XCON Working Group
Internet-Draft
Expires: July 3, 2006

O. Levin
Microsoft Corporation
R. Even
Polycom
P. Hagendorf
RADVISION

December 30, 2005

Centralized Conference Control Protocol
draft-levin-xcon-cccp-04

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on July 3, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

This document defines a Centralized Conferencing Control Protocol (CCCP) as a part of the XCON Working Group protocols suite. CCCP uses a client-server model for creation, querying, and manipulation of XCON entities, conference objects and sub-objects. An XCON entity, which implements a CCCP server, provides a means for

authorized CCCP clients (e.g. conference participants) to affect the behavior of a conference in the XCON system.

Table of Contents

1. Introduction 4
2. Terminology 4
3. Transport 4
4. The Protocol 4
4.1. Transaction Model 4
4.2. Multiple Primitives in a Single Operation 6
4.3. Response Codes and Failures 6
5. Using the Data Model 6
6. Design Rationale 7
6.1. Remote Procedure vs. Data Manipulation 7
6.2. CCCP Transactions vs. SOAP 8
7. Primitives 9
7.1. System Primitives 9
7.1.1. Cancel 9
7.1.2. Ping 10
7.1.3. getTemplates 11
7.1.4. getActiveConferences 11
7.2. Conference Primitives 11
7.2.1. addConference 11
7.2.2. modifyConference 13
7.2.3. deleteConference 14
7.2.4. getConference 15
7.3. User Primitives 16
7.3.1. addUser 16
7.3.2. modifyUser 17
7.3.3. modifyUserRoles 18
7.3.4. deleteUser 19
7.3.5. getUser 20
7.4. Endpoint Primitives 21
7.4.1. addEndpoint 21
7.4.2. modifyEndpoint 22
7.4.3. deleteEndpoint 23
7.4.4. getEndpoint 24
7.5. Endpoint Media Primitives 25
7.5.1. addEndpointMedia 26
7.5.2. modifyEndpointMedia 26
7.5.3. deleteEndpointMedia 27
7.5.4. getEndpointMedia 28
7.6. Sidebar Primitives 29
7.6.1. addSidebar 29
7.6.2. modifySidebar 30
7.6.3. deleteSidebar 30

7.6.4. addUserToSidebar 30
7.6.5. deleteUserFromSidebar 30
7.6.6. moveUserBetweenSidebars 30
7.7. Stimulus Primitives 30
8. The XML Schema 30
9. IANA Considerations 49
9.1. URN Sub-Namespace Registration for urn:ietf:params:xml:ns:cccp 49
9.2. XML Schema Registration 50
10. Security Considerations 50
11. Acknowledgements 50
12. References 51
12.1. Normative References 51
12.2. Informative References 51
Authors' Addresses 52
Intellectual Property and Copyright Statements 53

1. Introduction

The SIPPING Conference Framework [6] describes a general centralized conferencing architecture. The XCON Framework [7] defines how this architecture can be realized by an XCON compliant system. This document defines a Centralized Conferencing Control Protocol (CCCP) as a conference control protocol in the XCON protocols suite described in XCON Framework [7]

CCCP uses a client-server model for creation, querying, and manipulation of XCON entities, conference objects and sub-objects. By implementing a CCCP server, an XCON entity provides a means for authorized CCCP clients (e.g. conference participants) to affect the behavior of a conference in the XCON system. CCCP is a semantic-oriented protocol, which uses the XML types defined in the SIPPING Conference Package [2] for the representation of the conference object and its sub-objects . In future, the CCCP can be extended to include manipulations on additional conferencing objects being represented by XML schemas such as policies and templates.

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in BCP 14, RFC 2119 [1] and indicate requirement levels for compliant implementations.

3. Transport

The protocol design assumes that CCCP runs on a reliable transport protocol.

CCCP is agnostic to the details of the transport protocol being used beneath and does not rely on any information being conveyed on the transport level. This model allows for using different transport protocols based on the system requirements and also for multiplexing otherwise independent CCCP communications over a common transport channel.

4. The Protocol

4.1. Transaction Model

CCCP is a client-server protocol. The protocol defines two

operations: request and response.

A client issues requests to a server. The server MUST reply with a single final response. Two final responses are defined: "failure" and "success".

Before issuing the final response, the server MAY reply with multiple provisional responses. Currently a single provisional response "pending" is defined.

Editor's Note: Timeouts TBD.

A CCCP request carries the following attributes:

Attribute	Description
requestId	A unique string generated by the CCCP client across all its requests.
from	A transport URI which identifies the CCCP client.
to	A transport URI which identifies the CCCP server.
originator	A trusted ID of the originator of the request.

Table 1

The combination of the 'requestId', 'to', and 'from' attributes in the request constitutes the CCCP transaction ID.

A CCCP response carries the following attributes:

Attribute	Description
requestId	The original request ID generated by the client and echoed as is by the server.
from	A transport URI which identifies the CCCP server.
to	A transport URI which identifies the CCCP client.
code	The general response code: success, pending, or failure.
reason	The general CCCP failure reason.
timeout	The updated timeout used with pending responses (details TBD).
retryAfter	The suggested delay used with serverBusy responses.

Table 2

4.2. Multiple Primitives in a Single Operation

A CCCP operation (i.e. a request and a corresponding response) MAY contain multiple primitives. The CCCP MUST process the received primitives one-by-one in the order they appear in the request body. If the request contains multiple primitives, the corresponding response operation MUST contain the response primitive for each and in the same order as in the request.

Multiple primitives within the same request MUST be executed as an atomic operation. This means that if one primitive fails, the state of the object MUST be rolled back to its state before processing the request.

If a CCCP server is not willing to process a multi-primitive request, it MUST fail the transaction with the response code "notSupported".

4.3. Response Codes and Failures

CCCP defines the following reasons for failure of a request operation

Failure	Description
serverBusy	Optional retryAfter can be included in the response.
timeout	Operation took too long and was aborted by the server
unauthorized	Client is not authorized to perform the operation.
requestMalformed	The XML document in the request is either not well-formed or not compliant with the schema.
requestTooLarge	Result of the request operation length check.
requestCancelled	The pending request was canceled by CCCP.
notSupported	One of the primitives or their combination is not supported by the server.
otherFailure	Result of any other server fault condition.

Table 3

Most CCCP primitives define their own optional response codes. This allows communicating the detailed primitive result in addition to the CCCP general response code.

5. Using the Data Model

The CCCP operations are applied to the data objects expressed in terms of SIPPING Conference Package [2] XML types whenever possible. The considerations listed below MUST be taken into account when using the schema with CCCP.

The information included in the request expresses the desired data object state to be achieved after the operation is successfully completed. By definition, the CCCP primitives allow for inclusion of any information that can be expressed in terms of the conference-type and its sub-types.

Attributes 'state' and 'version' of the conference-type and its sub-types are never used with CCCP. The information in the response is provided as a feedback to the request only and typically does not carry the full state of the object.

For each primitive request, the CCCP explicitly defines (see Section 6) what information (i.e. attributes and elements) MUST be provided by the client and what information (when provided by the client) MUST NOT be ignored by the server. The rest of the information included by the client MAY be treated as optional by the server.

For each primitive response, the CCCP explicitly defines (see Section 6) what information (i.e. attributes and elements) if included by the server MUST NOT be ignored by the client. The rest of the information included by the client MAY be treated as optional by the server. If neither mandatory information nor new data is generated, the server MAY return minimum schema compliant construct, such as an element with empty body and the attributes identifying the corresponding request only. On the other hand, the CCCP server MAY include any rich dynamically generated information to the client (for example, to be displayed to a user or be logged in by the system) in the response. The client MAY ignore any information received in the response, unless stated otherwise in Section 6.

6. Design Rationale

6.1. Remote Procedure vs. Data Manipulation

The first step in the decision process was to compare between a data manipulation approach and a remote procedure call approach.

The advantages of the data manipulation approach are:

- o Mostly appropriate for simple lightweight clients using built on a stimulus-response model

- o The data model is the protocol -- any existing generic data manipulation protocol will work
- o Adding functionality does not require changes to the protocol, only to the data model. The server implementation needs to track these changes of course and implement the corresponding new functionality.

The advantages for the remote procedure call approach are:

- o Mostly appropriate for conferencing-aware client applications that are built to automate the experience and/or hide conferencing complexity from the end user
- o Makes compound operations and conference specific operations explicit and thus much easier and faster for conferencing server implementation
- o Allows for inclusion of data manipulation primitives when desired, e.g. for manipulating templates. In other words, a hybrid approach is easily built where it makes sense.

We came to a conclusion that there is a place for both approaches to co-exist in the industry. The decision of which to use in each case will be based on the client side requirements.

We also came to a conclusion that it is not necessary to define a new conference-specific protocol in order to meet the lightweight client requirements for a stimulus-response approach. Instead one of the existing standard data manipulation protocols can be used for this purpose. This approach will require standardizing the user interface in terms of a standard conferencing XML schema(s).

On the other hand, smart conference-aware conferencing clients cannot operate using abstract stimulus-response approach only. In order to achieve both efficient and flexible conference control, a truly application-specific, i.e. a conferencing control protocol, is needed. The CCCP is defined with this need in mind.

6.2. CCCP Transactions vs. SOAP

It is not difficult to map the CCCP primitives and functionality into a SOAP compliant protocol as shown in [TBD]. Apart from the pure syntax differences, the two protocols differ in the way they report the final result of a requested operation.

According to the SOAP specification, each transaction is comprised of a request and a single corresponding response (which are transported over the underlying HTTP transaction). This definition means that an application that uses SOAP needs to define its own conventions for handling requests that can not be completed immediately. Typically this is being achieved by introducing an additional notification

channel in the direction opposite to the request. It is important to note that this channel must not be mistaken with the conference state notification channel defined in [conf package].

The conference control notification channel should be provided to the client originating the request only and would not necessarily reflect any changes in the conference state. For example, in case a request transaction is completed with the "in progress" response and later a server fails to execute the request, the notification sent to the client will not convey any change in the conference state, but rather needs to convey the request ID and the failure reason.

CCCP is different at that sense from the SOAP architecture. CCCP does not use any dedicated notification channel. Instead CCCP has the notion of possible multiple pending responses always followed by the final (either success or failure) response. This approach simplifies the conferencing application and also makes CCCP truly independent from the underlying transport protocol.

It is important to note that a CCCP client is expected to support the Conference State event package as the means for maintaining the most current synchronized conference state. The client should not use the CCCP responses for updating the local copy of the conference state document.

7. Primitives

This section describes the defined CCCP primitives and includes valid XML document examples for each. The corresponding CCCP XML schema is provided in Section 7.

7.1. System Primitives

7.1.1. Cancel

Cancel a pending request.

This primitive can be issued by a client to cancel a pending transaction. The primitive is an independent transaction on its own. The body of the primitive MUST carry the requestId of one of the pending requests.

```
<request
    requestId="10"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <cancel>
                <requestId>5</requestId>
        </cancel>
</request>
```

Note that a valid response can contain an empty body.

```
<response
    requestId="10"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <cancel/>
</response>
```

7.1.2. Ping

Ping a CCCP Server.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <ping/>
</request>
```

A successful response is shown below.

```
<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <ping/>
</response>
```

7.1.3. getTemplates

Get the list of templates supported by the system. XML TBD.

7.1.4. getActiveConferences

Get the list of conference identifiers for active conference objects in the system.

XML TBD.

7.2. Conference Primitives

7.2.1. addConference

Create a conference.

The 'conferenceEntity' value in the request is a client's suggestion only. The CCCP server MAY replace the suggested value with a different 'conferenceEntity' value in the corresponding response.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <addConference>
     <conference-info
      entity="sip:McuConf1@company.com"
       xmlns="urn:ietf:params:xml:ns:conference-info">
       <conference-description>
        <subject>Design Review</subject>
         <conf-uris>
          <entry>
           <uri>tel:+1-8666119036</uri>
           <display-text>FFD bridge</display-text>
          </entry>
         </conf-uris>
         <service-uris>
          <entry>
           <uri>https://company.com/ConfServer</uri>
          </entry>
         </service-uris>
         <available-media>
          <entry label="1">
           <type>audio</type>
          </entry>
         </available-media>
        </conference-description>
       </conference-info>
      </addConference>
</request>
```

The CCCP server MAY replace the suggested 'conferenceEntity' with a different value in the corresponding response. In the case of a successful response, the CCCP client MUST use the new value and SHOULD use all the new parameters allocated by the server to the conference.

```
<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <addConference>
    <conference-info
      entity="sip:McuConf1@company.com"
       xmlns="urn:ietf:params:xml:ns:conference-info">
       <conference-description>
        <subject>Design Review</subject>
         <conf-uris>
          <entry>
           <uri>tel:+1-8666119036</uri>
           <display-text>FFD bridge</display-text>
          </entry>
         </conf-uris>
         <service-uris>
          <entry>
           <uri>https://company.com/ConfServer</uri>
          </entry>
         </service-uris>
         <available-media>
          <entry label="1">
           <type>audio</type>
          </entry>
         </available-media>
        </conference-description>
       </conference-info>
        </addConference>
</response>
```

7.2.2. modifyConference

Modify conference parameters.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyConference>
     <conference-info
      entity="sip:McuConf1@company.com"
      xmlns="urn:ietf:params:xml:ns:conference-info">
       <conference-description>
        <subject>Spec Review</subject>
       </conference-description>
      </conference-info>
     </modifyConference>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <modifyConference/>
</response>
```

7.2.3. deleteConference

Remove conference from the system.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteConference>
     <conferenceKeys
      confEntity="sip:McuConf1@company.com"/>
    </deleteConference>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <deleteConference/>
</response>
```

7.2.4. getConference

Retrieve the conference information.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <getConference>
                <conferenceKeys
                 confEntity="sip:conf233@example.com"/>
        </getConference>
</request>
```

```
<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
        <getConference>
    <conference-info
      entity="sip:McuConf1@company.com"
       xmlns="urn:ietf:params:xml:ns:conference-info">
       <conference-description>
        <subject>Design Review</subject>
         <conf-uris>
          <entry>
           <uri>tel:+1-8666119036</uri>
           <display-text>FFD bridge</display-text>
          </entry>
         </conf-uris>
         <service-uris>
          <entry>
           <uri>https://company.com/ConfServer</uri>
          </entry>
         </service-uris>
         <available-media>
          <entry label="1">
           <type>audio</type>
          </entry>
         </available-media>
        </conference-description>
       </conference-info>
        </getConference>
</response>
```

7.3. User Primitives

7.3.1. addUser

The client MUST provide the 'userEntity' value in the request.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addUser>
    <conferenceKeys
     confEntity="sip:conf233@example.com"/>
     <user entity="sip:bob@example.com"
     xmlns="urn:ietf:params:xml:ns:conference-info">
      <display-text>Bob Hoskins</display-text>
      <roles>
      <entry>presenter</entry>
      </roles>
     </user>
    </addUser>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addUser/>
</response>
```

7.3.2. modifyUser

Modify the user information.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyUser>
     <conferenceKeys
      confEntity="sip:conf233@example.com"/>
       <user entity="sip:bob@example.com"
        xmlns="urn:ietf:params:xml:ns:conference-info">
        <display-text>Bob Hoskins III</display-text>
        <associated-aors>
         <entry>
          <uri>tel:2562566</uri>
           <display-text>the description</display-text>
           <purpose>optional tbd values</purpose>
          </entry>
         </associated-aors>
         <roles>
          <entry>presenter</entry>
         </roles>
        </user>
     </modifyUser>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyUser/>
</response>
```

7.3.3. modifyUserRoles

This is a dedicated primitive to change user's roles. The same effect can be achieved by using the modifyUser primitive. Note that a CCCP server can choose to implement both approaches simultaneously

to be invoked by different clients.

Editor's Note: The dedicated primitive is defined to demonstrate that both approaches are possible with CCCP.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyUserRoles>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
      <user-roles
       xmlns="urn:ietf:params:xml:ns:conference-info">
       <entry>presenter</entry>
      </user-roles>
     </modifyUserRoles>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyUserRoles/>
</response>
```

7.3.4. deleteUser

Remove the user from the conference roster.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteUser>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
     </deleteUser>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteUser/>
</response>
```

7.3.5. getUser

Retrieve the information about a conference participant.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <getUser>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
     </getUser>
```

```
      </request>


      <response
          requestId="1"
          from= "https://Mcu55.company.com:444/MCU"
          to= "https://company.com:444/Client"
          code="success"
          xmlns="urn:ietf:params:xml:ns:cccp"
          xmlns:ci="urn:ietf:params:xml:ns:conference-info"
          xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
          <getUser>
           <conferenceKeys confEntity="sip:conf233@example.com"/>
             <user entity="sip:bob@example.com"
              xmlns="urn:ietf:params:xml:ns:conference-info">
              <display-text>Bob Hoskins III</display-text>
              <associated-aors>
               <entry>
                <uri>tel:2562566</uri>
                 <display-text>the description</display-text>
                 <purpose>optional tbd values</purpose>
                </entry>
               </associated-aors>
               <roles>
                <entry>presenter</entry>
               </roles>
              </user>
          </getUser>
      </response>
```

7.4. Endpoint Primitives

7.4.1. addEndpoint

Bring the specified user into a conference by establishing a call (signaling and media) with him/her/it.

The endpoint 'entity' value MAY be replaced or augmented by the CCCP server. The 'media-id' value MAY be replaced or augmented by the CCCP server. If the server returns this information in the response, the client MUST use the values provided by the server.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addEndpoint>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
       <endpoint entity="sip:bob@pc4.example.com"
        xmlns="urn:ietf:params:xml:ns:conference-info">
       <display-text>Bob's Laptop</display-text>
       <joining-method>dialed-out</joining-method>
       <media id="1">
       <display-text>main audio</display-text>
       <type>audio</type>
       </media>
      </endpoint>
     </addEndpoint>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addEndpoint/>
</response>
```

7.4.2. modifyEndpoint

Modify the call description or/and its behaivior.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyEndpoint>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
       <endpoint entity="sip:bob@pc4.example.com"
        xmlns="urn:ietf:params:xml:ns:conference-info">
        <display-text>Bob's Laptop</display-text>
       </endpoint>
      </modifyEndpoint>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyEndpoint/>
</response>
```

7.4.3. deleteEndpoint

Disconnect the call.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteEndpoint>
                <endpointKeys
                 confEntity="sip:conf233@example.com"
                 userEntity="sip:bob@example.com"
                 endpointEntity="sip:bob@pc33.example.com"/>
    </deleteEndpoint>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteEndpoint/>
</response>
```

7.4.4. getEndpoint

Retrieve the information about the call.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <getEndpoint>
                <endpointKeys
                 confEntity="sip:conf233@example.com"
                 userEntity="sip:bob@example.com"
                 endpointEntity="sip:bob@pc33.example.com"/>
    </getEndpoint>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <getEndpoint>
     <userKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"/>
       <endpoint entity="sip:bob@pc4.example.com"
        xmlns="urn:ietf:params:xml:ns:conference-info">
       <display-text>Bob's Laptop</display-text>
       <joining-method>dialed-out</joining-method>
       <media id="1">
       <display-text>main audio</display-text>
       <type>audio</type>
       </media>
      </endpoint>
    </getEndpoint>
</response>
```

7.5. Endpoint Media Primitives

7.5.1. addEndpointMedia

Add the specified media stream to the call.

The 'media-id' value MAY be replaced or augmented by the CCCP server. The CCCP client MUST use the new value if returned by the server in the response.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addEndpointMedia>
     <endpointKeys
      confEntity="sip:conf233@example.com"
      userEntity="sip:bob@example.com"
      endpointEntity="sip:bob@pc33.example.com"/>
      <media id="1"
       xmlns="urn:ietf:params:xml:ns:conference-info">
       <display-text>main audio</display-text>
       <type>audio</type>
      </media>
    </addEndpointMedia>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <addEndpointMedia/>
</response>
```

7.5.2. modifyEndpointMedia

Modify the media behavior. This primitive can be used to mute and un-mute the call.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyEndpointMedia>
        <endpointKeys
         confEntity="sip:conf233@example.com"
         userEntity="sip:bob@example.com"
         endpointEntity="sip:bob@pc33.example.com"/>
        <media id="1"
         xmlns="urn:ietf:params:xml:ns:conference-info">
                        <type>audio</type>
                        <status>recvonly</status>
                </media>
    </modifyEndpointMedia>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <modifyEndpointMedia/>
 </response>
```

7.5.3. deleteEndpointMedia

Remove the specified media stream from the call.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteEndpointMedia>
                <mediaKeys
                 confEntity="sip:conf233@example.com"
                 userEntity="sip:bob@example.com"
                 endpointEntity="sip:bob@pc33.example.com"
                 mediaId="1"/>
    </deleteEndpointMedia>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <deleteEndpointMedia/>
</response>
```

7.5.4. getEndpointMedia

Retrieve the information about the specified stream in the call.

```
<request
    requestId="1"
    from="https://company.com:444/Client"
    to="https://Mcu55.company.com:444/MCU"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <getEndpointMedia>
                <mediaKeys
                 confEntity="sip:conf233@example.com"
                 userEntity="sip:bob@example.com"
                 endpointEntity="sip:bob@pc33.example.com"
                 mediaId="1"/>
    </getEndpointMedia>
</request>


<response
    requestId="1"
    from= "https://Mcu55.company.com:444/MCU"
    to= "https://company.com:444/Client"
    code="success"
    xmlns="urn:ietf:params:xml:ns:cccp"
    xmlns:ci="urn:ietf:params:xml:ns:conference-info"
    xmlns:cis="urn:ietf:params:xml:ns:conference-info-separator"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="urn:ietf:params:xml:ns:cccp cccp.xsd">
    <getEndpointMedia>
                <endpointKeys
                 confEntity="sip:conf233@example.com"
                 userEntity="sip:bob@example.com"
                 endpointEntity="sip:bob@pc33.example.com"/>
        <media id="1"
         xmlns="urn:ietf:params:xml:ns:conference-info">
                        <type>audio</type>
                        <status>recvonly</status>
                </media>
    </getEndpointMedia>
</response>
```

7.6. Sidebar Primitives

7.6.1. addSidebar

Create a sidebar in the conference.

XML TBD.

7.6.2. modifySidebar

Modify the sidebar parameters in the conference.

XML TBD.

7.6.3. deleteSidebar

Remove the sidebar from the conference.

XML TBD.

7.6.4. addUserToSidebar

Add the specified conference participant to the sidebar.

XML TBD.

7.6.5. deleteUserFromSidebar

Remove the specified conference participant from the sidebar.

XML TBD.

7.6.6. moveUserBetweenSidebars

Move the the specified conference participant from sidebar A to sidebar B.

XML TBD.

7.7. Stimulus Primitives

This operation is used to convey opaque to the CCCP logic requests from a CCCP client to a CCCP server to be processed by applications above CCCP.

XML TBD.

8. The XML Schema

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema
    targetNamespace="urn:ietf:params:xml:ns:cccp"
    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"

          elementFormDefault="qualified"
          attributeFormDefault="unqualified"
          >
          <!--
             This import brings in the standard
             Conference Package definitions
          -->
          <xs:import
           namespace="urn:ietf:params:xml:ns:conference-info"
           schemaLocation="ci.xsd"/>

          <!--
             This import brings in the standard Conference Package
             Standard Separator definitions
          -->
          <xs:import
           namespace="urn:ietf:params:xml:ns:conference-info-separator"
           schemaLocation="ci-separator.xsd"/>

          <!--
              REQUEST ELEMENT
          -->
          <xs:element name="request" type="tns:request-type"/>

          <!--
              RESPONSE ELEMENT
          -->
          <xs:element name="response" type="tns:response-type"/>

          <!--
              REQUEST TYPE
          -->
          <xs:complexType name="request-type">
              <xs:sequence maxOccurs="unbounded">
                  <xs:choice>
                      <xs:element name="cancel"
                       type="tns:cancel-type"/>
                      <xs:element name="ping"
                      type="tns:ping-type"/>
                      <xs:element name="addConference"
                       type="tns:add-conference-type"/>
                      <xs:element name="modifyConference"
                       type="tns:add-conference-type"/>
                      <xs:element name="getConference"
```

```
                 type="tns:get-conference-type"/>
                <xs:element name="deleteConference"
                 type="tns:delete-conference-type"/>
                <xs:element name="addUser"
                 type="tns:add-user-type"/>
                <xs:element name="modifyUser"
                 type="tns:add-user-type"/>
                <xs:element name="modifyUserRoles"
                 type="tns:modify-user-roles-type"/>
                <xs:element name="getUser"
                 type="tns:get-user-type"/>
                <xs:element name="deleteUser"
                 type="tns:delete-user-type"/>
                <xs:element name="addEndpoint"
                 type="tns:add-endpoint-type"/>
                 <xs:element name="modifyEndpoint"
                 type="tns:add-endpoint-type"/>
                <xs:element name="getEndpoint"
                 type="tns:get-endpoint-type"/>
                <xs:element name="deleteEndpoint"
                 type="tns:delete-endpoint-type"/>
                <xs:element name="addEndpointMedia"
                 type="tns:add-endpoint-media-type"/>
                <xs:element name="modifyEndpointMedia"
                 type="tns:add-endpoint-media-type"/>
                <xs:element name="getEndpointMedia"
                 type="tns:get-endpoint-media-type"/>
                <xs:element name="deleteEndpointMedia"
                 type="tns:delete-endpoint-media-type"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0"
                 maxOccurs="unbounded"/>
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="requestId"
         type="xs:string" use="required"/>
        <!--
            The  URI of the CCCP client sending the CCCP request
            -->
        <xs:attribute name="from" type="xs:anyURI" use="required"/>
        <!--
            The URI of the CCCP server the request is destined to
            -->
        <xs:attribute name="to" type="xs:anyURI" use="required"/>

        <!--
             The trusted Identifier of the originator of the request
            -->
```

```
            <xs:attribute name="originator" type="xs:anyURI"
             use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              RESPONSE TYPE
            -->
        <xs:complexType name="response-type">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:choice>
                    <xs:element name="cancel"
                     type="tns:cancel-response-type"/>
                    <xs:element name="ping"
                     type="tns:ping-response-type"/>
                    <xs:element name="addConference"
                     type="tns:add-conference-response-type"/>
                    <xs:element name="modifyConference"
                     type="tns:add-conference-response-type"/>
                    <xs:element name="getConference"
                     type="tns:get-conference-response-type"/>
                    <xs:element name="deleteConference"
                     type="tns:delete-conference-response-type"/>
                    <xs:element name="addUser"
                     type="tns:add-user-response-type"/>
                    <xs:element name="modifyUser"
                     type="tns:add-user-response-type"/>
                    <xs:element name="modifyUserRoles"
                     type="tns:modify-user-roles-response-type"/>
                    <xs:element name="getUser"
                     type="tns:get-user-response-type"/>
                    <xs:element name="deleteUser"
                     type="tns:delete-user-response-type"/>
                    <xs:element name="addEndpoint"
                     type="tns:add-endpoint-response-type"/>
                     <xs:element name="modifyEndpoint"
                     type="tns:add-endpoint-response-type"/>
                    <xs:element name="getEndpoint"
                     type="tns:get-endpoint-response-type"/>
                    <xs:element name="deleteEndpoint"
                     type="tns:delete-endpoint-response-type"/>
                    <xs:element name="addEndpointMedia"
                     type="tns:add-endpoint-media-response-type"/>
                    <xs:element name="modifyEndpointMedia"
                     type="tns:add-endpoint-media-response-type"/>
                    <xs:element name="getEndpointMedia"
                     type="tns:get-endpoint-media-response-type"/>
                    <xs:element name="deleteEndpointMedia"
```

```
                    type="tns:delete-endpoint-media-response-type"/>
                   <xs:any namespace="##other" processContents="lax"
                    minOccurs="0"
                     maxOccurs="unbounded"/>
                </xs:choice>
            </xs:sequence>
            <xs:attribute name="requestId"
             type="xs:string" use="required"/>
            <!--
                The  URI of the CCCP server sending the CCCP response
                -->
            <xs:attribute name="from"
             type="xs:anyURI" use="required"/>
            <!--
                The  URI of the CCCP client the response is destined to
                -->
            <xs:attribute name="to"
             type="xs:anyURI" use="required"/>
            <!--
                 The trusted Identifier of the responder
                -->
            <xs:attribute name="responder"
             type="xs:anyURI" use="optional"/>
            <xs:attribute name="code"
             type="tns:response-code-type" use="required"/>
            <xs:attribute name="reason"
             type="tns:reason-code-type" use="optional"/>
            <xs:attribute name="displayString"
             type="xs:string" use="optional"/>
            <xs:attribute name="timeOut"
             type="xs:nonNegativeInteger" use="optional"/>
            <xs:attribute name="retryAfter"
             type="xs:nonNegativeInteger" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>
        <!--
            RESPONSE CODE TYPE
            -->
        <xs:simpleType name="response-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="success"/>
                <xs:enumeration value="pending"/>
                <xs:enumeration value="failure"/>
            </xs:restriction>
        </xs:simpleType>

        <!--
            REASON CODE TYPE
```

```
            -->
        <xs:simpleType name="reason-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="serverBusy"/>
                <xs:enumeration value="timeout"/>
                <xs:enumeration value="unauthorized"/>
                <xs:enumeration value="requestMalformed"/>
                <xs:enumeration value="requestTooLarge"/>
                <xs:enumeration value="requestCancelled"/>
                <xs:enumeration value="notSupported"/>
                <xs:enumeration value="otherFailure"/>
            </xs:restriction>
        </xs:simpleType>


        <!--
              CONFERENCE KEYS TYPE
            -->
        <xs:complexType name="conference-keys-type">
            <xs:attribute name="confEntity" type="xs:anyURI"/>
            <xs:anyAttribute namespace="##other"
             processContents="lax"/>
        </xs:complexType>

        <!--
              USER KEYS TYPE
            -->
        <xs:complexType name="user-keys-type">
            <xs:attribute name="confEntity" type="xs:anyURI"/>
            <xs:attribute name="userEntity" type="xs:anyURI"/>
            <xs:anyAttribute namespace="##other"
             processContents="lax"/>
        </xs:complexType>

        <!--
              ENDPOINT KEYS TYPE
            -->
        <xs:complexType name="endpoint-keys-type">
            <xs:attribute name="confEntity" type="xs:anyURI"/>
            <xs:attribute name="userEntity" type="xs:anyURI"/>
            <xs:attribute name="endpointEntity" type="xs:anyURI"/>
            <xs:anyAttribute namespace="##other"
             processContents="lax"/>
        </xs:complexType>

        <!--
              MEDIA KEYS TYPE
            -->
```

```
<xs:complexType name="media-keys-type">
    <xs:attribute name="confEntity" type="xs:anyURI"/>
    <xs:attribute name="userEntity" type="xs:anyURI"/>
    <xs:attribute name="endpointEntity" type="xs:anyURI"/>
    <xs:attribute name="mediaId" type="xs:string"/>
    <xs:anyAttribute namespace="##other"
     processContents="lax"/>
</xs:complexType>

<!--
      CANCEL TYPE
    -->
<xs:complexType name="cancel-type">
    <xs:sequence>
        <xs:element name="requestId" type="xs:string"/>
        <xs:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other"
     processContents="lax"/>
</xs:complexType>

<!--
      CANCEL RESPONSE TYPE
    -->
<xs:complexType name="cancel-response-type">
    <xs:sequence>
        <xs:element name="requestId"
         type="xs:string" minOccurs="0"/>
        <xs:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="reason"
     type="tns:cancel-reason-code-type" use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
    CANCEL REASON CODE TYPE
    -->
<xs:simpleType name="cancel-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="requestDoesntExist"/>
        <xs:enumeration value="otherFailure"/>
    </xs:restriction>
</xs:simpleType>

<!--
```

```
              PING TYPE
            -->
        <xs:complexType name="ping-type">
            <xs:sequence>
                <xs:any namespace="##other" processContents="lax"
                minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              PING RESPONSE TYPE

            -->
        <xs:complexType name="ping-response-type">
            <xs:sequence>
                <xs:element name="serverStatus"
                 type="tns:server-status-type" minOccurs="0"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

         <!--
              ADD CONFERENCE TYPE
            -->
        <xs:complexType name="add-conference-type">
            <xs:sequence>
                <xs:element ref="ci:conference-info"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              ADD CONFERENCE RESPONSE TYPE
            -->
        <xs:complexType name="add-conference-response-type">
            <xs:sequence>
                <xs:element ref="ci:conference-info" 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:add-conference-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
            ADD CONFERENCE REASON CODE TYPE
            -->
        <xs:simpleType name="add-conference-reason-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="conferenceExistsAlready"/>
                <xs:enumeration value="otherFailure"/>
            </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:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              GET CONFERENCE RESPONSE TYPE
            -->
        <xs:complexType name="get-conference-response-type">
            <xs:sequence>
                <xs:element ref="ci:conference-info" 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:get-conference-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              GET CONFERENCE REASON CODE TYPE
            -->
```

```
<xs:simpleType name="get-conference-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="conferenceDoesntExist"/>
        <xs:enumeration value="otherFailure"/>
    </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:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      DELETE CONFERENCE RESPONSE TYPE
    -->
<xs:complexType name="delete-conference-response-type">
    <xs:sequence>
        <xs:element ref="ci:conference-info" 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:delete-conference-reason-code-type"
      use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
    DELETE CONFERENCE REASON CODE TYPE
    -->
<xs:simpleType name="delete-conference-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="conferenceDoesntExist"/>
        <xs:enumeration value="otherFailure"/>
    </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 ref="ci:user"/>
        <xs:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      ADD USER RESPONSE TYPE
    -->
<xs:complexType name="add-user-response-type">
    <xs:sequence>
        <xs:element name="conferenceKeys"
         type="tns:conference-keys-type" minOccurs="0"/>
        <xs:element ref="ci:user" minOccurs="0"/>
        <xs:any namespace="##other" processContents="lax"
         maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="reason"
     type="tns:add-user-reason-code-type" use="optional"/>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
    ADD USER REASON CODE TYPE
    -->
<xs:simpleType name="add-user-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="conferenceDoesntExist"/>
        <xs:enumeration value="userExistsAlready"/>
        <xs:enumeration value="otherFailure"/>
    </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 ref="ci:user-roles"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              MODIFY USER ROLES RESPONSE TYPE
            -->
        <xs:complexType name="modify-user-roles-response-type">
            <xs:sequence>
                <xs:element name="conferenceKeys"
                 type="tns:conference-keys-type" minOccurs="0"/>
                <xs:element ref="ci:user" 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:get-user-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              GET USER TYPE
            -->
        <xs:complexType name="get-user-type">
            <xs:sequence>
                <xs:element name="userKeys"
                 type="tns:user-keys-type"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              GET USER RESPONSE TYPE
            -->
        <xs:complexType name="get-user-response-type">
            <xs:sequence>
                <xs:element name="conferenceKeys"
                 type="tns:conference-keys-type" minOccurs="0"/>
                <xs:element ref="ci:user" 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:get-user-reason-code-type" use="optional"/>
        <xs:anyAttribute namespace="##other"
         processContents="lax"/>
    </xs:complexType>

    <!--
        GET USER REASON CODE TYPE
        -->
    <xs:simpleType name="get-user-reason-code-type">
        <xs:restriction base="xs:string">
            <xs:enumeration value="conferenceDoesntExist"/>
            <xs:enumeration value="userDoesntExist"/>
            <xs:enumeration value="otherFailure"/>
        </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:any namespace="##other" processContents="lax"
             maxOccurs="unbounded"/>
         </xs:sequence>
         <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:complexType>

    <!--
          DELETE USER RESPONSE TYPE
        -->
    <xs:complexType name="delete-user-response-type">
        <xs:sequence>
            <xs:element name="conferenceKeys"
             type="tns:conference-keys-type" minOccurs="0"/>
            <xs:element ref="ci:user" 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:delete-user-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              DELETE USER REASON CODE TYPE
            -->
        <xs:simpleType name="delete-user-reason-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="conferenceDoesntExist"/>
                <xs:enumeration value="userDoesntExist"/>
                <xs:enumeration value="otherFailure"/>
            </xs:restriction>
        </xs:simpleType>

        <!--
              ADD ENDPOINT TYPE
            -->
        <xs:complexType name="add-endpoint-type">
            <xs:sequence>
                <xs:element name="userKeys"
                 type="tns:user-keys-type"/>
                <xs:element ref="ci:endpoint"/>
                <xs:sequence minOccurs="0">
                    <xs:element ref="cis:separator"/>
                    <xs:any namespace="##other" processContents="lax"
                     maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              ADD ENDPOINT RESPONSE TYPE
            -->
        <xs:complexType name="add-endpoint-response-type">
            <xs:sequence>
                <xs:element name="userKeys"
                 type="tns:user-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:add-user-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>
        <!--
              GET ENDPOINT TYPE
            -->
        <xs:complexType name="get-endpoint-type">
            <xs:sequence>
                <xs:element name="endpointKeys"
                 type="tns:endpoint-keys-type"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
             GET ENDPOINT REASON CODE TYPE
            -->
        <xs:simpleType name="get-endpoint-reason-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="conferenceDoesntExist"/>
                <xs:enumeration value="userDoesntExist"/>
                <xs:enumeration value="endpointDoesntExist"/>
                <xs:enumeration value="otherFailure"/>
            </xs:restriction>
        </xs:simpleType>

        <!--
              GET ENDPOINT RESPONSE TYPE
            -->
        <xs:complexType name="get-endpoint-response-type">
            <xs:sequence>
                <xs:element name="userKeys"
                 type="tns:user-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:get-endpoint-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>
```

```
<!--
      DELETE ENDPOINT TYPE
    -->
<xs:complexType name="delete-endpoint-type">
    <xs:sequence>
        <xs:element name="endpointKeys"
         type="tns:endpoint-keys-type"/>
        <xs:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      DELETE ENDPOINT REASON CODE TYPE
    -->
<xs:simpleType name="delete-endpoint-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="conferenceDoesntExist"/>
        <xs:enumeration value="userDoesntExist"/>
        <xs:enumeration value="endpointDoesntExist"/>
        <xs:enumeration value="otherFailure"/>
    </xs:restriction>
</xs:simpleType>

<!--
      DELETE ENDPOINT RESPONSE TYPE
    -->
<xs:complexType name="delete-endpoint-response-type">
    <xs:sequence>
        <xs:element name="userKeys"
         type="tns:user-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:delete-endpoint-reason-code-type"
      use="optional"/>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      ADD ENDPOINT MEDIA TYPE
    -->
```

```
<xs:complexType name="add-endpoint-media-type">
    <xs:sequence>
        <xs:element name="endpointKeys"
         type="tns:endpoint-keys-type"/>
        <xs:element ref="ci:media"/>
        <xs:any namespace="##other" processContents="lax"
         minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      ADD ENDPOINT MEDIA RESPONSE TYPE
    -->
<xs:complexType name="add-endpoint-media-response-type">
    <xs:sequence>
        <xs:element name="endpointKeys"
         type="tns:endpoint-keys-type" minOccurs="0"/>
        <xs:element ref="ci:media" 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:add-endpoint-media-reason-code-type"
      use="optional"/>
     <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<!--
      ADD ENDPOINT MEDIA REASON CODE TYPE
    -->
<xs:simpleType name="add-endpoint-media-reason-code-type">
    <xs:restriction base="xs:string">
        <xs:enumeration value="conferenceDoesntExist"/>
        <xs:enumeration value="userDoesntExist"/>
        <xs:enumeration value="endpointDoesntExist"/>
        <xs:enumeration value="mediaExistsAlready"/>
        <xs:enumeration value="otherFailure"/>
    </xs:restriction>
</xs:simpleType>

<!--
      GET ENDPOINT MEDIA TYPE
    -->
<xs:complexType name="get-endpoint-media-type">
```

```
            <xs:sequence>
                <xs:element name="mediaKeys"
                 type="tns:media-keys-type"/>
                <xs:any namespace="##other" processContents="lax"

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

        <!--
              GET ENDPOINT MEDIA REASON CODE TYPE
            -->
        <xs:simpleType name="get-media-reason-code-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="conferenceDoesntExist"/>
                <xs:enumeration value="userDoesntExist"/>
                <xs:enumeration value="endpointDoesntExist"/>
                <xs:enumeration value="mediaDoesntExist"/>
                <xs:enumeration value="otherFailure"/>
            </xs:restriction>
        </xs:simpleType>

        <!--
              GET ENDPOINT MEDIA RESPONSE TYPE
            -->
        <xs:complexType name="get-endpoint-media-response-type">
            <xs:sequence>
                <xs:element name="endpointKeys"
                 type="tns:endpoint-keys-type" minOccurs="0"/>
                <xs:element ref="ci:media" 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:get-media-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </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:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
              DELETE ENDPOINT MEDIA RESPONSE TYPE
            -->
        <xs:complexType name="delete-endpoint-media-response-type">
            <xs:sequence>
                <xs:element name="endpointKeys"
                 type="tns:endpoint-keys-type" minOccurs="0"/>
                <xs:element ref="ci:media" 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:get-media-reason-code-type" use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
          SERVER LIST TYPE
       -->
        <xs:complexType name="server-list-type">
            <xs:sequence>
                <xs:element name="entry"
                 type="tns:server-description-type"
                 minOccurs="0" maxOccurs="unbounded"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute ref="ci:state" use="optional" default="full"/>
            <xs:attribute name="version" type="xs:unsignedInt"
             use="optional"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
          SERVER DESCRIPTION TYPE
        -->
        <xs:complexType name="server-description-type">
            <xs:sequence>
```

```
                <xs:element name="status" type="tns:server-status-type"
                 minOccurs="0"/>
                <xs:element name="modality" type="xs:string"/>
                <xs:element name="vendor" type="xs:string"/>
                <xs:any namespace="##other" processContents="lax"
                 minOccurs="0" maxOccurs="unbounded"/>
            </xs:sequence>
            <xs:attribute name="entity" type="xs:anyURI" use="required"/>
            <xs:anyAttribute namespace="##other" processContents="lax"/>
        </xs:complexType>

        <!--
          SERVER STATUS TYPE
        -->
        <xs:simpleType name="server-status-type">
            <xs:restriction base="xs:string">
                <xs:enumeration value="full"/>
                <xs:enumeration value="loaded"/>
                <xs:enumeration value="normal"/>
                <xs:enumeration value="unavailable"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:schema>
```

Figure 39

9. IANA Considerations

This document registers a new XML namespace and a new XML schema.

9.1. URN Sub-Namespace Registration for urn:ietf:params:xml:ns:cccp

This section registers a new XML namespace, as per the guidelines in RFC 3688 [5].

```
URI: The URI for this namespace is urn:ietf:params:xml:ns:cccp
Registrant Contact: IETF XCON Working Group <xcon@ietf.org>, as
   designated by the IESG <iesg@ietf.org>
XML:

BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
          "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type"
     content="text/html;charset=iso-8859-1"/>
  <title>Centralized Conference Information Namespace</title>

</head
<body>
  <h1>Namespace for Centralized Conference Control Protocol</h1>
  <h2>urn:ietf:params:xml:ns:cccp</h2>
  <p>See <a href="[[[URL of published RFC]]]">RFCXXXX</a>.</p>
</body>
</html>
END
```

9.2. XML Schema Registration

This specification registers a schema, as per the guidelines in RFC 3688 [5].

```
URI: please assign
Registrant Contact: IETF XCON Working Group <xcon@ietf.org>, as
designated by the IESG <iesg@ietf.org>
XML: The XML can be found as the sole content of Section 7
```

10. Security Considerations

Manipulation of conference state and policy information through the conference control protocol require a strong means for authentication, conference information protection, and applying comprehensive authorization rules by a focus. Users of this specification MUST use encrypted transport means and comply with the security considerations discussed in the XCON Framework [7] and the SIP Event Package for Conference State [2] .

11. Acknowledgements

The author would like to thank Gur Kimchi for his earlier work that served as the starting point for this specification.

12. References

12.1. Normative References

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

[2] Rosenberg, J., "A Session Initiation Protocol (SIP) Event Package for Conference State", draft-ietf-sipping-conference-package-12 (work in progress), July 2005.

[3] Levin, O., "Extensions to the Session Initiation Protocol (SIP) Event Package for Conference State", draft-levin-xcon-conference-package-ext-00 (work in progress), October 2005.

12.2. Informative References

[4] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[5] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[6] Rosenberg, J., "A Framework for Conferencing with the Session Initiation Protocol", draft-ietf-sipping-conferencing-framework-05 (work in progress), May 2005.

[7] Barnes, M., "A Framework and Data Model for Centralized Conferencing", draft-ietf-xcon-framework-01 (work in progress), July 2005.

Authors' Addresses

Orit Levin
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052, USA

Email: oritl@microsoft.com

Roni Even
Polycom
94 Derech Em Hamoshavot
Petach Tikva, 49130, Israel

Email: roni.even@polycom.co.il

Pierre Hagendorf
RADVISION
24, Raul Wallenberg St.
Tel-Aviv, 69719, Israel

Email: pierre@radvision.com

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.