOccurs="0"></xs:element>
      <xs:element name="available-media" type="conference-media-type"
minOccurs="0"></xs:element>
      <xs:element ref="msci:disclaimer" minOccurs="0"></xs:element>
      <xs:element ref="msci:organizer" minOccurs="0"></xs:element>
      <xs:element ref="msci:conference-id" minOccurs="0"></xs:element>
      <xs:element ref="msci:conference-key" minOccurs="0"></xs:element>
      <xs:element ref="msci:last-update" minOccurs="0"></xs:element>
      <xs:element ref="msci:last-activate" minOccurs="0"></xs:element>
      <xs:element ref="msci:is-active" minOccurs="0"></xs:element>
      <xs:element ref="msci:expiry-time" minOccurs="0"></xs:element>
      <xs:element ref="msci:admission-policy" minOccurs="0"></xs:element>
      <xs:element ref="msci:organizer-roaming-data" minOccurs="0"></xs:element>
      <xs:element ref="msci:notification-data" minOccurs="0"></xs:element>

      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:element ref="msci:pstn-access" minOccurs="0"></xs:element>
        <xs:sequence minOccurs="0">
          <xs:element ref="cis:separator"></xs:element>
```

```
            <xs:element ref="msci:lobby-capable" minOccurs="0"></xs:element>
            <xs:element ref="msci:anonymous-type-allowed"></xs:element>
            <xs:element ref="msci:join-url" minOccurs="0"></xs:element>
            <xs:element ref="msci:autopromote" minOccurs="0"></xs:element>
            <xs:element ref="msci:autopromote-allowed" minOccurs="0"></xs:element>
            <xs:element ref="msci:pstn-lobby-bypass" minOccurs="0"></xs:element>
            <xs:element ref="msci:pstn-lobby-bypass-allowed"
```

minOccurs="0"></xs:element>

```
            <xs:element ref="msci:disclaimer-title"
  minOccurs="0"></xs:element>
            <xs:element ref="msci:recording-allowed"
```

minOccurs="0"></xs:element>

```
            <xs:element ref="msci:externaluser-recording-allowed"
```

minOccurs="0"></xs:element>

```
            <xs:element ref="msci:server-mode"
  minOccurs="0"></xs:element>
            <xs:element ref="msci:recording-notification"
  minOccurs="0"></xs:element>
            <xs:sequence minOccurs="0">
              <xs:element ref="cis:separator"></xs:element>
              <xs:any namespace="##other" processContents="lax"
               maxOccurs="unbounded"></xs:any>
            </xs:sequence>
          </xs:sequence>
      </xs:sequence>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

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

  <!--
      CONFERENCE STATE TYPE
    -->
  <xs:complexType name="conference-state-type">
    <xs:sequence>
      <xs:element name="user-count" type="xs:unsignedInt" minOccurs="0"></xs:element>
```

```
      <xs:element name="active" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="locked" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

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

  <!--
      URIs TYPE
    -->
  <xs:complexType name="uris-type">
    <xs:sequence>
      <xs:element name="entry" type="uri-type" maxOccurs="unbounded"></xs:element>
    </xs:sequence>
    <xs:attribute name="state" type="state-type" use="optional"
default="full"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      URI TYPE
    -->
  <xs:complexType name="uri-type">
    <xs:sequence>
      <xs:element name="uri" type="xs:anyURI"></xs:element>
      <xs:element name="display-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="purpose" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="modified" type="execution-type" minOccurs="0"></xs:element>
      <xs:element ref="msci:hash-code" minOccurs="0"></xs:element>
```

```
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
       <xs:element ref="msci:encrypted-uri" 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:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      KEWORDS TYPE
    -->
  <xs:simpleType name="keywords-type">
    <xs:list itemType="xs:string"></xs:list>
  </xs:simpleType>

  <!--
      USERS TYPE
    -->
  <xs:complexType name="users-type">
    <xs:sequence>
      <xs:element name="user" type="user-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute name="state" type="state-type" use="optional"
default="full"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      USER TYPE
    -->
  <xs:complexType name="user-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="associated-aors" type="uris-type" minOccurs="0"></xs:element>
      <xs:element name="roles" type="user-roles-type" minOccurs="0"></xs:element>
      <xs:element name="languages" type="user-languages-type" minOccurs="0"></xs:element>
      <xs:element name="cascaded-focus" type="xs:anyURI" minOccurs="0"></xs:element>
      <xs:element name="endpoint" type="endpoint-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:element ref="msci:designated-presenter" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="entity" type="xs:anyURI"></xs:attribute>
    <xs:attribute ref="msci:smtp-address"></xs:attribute>
    <xs:attribute name="state" type="state-type" use="optional"
default="full"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
```

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

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

  <!--
      ENDPOINT TYPE
    -->
  <xs:complexType name="endpoint-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="referred" type="execution-type" minOccurs="0"></xs:element>
      <xs:element name="status" type="endpoint-status-type" minOccurs="0"></xs:element>
      <xs:element name="joining-method" type="joining-type" minOccurs="0"></xs:element>
      <xs:element name="joining-info" type="execution-type" minOccurs="0"></xs:element>
      <xs:element name="disconnection-method" type="disconnection-type"
minOccurs="0"></xs:element>
      <xs:element name="disconnection-info" type="execution-type" minOccurs="0"></xs:element>
      <xs:element name="media" type="media-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:element name="call-info" type="call-type" minOccurs="0"></xs:element>
      <xs:element ref="msci:roles" minOccurs="0"></xs:element>
      <xs:element ref="msci:authMethod" minOccurs="0"></xs:element>
      <xs:element ref="msci:accessMethod" minOccurs="0"></xs:element>
      <xs:element ref="msci:clientInfo" minOccurs="0"></xs:element>
      <xs:element ref="msci:post-dial" minOccurs="0"></xs:element>
      <xs:element ref="msci:pstnRole" minOccurs="0"></xs:element>
      <xs:element ref="msci:pstnLeaderPasscode" minOccurs="0"></xs:element>
      <xs:element ref="msci:endpoint-capabilities" minOccurs="0"></xs:element>
      <xs:element ref="msci:is-robot" minOccurs="0"></xs:element>
      <xs:element ref="msci:current-sidebar" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:element ref="msci:session-on-behalf-of" minOccurs="0"></xs:element>
    <xs:element ref="msci:in-conferencing-services" minOccurs="0"></xs:element>
    <xs:element ref="msci:languages" minOccurs="0"></xs:element>
    <xs:element ref="msci:is-pstn-endpoint" minOccurs="0"></xs:element>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"></xs:element>
      <xs:element ref="msci:client-recording" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
```

</xs:sequence>

```
  </xs:sequence>
      </xs:sequence>
      <xs:attribute name="entity" type="xs:string"></xs:attribute>
      <xs:attribute name="state" type="state-type" use="optional"
  default="full"></xs:attribute>
      <xs:attribute ref="msci:session-type" use="optional"></xs:attribute>
      <xs:attribute ref="msci:epid" use="optional"></xs:attribute>
      <xs:attribute ref="msci:sip-instance" use="optional"></xs:attribute>
      <xs:attribute ref="msci:endpoint-uri" use="optional"></xs:attribute>
      <xs:attribute ref="msci:refer-to-uri" use="optional"></xs:attribute>
      <xs:attribute ref="msci:asserted-identity" use="optional"></xs:attribute>
      <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
    </xs:complexType>

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

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

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

    <!--
        EXECUTION TYPE
```

```
    -->
  <xs:complexType name="execution-type">
    <xs:sequence>
      <xs:element name="when" type="xs:dateTime" minOccurs="0"></xs:element>
      <xs:element name="reason" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="by" type="xs:anyURI" minOccurs="0"></xs:element>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      CALL TYPE
    -->
  <xs:complexType name="call-type">
    <xs:choice>
      <xs:element name="sip" type="sip-dialog-id-type"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:choice>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </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>
      <xs:element name="call-id" type="xs:string"></xs:element>
      <xs:element name="from-tag" type="xs:string"></xs:element>
      <xs:element name="to-tag" type="xs:string"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      MEDIA TYPE
    -->
  <xs:complexType name="media-type">
    <xs:sequence>
      <xs:element name="display-text" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="type" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="label" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="src-id" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="status" type="media-status-type" minOccurs="0"></xs:element>
      <xs:element ref="msci:media-ingress-filter" minOccurs="0">
        <xs:annotation>
          <xs:documentation>
                      If this element is not present, a value of 'unblock'
                      is assumed

          </xs:documentation>
        </xs:annotation>
      </xs:element>
      <xs:element ref="msci:media-egress-filter" minOccurs="0"></xs:element>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
```

```
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:attribute name="id" type="xs:string" use="required"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      MEDIA STATUS TYPE
    -->
  <xs:simpleType name="media-status-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="recvonly"></xs:enumeration>
      <xs:enumeration value="sendonly"></xs:enumeration>
      <xs:enumeration value="sendrecv"></xs:enumeration>
      <xs:enumeration value="inactive"></xs:enumeration>
    </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:element>
    </xs:sequence>
    <xs:attribute name="state" type="state-type" use="optional"
default="full"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
</xs:schema>
```

8.2 confinfoextensions Namespace

This namespace is identified by the following URN:

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

```
<?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:msdata="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
xmlns:msim="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions">

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

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

  <!--
       This import brings the MCU settings definitions.
    -->
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
schemaLocation="acpmcusettings.xsd"></xs:import>
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
schemaLocation="avmcusettings.xsd"></xs:import>
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
schemaLocation="datamcusettings.xsd"></xs:import>
  <xs:import namespace="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
schemaLocation="immcusettings.xsd"></xs:import>
  <xs:element name="disclaimer" type="xs:string"></xs:element>
  <xs:element name="designated-presenter" type="xs:boolean"></xs:element>
  <xs:attribute name="conference-id" type="xs:string"></xs:attribute>
  <xs:element name="conference-id" type="xs:string"></xs:element>
  <xs:element name="conference-key" type="tns:conference-key-type"></xs:element>
  <xs:element name="current-sidebar" type="xs:anyURI"></xs:element>
  <xs:element name="last-update" type="xs:dateTime"></xs:element>
  <xs:element name="last-activate" type="xs:dateTime"></xs:element>
  <xs:element name="is-active" type="xs:boolean"></xs:element>
  <xs:element name="expiry-time" type="xs:dateTime"></xs:element>
  <xs:element name="organizer-roaming-data" type="tns:organizer-roaming-data-
type"></xs:element>
  <xs:element name="notification-data" type="tns:notification-data-type"></xs:element>
  <xs:element name="encryption-key" type="tns:encryption-key-type"></xs:element>
  <xs:element name="opaque" type="tns:encryption-key-opaque-type"></xs:element>
  <xs:attribute name="mcu-type" type="xs:string"></xs:attribute>
  <xs:element name="roles" type="ci:user-roles-type"></xs:element>        <xs:attribute
name="smtp-address" type="xs:anyURI"></xs:attribute>
  <xs:element name="encrypted-uri" type="tns:encrypted-content-type"></xs:element>
  <xs:element name="languages" type="ci:user-languages-type"></xs:element>
  <xs:element name="is-pstn-endpoint" type="xs:boolean"></xs:element>
  <xs:element name="anonymous-type-allowed" type="xs:boolean"></xs:element>
  <xs:element name="lobby-capable" type="xs:boolean"></xs:element>
  <xs:element name="join-url" type="xs:anyURI"></xs:element>
  <xs:element name="autopromote-allowed" type="tns:autopromote-type" ></xs:element>
  <xs:element name="autopromote" type="tns:autopromote-type"></xs:element>
  <xs:simpleType name="autopromote-type">
    <xs:restriction base="xs:unsignedInt"></xs:restriction>
  </xs:simpleType>
```

```
<xs:element name="pstn-lobby-bypass-allowed" type="xs:boolean"></xs:element>

<xs:element name="pstn-lobby-bypass" type="xs:boolean"></xs:element>

<xs:element name="disclaimer-title" type="xs:string"></xs:element>

<xs:element name="recording-allowed" type="xs:boolean"></xs:element>

<xs:element name="externaluser-recording-allowed" type="xs:boolean" />
```

```
  <xs:element name="recording-notification" type="xs:boolean"></xs:element>
  <xs:element name="server-mode" type="xs:unsignedInt"></xs:element>
```

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

  <!--
      ENCRYPTION KEY TYPE
    -->
  <xs:complexType name="encryption-key-type">
    <xs:sequence>
      <xs:element name="x509-certificate" type="xs:base64Binary"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      CONFERENCE KEY TYPE
    -->
  <xs:complexType name="conference-key-type">
    <xs:sequence>
      <xs:element name="cms-data" type="xs:base64Binary"></xs:element>
      <xs:element name="opaque" type="tns:encryption-key-opaque-type"
minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>



<!--
      ENCRYPTED CONTENT TYPE
    -->
    <xs:complexType name="encrypted-content-type">
        <xs:sequence>
            <xs:element name="cms-data" type="xs:base64Binary"/>
            <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">
    <xs:sequence>
```

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

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

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

<!--
      PSTN bridging access information for the conference
-->
<xs:element name="pstn-access" type="tns:pstn-access-type"></xs:element>
<xs:simpleType name="pstn-meeting-id-type">
  <xs:restriction base="xs:string">
    <xs:pattern value="[1-9][0-9]*"></xs:pattern>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="pstn-access-type">
  <xs:sequence>
    <xs:element name="id" type="tns:pstn-meeting-id-type" minOccurs="0"></xs:element>
       <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
</xs:complexType>


  <!--
      CONFERENCE VIEW TYPE
    -->
  <xs:element name="conference-view" type="tns:conference-view-type"></xs:element>
  <xs:complexType name="conference-view-type">
    <xs:sequence>
      <xs:element name="entity-view" type="entity-view-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
```

```
    <xs:attribute ref="ci:state" default="full"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      ENTITY VIEW TYPE
    -->
  <xs:complexType name="entity-view-type">
    <xs:sequence>
      <xs:element name="entity-capabilities" type="entity-capabilities-type"
minOccurs="0"></xs:element>
      <xs:element name="entity-policy" type="entity-policy-type" minOccurs="0"></xs:element>
      <xs:element name="entity-settings" type="entity-settings-type"
minOccurs="0"></xs:element>
      <xs:element name="entity-state" type="entity-state-type" minOccurs="0"></xs:element>
      <xs:element name="entity-shared-data" type="entity-shared-data-type"
minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute ref="ci:state" default="full"></xs:attribute>
    <xs:attribute name="entity" type="xs:anyURI"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
  <xs:element name="modal-parameters" type="tns:modal-parameters-type"></xs:element>
  <xs:complexType name="modal-parameters-type">
    <xs:sequence>
      <xs:choice>
        <xs:element name="audio-parameters" type="msav:audio-parameters-type"
minOccurs="0"></xs:element>
        <xs:element name="video-parameters" type="msav:video-parameters-type"
minOccurs="0"></xs:element>
      </xs:choice>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>


  <!--
      ENTITY CAPABILITIES TYPE
    -->
  <xs:complexType name="entity-capabilities-type">
    <xs:sequence>
      <xs:choice minOccurs="0">
        <xs:element ref="msacp:capabilities"></xs:element>
        <xs:element ref="msav:capabilities"></xs:element>

        <!-- <xs:element ref="msdata:capabilities"/> -->

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

```
  <!--
      ENTITY POLICY TYPE
    -->
  <xs:complexType name="entity-policy-type">
    <xs:sequence>
      <xs:element name="guid" type="xs:anyURI" minOccurs="0" nillable="true"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
      ENTITY SETTINGS TYPE
    -->
  <xs:complexType name="entity-settings-type">
    <xs:sequence>
      <xs:element name="mediaFiltersRules" type="tns:media-filters-rules-type"
minOccurs="0"></xs:element>
      <xs:element name="media" type="ci:conference-media-type" minOccurs="0"></xs:element>
      <xs:choice minOccurs="0">
        <xs:element ref="msacp:settings"></xs:element>
        <xs:element ref="msav:settings"></xs:element>
        <xs:element ref="msdata:settings"></xs:element>
        <xs:element ref="msim:settings"></xs:element>
      </xs:choice>
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

  <!--
        ENTITY STATE TYPE
     -->
  <xs:complexType name="entity-state-type">
    <xs:sequence>
      <xs:element name="displayText" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="userCount" type="xs:unsignedInt" minOccurs="0"></xs:element>
      <xs:element name="active" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="locked" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="mediaFiltersRules" type="tns:media-filters-rules-type"
minOccurs="0"></xs:element>
      <xs:element name="media" type="ci:conference-media-type" minOccurs="0"></xs:element>
      <xs:choice minOccurs="0">
        <xs:element ref="msacp:state"></xs:element>
        <xs:element ref="msav:state"></xs:element>
      </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: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:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </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>

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

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

  <!--
      MS Authentication Type
    -->
  <xs:element name="authMethod" type="tns:auth-method-type"></xs:element>
  <xs:simpleType name="auth-method-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="enterprise"></xs:enumeration>
      <xs:enumeration value="anonymous"></xs:enumeration>
      <xs:enumeration value="federated"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>

  <!--
      MS Role Type
    -->
  <xs:simpleType name="ms-role-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="attendee"></xs:enumeration>
      <xs:enumeration value="presenter"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="pstnRole" type="tns:ms-role-type"></xs:element>

  <!--
      USER ACCESS TYPE
    -->
  <xs:element name="accessMethod" type="tns:access-method-type"></xs:element>
  <xs:simpleType name="access-method-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="external"></xs:enumeration>
      <xs:enumeration value="internal"></xs:enumeration>
    </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">
   <xs:complexType name="client-info-type">
    <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">
      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator" />
        <xs:element ref="msci2:user-agent" minOccurs="0" maxOccurs="1" />
        <xs:element ref="msci2:lobby-capable" minOccurs="0" />
```

```
        <xs:element ref="msci2:lobby-capable" minOccurs="0"/>
        <xs:sequence minOccurs="0">
          <xs:element ref="cis:separator" />
          <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded" />
        </xs:sequence>
      </xs:sequence>
    </xs:sequence>
  </xs:complexType>
  <!--
      ORGANIZER TYPE
    -->
  <xs:element name="organizer" type="tns:organizer-type"></xs:element>
  <xs:complexType name="organizer-type">
    <xs:sequence>
      <xs:element name="entity" type="xs:anyURI" minOccurs="0"></xs:element>
      <xs:element name="display-name" type="xs:string" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>

<!--
      SESSION-ON-BEHALF-OF TYPE
    -->
  <xs:element name="session-on-behalf-of" type="tns:session-on-behalf-of-type"></xs:element>
  <xs:complexType name="session-on-behalf-of-type">
    <xs:sequence>
      <xs:element name="entity" type="xs:anyURI" minOccurs="1"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>

<!--
      CLIENT RECORDING
-->
<xs:complexType name="client-recording-type">
  <xs:sequence>
    <xs:any namespace="##other" minOccurs="0" maxOccurs="unbounded"
processContents="lax"></xs:any>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
</xs:complexType>

   <xs:element name="client-recording" type="tns:client-recording-type"></xs:element>

   <!--
      MEDIA FILTER
    -->
  <xs:element name="media-filter" type="tns:media-filter-type"></xs:element>
  <xs:element name="media-ingress-filter" type="tns:media-filter-type"></xs:element>
  <xs:element name="media-egress-filter" type="tns:media-filter-type"></xs:element>
  <xs:simpleType name="media-filter-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="block"></xs:enumeration>
      <xs:enumeration value="unblock"></xs:enumeration>
    </xs:restriction>
```

```
</xs:simpleType>

<!--
    Post dial strings
  -->
<xs:element name="post-dial" type="xs:string"></xs:element>

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

<!--
    endpoint capabilities
  -->
<xs:element name="endpoint-capabilities" type="endpoint-capabilities-type"></xs:element>
<xs:complexType name="endpoint-capabilities-type">
  <xs:sequence>
    <xs:choice minOccurs="0">
      <xs:element ref="msacp:endpoint-capabilities"></xs:element>
      <xs:element ref="msav:endpoint-capabilities"></xs:element>
      <xs:element ref="msdata:endpoint-capabilities"></xs:element>
      <xs:element ref="msim:endpoint-capabilities"></xs:element>
    </xs:choice>
    <xs:sequence minOccurs="0">
      <xs:element ref="cis:separator"></xs:element>
      <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
</xs:complexType>

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

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

<!-- ERROR ELEMENT -->
<xs:element name="error" type="error-type"></xs:element>

<!-- ERROR TYPE -->
<xs:complexType name="error-type">
  <xs:sequence>
    <xs:element name="code" type="xs:string"></xs:element>
    <xs:element name="description" type="xs:string" minOccurs="0"></xs:element>
  </xs:sequence>
</xs:complexType>

<!-- session-type string -->
<xs:attribute name="session-type" type="xs:string"></xs:attribute>

<!-- epid -->
<xs:attribute name="epid" type="xs:string"></xs:attribute>

<!-- sip-instance-->
<xs:attribute name="sip-instance" type="xs:string"></xs:attribute>
```

```
  <!-- endpoint-uri uri -->
  <xs:attribute name="endpoint-uri" type="xs:anyURI"></xs:attribute>

  <!-- refer-to-uri uri -->
  <xs:attribute name="refer-to-uri" type="xs:anyURI"></xs:attribute>

  <!-- asserted-identity -->
  <xs:attribute name="asserted-identity" type="xs:string"></xs:attribute>
</xs:schema>
```

8.3 conference-info-separator Namespace

This namespace is identified by the following URN:

urn:ietf:params:xml:ns:conference-info-separator

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:conference-info-separator"
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:tns="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="urn:ietf:params:xml:ns:conference-info-
separator">

  <!--
    This defines a separator marking the beginning of extensions
    -->
  <xs:element name="separator">
    <xs:complexType></xs:complexType>
  </xs:element>
</xs:schema>
```

8.4 dataconfinfoextensions Namespace

This namespace is identified by the following URN:

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

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/dataconfinfoextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="app-viewing-behavior-type">
    <xs:annotation>
      <xs:documentation>
                Determines what form of application / desktop sharing to use for a
conference.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
```

```
      <xs:enumeration value="disabled"></xs:enumeration>
      <xs:enumeration value="enableWithoutSharingControl"></xs:enumeration>
      <xs:enumeration value="enableWithSharingOfOnlyASingleApplication"></xs:enumeration>
      <xs:enumeration value="enableWithFullSharing"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="conferencing-type-type">
    <xs:annotation>
      <xs:documentation>
                Determines whether this is a presentation or escalation conference.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="presentation"></xs:enumeration>
      <xs:enumeration value="collaboration"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="settings-type">
    <xs:sequence>
      <xs:element name="app-viewing-behavior" type="tns:app-viewing-behavior-
type"></xs:element>
      <xs:element name="conferencing-type" type="tns:conferencing-type-type" minOccurs="0"
maxOccurs="1"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="settings" type="tns:settings-type"></xs:element>
  <xs:complexType name="endpoint-capabilities-type">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="endpoint-capabilities" type="tns:endpoint-capabilities-
type"></xs:element>
  <xs:complexType name="data-mcu-state-type" ms:className="C3PDataMcuStateType">
    <xs:sequence>
      <xs:element name="hasContent" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="hasContentInMeeting" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="hasPresentedContent" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="data-mcu-state" type="tns:data-mcu-state-type"
ms:ignore="true"></xs:element>
</xs:schema>
```

8.5 avconfinfoextensions Namespace

This namespace is identified by the following URN:

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

```
<?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:tns="http://schemas.microsoft.com/rtc/2005/08/avconfinfoextensions"
xmlns:msmcu="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <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 through policy.  (entity-policy
element of entity-view )
      -->
      <xs:element name="supports-audio" type="xs:boolean"></xs:element>
      <xs:element name="supports-video" type="xs:boolean"></xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="capabilities" type="tns:capabilities-type"></xs:element>
  <xs:complexType name="audio-settings-type">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="video-settings-type">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="settings-type">
    <xs:sequence>
      <xs:element name="audio" type="tns:audio-settings-type" minOccurs="0"></xs:element>
      <xs:element name="video" type="tns:video-settings-type" minOccurs="0"></xs:element>
      <xs:element ref="msmcu:entry-exit-announcements" minOccurs="0"></xs:element>

      <xs:sequence minOccurs="0">
        <xs:element ref="cis:separator"></xs:element>
        <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
      </xs:sequence>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="settings" type="tns:settings-type"></xs:element>
  <xs:complexType name="contributing-sources-type">
    <xs:sequence>
      <xs:element name="entry" type="xs:anyURI" minOccurs="0"
maxOccurs="unbounded"></xs:element>
    </xs:sequence>
    <xs:attribute name="empty" type="xs:boolean" use="optional"></xs:attribute>
  </xs:complexType>
  <xs:complexType name="dominant-speaker-source-type">
```

```
    <xs:sequence>
      <xs:element name="entry" type="xs:anyURI" minOccurs="0"></xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="video-parameters" type="tns:video-parameters-type"></xs:element>
  <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"></xs:element>

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

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

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

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

8.6 imconfinfoextensions Namespace

This namespace is identified by the following URN:

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

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/imconfinfoextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:complexType name="settings-type">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="settings" type="tns:settings-type"></xs:element>
  <xs:simpleType name="supported-im-formats-type">
    <xs:annotation>
      <xs:documentation>
                A string indicating the im content types that can be rendered by the
endpoint.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:maxLength value="512"></xs:maxLength>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="user-agent-type">
    <xs:annotation>
      <xs:documentation>
                A string indicating the user agent of the endpoint.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:maxLength value="128"></xs:maxLength>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="endpoint-capabilities-type">
    <xs:sequence>
      <xs:element name="supported-im-formats" type="tns:supported-im-formats-type"
minOccurs="0"></xs:element>
      <xs:element name="user-agent" type="tns:user-agent-type" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="endpoint-capabilities" type="tns:endpoint-capabilities-
type"></xs:element>
</xs:schema>
```

8.7 acpconfinfoextensions Namespace

This namespace is identified by the following URN:

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

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema targetNamespace="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
elementFormDefault="qualified" attributeFormDefault="unqualified" version="1.0"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:msci="http://schemas.microsoft.com/rtc/2005/08/confinfoextensions"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/acpconfinfoextensions"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="entry-exit-announcement-type">
    <xs:annotation>
      <xs:documentation>
                The different kinds of audio announcement which can
                be played when a user joins or leaves the conference.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="silence"></xs:enumeration>
      <xs:enumeration value="tone"></xs:enumeration>
      <xs:enumeration value="recordedName"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="telephone-number-type">
    <xs:annotation>
      <xs:documentation>
                A telephone number.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:maxLength value="128"></xs:maxLength>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="passcode-type">
    <xs:annotation>
      <xs:documentation>
                A passcode for an Audio Conferencing Provider.

      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:maxLength value="128"></xs:maxLength>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="capability-name-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="dialIn"></xs:enumeration>
      <xs:enumeration value="dialOut"></xs:enumeration>
      <xs:enumeration value="disconnectUser"></xs:enumeration>
      <xs:enumeration value="muteUser"></xs:enumeration>
      <xs:enumeration value="joinMuted"></xs:enumeration>
      <xs:enumeration value="muteLock"></xs:enumeration>
```

```
      <xs:enumeration value="lockConference"></xs:enumeration>
      <xs:enumeration value="changeEntryExitAnnouncement"></xs:enumeration>
      <xs:enumeration value="silenceEntryExitAnnouncement"></xs:enumeration>
      <xs:enumeration value="toneEntryExitAnnouncement"></xs:enumeration>
      <xs:enumeration value="recordedNameEntryExitAnnouncement"></xs:enumeration>
      <xs:enumeration value="recordNames"></xs:enumeration>
      <xs:enumeration value="playRecordedName"></xs:enumeration>
      <xs:enumeration value="multipleLeaders"></xs:enumeration>
      <xs:enumeration value="hashCodeReconciliation"></xs:enumeration>
      <xs:enumeration value="activateConference"></xs:enumeration>
      <xs:enumeration value="sidebars"></xs:enumeration>
      <xs:enumeration value="robotDialInInformation"></xs:enumeration>
      <xs:enumeration value="changeUserPstnRole"></xs:enumeration>
      <xs:enumeration value="firstPartyDialOut"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
  <xs:complexType name="capability-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
    <xs:attribute name="name" type="tns:capability-name-type" use="required"></xs:attribute>
    <xs:anyAttribute namespace="##other" processContents="lax"></xs:anyAttribute>
  </xs:complexType>
  <xs:complexType name="capabilities-type">
    <xs:sequence>
      <xs:element name="capability" type="tns:capability-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="capabilities" type="tns:capabilities-type"></xs:element>
  <xs:complexType name="telephone-numbers-type">
    <xs:sequence>
      <xs:element name="entry" type="tns:telephone-number-type" minOccurs="0"
maxOccurs="unbounded"></xs:element>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="settings-type">
    <xs:sequence>
      <xs:element name="tollNumber" type="tns:telephone-number-type"
minOccurs="0"></xs:element>
      <xs:element name="tollFreeNumber" type="tns:telephone-number-type"
minOccurs="0"></xs:element>
      <xs:element name="domain" type="xs:string" minOccurs="0"></xs:element>
      <xs:element name="leaderPasscode" type="tns:passcode-type" minOccurs="0"></xs:element>
      <xs:element name="participantPasscode" type="tns:passcode-type"></xs:element>
      <xs:element name="showTollNumber" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="showTollFreeNumber" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="enableCallMe" type="xs:boolean" minOccurs="0"></xs:element>
      <xs:element name="robotCallerIds" type="tns:telephone-numbers-type"
minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="settings" type="tns:settings-type"></xs:element>
  <xs:complexType name="state-type">
    <xs:sequence>
```

```
      <xs:element name="tollNumber" type="tns:telephone-number-type"
minOccurs="0"></xs:element>
      <xs:element name="tollFreeNumber" type="tns:telephone-number-type"
minOccurs="0"></xs:element>
      <xs:element name="participantPasscode" type="tns:passcode-type"
minOccurs="0"></xs:element>
      <xs:element name="entryExitAnnouncement" type="tns:entry-exit-announcement-type"
minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="state" type="tns:state-type"></xs:element>
  <xs:complexType name="shared-data-type">
    <xs:sequence>
      <xs:element name="robotDialInPhoneNumber" type="tns:telephone-number-type"
minOccurs="0"></xs:element>
      <xs:element name="robotDialInPostDial" type="xs:string" minOccurs="0"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="shared-data" type="tns:shared-data-type"></xs:element>
  <xs:complexType name="endpoint-capabilities-type">
    <xs:sequence>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="endpoint-capabilities" type="tns:endpoint-capabilities-
type"></xs:element>
</xs:schema>
```

8.8 asconfinfoextensions Namespace

This namespace is identified by the following URN:

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

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

<!--Extensions for Application Sharing -->
<xs:schema version="1.0"
targetNamespace="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
elementFormDefault="qualified" attributeFormDefault="unqualified"
xmlns:tns="http://schemas.microsoft.com/rtc/2005/08/asconfinfoextensions"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:xs="http://www.w3.org/2001/XMLSchema">

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

  <!-- SESSION IDS TYPE -->
  <xs:complexType name="session-ids-type">
    <xs:sequence>
```

```
      <xs:element name="session-id" type="xs:string" minOccurs="1"
maxOccurs="unbounded"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="session-ids" type="tns:session-ids-type"></xs:element>

  <!-- CABILITIES TYPE -->
  <xs:complexType name="capabilities-type">
    <xs:sequence>
      <xs:element name="control-permission" type="tns:control-permission-type-
ex"></xs:element>
      <xs:any namespace="##other" processContents="lax" minOccurs="0"
maxOccurs="unbounded"></xs:any>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="capabilities" type="tns:capabilities-type"></xs:element>

  <!-- CONTROL PERMISSION TYPE -->
  <xs:simpleType name="control-permission-type-ex">
    <xs:union memberTypes="tns:control-permission-type xs:string"></xs:union>
  </xs:simpleType>
  <xs:simpleType name="control-permission-type">
    <xs:restriction base="xs:string">
      <xs:enumeration value="All"></xs:enumeration>
      <xs:enumeration value="ActiveDirectoryUsers"></xs:enumeration>
      <xs:enumeration value="None"></xs:enumeration>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

8.9 commonmcuextensions

This namespace is identified by the following URN:

http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions

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

<xs:schema version="1.0"
targetNamespace="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:tns="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:ms="urn:microsoft-cpp-xml-serializer"
xmlns="http://schemas.microsoft.com/rtc/2009/03/commonmcuextensions"
xmlns:ci="urn:ietf:params:xml:ns:conference-info"
xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
  <!-- C3P schema extension for elements common to multiple MCUs -->

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

  <!--
```

```
    Sequence of possible capabilities, to be inserted (with
    reference to the capability-value-type type) as they are defined in the
    future.  The structure of this XML is similar to media-capabilities
    (see ms-ci-ext.xsd)
  -->
  <xs:complexType name="session-capabilities-type"
ms:className="C3PMcuCommonSessionCapabilitiesType">
    <xs:sequence>
      <xs:any namespace="##any" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:element name="session-capabilities" type="tns:session-capabilities-type"
ms:ignore="true"/>

  <xs:simpleType name="capability-value-type">
    <xs:annotation>
      <xs:documentation>
        Possible capability values (for both endpoint-capabilities and media-capabilities)
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="sendrecv"/>
      <xs:enumeration value="sendonly"/>
      <xs:enumeration value="recvonly"/>
      <xs:enumeration value="none"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="media-capability-type">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="value" type="tns:capability-value-type"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!--
    PERMISSION OPTIONS TYPE
  -->
  <xs:element name="permission-options" type="tns:permission-options-type" ms:ignore="true"/>

  <xs:complexType name="permission-options-type">
    <xs:sequence>
      <xs:element name="permission-option" type="tns:permission-option-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>

  <!--
    PERMISSION OPTION TYPE
  -->
  <xs:complexType name="permission-option-type">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="value" type="xs:string"/>
      <xs:element name="mutable" type="xs:boolean"/>
```

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

  <!--
    PERMISSIONS TYPE
  -->
  <xs:element name="permissions" type="tns:permissions-type" ms:ignore="true"/>

  <xs:complexType name="permissions-type">
    <xs:sequence>
      <xs:element name="permission-type" type="tns:permission-type" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <!--
    PERMISSION TYPE
  -->
  <xs:complexType name="permission-type">
    <xs:sequence>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="value" type="xs:string"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
  <xs:complexType name="entry-exit-announcements-type">
    <xs:sequence>
      <xs:element name="modifiable" type="xs:boolean" minOccurs="0"/>
      <xs:element name="enabled" type="xs:boolean"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>

  <xs:element name="entry-exit-announcements" type="tns:entry-exit-announcements-type"
ms:ignore="true"/>

</xs:schema>
```

9 Appendix D: 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.

<1> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<2> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<3> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<4> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<5> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported. Federated users are always treated as authenticated for purposes of joining a conference.

<6> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<7> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<8> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<9> Section 2.2.2.3: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<10> Section 2.2.2.4: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<11> Section 2.2.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<12> Section 2.2.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<13> Section 2.2.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<14> Section 2.2.2.4: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<15> Section 2.2.2.6: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<16> Section 2.2.3.3: Office Communications Server 2007, Office Communicator 2007: The **endpoint** element cannot contain child elements. All other products can contain a **session-on-behalf-of** element within the **endpoint** element.

<17> Section 2.2.3.17: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<18> Section 2.2.3.18: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<19> Section 2.2.3.19: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<20> Section 2.2.3.20: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<21> Section 2.2.6: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<22> Section 3.2: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<23> Section 3.2.4.1.1: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, the client is required to add a **session-on-behalf-of** element to the **addUser** Request Body.

<24> Section 3.2.4.2: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported. For all other products, if the validation fails, the focus is required to respond with a 403 Forbidden response.

<25> Section 3.3.4.1: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<26> Section 3.11: Office Communications Server 2007, Office Communicator 2007: This behavior is not supported.

<27> Section 3.13: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

<28> Section 3.14: Office Communications Server 2007, Office Communicator 2007, Office Communications Server 2007 R2, Office Communicator 2007 R2: This behavior is not supported.

10 Change Tracking

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

11 Index

A

C

D

J

L

M

N

O

P

R

S

T

V