XCON Working Group	M. Barnes	
Internet-Draft	Nortel	
Intended status: Informational	C. Boulton	
Expires: July 2, 2010	NS-Technologies	
	L. Miniero	
	Meetecho	
	R. Presta	
	S P. Romano	
	University of Napoli	
	December 29, 2009	

TOC

Centralized Conferencing Manipulation Protocol (CCMP) Call Flow Examples
draft-ietf-xcon-examples-02

Abstract

This document provides detailed call flows for the scenarios documented in the Centralized Conferencing (XCON) Framework and the XCON Scenarios. The call flows document the use of the interface between a conference control client and a conference control server using the Centralized Conferencing Manipulation Protocol (CCMP). The objective is to provide a base reference for both protocol researchers and developers.

Status of this Memo

Table of Contents

---

1. Introduction

TOC

This document provides detailed call flows for the scenarios documented in the Framework for Centralized Conferencing (XCON Framework) [RFC5239] (Barnes, M., Boulton, C., and O. Levin, “A Framework for Centralized Conferencing,” June 2008.) and the XCON Scenarios [RFC4597] (Even, R. and N. Ismail, “Conferencing Scenarios,” August 2006.). The XCON scenarios describe a broad range of use cases taking advantage of the advanced conferencing capabilities provided by a system realization of the XCON framework. The call flows document the use of the interface between a conference control client and a conference control server using the Centralized Conferencing Manipulation Protocol (CCMP) [I-D.ietf-xcon-ccmp] (Barnes, M., Boulton, C., Romano, S., and H. Schulzrinne, “Centralized Conferencing Manipulation Protocol,” April 2010.).
Due to the broad range of functionality provided by the XCON Framework and the flexibility of the CCMP messaging, these call flows should not be considered inclusive of all the functionality that can provided by the XCON Framework and protocol implementations. These flows represent a sample to provide an overview of the feature rich capabilities of the XCON framework and CCMP messaging for protocol developers, software developers and researchers.

---

2. Conventions

TOC

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 [RFC2119] (Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” March 1997.) and indicate requirement levels for compliant implementations. In this document, these key words are used when describing normative functionality based on the XCON Framework and CCMP.
Note that due to RFC formatting conventions, this document often splits message details whose content would exceed 72 characters. A backslash character marks where this line folding has taken place. This backslash and its trailing CRLF and whitespace would not appear in the actual protocol contents.

3. Terminology

TOC

This document uses the same terminology as found in the referenced documents, with the following terms and abbreviations used in the call flows. Also, note that the term "call flows" is used in a very generic sense in this document since the media is not limited to voice. The calls supported by the XCON framework and CCMP can consist of media such as text, voice and video, including multiple media types in a single active conference.

Conferencing and Media Client Client (CMCC): As defined in the XCON Framework. In the flows in this document, the CMCC is logically equivalent to the use of UAC as the client notation in the media control call flows [I-D.ietf-mediactrl-call-flows] (Amirante, A., Castaldi, T., Miniero, L., and S. Romano, "Media Control Channel Framework (CFW) Call Flow Examples," February 2010.).

Conferencing Server (ConfS): As defined in the XCON Framework. In this document, the conferencing server is used interchangeably with the term Application Server (AS) as used in the Media Control Architectural Framework [RFC5567] (Melanchuk, T., "An Architectural Framework for Media Server Control," June 2009.). However, these need not be the same entities in an implementation.

Media Server (MS): As defined in the Media Control Architectural Framework [RFC5567] (Melanchuk, T., "An Architectural Framework for Media Server Control," June 2009.).

4. Overview

TOC

This document provides a sampling of detailed call flows that can be implemented based on a system realization of [RFC5239] (Barnes, M., Boulton, C., and O. Levin, "A Framework for Centralized Conferencing," June 2008.) and implementation of [I-D.ietf-xcon-ccmp] (Barnes, M., Boulton, C., Romano, S., and H. Schulzrinne, "Centralized Conferencing Manipulation Protocol," April 2010.). This is intended to be a simple guide on the use of the conference control protocol between the Conference Server and the Conference Control Client. The objective is

to provide an informational base reference for protocol developers, software developers and researchers.
This document focuses on the interaction between the Conference (and Media) Control Client and the Conferencing system, specifically the Conference Server. The scenarios are based on those described in the XCON framework, many of which are based on the advanced conferencing capabilities described in the XCON scenarios. Additional scenarios have been added to provide examples of other real life scenarios that are anticipated to be supported by the framework. With the exception of an initial example with media control messaging, the examples do not include the details for the media control [I-D.ietf-mediactrl-mixer-control-package] (McGlashan, S., Melanchuk, T., and C. Boulton, "A Mixer Control Package for the Media Control Channel Framework," February 2010.), call signaling or binary floor control [RFC4582] (Camarillo, G., Ott, J., and K. Drage, "The Binary Floor Control Protocol (BFCP)," November 2006.) protocols. This document references the scenarios in the Media Control call flows [I-D.ietf-mediactrl-call-flows] (Amirante, A., Castaldi, T., Miniero, L., and S. Romano, "Media Control Channel Framework (CFW) Call Flow Examples," February 2010.), SIP Call Control Conferencing [RFC4579] (Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents," August 2006.) and binary floor control protocol documents.

---

5. Conference Creation

TOC

This section provides the details associated with the various ways in which a conference can be created using CCMP and the XCON framework constructs. As previously mentioned the details of the media control, call signaling and floor control protocols, where applicable, are annotated in the flows without showing all the details. However, for clarification purposes, the first example Section 5.1 (Basic Conference Creation) provides the details of the media control messaging along with an example of the standard annotation used throughout the remainder of this document. In subsequent flows, only this annotation (identified by lower case letters) is included and the reader is encouraged to refer to the call flows in the relevant documents for details for the other protocols. The annotations for the call signaling are on the left side of the conferencing server vertical bar and those for the media control messaging are on the right side.

TOC

5.1. Basic Conference Creation

The simplest manner in which a conference can be created is accomplished by the client sending a "confRequest" message with the "create" operation as the only parameter to the conference server, together with the "confUserID" associated with the requesting client itself. This results in the creation of a default conference, with an XCON-URI in the form of the "confObjID" parameter, the XCON-UserID in the form of the "confUserID" parameter (the same already present in the request) and the data for the conference object in the "confInfo" parameter all returned in the "confResponse" message. According to the implementation of the framework, this example may also add the user that invoked the conference upon creation to the conference object (e.g., "method" attribute in the "target" element of "allowed-users-list" may be set to "dial out" for this client based on the particular conferencing systems default). This is exactly the case depicted in the figure, which is presented to enrich the scenario. Note that, depending upon the conferencing system, this default conference could be specific to the client requesting the conference and thus may be different for the initiator than other participants (e.g., IVR interactions in this case which are not shown).

The specific data for the conference object is returned in the "confResponse" message in the "confInfo" parameter. This allows the client (with the appropriate authorization) to manipulate this data and add additional participants to the conference, as well as change the data during the conference. In addition, the client may distribute the conferencing information to other participants allowing them to join, the details of which are provided in additional flows. Please notice that, according to CCMP specification, the restitution of the new conference data in the "confInfo" parameter is not mandatory: if the "confInfo" parameter of the successful confResponse/create is void, a following confRequest/retrieve of the returned "confObjID" can be triggered to provide the requesting client with the detailed conference description.

Clients that are not XCON-aware may join the conference using a specific signaling interface such as SIP [RFC3261] (Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, “SIP: Session Initiation Protocol,” June 2002.), using the signaling interface to the conference focus as described in [RFC4579] (Johnston, A. and O. Levin, “Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents,” August 2006.). However, these details are not shown in the message flows. The message flows in this document identity the point in the message flows at which this signaling occurs via the lower case letter items (i.e., (a)...(x)) along with the appropriate text for the processing done by the conferencing server.

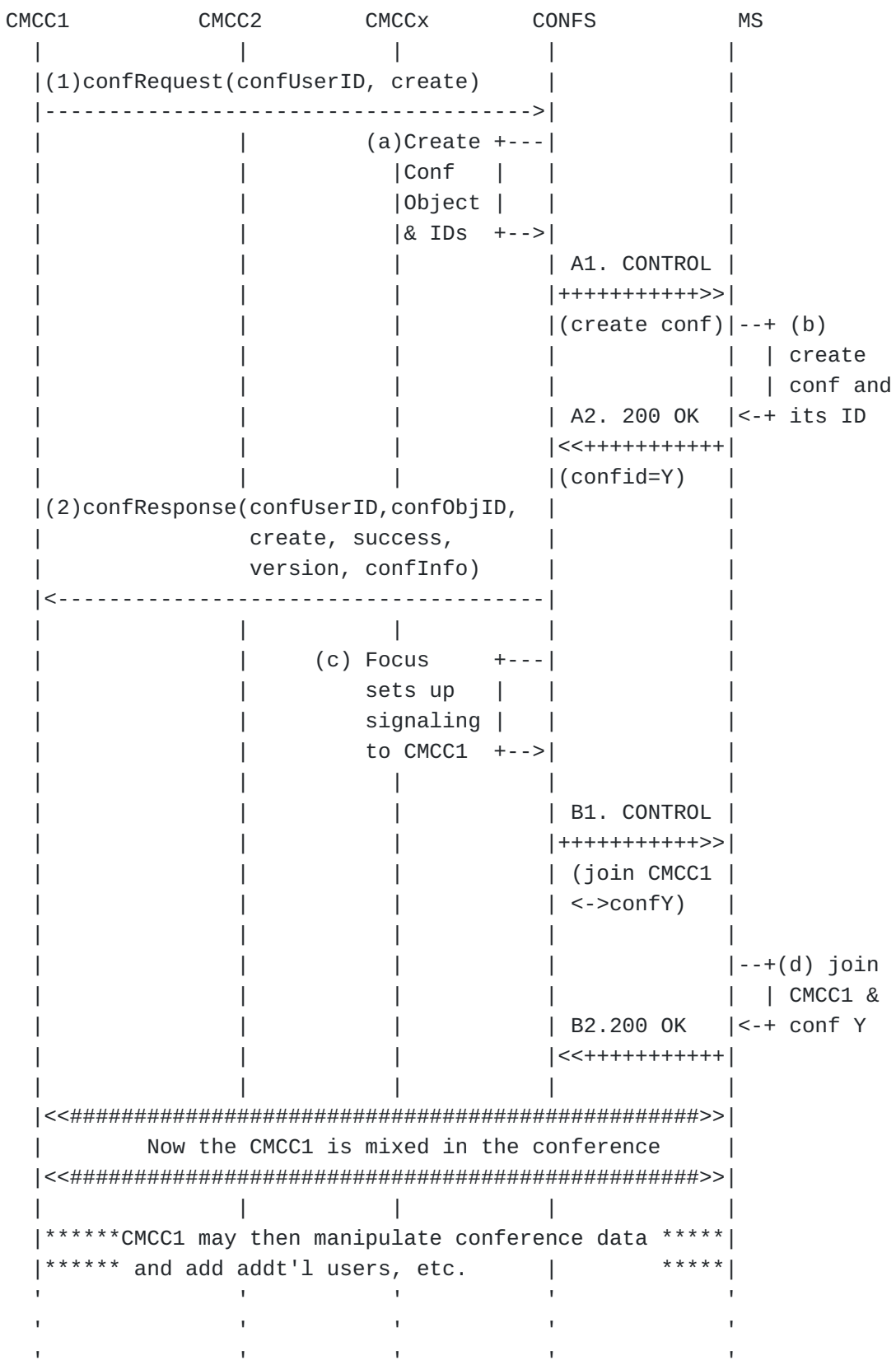

CMCC1
CMCC2
CMCCx
CONFS
MS
(1)confRequest(confUserID, create)
(a)Create Conf Object & IDs
A1. CONTROL (create conf)
(b) create conf and its ID
A2. 200 OK (confid=Y)
(2)confResponse(confUserID,confObjID, create, success, version, confInfo)
(c) Focus sets up signaling to CMCC1
B1. CONTROL (join CMCC1 <->confY)
(d) join CMCC1 & conf Y
B2.200 OK
Now the CMCC1 is mixed in the conference
******CMCC1 may then manipulate conference data *****
****** and add addt'l users, etc.

Figure 1: Create Basic Conference - Complete flow

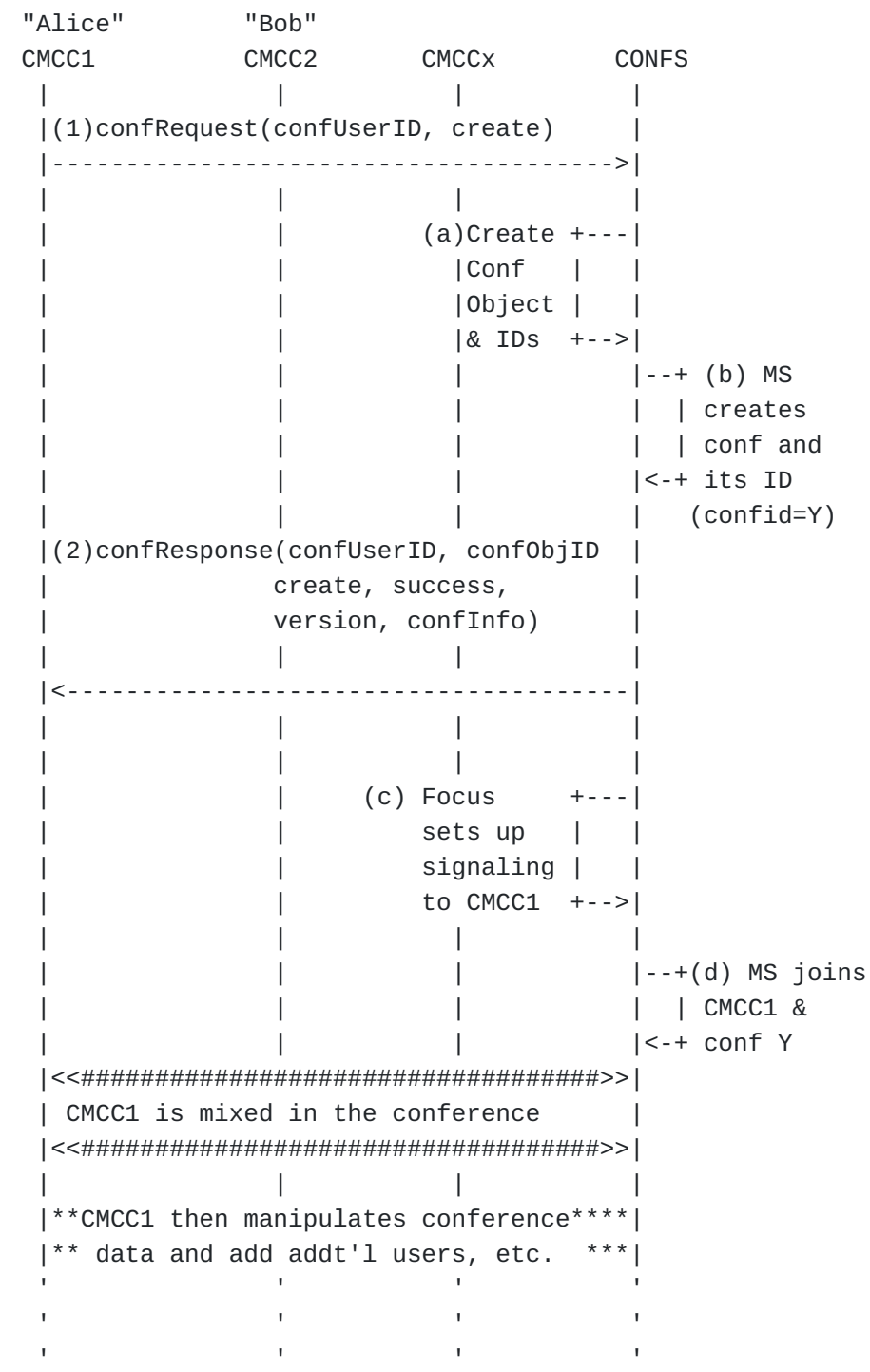


Figure 2: Create Basic Conference - Annotated Flow

1. confRequest/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
      xmlns:info="urn:ietf:params:xml:ns:conference-info"
      xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
      xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
   <ccmpRequest
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ccmp:ccmp-conf-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <operation>create</operation>
   </ccmpRequest>
</ccmp:ccmpRequest>
```

2. confResponse/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
     xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
     xmlns:info="urn:ietf:params:xml:ns:conference-info"
     xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpResponse
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:type="ccmp:ccmp-conf-response-message-type">
     <confUserID>xcon-userid:Alice@example.com</confUserID>
     <confObjID>xcon:8977794@example.com</confObjID>
     <operation>create</operation>
     <response-code>success</response-code>
     <version>1</version>
     <ccmp:confResponse>
         <confInfo entity="xcon:8977794@example.com">
            <info:conference-description>
                <info:display-text>
                   Default conference initiated by Alice
                </info:display-text>
                <info:conf-uris>
                   <info:entry>
                      <info:uri>
                          xcon:8977794@example.com
                      </info:uri>
                      <info:display-text>
                          Conference XCON-URI
                      </info:display-text>
                   </info:entry>
                </info:conf-uris>
```

```
                    <info:maximum-user-count>10</info:maximum-user-count>
                    <info:available-media>
                           <info:entry label="11">
                               <info:type>audio</info:type>
                           </info:entry>
                    </info:available-media>
                     <info:conference-state>
                       <active>false</active>
                     </info:conference-state>
                </info:conference-description>
                <info:users>
                   <xcon:join-handling>allow</xcon:join-handling>
                   <xcon:allowed-users-list>
                      <xcon:target uri="xcon-userid:Alice@example.com"
                                   method="dial-out"/>
                    </xcon:allowed-users-list>
                </info:users>
            </confInfo>
        </ccmp:confResponse>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 3: Create Basic Conference (Annotated) Detailed Messaging

---

5.2. Basic Conference Creation for a specific instance of Conference Information

TOC

A conference can also be created by the client sending a "confRequest" message with the "create" operation, along with the desired data in the form of the "confInfo" parameter for the conference to be created. The request also includes the "confUserID" of the requesting entity. If the conferencing system can support that specific type of conference (capabilities, etc.), then the request results in the creation of a conference. In this success case, an XCON-URI in the form of the "confObjID" parameter and the XCON-UserID in the form of the "confUserID" parameter (again, the same as the requesting entity) are returned in the "confResponse" message. In this example, we choose not to return the created conference object in the successful "confResponse" in the "confInfo" parameter.
This example also activates the conference upon creation (i.e., "method" attribute is set to "dial out" for this client based on the particular conferencing systems default). Just as before, this is not

to be considered mandatory, since it depends on the implementation choices of the framework. Note that, depending upon the conferencing system, this default conference could be specific to the client requesting the conference and thus may be different for the initiator than other participants (e.g., IVR interactions in this case which are not shown).

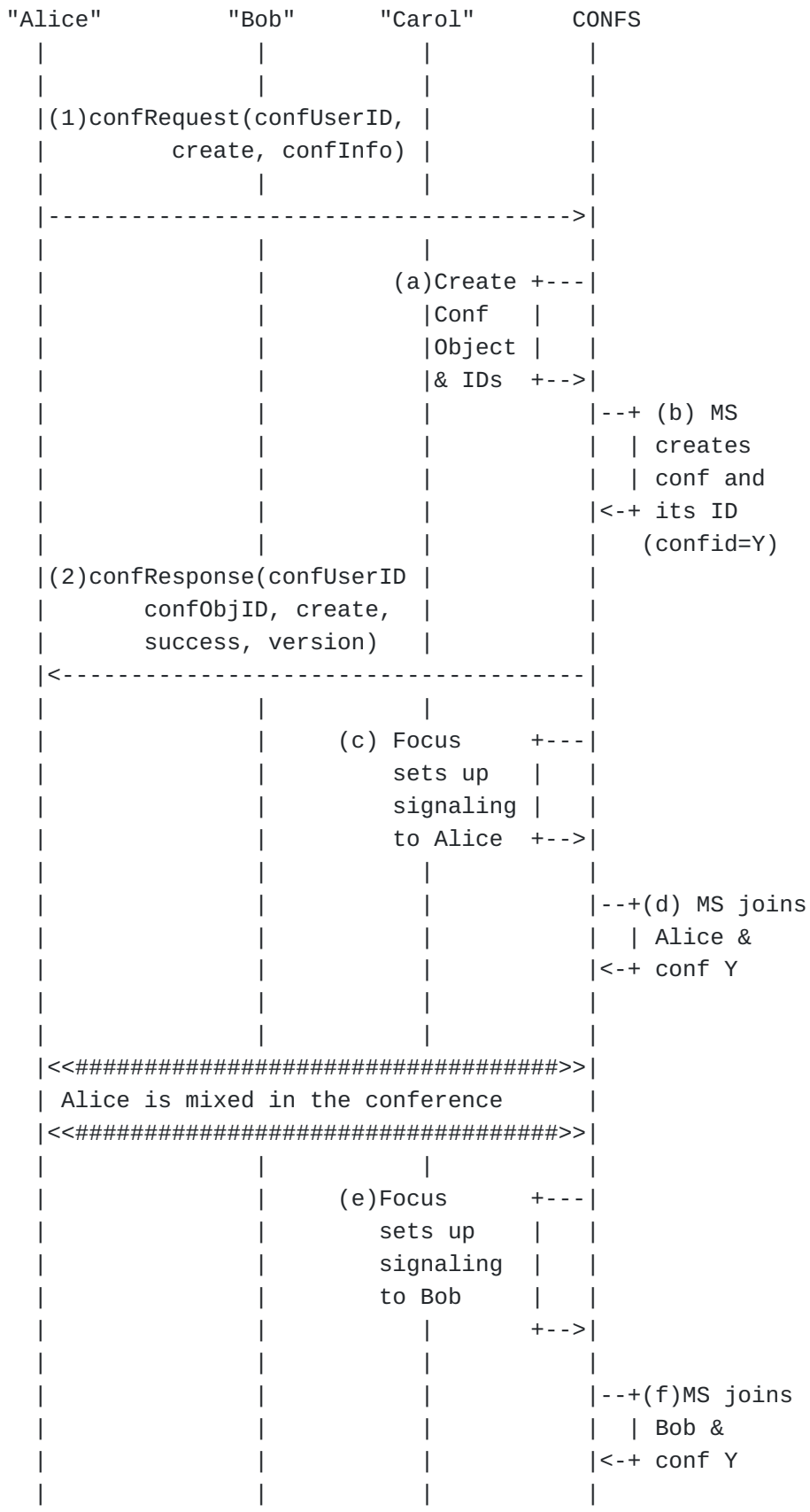

"Alice"
"Bob"
"Carol"
CONFS
(1)confRequest(confUserID, create, confInfo)
(a)Create Conf Object & IDs
(b) MS creates conf and its ID (confid=Y)
(2)confResponse(confUserID confObjID, create, success, version)
(c) Focus sets up signaling to Alice
(d) MS joins Alice & conf Y
Alice is mixed in the conference
(e)Focus sets up signaling to Bob
(f)MS joins Bob & conf Y

Figure 4: Create Basic Conference from user provided conference-info

1. confRequest/create message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
      xmlns:info="urn:ietf:params:xml:ns:conference-info"
      xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
      xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
   <ccmpRequest
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ccmp:ccmp-conf-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <operation>create</operation>
      <ccmp:confRequest>
         <confInfo>
            <info:conference-description>
                <info:display-text>
                   Dial-out conference initiated by Alice
                </info:display-text>
                 <info:maximum-user-count>10</info:maximum-user-count>
                 <info:available-media>
                        <info:entry>
                            <info:type>audio</info:type>
                        </info:entry>
                 </info:available-media>
             </info:conference-description>
             <info:users>
                <xcon:join-handling>allow</xcon:join-handling>
                <xcon: allowed-users-list>
                   <xcon:target uri="sip:alice@example.com"
                                method="dial-out"/>
                   <xcon:target uri="sip:bob83@example.com"
                                method="dial-out"/>
                   <xcon:target uri="sip:carol@example.com"
                                method="dial-out"/>
                 </xcon:allowed-users-list>
             </info:users>
         </confInfo>
      </ccmp:confRequest>
   </ccmpRequest>
</ccmp:ccmpRequest>
```

2. confResponse message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
     xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
```

```
        xmlns:info="urn:ietf:params:xml:ns:conference-info"
        xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
      <ccmpResponse
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ccmp:ccmp-conf-response-message-type">
       <confUserID>xcon-userid:Alice@example.com</confUserID>
       <confObjID>xcon:6845432@example.com</confObjID>
       <operation>create</operation>
       <response-code>success<response-code>
       <version>1</version>
      </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 5: Create Basic Conference Detailed Messaging

5.3. Basic Conference Creation - Cloning an existing Conference

TOC

A client can also create another conference by cloning an existing conference, such as an active conference or conference reservation. In this example, the client sends a "confRequest" message with the "create" operation, along with the "confUserID" and a specific "confObjID", from which a new conference is to be created by cloning an existing conference.

An example of how a client can create a conference based on a blueprint is provided in Section 5.4 (Conference Creation using Blueprints). The manner by which a client in this example might learn about a conference reservation or active conferences is similar to the first step in the blueprint example, with the exception of specifying querying for different types of conference objects supported by the specific conferencing system. For example, in this example, the client clones a conference reservation (i.e., an inactive conference).

If the conferencing system can support a new instance of the specific type of conference(capabilities, etc.), then the request results in the creation of a conference, with an XCON-URI in the form of a new value in the "confObjID" parameter to reflect the newly cloned conference object returned in the "confResponse" message.

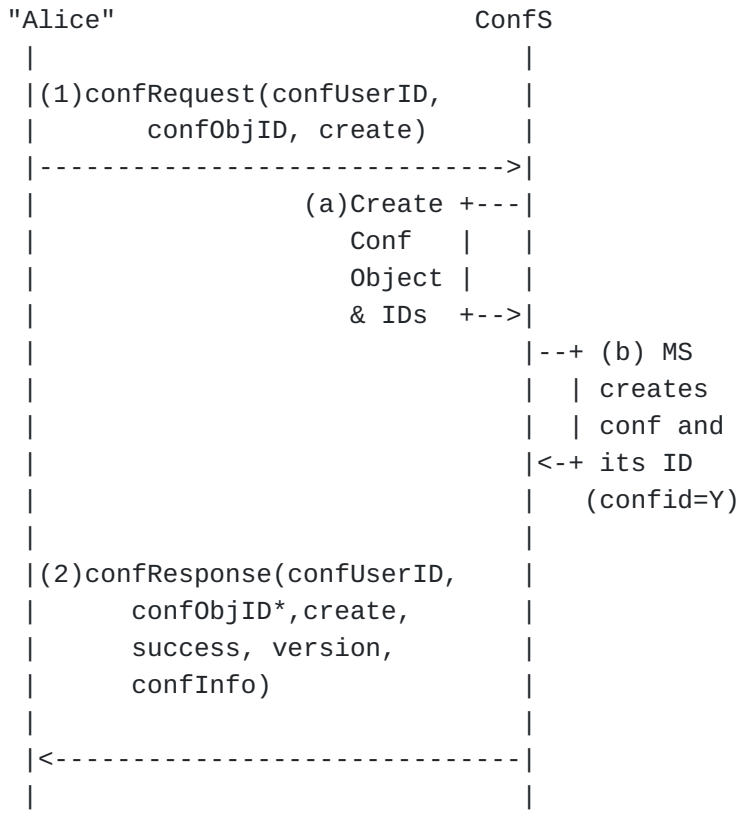


Figure 6: Create Basic Conference - Clone

---

1. "Alice" sends a confRequest message to clone a conference based on an existing conference reservation. "Alice" indicates this conference should be cloned from the specified parent conference represented by the "confObjID" in the request.

2. Upon receipt of the confRequest message containing a "create" operation and "confObjID", the conferencing system ensures that the "confObjID" received is valid. The conferencing system determines the appropriate read/write access of any users to be added to a conference based on this "confObjID" (using membership, roles, etc.). The conferencing system uses the received "confObjID" to clone a conference reservation. The conferencing system also reserves or allocates a new "confObjID" (called "confObjID*" in Figure 6 (Create Basic Conference - Clone)) to be used for the cloned conference object. This new identifier is of course different from the one associated with the conference to be cloned, since it represents a different conference object. Any subsequent

protocol requests from any of the members of the conference must address this new identifier. The conferencing system maintains the mapping between this conference ID and the parent conference object ID associated with the reservation through the conference instance, and this mapping is explicitly addressed through the "cloning-parent" element of the "conference-description" in the new conference object.

1. confRequest/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
      xmlns:info="urn:ietf:params:xml:ns:conference-info"
      xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
      xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
   <ccmpRequest
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ccmp:ccmp-conf-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:6845432@example.com</confObjID>
      <operation>create</operation>
   </ccmpRequest>
</ccmp:ccmpRequest>
```

2. confResponse/create message

```
 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
     xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
     xmlns:info="urn:ietf:params:xml:ns:conference-info"
     xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
  <ccmpResponse
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:type="ccmp:ccmp-conf-response-message-type">
     <confUserID>xcon-userid:Alice@example.com</confUserID>
     <confObjID>xcon:8977794@example.com</confObjID>
     <operation>create</operation>
     <response-code>success</response-code>
     <version>1</version>
     <ccmp:confResponse>
          <confInfo entity="xcon:8977794@example.com">
            <info:conference-description>
                <info:display-text>
                   New conference by Alice cloned from 6845432
                </info:display-text>
                <xcon:cloning-parent>
                    xcon:6845432@example.com
                </xcon:cloning-parent>
                 <info:maximum-user-count>10</info:maximum-user-count>
                 <info:available-media>
                        <info:entry label="11">
                            <info:type>audio</info:type>
                        </info:entry>
```

```
                    </info:available-media>
                </info:conference-description>
                <info:users>
                    <xcon:join-handling>allow</xcon:join-handling>
                       <xcon: allowed-users-list>
                      <xcon:target uri="sip:alice@example.com"
                                   method="dial-out"/>
                      <xcon:target uri="sip:bob83@example.com"
                                   method="dial-out"/>
                      <xcon:target uri="sip:carol@example.com"
                                   method="dial-out"/>
                    </xcon:allowed-users-list>
                </info:users>
                   <xcon:floor-information>
                      <xcon:floor-request-handling>
                         confirm</xcon:floor-request-handling>
                      <xcon:conference-floor-policy>
                        <xcon:floor id="11"/>
                      </xcon:conference-floor-policy>
                   </xcon:floor-information>
               </confInfo>
           </ccmp:confResponse>
       </ccmpResponse>
   </ccmp:ccmpResponse>
```

Figure 7: Create Basic Conference (Clone) Detailed Messaging

5.4. Conference Creation using Blueprints

TOC

Figure 8 (Client Creation of Conference using Blueprints) provides an example of one client "Alice" determining the conference blueprints available for a particular conferencing system and creating a conference based on the desired blueprint.

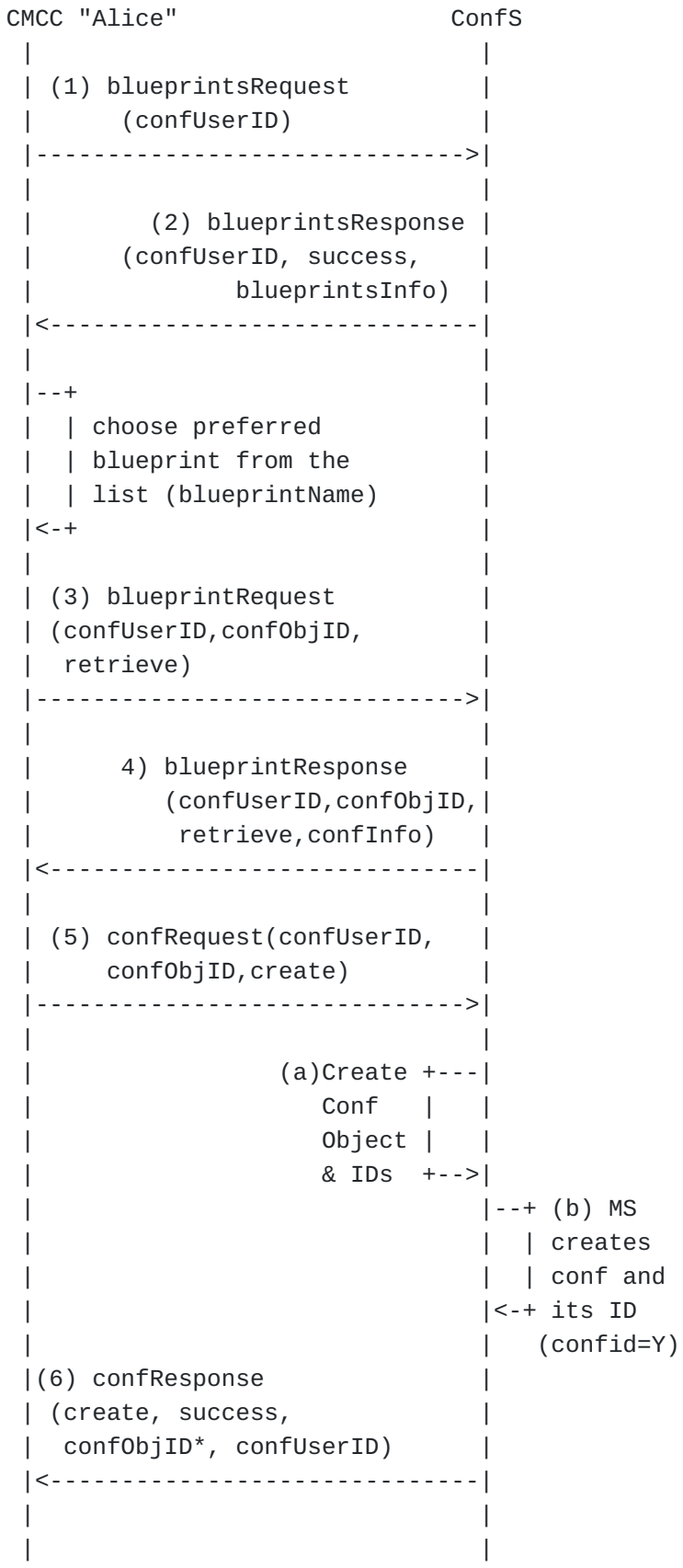

CMCC "Alice"
ConfS
(1) blueprintsRequest
(confUserID)
(2) blueprintsResponse
(confUserID, success,
blueprintsInfo)
choose preferred
blueprint from the
list (blueprintName)
(3) blueprintRequest
(confUserID,confObjID,
retrieve)
4) blueprintResponse
(confUserID,confObjID,
retrieve,confInfo)
(5) confRequest(confUserID,
confObjID,create)
(a)Create
Conf
Object
& IDs
(b) MS
creates
conf and
its ID
(confid=Y)
(6) confResponse
(create, success,
confObjID*, confUserID)

Figure 8: Client Creation of Conference using Blueprints

1. "Alice" first sends a "blueprintsRequest" message to the conferencing system identified by the conference server discovery process. Upon receipt of the "blueprintsRequest", the conferencing system would first authenticate "Alice" and then ensure that "Alice" has the appropriate authority based on system policies to receive any blueprints supported by that system.

2. Any blueprint that "Alice" is authorized to use are returned in a "blueprintsResponse" message in the "blueprintsInfo" attribute.

3. Upon receipt of the "blueprintsResponse" containing the blueprints, "Alice" determines which blueprint to use for the conference to be created. "Alice" sends a "blueprintRequest" message to get the specific blueprint as identified by the "confObjID".

4. The conferencing system returns the "confInfo" associated with the specific blueprint as identified by the 'confObjID' in the "blueprintResponse" message.

5. "Alice" then sends a "confRequest" with a "create" operation to the conferencing system to create a conference reservation, including the appropriate "blueprintName" and associated "confObjID".

6. Upon receipt of the "confRequest" message with a "create" operation, the conferencing system uses the received blueprint to clone a conference, allocating a new "confObjID" (again called "confObjID* in the example). The conferencing server then sends a "confResponse" message including the new "confObjID*" associated with the newly created conference instance. Upon receipt of the "confResponse" message, "Alice" can now add other users to the conference .

1. blueprintsRequest message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
     xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
     xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="xcon:ccmp-blueprints-request-message-type">
       <confUserID>xcon-userid:Alice@example.com</confUserID>
    </ccmpRequest>
   </ccmp:ccmpRequest>
```

2. blueprintsResponse message

```
   <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <ccmp:ccmpResponse
    xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
    xmlns:info="urn:ietf:params:xml:ns:conference-info"
    xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
   <ccmpResponse
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:type="ccmp:ccmp-blueprints-response-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <response-code>success</response-code>
        <ccmp:blueprintsResponse>
         <blueprintsInfo>
          <info:entry>
           <info:uri>xcon:AudioRoom@example.com</info:uri>
           <info:display-text>AudioRoom</info:display-text>
           <info:purpose>Simple Room:
              conference room with public access,
              where only audio is available, more users
              can talk at the same time
              and the requests for the AudioFloor
              are automatically accepted.
           </info:purpose>
          </info:entry>
          <info:entry>
           <info:uri>xcon:VideoRoom@example.com</info:uri>
           <info:display-text>VideoRoom</info:display-text>
           <info:purpose>Video Room:
               conference room with public access,
               where both audio and video are available,
               8 users can talk and be seen at the same time,
               and the floor requests are automatically accepted.
           </info:purpose>
          </info:entry>
```

```
        <info:entry>
         <info:uri>xcon:AudioConference1@example.com</info:uri>
         <info:display-text>AudioConference1</info:display-text>
         <info:purpose>Public Audio Conference:
              conference with public access,
              where only audio is available,
              only one user can talk at the same time,
              and the requests for the AudioFloor MUST
              be accepted by a Chair.
         </info:purpose>
        </info:entry>
        <info:entry>
         <info:uri>xcon:VideoConference1@example.com</info:uri>
         <info:display-text>VideoConference1</info:display-text>
           <info:purpose>Public Video Conference: conference
               where both audio and video are available,
               only one user can talk
           </info:purpose>
         </info:entry>
         <info:entry>
          <info:uri>xcon:AudioConference2@example.com</info:uri>
          <info:display-text>AudioConference2</info:display-text>
          <info:purpose>Basic Audio Conference:
               conference with private access,
               where only audio is available,
               only one user can talk at the same time,
               and the requests for the AudioFloor MUST
               be accepted by a Chair.
          </info:purpose>
         </info:entry>
      </blueprintsInfo>
    </ccmp:blueprintsResponse>
  </ccmpResponse>
</ccmp:ccmpResponse>
```

3. blueprintRequest/retrieve message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
      xmlns:info="urn:ietf:params:xml:ns:conference-info"
      xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
      xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
  <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-blueprint-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:AudioRoom@example.com</confObjID>
        <operation>retrieve</operation>
        <ccmp:blueprintRequest/>
```

```
     </ccmpRequest>
   </ccmp:ccmpRequest>
```

4. blueprintResponse/retrieve message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <ccmp:ccmpResponse
         xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
         xmlns:info="urn:ietf:params:xml:ns:conference-info"
         xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
     <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-blueprint-response-message-type">
       <confUserID>xcon-userid:Alice@example.com</confUserID>
       <confObjID>xcon:AudioRoom@example.com</confObjID>
       <operation>retrieve</operation>
       <response-code>success</response-code>
       <ccmp:blueprintResponse>
         <blueprintInfo entity="xcon:AudioRoom@example.com">
           <info:conference-description>
              <info:display-text>AudioRoom</info:display-text>
              <info:maximum-user-count>2</info:maximum-user-count>
              <info:available-media>
                <info:entry label="audioLabel">
                    <info:type>audio</info:type>
                </info:entry>
                </info:available-media>
           </info:conference-description>
           <info:users>
              <xcon:join-handling>allow</xcon:join-handling>
           </info:users>
           <xcon:floor-information>
             <xcon:floor-request-handling>confirm
             </xcon:floor-request-handling>
             <xcon:conference-floor-policy>
                   <xcon:floor id="audioLabel"></xcon:floor>
             </xcon:conference-floor-policy>
           </xcon:floor-information>
         </blueprintInfo>
       </ccmp:blueprintResponse>
     </ccmpResponse>
   </ccmp:ccmpResponse>
```

5. confRequest/create message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest
        xmlns:info="urn:ietf:params:xml:ns:conference-info"
        xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
        xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
```

```
    <ccmpRequest
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:type="ccmp:ccmp-conf-request-message-type">
       <confUserID>xcon-userid:Alice@example.com</confUserID>
       <confObjID>xcon:AudioRoom@example.com</confObjID>
       <operation>create</operation>
       <ccmp:confRequest/>
    </ccmpRequest>
  </ccmp:ccmpRequest>
```

6. confResponse/create message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpResponse
       xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
       xmlns:info="urn:ietf:params:xml:ns:conference-info"
       xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpResponse
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:type="ccmp:ccmp-conf-response-message-type">
       <confUserID>xcon-userid:Alice@example.com</confUserID>
       <confObjID>xcon:8977794@example.com</confObjID>
       <operation>create</operation>
       <response-code>success</response-code>
       <version>1</version>
       <ccmp:confResponse>
            <confInfo entity="xcon:8977794@example.com">
              <info:conference-description>
                  <info:display-text>
                     New conference by Alice cloned from AudioRoom
                  </info:display-text>
                  <info:conf-uris>
                     <info:entry>
                        <info:uri>
                            xcon:8977794@example.com
                        </info:uri>
                        <info:display-text>
                            conference xcon-uri
                        </info:display-text>
                        <xcon:conference-password>
                            8601
                        </xcon:conference-password>
                     </info:entry>
                  </info:conf-uris>
                  <info:maximum-user-count>10</info:maximum-user-count>
                  <info:available-media>
                         <info:entry label="11">
                             <info:type>audio</info:type>
```

```
                        </info:entry>
                    </info:available-media>
                </info:conference-description>
                <info:users>
                    <xcon:join-handling>allow</xcon:join-handling>
                </info:users>
                   <xcon:floor-information>
                      <xcon:floor-request-handling>
                         confirm</xcon:floor-request-handling>
                      <xcon:conference-floor-policy>
                        <xcon:floor id="11"/>
                      </xcon:conference-floor-policy>
                   </xcon:floor-information>
               </confInfo>
           </ccmp:confResponse>
       </ccmpResponse>
   </ccmp:ccmpResponse>
```

Figure 9: Create Conference (Blueprint) Detailed Messaging

6. General Conference scenarios and examples

TOC

The following scenarios are based on those documented in the XCON framework. The examples assume that a conference has already been correctly established, with media, if applicable, per one of the examples in Section 5 (Conference Creation).

6.1. Conference Announcements and Recordings

TOC

In this example, as shown in Figure 10 (Recording and Announcements) "Alice" is joining "Bob"'s conference that requires that she first enter a pass code. After successfully entering the passcode, an announcement prompts "Alice to speak her name so it can be recorded. When "Alice" is added to the active conference, the recording is played back to all the existing participants. A very similar example is presented in Figure 33 of [I-D.ietf-mediactrl-call-flows] (Amirante, A., Castaldi, T., Miniero, L., and S. Romano, “Media Control Channel Framework (CFW) Call Flow Examples,” February 2010.).


CCMC "Alice"
ConfS
MS
(1)userRequest(confUserID, confObjID,create,userInfo)
--+ Alice is
| new in the
<-+ system (create
confUserID)
ConfS handles SIP signaling (Alice<->ConfS<->MS)
--+ A password is
| required for
<-+ that conference
Request IVR menu (PIN)
<========== MS gets PIN from Alice through DTMF ==========>
Provided PIN is...
Check PIN
--+ Alice must
| record her
<-+ name
Request name recording
<========== MS records Alice's audio RTP (name) ==========>
Audio recording
Complete creation of Alice
(2)userResponse(confUserID, confObjID, create, success, version)

Figure 10: Recording and Announcements

1. Upon receipt of the userRequest from "Alice" to be added to "Bob's" conference (i.e., an "update" operation on "Bob's" conference object), the conferencing system determines that a password is required for this specific conference. Thus an announcement asking "Alice" to enter the password is provided to "Alice". This may be achieved by means of typical IVR fucntionality. Once "Alice" enters the password, it is validated against the policies associated with "Bob's" active conference. The conferencing system then connects to a server which prompts and records "Alice's" name. The conferencing system must also determine whether "Alice" is already a user of this conferencing system or whether she is a new user. In this case, "Alice" is a new user for this conferencing system, so a conference user identifier is created for "Alice". Based upon the addressing information provided by "Alice", the call signaling to add "Alice" to the conference is instigated through the Focus.

2. The conference server sends "Alice" a userResponse message which includes the "confUserID" assigned by the conferencing system for "Alice". This would allow "Alice" to later perform operations on the conference (if she were to have the appropriate policies), including registering for event notifications associated with the conference. Once the call signaling indicates that "Alice" has been successfully added to the specific conference, per updates to the state, and depending upon the policies, other participants (e.g., "Bob") are notified of the addition of "Alice" to the conference via the conference notification service and an announcement is provided to all the participants indicating that "Alice" has joined the conference.

1. userRequest/create message (Alice tries to enter the conference without providing the

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
     xmlns:info="urn:ietf:params:xml:ns:conference-info"
           xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
           xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="ccmp:ccmp-user-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>create</operation>
        <ccmp:userRequest>
            <userInfo entity="xcon-userid:Alice@example.com">
                <info:associated-aors>
                    <info:entry>
                        <info:uri>
                           mailto:Alice83@example.com
                        </info:uri>
                        <info:display-text>email</info:display-text>
                    </info:entry>
                </info:associated-aors>
                <info:endpoint entity="sip:alice_789@example.com"/>
            </userInfo>
        </ccmp:userRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>
```

2. userResponse/create message ("passwordRequired")

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
    xmlns:info="urn:ietf:params:xml:ns:conference-info"
    xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
    xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
   <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-user-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>create</operation>
        <ccmp:response-code>passwordRequired</ccmp:response-code>
        <ccmp:userResponse/>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

3. userRequest/create message (Alice provides the password)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
     xmlns:info="urn:ietf:params:xml:ns:conference-info"
           xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
           xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xsi:type="ccmp:ccmp-user-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>create</operation>
        <password>8601</password>
        <ccmp:userRequest>
            <userInfo entity="xcon-userid:Alice@example.com">
                <info:associated-aors>
                    <info:entry>
                        <info:uri>
                           mailto:Alice83@example.com
                        </info:uri>
                        <info:display-text>email</info:display-text>
                    </info:entry>
                </info:associated-aors>
                <info:endpoint entity="sip:alice_789@example.com"/>
            </userInfo>
        </ccmp:userRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>

4.      userResponse/create message ("success")

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
    xmlns:info="urn:ietf:params:xml:ns:conference-info"
    xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
    xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
   <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-user-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>create</operation>
        <response-code>success</response-code>
        <version>10</version>
        <ccmp:userResponse/>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 11: Announcement Messaging Details

6.2. Monitoring for DTMF

TOC

The conferencing system also needs the capability to monitor for DTMF from each individual participant. This would typically be used to enter the identifier and/or access code for joining a specific conference. An example of DTMF monitoring, within the context of the framework elements, is shown in Figure 10 (Recording and Announcements). A typical way for the conferencing system to be aware of all the DTMF interactions within the context of conferences it is responsible for, is making use of the MEDIACTRL architecture for what regards media manipulation. Examples in that sense (specifically for what concerns DTMF interception in conference instances) are presented in [I-D.ietf-mediactrl-call-flows] (Amirante, A., Castaldi, T., Miniero, L., and S. Romano, "Media Control Channel Framework (CFW) Call Flow Examples," February 2010.).

6.3. Adding a Party

TOC

In this example, "Alice" wants to add "Bob" to an established conference. In the following example we assume "Bob" is a new user to the system, which means "Alice" also needs to provide details about him. In fact, the case of "Bob" already present as a user in the conferencing system is much easier to address, and will be discussed later on.

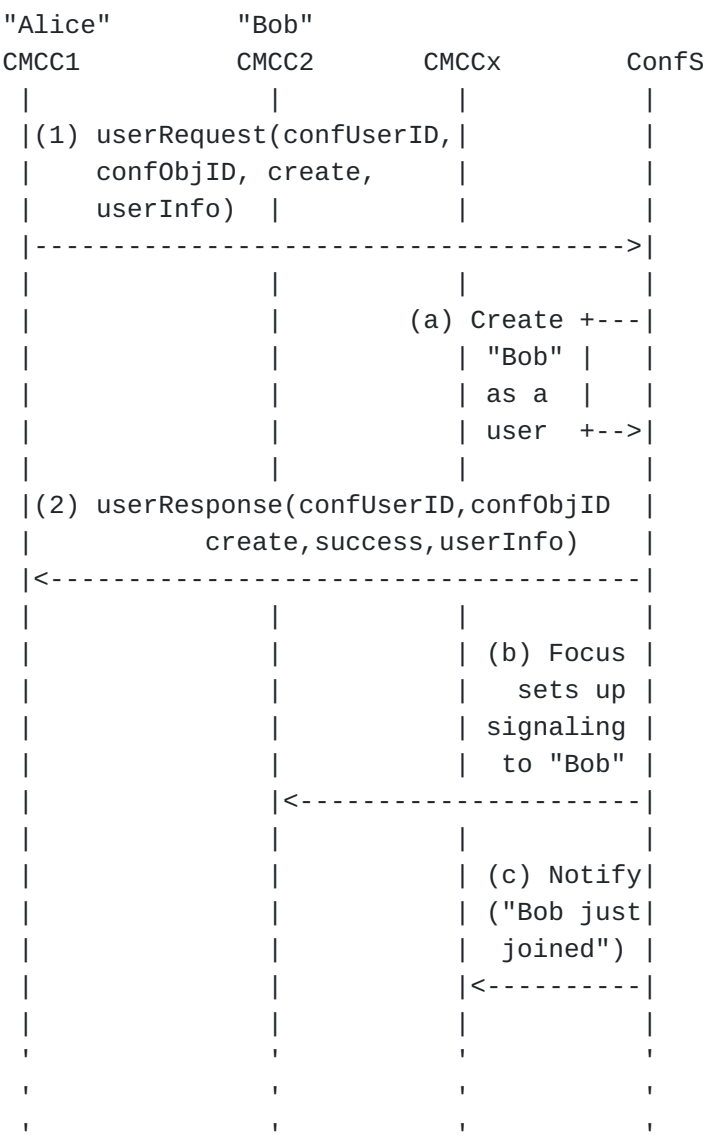


Figure 12: Client Manipulation of Conference - Add a party

---

1. "Alice" sends a userRequest message with an operation of "create" to add "Bob" to the specific conference as identified by the confObjID. The "create" operation also makes sure that "Bob" is created as a user in the whole conferencing system. This is done by adding a "userInfo" element describing "Bob" as a user. This is needed in order to let the conferencing system be aware of "Bob"'s characteristics. In case Bob was already a registered user, "Alice" would just have referenced him through his XCON UserID, without providing the additional "userInfo". In fact, that information (including, for instance, "Bob"'s SIP

URI to be used subsequently for dial-out) would be obtained by referencing the extant registration. The conference server ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. As mentioned before, a new Conference User Identifier is created for Bob, and the "userInfo" is used to update the conference object accordingly.

2. In the presented example, the call signaling to add "Bob" to the conference is instigated through the Focus as well. Please notice that this is implementation specific. In fact, a conferencing system may accomplish different actions after the user creation, just as it may do nothing at all. Among the possible actions, for instance "Bob" may be added as a "target" element to the "allowed-users-list" element, whose joining "method" may be either "dial-in" or "dial-out". Besides, out-of-band notification mechanisms may be involved as well, e.g. to notify "Bob" via mail of the new conference, including details as the date, password, expected participants and so on.

> To conclude the overview on this scenario, once "Bob" has been successfully added to the specified conference, per updates to the state, and depending upon the policies, other participants (including "Bob" himself) may be notified of the addition of "Bob" to the conference via the Conference Notification Service.

---

1. userRequest/create message (Alice adds Bob)

```
 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
        xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
        xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:type="ccmp:ccmp-user-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977878@example.com</confObjID>
        <operation>create</operation>
        <ccmp:userRequest>
          <userInfo>
              <info:display-text>Bob</info:display-text>
              <info:associated-aors>
                  <info:entry>
                    <info:uri>mailto:bob.depippis@example.com</info:uri>
                    <info:display-text>Bob's email</info:display-text>
                  </info:entry>
              </info:associated-aors>
              <info:endpoint entity="sip:bob83@example.com">
                  <info:display-text>Bob's laptop</info:display-text>
              </info:endpoint>
          </userInfo>
        </ccmp:userRequest>
    </ccmpRequest>
  </ccmp:ccmpRequest>
```

2. userResponse/create message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpResponse xmlns:info="urn:ietf:params:xml:ns:conference-info"
         xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
         xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:type="ccmp:ccmp-user-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977878@example.com</confObjID>
        <operation>create</operation>
        <response-code>success</response-code>
        <version>10</version>
        <ccmp:userResponse>
          <userInfo entity="xcon-userid:Bob@example.com">
              <info:display-text>Bob</info:display-text>
              <info:associated-aors>
                  <info:entry>
```

```
                    <info:uri>mailto:bob.depippis@example.com</info:uri>
                    <info:display-text>Bob's email</info:display-text>
                </info:entry>
            </info:associated-aors>
            <info:endpoint entity="sip:bob83@example.com">
                <info:display-text>Bob's laptop</info:display-text>
            </info:endpoint>
        </userInfo>
    </ccmp:userResponse>
  </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 13: Add Party Message Details

6.4. Muting a Party

TOC

This section provides an example of the muting of a party in an active conference. The unmuting would involve the identical CCMP message flow. Although, in the case that floor control is involved, whether or not a particular conference client can unmute itself must be considered by the conferencing system.

> Please notice that interaction between CCMP and floor control should be carefully considered. In fact, handling CCMP- and BFCP-based media control has to be considered as multiple layers: i.e., a participant may have the BFCP floor granted, but be muted by means of CCMP. If so, he would still be muted in the conference, and would only be unmuted if both the protocols allowed for this.

Figure 14 (Client Manipulation of Conference - Mute a party) provides an example of one client "Alice" impacting the media state of another client "Bob". This example assumes an established conference. In this example, the client, "Alice" whose Role is "moderator" of the conference, wants to mute "Bob" on a medium-size multi-party conference, as his device is not muted (and he's obviously not listening to the call) and background noise in his office environment is disruptive to the conference. BFCP floor control is assumed not to be involved.

Figure 14: Client Manipulation of Conference - Mute a party

---

1. Upon receipt of userRequest message with an update operation and the userInfo with the "status" field in the "media" element for "Bob" set to "revconly". The Conference Server ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation and updates the userInfo in the conference object reflect that "Bob"s media is not to be mixed with the conference media. In case the Conference Server relies on a remote Media Server for its multimedia functionality, it subsequently changes "Bob"'s media profile accordingly by means

of the related protocol interaction with the MS. An example describing a possible way of dealing with such a situation using the Media Server Control architecture is described in [I-D.ietf-mediactrl-call-flows] (Amirante, A., Castaldi, T., Miniero, L., and S. Romano, “Media Control Channel Framework (CFW) Call Flow Examples,” February 2010.), at "Simple Bridging: Framework Transactions (2)".

2. A userResponse message with a responseCode of "success" is then sent to "Alice". Depending upon the policies, tne conference server may notify other participants (including "Bob") of this update via the Conference Notification Service.

1. userRequest/update message (Alice mutes Bob)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
      xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
      xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
  <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:type="ccmp:ccmp-user-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8977878@example.com</confObjID>
      <operation>update</operation>
      <ccmp:userRequest>
          <userInfo entity="xcon-userid:Bob@example.com">
              <info:endpoint entity="sip:bob83@example.com">
                  <info:media id="1">
                      <info:label>123</info:label>
                      <info:status>recvonly</info:status>
                  </info:media>
              </info:endpoint>
          </userInfo>
      </ccmp:userRequest>
  </ccmpRequest>
</ccmp:ccmpRequest>
```

2. userResponse/update message

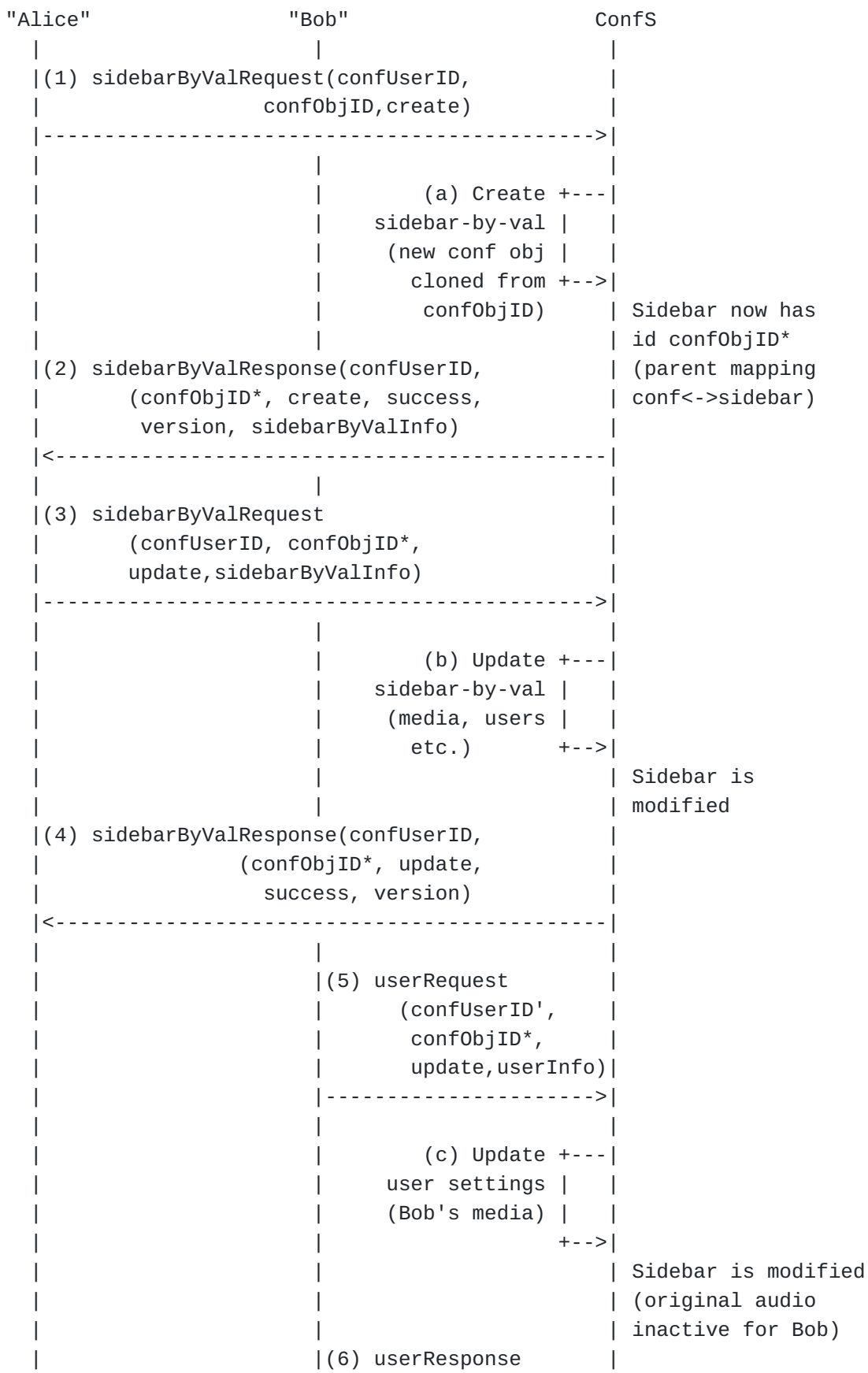
```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse xmlns:info="urn:ietf:params:xml:ns:conference-info"
            xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
            xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
  <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:type="ccmp:ccmp-user-response-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8977878@example.com</confObjID>
      <operation>update</operation>
      <response-code>success</response-code>
      <version>7</version>
      <ccmp:userResponse/>
  </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 15: Mute Message Details

6.5. Internal Sidebar

TOC

Figure 16 (Client Creation of a Sidebar Conference) provides an example of one client "Alice" involved in active conference with "Bob" and "Carol". "Alice" wants to create a sidebar to have a side discussion with "Bob" while still viewing the video associated with the main conference. Alternatively, the audio from the main conference could be maintained at a reduced volume. "Alice" initiates the sidebar by sending a request to the conferencing system to create a conference reservation based upon the active conference object. "Alice" and "Bob" would remain on the roster of the main conference, such that other participants could be aware of their participation in the main conference, while an internal-sidebar conference is occurring. Besides, "Bob" decides that he is not interested in still receiving the conference audio in background (not even at a lower volume as "Alice" configured) and so modifies the sidebar in order to make that stream inactive for him.


"Alice"
"Bob"
ConfS
(1) sidebarByValRequest(confUserID, confObjID,create)
(a) Create sidebar-by-val (new conf obj cloned from confObjID)
Sidebar now has id confObjID* (parent mapping conf<->sidebar)
(2) sidebarByValResponse(confUserID, (confObjID*, create, success, version, sidebarByValInfo)
(3) sidebarByValRequest (confUserID, confObjID*, update,sidebarByValInfo)
(b) Update sidebar-by-val (media, users etc.)
Sidebar is modified
(4) sidebarByValResponse(confUserID, (confObjID*, update, success, version)
(5) userRequest (confUserID', confObjID*, update,userInfo)|
(c) Update user settings (Bob's media)
Sidebar is modified (original audio inactive for Bob)
(6) userResponse

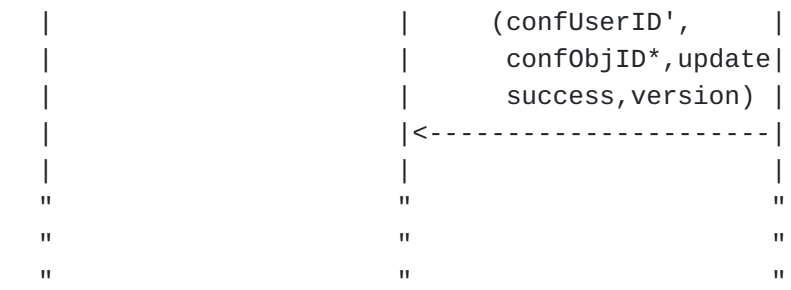


Figure 16: Client Creation of a Sidebar Conference

---

1. Upon receipt of CCMP sidebarByValRequest message to "reserve" a new sidebar conference based upon the confObjID received in the request, the conferencing system uses the confObjID to clone a conference reservation for the sidebar. The sidebar reservation is NOT independent of the active conference (i.e., parent). The conferencing system also reserves or allocates a new confObjID to be used for any subsequent protocol requests from any of the members of the conference.

2. The relationship information is provided in the sidebarByValResponse message, specifically in the "sidebar-parent" element. A dump of the complete representation of the main/parent conference is provided below as well to show how the cloning process for the creation of the sidebar could take place.

3. Upon receipt of the sidebarByValResponse message to reserve the conference, "Alice" can now create an active conference using that reservation, or create additional reservations based upon the existing reservations. In this example, "Alice" wants only "Bob" to be involved in the sidebar, thus she manipulates the membership so that only the two of them appear in the "allowed-users-list" section. "Alice" also wants both audio and the video from the original conference to be available in the sidebar. For what concerns the media belonging to the sidebar itself, "Alice" wants the audio to be restricted to the participants in the sidebar (that is, her and "Bob"). Additionally, "Alice" manipulates the media values to recieve the audio from the main conference at a reduced volume, so that the communication between her and "Bob" isn't affected. "Alice" sends a sidebarByValRequest message with an operation of "update" along with the sidebarByValInfo in the reservation, to create an active conference.

4. Upon receipt of the sidebarByValRequest to update the reservation to create an active conference for the sidebar, as identified by the sidebar conference object ID, the conference server ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conference server must also validate the updated information in the reservation, ensuring that a member like "Bob" is already a user of this conference server. Once the data for the confObjID is updated, the conference server sends a sidebarByValResponse to "Alice". Depending upon the policies, the initiator of the request (i.e., "Alice") and the participants in the sidebar (i.e., "Bob") may be notified of his addition to the sidebar via the conference notification service.

5. At this point, "Bob" sends a userRequest message to the conference server with an operation of "update" to completely disable the background audio from the parent conference, since it prevents him from understanding what "Alice" says in the sidebar.

6. Notice that "Bob's" request only changes the media perspective for "Bob". "Alice" keeps on receiving both the audio from "Bob" and the background from the parent conference. This request may be relayed by the conference server to the Media Server handling the mixing, if present. Upon completion of the change, the conference server sends a "userResponse" message to "Bob". Depending upon the policies, the initiator of the request (i.e., "Bob") and the participants in the sidebar (i.e., "Alice") may be notified of this change via the conference notification service.

That said, let's consider the following conference object:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <info:conference-info
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
                entity="xcon:8977878@example.com">
     <info:conference-description>
        <info:display-text>MAIN CONFERENCE</info:display-text>
        <info:conf-uris>
            <info:entry>
               <info:uri>sip:8977878@example.com</info:uri>
               <info:display-text>conference sip uri</info:display-text>
            </info:entry>
        </info:conf-uris>
        <info:available-media>
          <info:entry label="123">
            <info:display-text>main conference audio</info:display-text>
            <info:type>audio</info:type>
            <info:status>sendrecv</info:status>
          </info:entry>
          <info:entry label="456">
            <info:display-text>main conference video</info:display-text>
            <info:type>video</info:type>
            <info:status>sendrecv</info:status>
            <xcon:controls>
                    <xcon:video-layout>single-view</xcon:video-layout>
           </xcon:controls>
          </info:entry>
        </info:available-media>
    </info:conference-description>
    <info:conference-state>
        <info:active>true</info:active>
    </info:conference-state>
    <info:users>
        <info:user entity="xcon-userid:Alice@example.com">
            <info:display-text>Alice</info:display-text>
            <info:endpoint entity="sip:Alice@example.com">
                <info:media id="1">
                    <info:label>123</info:label>
                    <info:status>sendrecv</info:status>
                </info:media>
                <info:media id="2">
                    <info:label>456</info:label>
                    <info:status>sendrecv</info:status>
                </info:media>
            </info:endpoint>
        </info:user>
```

```
            <info:user entity="xcon-userid:Bob@example.com">
                <info:display-text>Bob</info:display-text>
                <info:endpoint entity="sip:bob83@example.com">
                    <info:media id="1">
                        <info:label>123</info:label>
                        <info:status>sendrecv</info:status>
                    </info:media>
                    <info:media id="2">
                        <info:label>456</info:label>
                        <info:status>sendrecv</info:status>
                    </info:media>
                </info:endpoint>
            </info:user>
            <info:user entity="xcon-userid:Carol@example.com">
                <info:display-text>Carol</info:display-text>
                <info:endpoint entity="sip:carol@example.com">
                    <info:media id="1">
                        <info:label>123</info:label>
                        <info:status>sendrecv</info:status>
                    </info:media>
                    <info:media id="2">
                        <info:label>456</info:label>
                        <info:status>sendrecv</info:status>
                    </info:media>
                </info:endpoint>
            </info:user>
        </info:users>
    </info:conference-info>
```

This is the representation of the conference the sidebar is going to be created in. As such, it will be used by the conferencing system in order to create the new conference object associated with the sidebar. In fact, the sidebar creation happens through a cloning of the parent conference. Once the sidebar is created, an "update" makes sure that the sidebar is customized as needed. The following protocol dump makes the process clearer.

---

1. sidebarByValRequest/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
             xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
             xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
  <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="ccmp:ccmp-sidebarByVal-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8977878@example.com</confObjID>
      <operation>create</operation>
      <ccmp:sidebarByValRequest/>
  </ccmpRequest>
</ccmp:ccmpRequest>
```

2. sidebarByValResponse/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
            xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
            xmlns:info="urn:ietf:params:xml:ns:conference-info"
            xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
  <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="ccmp:ccmp-sidebarByVal-response-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8974545@example.com</confObjID>
      <response-code>success</response-code>
          <version>1</version>
      <ccmp:sidebarByValResponse>
          <sidebarByValInfo entity="xcon:8974545@example.com">
              <info:conference-description>
                  <info:display-text>
                       SIDEBAR CONFERENCE registered by Alice
                  </info:display-text>
                  <xcon:sidebar-parent>
                       xcon:8977878@example.com
                  </xcon:sidebar-parent>
                  <info:available-media>
                      <info:entry label="123">
                          <info:display-text>
                                main conference audio
                          </info:display-text>
                          <info:type>audio</info:type>
                          <info:status>sendrecv</info:status>
                      </info:entry>
                      <info:entry label="456">
```

```
                        <info:display-text>
                              main conference video
                        </info:display-text>
                        <info:type>video</info:type>
                        <info:status>sendrecv</info:status>
                    </info:entry>
                </info:available-media>
            </info:conference-description>
            <info:conference-state>
                <info:active>false</info:active>
            </info:conference-state>
            <info:users>
                <xcon:allowed-users-list>
                    <xcon:target method="dial-out"
                          uri="xcon-userid:Alice@example.com"/>
                    <xcon:target method="dial-out"
                          uri="xcon-userid:Bob@example.com"/>
                    <xcon:target method="dial-out"
                          uri="xcon-userid:Carol@example.com"/>
                </xcon:allowed-users-list>
            </info:users>
        </sidebarByValInfo>
    </ccmp:sidebarByValResponse>
  </ccmpResponse>
</ccmp:ccmpResponse>
```

3. sidebarByValRequest/update message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest
            xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
            xmlns:info="urn:ietf:params:xml:ns:conference-info"
            xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-sidebarByVal-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8974545@example.com</confObjID>
        <operation>update</operation>
        <ccmp:sidebarByValRequest>
            <sidebarByValInfo entity="xcon:8974545@example.com">
                <info:conference-description>
                  <info:display-text>
                        private sidebar Alice - bob
                  </info:display-text>
                  <info:available-media>
                        <info:entry label="123">
                            <info:display-text>
                                main conference audio
```

```
                            </info:display-text>
                            <info:type>audio</info:type>
                            <info:status>recvonly</info:status>
                            <xcon:controls>
                                <xcon:gain>-60</xcon:gain>
                            </xcon:controls>
                        </info:entry>
                        <info:entry label="456">
                            <info:display-text>
                                main conference video
                            </info:display-text>
                            <info:type>video</info:type>
                            <info:status>recvonly</info:status>
                        </info:entry>
                        <info:entry label="sidebarAudioLabel">
                            <info:display-text>
                                sidebar audio
                            </info:display-text>
                            <info:type>audio</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                        <info:entry label="sidebarVideoLabel">
                            <info:display-text>
                                sidebar video
                            </info:display-text>
                            <info:type>video</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                    </info:available-media>
                </info:conference-description>
                <info:users>
                    <xcon:allowed-users-list>
                        <xcon:target method="dial-in"
                              uri="xcon-userid:Alice@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Bob@example.com"/>
                    </xcon:allowed-users-list>
                </info:users>
            </sidebarByValInfo>
        </ccmp:sidebarByValRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>
```

4. sidebarByValResponse/update message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpResponse
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
```

```
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="ccmp:ccmp-sidebarByVal-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8974545@example.com</confObjID>
        <operation>update</operation>
        <response-code>success</response-code>
        <version>2</version>
        <ccmp:sidebarByValResponse/>
    </ccmpResponse>
  </ccmp:ccmpResponse>
```

5. userRequest/update message (Alice updates Bob's media)

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest
           xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
           xmlns:info="urn:ietf:params:xml:ns:conference-info"
           xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
      <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-user-request-message-type">
        <confUserID>xcon-userid:Bob@example.com</confUserID>
        <confObjID>xcon:8974545@example.com</confObjID>
        <operation>update</operation>
        <ccmp:userRequest>
            <userInfo entity="xcon-userid:Bob@example.com">
                <info:endpoint entity="sip:bob83@example.com">
                    <info:media id="1">
                        <info:display-text>
                            main conference audio
                        </info:display-text>
                        <info:label>123</info:label>
                        <info:status>inactive</info:status>
                    </info:media>
                </info:endpoint>
            </userInfo>
        </ccmp:userRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>
```

6. userResponse/update message

```
  <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpResponse xmlns:info="urn:ietf:params:xml:ns:conference-info"
              xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
              xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
```

```
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-user-response-message-type">
        <confUserID>xcon-userid:Bob@example.com</confUserID>
        <confObjID>xcon:8974545@example.com</confObjID>
        <operation>update</operation>
        <response-code>success</response-code>
        <version>3</version>
        <ccmp:userResponse/>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 17: Internal Sidebar Messaging Details

6.6. External Sidebar

TOC

Figure 18 (Client Creation of an External Sidebar) provides an example of a different approach towards sidebar. In this scenario, one client, "Alice", is involved in an active conference with "Bob", "Carol", "David" and "Ethel". "Alice" gets an important text message via a whisper from "Bob" that a critical customer needs to talk to "Alice", "Bob" and "Ethel". "Alice" creates a sidebar to have a side discussion with the customer "Fred" including the participants in the current conference with the exception of "Carol" and "David", who remain in the active conference. The difference from the previous scenario is that "Fred" is not part of the parent conference: this means that different policies might be involved, considering that "Fred" may access information coming from the parent conference, in case the sidebar was configured accordingly. For this reason, in this scenario we assume that "Alice" disables all the media from the original (parent) conference within the sidebar. This means that, while in the previous example "Alice" and "Bob" still heard the audio from the main conference in background, this time no background is made available. "Alice" initiates the sidebar by sending a request to the conferencing system to create a conference reservation based upon the active conference object. "Alice", "Bob" and "Ethel" would remain on the roster of the main conference in a hold state. Whether or not the hold state of these participants is visible to other participants depends upon the individual and local policy.


"Alice"
"Bob"
ConfS
(1) sidebarByRefRequest(confUserID, confObjID, create)
(a) Create sidebar-by-ref (new conf obj cloned from confObjID)
Sidebar now has id confObjID* (parent mapping conf<->sidebar)
(2) sidebarByRefResponse(confUserID, confObjID*, create, success, version, sidebarByRefInfo)
(3) sidebarByRefRequest(confUserID, confObjID*,update,sidebarByRefInfo)
(b) Create new user for "Fred"
(c) Update sidebar-by-val (media, users policy, etc.)
Sidebar is modified: no media from the parent conference is available to anyone
(4) sidebarByRefResponse(confUserID, confObjID*, update, success, version)
Notify ("Fred" added to sidebar users)

" " "

Figure 18: Client Creation of an External Sidebar

---

1. Upon receipt of the "sidebarByRefRequest" message to create a new sidebar conference, based upon the active conference specified by "confObjID" in the request, the conferencing system uses the received active conference to clone a conference reservation for the sidebar. The sidebar reservation is NOT independent of the active conference (i.e., parent). The conferencing system as before reserves or allocates a conference ID (confObjID*) to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the sidebar reservation through the conference instance. Just as before, this mapping is mantained in "sidebar-parent".

2. Upon receipt of the "sidebarByRefResponse" message, which acknowledges the successful creation of the sidebar object, "Alice" decides that only "Bob" and "Ethel", along with the new participant "Fred" are to be involved in the sidebar. Thus she manipulates the membership accordingly. "Alice" also sets the media in the "conference-info" such that the participants in the sidebar don't receive any media from the main conference. All these settings are provided to the conferencing system by means of a new "sidebarByRefRequest" message, with an "update" operation.

3. "Alice" sends the aforementioned "sidebarByRefRequest" to update the information in the reservation and to create an active conference. Upon receipt of the "sidebarByRefRequest" with an operation of "update", the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conferencing system also validates the updated information in the reservation. Since "Fred" is a new user for this conferencing system, a conference user identifier is created for "Fred". Specifically, "Fred" is added to the conference by only providing his SIP URI. Based upon the addressing information provided for "Fred" by "Alice", the call signaling to add "Fred" to the conference may be instigated through the Focus (e.g. if "Fred" had a "dial-out" method set as the target for him) at the actual activation of the sidebar.

4. The conference server sends a "sidebarByRefResponse" message and, depending upon the policies, the initiator of the request (i.e., "Alice") and the participants in the sidebar (i.e., "Bob" and "Ethel") may be notified of his addition to the sidebar via the conference notification service.

1. sidebarByRefRequest/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
             xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
             xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
  <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="ccmp:ccmp-sidebarByRef-request-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8977878@example.com</confObjID>
      <operation>create</operation>
      <ccmp:sidebarByRefRequest/>
  </ccmpRequest>
</ccmp:ccmpRequest>
```

2. sidebarByRefResponse/create message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
              xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
              xmlns:info="urn:ietf:params:xml:ns:conference-info"
              xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
  <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:type="ccmp:ccmp-sidebarByref-response-message-type">
      <confUserID>xcon-userid:Alice@example.com</confUserID>
      <confObjID>xcon:8971212@example.com</confObjID>
      <operation>success</operation>
      <response-code>success</response-code>
      <version>1</version>
      <ccmp:sidebarByRefResponse>
          <sidebarByRefInfo entity="xcon:8971212@example.com">
              <info:conference-description>
                  <info:display-text>
                      SIDEBAR CONFERENCE registered by Alice
                  </info:display-text>
                  <xcon:sidebar-parent>
                      xcon:8977878@example.com
                  </xcon:sidebar-parent>
                  <info:available-media>
                      <info:entry label="123">
                          <info:display-text>
                               main conference audio
                          </info:display-text>
                          <info:type>audio</info:type>
                          <info:status>sendrecv</info:status>
```

```
                        </info:entry>
                        <info:entry label="456">
                            <info:display-text>
                                 main conference video
                            </info:display-text>
                            <info:type>video</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                    </info:available-media>
                </info:conference-description>
                <info:conference-state>
                    <info:active>false</info:active>
                </info:conference-state>
                <info:users>
                    <xcon:allowed-users-list>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Alice@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Bob@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Carol@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:David@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Ethel@example.com"/>
                    </xcon:allowed-users-list>
                </info:users>
            </sidebarByRefInfo>
        </ccmp:sidebarByRefResponse>
    </ccmpResponse>
  </ccmp:ccmpResponse>
```

3. sidebarByRefRequest/update message

```
  <ccmp:ccmpRequest
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-sidebarByRef-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8971212@example.com</confObjID>
        <operation>update</operation>
        <ccmp:sidebarByRefRequest>
            <sidebarByRefInfo entity="xcon:8971212@example.com">
                <info:conference-description>
                    <info:display-text>
                        sidebar Alice, bob, ethel & fred
```

```
    </info:display-text>
    <info:available-media>
        <info:entry label="123">
            <info:display-text>
                 main conference audio
            </info:display-text>
            <info:type>audio</info:type>
            <info:status>inactive</info:status>
        </info:entry>
        <info:entry label="456">
            <info:display-text>
                 main conference video
        </info:display-text>
            <info:type>video</info:type>
            <info:status>inactive</info:status>
        </info:entry>
        <info:entry>
            <info:display-text>
                 sidebar audio
            </info:display-text>
            <info:type>audio</info:type>
            <info:status>sendrecv</info:status>
        </info:entry>
        <info:entry>
            <info:display-text>
                 sidebar video
            </info:display-text>
            <info:type>video</info:type>
            <info:status>sendrecv</info:status>
            <xcon:controls>
                 <xcon:video-layout>
                       single-view
                 </xcon:video-layout>
            </xcon:controls>
        </info:entry>
    </info:available-media>
</info:conference-description>
<info:conference-state>
    <info:active>false</info:active>
</info:conference-state>
<info:users>
    <xcon:allowed-users-list>
        <xcon:target method="dial-in"
              uri="xcon-userid:Alice@example.com"/>
        <xcon:target method="dial-out"
              uri="xcon-userid:Bob@example.com"/>
        <xcon:target method="dial-out"
              uri="sip:fred@example.com"/>
    </xcon:allowed-users-list>
```

```
                </info:users>
              </sidebarByRefInfo>
          </ccmp:sidebarByRefRequest>
      </ccmpRequest>
    </ccmp:ccmpRequest>
```

4. sidebarByRefResponse/update message

```
    <ccmp:ccmpResponse
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
      <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:type="ccmp:ccmp-sidebarByref-response-message-type">
          <confUserID>xcon-userid:Alice@example.com</confUserID>
          <confObjID>xcon:8971212@example.com</confObjID>
          <operation>update</operation>
          <response-code>success</response-code>
          <version>2</version>
          <ccmp:sidebarByRefResponse/>
      </ccmpResponse>
    </ccmp:ccmpResponse>
```

Figure 19: External Sidebar Messaging Details

6.7. Floor control using sidebars

TOC

Floor control with sidebars can be used to realize conferencing scenario such as an analyst briefing. In this scenario, the conference call has a panel of speakers who are allowed to talk in the main conference. The other participants are the analysts, who are not allowed to speak unless they have the floor. To request access to the floor, they have to join a new sidebar with the moderator and ask their question. The moderator can also whisper to each analyst what their status/position in the floor control queue, similar to the example in Figure 23 (Whisper). It should be noted that other mechanisms which don't make use of sidebars could be used for floor control such as those detailed in BFCP.

Figure 20 (Floor Control with sidebars) provides an example of the configuration involved for this type of conference. As in the previous sidebar examples, there is the main conference along with a sidebar. "Alice" and "Bob" are the main participants in the conference, with

"A1", "A2" and "A3" representing the analysts. The sidebar remains active throughout the conference, with the moderator, "Carol", serving as the chair. As discussed previously, the sidebar conference is NOT independent of the active conference (i.e., parent). The analysts are provided the conference object ID associated with the active sidebar when they join the main conference. The conferencing system also allocates a conference ID to be used for any subsequent manipulations of the sidebar conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the active sidebar conference through the conference instance. The analysts are permanently muted while in the main conference. The analysts are moved to the sidebar when they wish to speak. Only one analyst is given the floor at a given time. All participants in the main conference receive audio from the sidebar conference, as well as audio provided by the panelists in the main conference.

(To Be added).

Figure 20: Floor Control with sidebars

---

1. "A1" wishes to ask a question, so he sends a Floor Request message to the floor control server.

2. Upon receipt of the request, the floor control server notifies the moderator, "Carol" of the active sidebar conference, whose serving as the floor chair.

3. Since no other analysts have yet requested the floor, "Carol" indicates to the floor control server that "A1" may be granted the floor.

(CCMP Messaging details not available yet).

Figure 21: Floor Control Messaging Details

6.8. Whispering or Private Messages

TOC

The case of private messages can be handled as a sidebar with just two participants, similarly to the example in section Section 6.5 (Internal Sidebar). Unlike the previous example, anyway, rather than using audio within the sidebar, "Alice" could just add an additional text based media stream to the sidebar in order to convey her whisper. From the protocol point of view, with reference to the messages described in Figure 17 (Internal Sidebar Messaging Details), only the third CCMP message (a sidebarByValRequest/update) changes, as depicted in Figure 22 (SidebarByVal Update for private messages scenario).

3. sidebarByValRequest/update message

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest
            xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
            xmlns:info="urn:ietf:params:xml:ns:conference-info"
            xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-sidebarByVal-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8974545@example.com</confObjID>
        <operation>update</operation>
        <ccmp:sidebarByValRequest>
            <sidebarByValInfo entity="xcon:8974545@example.com">
                <info:conference-description>
                  <info:display-text>
                        private sidebar alice - bob
                  </info:display-text>
                  <info:available-media>
                        <info:entry label="123">
                            <info:display-text>
                                main conference audio
                            </info:display-text>
                            <info:type>audio</info:type>
                            <info:status>recvonly</info:status>
                        </info:entry>
                        <info:entry label="456">
                            <info:display-text>
                                main conference video
                            </info:display-text>
                            <info:type>video</info:type>
                            <info:status>recvonly</info:status>
                        </info:entry>
                        <info:entry label="sidebarTextLabel">
                            <info:display-text>
                                sidebar text
                            </info:display-text>
                            <info:type>text</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                    </info:available-media>
                </info:conference-description>
                <info:users>
                    <xcon:allowed-users-list>
                        <xcon:target method="dial-in"
                              uri="xcon-userid:Alice@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:Bob@example.com"/>
```

```
                    </xcon:allowed-users-list>
                </info:users>
            </sidebarByValInfo>
        </ccmp:sidebarByValRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>
```

Figure 22: SidebarByVal Update for private messages scenario

The other context, referred to as whisper, in this document refers to situations involving one time media targetted to specific user(s). An example of a whisper would be an announcement injected only to the conference chair or to a new participant joining a conference. Please notice that such an announcement would not be conveyed by means of CCMP, while rather by means of a notification protocol related to it, e.g. a SIP event package, XMPP, or even a multimedia announcement. CCMP would only be involved with respect to the creation of an ad-hoc sidebar, as it will be clearer in the following lines.
Figure 23 (Whisper) provides an example of one user "Alice" who's chairing a fixed length conference with "Bob" and "Carol". The configuration is such that only the chair is providing a warning when there is only 10 minutes left in the conference. At that time, "Alice" is moved into a sidebar created by the conferencing system and only "Alice" receives the announcement.

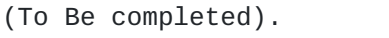


Figure 23: Whisper

---

1. When the conferencing system determines that there is only 10 minutes left in the conference which "Alice" is chairing, the conferencing system directly creates an active sidebar conference, based on the active conference associated with "Alice". This sidebar conference is NOT independent of the active conference (i.e., parent). The conferencing system also allocates a conference ID to be used for any subsequent manipulations of the sidebar conference.

2. Immediately upon creation of the active sidebar conference, the announcement media is provided to "Alice". Depending upon the policies, Alice may be notified of her addition to the sidebar via the conference notification service. "Alice" continues to receive the media from the main conference.

3. Upon completion of the announcement, "Alice" is removed from the siebar and the sidebar conference is deleted.

4. "Alice" is notified of her removal from the sidebar via the conference notification service.

6.9. Observing and Coaching

TOC

An example of observing and coaching is shown in figure Figure 25 (Supervisor Creating a Sidebar for Observing/Coaching). In this example, call center agent "Bob" is involved in a conference with customer "Carol". Since "Bob" is a new agent and "Alice" sees that he has been on the call with "Carol" for longer than normal, she decides to observe the call and coach "Bob" as necessary.
Consider the following as the conference document associated with the video conference involving Bob (the call agent) and Carol (the customer) (Figure 24 (A call-center conference object example)):

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <info:conference-info
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
                entity="xcon:8978383@example.com">
     <info:conference-description>
        <info:display-text>CUSTOMER SERVICE conference</info:display-text>
        <info:conf-uris>
            <info:entry>
               <info:uri>sip:8978383@example.com</info:uri>
               <info:display-text>conference sip uri</info:display-text>
            </info:entry>
        </info:conf-uris>
        <info:available-media>
          <info:entry label="123">
            <info:display-text>service audio</info:display-text>
            <info:type>audio</info:type>
            <info:status>sendrecv</info:status>
          </info:entry>
          <info:entry label="456">
            <info:display-text>service video</info:display-text>
            <info:type>video</info:type>
            <info:status>sendrecv</info:status>
            <xcon:controls>
                    <xcon:video-layout>single-view</xcon:video-layout>
           </xcon:controls>
          </info:entry>
        </info:available-media>
    </info:conference-description>
    <info:conference-state>
        <info:active>true</info:active>
    </info:conference-state>
    <info:users>
        <info:user entity="xcon-userid:bob@example.com">
            <info:display-text>Bob – call agent</info:display-text>
            <info:endpoint entity="sip:bob@example.com">
                <info:media id="1">
                    <info:label>123</info:label>
                    <info:status>sendrecv</info:status>
                </info:media>
                <info:media id="2">
                    <info:label>456</info:label>
                    <info:status>sendrecv</info:status>
                </info:media>
            </info:endpoint>
        </info:user>
```

```
      <info:user entity="xcon-userid:carol@example.com">
          <info:display-text>Carol - customer</info:display-text>
          <info:endpoint entity="sip:carol@example.com">
              <info:media id="1">
                  <info:label>123</info:label>
                  <info:status>sendrecv</info:status>
              </info:media>
              <info:media id="2">
                  <info:label>456</info:label>
                  <info:status>sendrecv</info:status>
              </info:media>
          </info:endpoint>
      </info:user>
  </info:users>
</info:conference-info>
```

Figure 24: A call-center conference object example

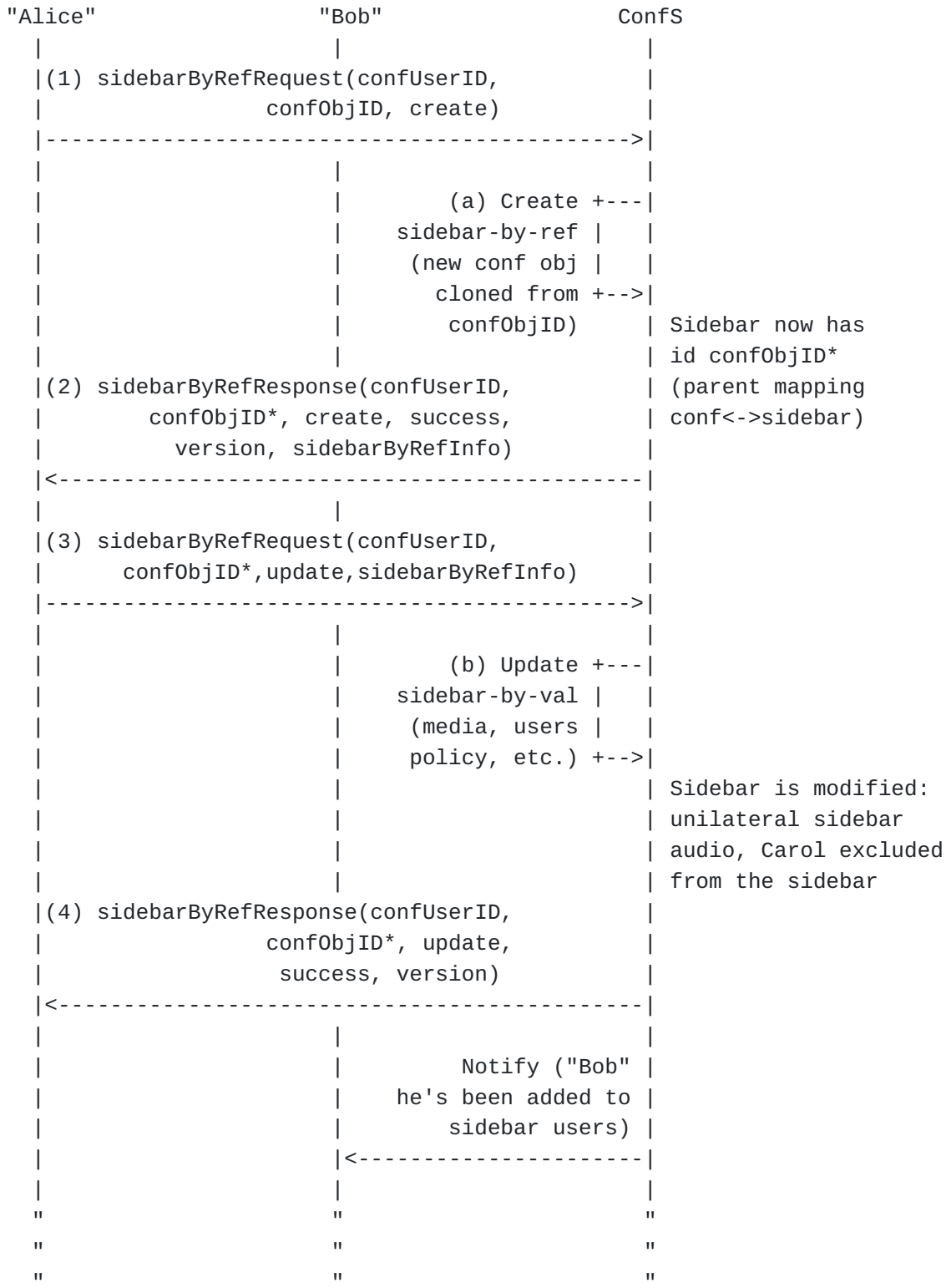


Figure 25: Supervisor Creating a Sidebar for Observing/Coaching

1. Upon receipt of the sidbarByRefRequest/create from "Alice" to "create" a new sidebar conference from the confObjID received in the request, the conferencing system uses the received active conference to clone a conference reservation for the sidebar. The conferencing system also allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the confObjID associated with the sidebar reservation through the conference instance. The conference server sends a sidebarByRefResponse message with the new confObjID and relevant confInfo.

2. Upon receipt of the confResponse message, "Alice" manipulates the data received in the confInfo in the response. "Alice" wants only "Bob" to be involved in the sidebar, thus she updates the "allowed-users-list" to include only "Bob". "Alice" also wants the audio to be received by herself and "Bob" from the original conference, but wants any outgoing audio from herself to be restricted to the participants in the sidebar, whereas "Bob's" outgoing audio should go to the main conference, so that both "Alice" and the customer "Carol" hear the same audio from "Bob". "Alice" sends a sidebarByRefRequest message with an "update" operation including the updated conference information.

3. Upon receipt of the sidbarByRefRequest message with an "update" operation, the conferencing system ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation.

4. After validating the data, the conference server sends a sidebarByRefResponse message. Based upon the addressing information provided for "Bob" by "Alice", the call signaling to add "Bob" to the sidebar with the appropriate media characteristics is instigated through the Focus. "Bob" is notified of his addition to the sidebar via the conference notification service, thus he is aware that "Alice" the supervisor is available for coaching him through this call.

1. sidebarByRefRequest/create message (Alice - the coach - creates a sidebar)

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpRequest xmlns:info="urn:ietf:params:xml:ns:conference-info"
              xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
              xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-sidebarByRef-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8978383@example.com</confObjID>
        <operation>create</operation>
        <ccmp:sidebarsByRefRequest/>
    </ccmpRequest>
</ccmp:ccmpRequest>
```

2. sidebarByRefRequest/create message ("success")

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
  <ccmp:ccmpResponse
               xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
               xmlns:info="urn:ietf:params:xml:ns:conference-info"
               xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:type="ccmp:ccmp-sidebarByref-response-message-type">
        <confUserID>xcon-userid:alice@example.com</confUserID>
        <confObjID>xcon:8971313@example.com</confObjID>
        <ccmp:response-code>success</ccmp:response-code>
        <version>1</version>
        <ccmp:sidebarByRefResponse>
            <sidebarByRefInfo entity="xcon:8971313@example.com">
                <info:conference-description>
                    <info:display-text>
                        SIDEBAR CONFERENCE registered by alice
                    </info:display-text>
                    <xcon:sidebar-parent>
                        xcon:8971313@example.com
                    </xcon:sidebar-parent>
                    <info:available-media>
                        <info:entry label="123">
                            <info:display-text>
                                 main conference audio
                            </info:display-text>
                            <info:type>audio</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                        <info:entry label="456">
                            <info:display-text>
```

```
                                main conference video
                            </info:display-text>
                            <info:type>video</info:type>
                            <info:status>sendrecv</info:status>
                        </info:entry>
                    </info:available-media>
                </info:conference-description>
                <info:conference-state>
                    <info:active>false</info:active>
                </info:conference-state>
                <info:users>
                    <xcon:allowed-users-list>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:alice@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:bob@example.com"/>
                        <xcon:target method="dial-out"
                              uri="xcon-userid:carol@example.com"/>
                    </xcon:allowed-users-list>
                </info:users>
            </sidebarByRefInfo>
        </ccmp:sidebarByRefResponse>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

3. sidebarByRefRequest/update message (Alice introduces unilateral sidebar audio and excludes Carol from the sidebar)

```
<ccmp:ccmpRequest
                xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info"
                xmlns:info="urn:ietf:params:xml:ns:conference-info"
                xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-sidebarByRef-request-message-type">
        <confUserID>xcon-userid:alice@example.com</confUserID>
        <confObjID>xcon:8971313@example.com</confObjID>
        <operation>update</operation>
        <ccmp:sidebarByRefRequest>
            <sidebarByRefInfo entity="xcon:8971313@example.com">
                <info:conference-description>
                    <info:display-text>
                        Coaching sidebar alice(supervisor) – bob(call agent)
                    </info:display-text>
                    <info:available-media>
                        <info:entry label="sidebarAudio">
                            <info:display-text>
                                 Alice-to-Bob audio
                            </info:display-text>
                            <info:type>audio</info:type>
```

```
                        <info:status>sendrecv</info:status>
                    </info:entry>
                </info:available-media>
            </info:conference-description>
            <info:conference-state>
                <info:active>false</info:active>
            </info:conference-state>
            <info:users>
                <xcon:allowed-users-list>
                    <xcon:target method="dial-in"
                          uri="xcon-userid:alice@example.com"/>
                    <xcon:target method="dial-out"
                          uri="xcon-userid:bob@example.com"/>
                </xcon:allowed-users-list>
                <user entity="xcon-userid:Alice@example.com>
                  <info:endpoint>
                    <info:media>
                     <info:label>sidebarAudio</info:label>
                     <info:status>sendonly</info:status>
                    </info:media>
                  </info:endpoint>
                </user>
                <user entity="xcon-userid:Bob@example.com>
                  <info:endpoint>
                    <info:media>
                     <info:label>sidebarAudio</info:label>
                     <info:status>recvonly</info:status>
                    </info:media>
                  </info:endpoint>
                </user>
            </info:users>
        </sidebarByRefInfo>
    </ccmp:sidebarByRefRequest>
  </ccmpRequest>
</ccmp:ccmpRequest>
```

Figure 26: Coaching and Observing Messaging details

TOC

7. Removing participants and deleting conferences

The following scenarios detail the basic operations associated with removing participants from conferences and entirely deleting conferences. The examples assume that a conference has already been correctly established, with media, if applicable, per one of the examples in Section 5 (Conference Creation).

7.1. Removing a Party

TOC

Figure 27 (Client Manipulation of Conference - Remove a party) provides an example of one client "Alice" removing another participant "Bob" from a conference. This example assumes an established conference with "Alice", "Bob", "Claire" and "Duck". In this example, "Alice" wants to remove "Bob" from the conference so that the group can continue in the same conference without "Bob"'s participation.

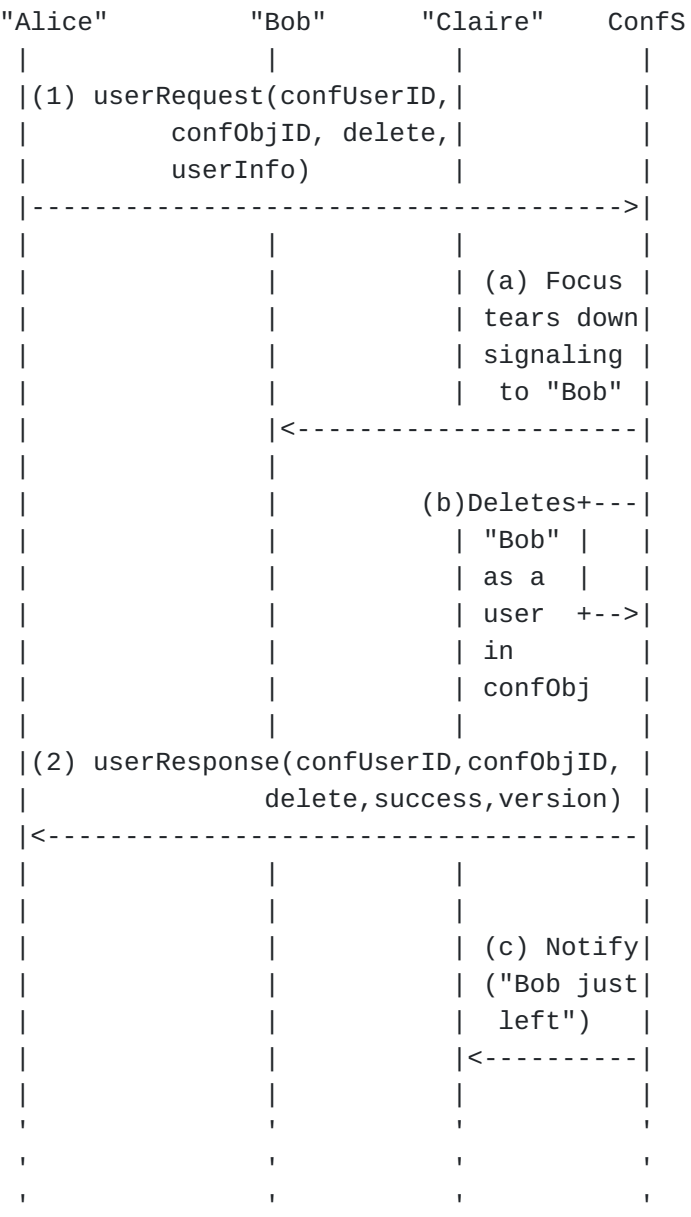


Figure 27: Client Manipulation of Conference - Remove a party

---

1. "Alice" sends a userRequest message, with a "delete" operation. The conference server ensures that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation.

2. Based upon the addressing and media information in the conference object for "Bob" in the "user" element, the conference instigates the process to remove "Bob" (e.g., the

call signaling to remove "Bob" from the conference is instigated through the Focus). The conference server updates the data in the conference object, thus removing "Bob" from the "users" list. After updating the data, the conference server sends a userResponse message to "Alice". Depending upon the policies, other participants (e.g. "Claire") may be notified of the removal of "Bob" from the conference via the Conference Notification Service.

```
1. userRequest/delete message (Alice deletes Bob):

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
       xmlns:info="urn:ietf:params:xml:ns:conference-info"
       xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
       xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-user-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>delete</operation>
        <ccmp:userRequest>
            <userInfo entity="xcon-userid:Bob@example.com"/>
        </ccmp:userRequest>
    </ccmpRequest>
</ccmp:ccmpRequest>


2. userResponse/delete message

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
       xmlns:info="urn:ietf:params:xml:ns:conference-info"
       xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
       xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-user-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
         <operation>delete</operation>
        <response-code>success</response-code>
        <version>17</version>
        <ccmp:userResponse/>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 28: Removing a Participant Messaging Details

TOC

7.2. Deleting a Conference

Details to be added.

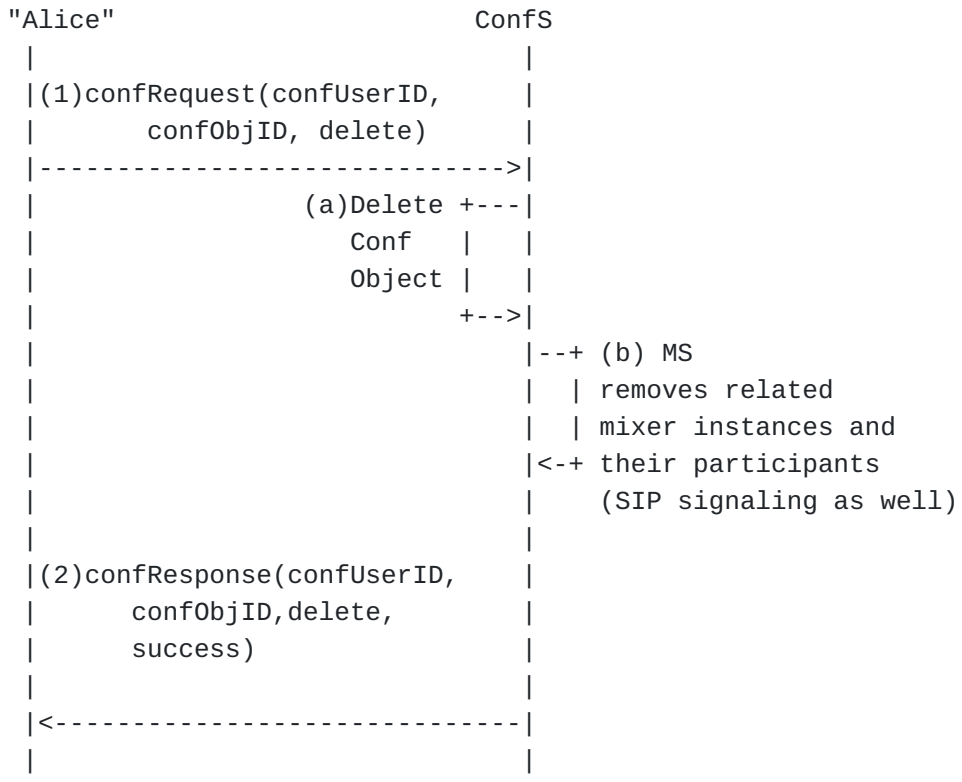


Figure 29: Deleting a conference

(Text description to be added).

```
1. confRequest/delete message (Alice deletes a conference)

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpRequest
       xmlns:info="urn:ietf:params:xml:ns:conference-info"
       xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
       xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:type="ccmp:ccmp-conf-request-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>delete</operation>
        <ccmp:confRequest/>
    </ccmpRequest>
</ccmp:ccmpRequest>


2. confResponse/delete message ("success")

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ccmp:ccmpResponse
       xmlns:info="urn:ietf:params:xml:ns:conference-info"
       xmlns:ccmp="urn:ietf:params:xml:ns:xcon:ccmp"
       xmlns:xcon="urn:ietf:params:xml:ns:xcon-conference-info">
    <ccmpResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="ccmp:ccmp-conf-response-message-type">
        <confUserID>xcon-userid:Alice@example.com</confUserID>
        <confObjID>xcon:8977794@example.com</confObjID>
        <operation>delete</operation>
        <response-code>success</response-code>
        <ccmp:confResponse/>
    </ccmpResponse>
</ccmp:ccmpResponse>
```

Figure 30: Deleting a Conference Messaging Details

---

8. Additional Conference Scenarios and Examples

TOC

The following are additional scenarios making use of the XCON framework and associated protocols. In some cases, these examples make use of

some of the building block scenarios detailed in the previous example sections, in which case the appropriate scenario is referenced rather than duplicating details. In addition, in cases where the scenarios make use of other protocols, as in the previous section, the appropriate reference in the form of a title to the specific flow in the appropriate protocol document is included.

8.1. Chat

TOC

The chat functionality described in this section of the document allows clients that use the XCON framework and protocols for other media types (e.g. voice/video) to utilize the same conference control mechanisms and conferencing system to establish, update and delete a conference instance associated with an Instant Messaging (IM) chat session, independent of the IM chat protocol. In some cases(e.g., Message Session Relay Protocol (MSRP) chat), this would provide additional capabilities, such as sidebars. This approach also allows the conferencing system to provide a natural interworking point for various IM protocols, the details of the interworking are outside the scope of this document.

An IM client wishing to join a conference uses standardized centralized conferencing mechanisms for creating and joining a conference, as identified in the previous sections. The request to send an IM to an IM media session is specific to the IM protocol (e.g., MSRP SEND), just as there is specific media control messaging for other types of sessions.

An IM client connecting to a conferencing system has a 1:1 relationship with the IM media signaling entity in the conferencing system. This relationship is referred to as an IM session. Further details of the correlation of the IM session identifiers with the XCON session identifiers is provided in [I-D.boulton-xcon-session-chat] (Barnes, M., Boulton, C., and S. Loreto, “Chatrooms within a Centralized Conferencing (XCON) System,” July 2009.). The IM media signaling entity is responsible for distribution of all the messages to the other participants.

As with the other example conferences created, each IM session is logically associated with a specific conference. The conference itself has a specific identifier in the form of the XCON-URI, which is passed in the "confObjID" element in the CCMP messages. This provides the relevant association between IM session and a centralized conference.

An IM client wishing to delete a chat room uses standardized mechanisms for deleting a conference instance, such as those detailed in Section 7.2 (Deleting a Conference).

TOC

8.1.1. Basic Chat Operations

This section provides details of the realization of the Multi-party IM (chat) within the context of the centralized conferencing framework. A brief discussion and diagrams are provided for creating, joining, and deleting a chat based conference. The discovery of chat rooms available on a specific conferencing system is inherent in the blueprint capability provided by the conferencing system. The objective of this section is to further illustrate the model, mechanisms and protocols presented in the previous sections and also serves to validate that the model, mechanisms and protocols are sufficient to support IM chat.
It should be noted that not all entities impacted by the request are shown in the diagram (e.g., Focus), but rather the emphasis is on the new entities introduced by this centralized conferencing framework.

8.1.1.1. Creating a Chat Room

TOC

There are different ways to create a conference. A participant can create a conference using call signaling means only, such as SIP, as detailed in [RFC4579] (Johnston, A. and O. Levin, “Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents,” August 2006.). For a conferencing client to have more flexibility in defining the charaterisitics and capabilities of a chat based conference, a conferencing client would implement a conference control protocol client. By using a conference control protocol, the client can determine the capabilities of a conferencing system and its various resources.
Figure 31 (Client Creation of Chat room) provides an example of one client "Alice" determining the conference blueprints available to support various types of chat rooms for a particular conferencing system and creating a chat based conference using the desired blueprint.

Details to be added.

Figure 31: Client Creation of Chat room

Upon receipt of the Conference Control Protocol request for blueprints associated with chat rooms, the conferencing system would first authenticate "Alice" (and allocate a conference user identifier, if necessary) and then ensure that "Alice" has the appropriate authority based on system policies to receive any chat room based blueprints supported by that system. Any blueprints that "Alice" is authorized to use are returned in a response, along with the conference user ID.
Upon receipt of the Conference Control Protocol response containing the blueprints, "Alice" determines which blueprint to use for the conference to be created. "Alice" creates a conference object based on the blueprint (i.e., clones) and modifies applicable fields, such as membership list, topic details, and start time. "Alice" then sends a request to the conferencing system to create a conference reservation based upon the updated blueprint.
Upon receipt of the Conference Control Protocol request to "create" a conference based upon the blueprint in the request, the conferencing system ensures that the blueprint received is a valid blueprint (i.e. the values of the various field are within range). The conferencing system determines the appropriate read/write access of any users to be added to a conference based on this blueprint (using membership, roles, etc.). The conferencing system uses the received blueprint to clone a conference reservation. The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the reservation through the conference instance.
Upon receipt of the conference control protocol response to reserve the conference, "Alice" now creates an active chat room using that reservation. "Alice" provides the conference information, including the necessary conference ID, to desired participants to allow them to join the chat room. "Alice" may also add other users to the chat room. When the first participant, including "Alice", requests to be added to the conference, an active conference and focus are created. The focus is associated with the conference ID received in the request.

(CCMP Messaging details not available yet.
Plan is to reference detailed flows in
previous sections
in the example.)

Figure 32: Chatroom Creation Messaging Details

8.1.1.2. Joining a Chat Room

TOC

A participant can join and leave the conference using call signaling means only, such as SIP. However, in order to perform richer conference control a user client can implement a conference control protocol client. By using a conference control protocol, the client can affect its own state and the state of other participants, depending upon policies, which may indirectly affect the state of any of the conference participants.

In the example in section Section 8.1.1.1 (Creating a Chat Room), "Alice" has reserved a chat room . "Alice" has also already joined the conference and made the chat room active. "Alice" can either add additional participants to the chat room or provide the conference information, including the necessary conference ID, to desired participants and allow them to request to join themselves. Any participants that have the authority to manipulate the conference would receive the conference object identifier of the active conference object in the response to their request to join.

Figure 33 (Joining a chat room) provides an example of "Bob" joining the chat room using the conference ID provided by "Alice" (e.g., in an IM).

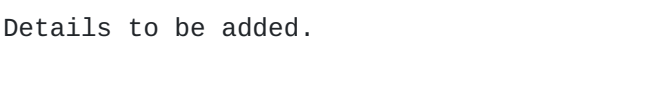


Figure 33: Joining a chat room

Upon receipt of the Conference Control Protocol request to "add" a party ("Bob") in the specific conference as identified by the conference object ID, the conferencing system must determine whether "Bob" is already a user of this conferencing system or whether he is a new user. If "Bob" is a new user for this conferencing system, a Conference User Identifier is created for Bob. The conferencing system must also ensure that "Bob" has the appropriate authority based on the policies associated with that specific conference object to perform the operation.

Once "Bob" has been successfully added to the chat room, a response is sent to "Bob". Depending upon the policies, other participants

(including "Bob") may be notified of the addition of "Bob" to the conference via the Conference Notification Service.

(CCMP Messaging details not available yet.
Plan is to reference detailed flows in
previous sections as appropriate
in the example.)

Figure 34: Chatroom Join Messaging Details

8.1.1.3. Deleting a Chat Room

TOC

Depending upon the conferencing system policies and policies specific to the chat room, the creator of the chat would typically be the participant authorized to delete the chat room.
In the example in section Section 8.1.1.1 (Creating a Chat Room), "Alice" has created a chat room and provided the conference information, including the necessary conference ID, to desired participants and allow them to request to join themselves. "Bob" and others are participants in the chat. Figure 35 (Deleting a chat room) provides an example of "Alice" later deleting this same chat room.

Details to be added.

Figure 35: Deleting a chat room

Upon receipt of the Conference Control Protocol request to "delete" the specific chat room as identified by the conference object ID, the conferencing system must determine whether "Alice" has the authority to delete this conference. Since "Alice" is the creator of the conference, the "delete" operation is performed, with the appropriate signaling sent to the participants, including a response to "Alice" indicating that the chat room has been deleted.

One step in the deletion of the chat room may include notifitying the participants (including "Bob") that they have been removed via the Conference Notification Service.

(CCMP Messaging details not available yet. Plan is to reference detailed flows in previous sections .)

Figure 36: Chatroom Deletion Messaging Details

8.1.2. Advanced Operations

TOC

This section provides details of the realization of advanced chat features, such as sidebars and private messages, within the context of the centralized conferencing framework. As with Section 8.1.1 (Basic Chat Operations), the objective of this section is to further illustrate the model, mechanisms and protocols presented in the previous sections and also serves to validate that the model, mechanisms and protocols are sufficient to support advance IM chat features.

8.1.2.1. Text Sidebar

TOC

The concept of a 'sidebar' in conferencing system is fully described in the Sidebar section and related subsections within the Conferencing Scenarios Realization section of the centralized conferencing framework document [RFC5239] (Barnes, M., Boulton, C., and O. Levin, “A Framework for Centralized Conferencing,” June 2008.). The creation, manipulation and deletion of sidebars for chat rooms follows the same principles. A conference object representing a sidebar is created by cloning the parent associated with the existing conference and updating any information specific to the sidebar. A sidebar conference object is implicitly linked to the parent conference object (i.e. it is not an independent object) and is associated with the parent conference object identifier. A conferencing system manages and enforces the parent and appropriate localized restrictions on the sidebar conference object (e.g., no members from outside the parent conference instance can join,

sidebar conference can not exist if parent conference is terminated, etc.).
Figure 37 (Client Creation of a Sidebar Conference) provides an example of one client "Alice" involved in active chat room with "Bob" and "Carol". "Alice" wants to create a sidebar to have a side discussion with "Bob" while still receiving the session based messaging associated with the main chat room. Whether the text is interleaved with the main chat or whether a separate window is created for the sidebar is implementation specific. "Alice" initiates the sidebar by sending a request to the conferencing system to create a conference chat reservation based upon the active chat conference object. "Alice" and "Bob" would remain on the roster of the main conference, such that other participants could be aware of their participation in the main conference, while the text sidebar conference is occurring.

Figure 37: Client Creation of a Sidebar Conference

Upon receipt of the Conference Control Protocol request to "reserve" a new sidebar chat conference, based upon the active chat conference received in the request, the conferencing system uses the received active chat conference to clone a conference chat reservation for the sidebar. As discussed previously, the sidebar reservation is NOT independent of the active conference (i.e., parent). The conferencing system also reserves or allocates a conference ID to be used for any subsequent protocol requests from any of the members of the conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the sidebar reservation through the conference instance.
Upon receipt of the conference control protocol response to reserve the conference, "Alice" can now create an active chat conference using that reservation or create additional reservations based upon the existing reservations. In this example, "Alice" wants only "Bob" to be involved in the sidebar, thus she manipulates the membership. "Alice" also only wants the text from the original conference, but wants the text within the sidebar to be restricted to the participants in the sidebar.
"Alice" sends a conference control protocol request to update the information in the reservation and to create an active conference.
Upon receipt of the conference control protocol request to update the reservation and to create an active chat conference for the sidebar, as identified by the conference object ID, the conferencing system ensures

that "Alice" has the appropriate authority based on the policies associated with that specific conference object to perform the operation. The conferencing system must also validate the updated information in the reservation, ensuring that a member like "Bob" is already a user of this conferencing system.
Depending upon the policies, the initiator of the request (i.e., "Alice") and the participants in the sidebar (i.e., "Bob") may be notified of his addition to the sidebar via the conference notification service.

Details to be added.

Figure 38: Chatroom Sidebar Messaging Details

8.1.2.2. Private Message

TOC

The case of private messages can be handled as a sidebar with just two participants, identical to the example in section Section 8.1.2.1 (Text Sidebar). The other context, referred to as whisper, in this document refers to situations involving one time media targetted to specific user(s). An example of a whisper would be a text message injected only to the conference chair or to a new participant joining a conference. Figure 39 (Whisper) provides an example of one user "Alice" who's chairing a fixed length conference with "Bob" and "Carol". The configuration is such that only the chair is providing a warning when there is only 10 minutes left in the conference. At that time, "Alice" is moved into a sidebar created by the conferencing system and only "Alice" receives that text message announcing the 10 minute warning.

Details to be added.

Figure 39: Whisper

When the conferencing system determines that there is only 10 minutes left in the conference which "Alice" is chairing, rather than creating a reservation as was done for the sidebar in Section 8.1.2.1 (Text Sidebar), the conferencing system directly creates an active chat sidebar conference, based on the active chat conference associated with "Alice". As discussed previously, the sidebar conference is NOT independent of the active conference (i.e., parent). The conferencing system also allocates a conference ID to be used for any subsequent manipulations of the sidebar chat conference. The conferencing system maintains the mapping between this conference ID and the conference object ID associated with the active sidebar conference through the conference instance.
Immediately upon creation of the active chat sidebar conference, the text announcement is provided to "Alice". Depending upon the policies, Alice may be notified of her addition to the sidebar via the conference notification service. "Alice" continues to receive the text messages from the main conference.
Upon delivery of the text announcement, "Alice" is removed from the sidebar and the sidebar conference is deleted. Depending upon the policies, "Alice" may be notified of her removal from the sidebar via the conference notification service.

Details to be added.

Figure 40: Chatroom Sidebar Messaging Details

---

9. IANA Considerations

TOC

This document has no IANA considerations.

---

10. Security Considerations

TOC

The security considerations applicable to the implementation of these call flows is documented in the XCON Framework, with additional security considerations documented in the CCMP document. Where applicable, statements with regards to the necessary security are discussed in particular flows, however, since this is only an

informational document, readers are strongly recommended to carefully consider the security considerations defined in the XCON Framework and the CCMP document.

---

11. Change Summary

TOC

NOTE TO THE RFC-Editor: Please remove this section prior to publication as an RFC.
The following are the major changes between the 01 and the 02 versions of the draft:

- updated the call flows in order to take into account the new versioning mechanism of the CCMP;

- clarified, per agreement in Stockholm, that cloning from a blueprint does not need a cloning-parent to be made available in the response;

- clarified that BFCP and CCMP-based media control are neither in conflict nor one the wrapper of the other; they act at different levels, and when both are involved, it is required that both grant a resource before it can be used by an interested participant;

- changed all the domains involved in the flows to make them compliant with [RFC2606] (Eastlake, D. and A. Panitz, “Reserved Top Level DNS Names,” June 1999.);

- clarified that a successful creation of a new conference object may or may not contain the whole confInfo object in the response; in case it doesn't, a retrieve of the updated object can be achieved by issuing a confRequest/retrieve;

*clarified that the scenario in Section 6.1 (Conference Announcements and Recordings) only involves CCMP in adding the user to a conference; this includes requiring the use of a password only in adding the user to the conference object; the actual request for PIN/Password when joining thw conference is handled by means of out-of-band mechanisms (in this case at the media level, with the help of the MEDIACTRL framework);

*added and corrected Sidebars-related scenarios;

*added flows for some previously missing scenarios: Private Message/Whisper, Coaching Scenario, Removing a Party, Deleting a Conference;

*

The following are the major changes between the 00 and the 01 versions of the draft:

*Updates to reflect change of CCMP to HTTP transport model.

The following are the major changes between the individual 01 version to the WG 00:

*Updates to reflect most recent version of CCMP, including parameter names, etc.

*Added protocol details to many of the examples.

*Editorial: Simplifying intro, terms, etc.

12. Acknowledgements

TOC

The detailed content for this document is derived from the prototype work of Lorenzo Miniero, Simon Pietro-Romano, Tobia Castaldi and their colleagues at the University of Napoli.

---

13. References

TOC

13.1. Normative References

TOC

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC5239]	Barnes, M., Boulton, C., and O. Levin, "A Framework for Centralized Conferencing," RFC 5239, June 2008 (TXT).
[I-D.ietf-xcon-ccmp]	Barnes, M., Boulton, C., Romano, S., and H. Schulzrinne, "Centralized Conferencing Manipulation Protocol," draft-ietf-xcon-ccmp-07 (work in progress), April 2010 (TXT).

13.2. Informative References

TOC

[RFC2606]	Eastlake, D. and A. Panitz, "Reserved Top Level DNS Names," BCP 32, RFC 2606, June 1999 (TXT).
[RFC3261]	Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261, June 2002 (TXT).
[RFC4579]	Johnston, A. and O. Levin, "Session Initiation Protocol (SIP) Call Control - Conferencing for User Agents," BCP 119, RFC 4579, August 2006 (TXT).
[RFC4597]	Even, R. and N. Ismail, "Conferencing Scenarios," RFC 4597, August 2006 (TXT).
[RFC4582]	Camarillo, G., Ott, J., and K. Drage, "The Binary Floor Control Protocol (BFCP)," RFC 4582, November 2006 (TXT).
[I-D.ietf-xcon-event-package]	Camarillo, G., Srinivasan, S., Even, R., and J. Urpalainen, "Conference Event Package Data Format Extension for Centralized Conferencing (XCON)," draft-ietf-xcon-event-package-01 (work in progress), September 2008 (TXT).

[I-D.ietf-xcon-common-data-model]	Novo, O., Camarillo, G., Morgan, D., and J. Urpalainen, "Conference Information Data Model for Centralized Conferencing (XCON)," draft-ietf-xcon-common-data-model-18 (work in progress), February 2010 (TXT).
[I-D.ietf-mediactrl-call-flows]	Amirante, A., Castaldi, T., Miniero, L., and S. Romano, "Media Control Channel Framework (CFW) Call Flow Examples," draft-ietf-mediactrl-call-flows-03 (work in progress), February 2010 (TXT).
[RFC5567]	Melanchuk, T., "An Architectural Framework for Media Server Control," RFC 5567, June 2009 (TXT).
[I-D.ietf-mediactrl-mixer-control-package]	McGlashan, S., Melanchuk, T., and C. Boulton, "A Mixer Control Package for the Media Control Channel Framework," draft-ietf-mediactrl-mixer-control-package-11 (work in progress), February 2010 (TXT).
[I-D.boulton-xcon-session-chat]	Barnes, M., Boulton, C., and S. Loreto, "Chatrooms within a Centralized Conferencing (XCON) System," draft-boulton-xcon-session-chat-04 (work in progress), July 2009 (TXT).

Authors' Addresses

TOC

	Mary Barnes
	Nortel
	2201 Lakeside Blvd
	Richardson, TX
Email:	mary.barnes@nortel.com
	Chris Boulton
	NS-Technologies
Email:	chris@ns-technologies.com
	Lorenzo Miniero
	Meetecho
	Via Carlo Poerio 89/a
	Napoli 80121
	Italy
Email:	lorenzo@meetecho.com
	Roberta Presta
	University of Napoli
	Via Claudio 21
	Napoli 80125
	Italy
Email:	roberta.presta@unina.it

	Simon Pietro Romano
	University of Napoli
	Via Claudio 21
	Napoli 80125
	Italy
Email:	spromano@unina.it