Implementing XMPP Multi-User Chat for UPnP Cloud

Document Date: December 31, 2015

© 2015 UPnP Forum. All rights reserved.
The UPnP® Word Mark and UPnP® Logo are certification marks owned by UPnP Forum.
CONTENTS

1 Introduction to Multi-User Chat (MUC) ................................................................. 3
  1.1 Audience .................................................................................................................. 3
    1.1.1 Acronyms ............................................................................................................ 3
    1.1.2 General Cloud Terms and Definitions ............................................................... 4
    1.1.3 Device and Control Point Terms and Definitions .................................................. 4
    1.1.4 Multi-User Chat Terms and Definitions ............................................................... 4
  1.2 Notation .................................................................................................................... 5
  1.3 The MUC Scenarios .................................................................................................. 6

2 The MUC Protocol ........................................................................................................... 9
  2.1 MUC Rooms ............................................................................................................. 9
    2.1.1 MUC Room Creation ......................................................................................... 9
    2.1.2 Configuring a MUC Room .................................................................................. 11
    2.1.3 MUC Room Invitations and Acceptance ............................................................ 13
  2.2 The MUC Session ..................................................................................................... 17
    2.2.1 Description ......................................................................................................... 17
    2.2.2 Control ............................................................................................................... 19
    2.2.3 Eventing ............................................................................................................. 20
  2.3 MUC with Other Users ............................................................................................ 21

3 Other Considerations ..................................................................................................... 23
  3.1.1 Room Maintenance ............................................................................................. 23
  3.1.2 Room Moderation ............................................................................................... 23
  3.1.3 Room Teardown .................................................................................................. 23
  3.1.4 Roles and Affiliations ......................................................................................... 24

4 Summary of Suggested Capabilities ............................................................................. 25

5 References ..................................................................................................................... 25

Figure 1-1: Single User MUC Scenario (jeffrey@mycloud.org) ........................................... 7
Figure 1-2: Multi User MUC Scenario (jeffrey@mycloud.org, mary@theircloud.org) ............. 8

Table 1-1: Acronyms ........................................................................................................ 3
Table 2-1: Room Configuration and Suggested Defaults ..................................................... 12
Table 3-1: Roles and Associated Privileges ........................................................................ 24
Table 3-2: Affiliations and Associated Privileges ............................................................... 24
Table 4-1: MUC Stanza Summary for UCC-CP and UCCD .................................................. 25
1 Introduction to Multi-User Chat (MUC)

This document describes an application of XMPP Multi-User Chat protocol for devices and control points supporting UPnP 2.0 UCA capabilities. Multi-User Chat or MUC is a well-known application of XMPP. Originally named "groupchat" when XMPP was still "Jabber", it is now a standards track XEP of the XMPP Standards Foundation, specifically [XEP0045]. A natural extension of presence and instant messaging, the typical use case enables the following scenario:

- let's discuss a specific subject, or "chat",
- by creating a digital (XMPP) meeting place or "Room" for the chat,
- and inviting interested "users" to the Room for the chat,
- while moderating the "chat", if desired, including the teardown of the Room.
- In the scenario above, the "chat" typically involves a broadcast of textual information - in a <message> stanza of type "groupchat" - to all occupants of a Room via sending the stanza directly to the Room JID.

Although, the specific chat messaging is not central to the UPnP™ Cloud use-case the mechanisms for managing Rooms and Room invitations can be leveraged to enable compelling experiences such as virtual sharing Rooms, or - to recast the above scenario -

- let's share some family videos,
- by creating a virtual "Family" Room for the sharing,
- and inviting two UCA TVs (UCCD MediaRenderers [MR3]) and a NAS (UCCD MediaServer [MS4]) to the "Family" Room,
- and playing the content with a UCC-CP (the moderator) to both TVs from the NAS.
- Thus, the purpose of this document is to show the details of how UCA and XMPP MUC can be combined to realize the virtual Room sharing scenario.

1.1 Audience
The reader is assumed to be familiar with UPnP Device Architecture 2.0 [UDA] and specifically Annex C or "UPnP Cloud". The following Acronyms, Terms and Definitions also apply:

1.1.1 Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MUC</td>
<td>Multi-User Chat</td>
</tr>
<tr>
<td>UCA</td>
<td>UPnP Cloud Annex</td>
</tr>
<tr>
<td>UCC</td>
<td>UPnP Cloud Capable</td>
</tr>
<tr>
<td>UCC-CP</td>
<td>UPnP Cloud Capable Control Point</td>
</tr>
<tr>
<td>UCCD</td>
<td>UPnP Cloud Capable Device</td>
</tr>
<tr>
<td>UCCD-M</td>
<td>UPnP Cloud Capable Device [Mobile]</td>
</tr>
<tr>
<td>UCS</td>
<td>UPnP Cloud Server</td>
</tr>
<tr>
<td>XMPP</td>
<td>Extensible Messaging and Presence Protocol</td>
</tr>
</tbody>
</table>
1.1.2 General Cloud Terms and Definitions

Cloud, in the context of UPnP, is the logical domain, not in the user's home(s), where their UCODs, UCCDs and UCC-CPs connect, their UCBSs execute, and their cloud based content resides.

Domain is a scoped access to a subset of all available cloud devices, services and users.

Account is a domain in the cloud consisting of a username and a credential. Also, the account can contain ancillary information such as address, telephone number, email information and possibly transactional information such as credit card information. An extended concept of an account is the combination of devices, services and users registered or interacting with the account.

User, in the context of UPnP cloud, is a uniquely identifiable participant that interacts with the UPnP cloud ecosystem. A user can be an account owner or participant and can be associated with multiple Cloud Accounts.

Login is an identification process that allows a specific user to access their Cloud Account by confirming their username and credential. Login can also refer to the act of starting an active session with the Cloud Account. The login can be automated once initial login succeeds. Some minimal behavior equivalent to UPnP Public Role [DP] could be identified.

Owner is a user that has management rights over an account (or group) and the devices, services, and users allowed access within that account.

Home, in the context of UPnP cloud, is the logical network(s) or LAN(s) where a user’s UHOD and CPDevs are connected.

UPnP Cloud Capable (UCC) means that the device or control point (CP) is capable of interaction with UPnP Cloud Based Services.

Invitation is the initiation part of registering a device, service, or user to a Cloud Account or cloud group.

Invited User is a user from a Cloud Account that has been invited to have access to devices and services in a different Cloud Account. The devices and services access can be granted with a per-device, per-group, or per-service granularity.

1.1.3 Device and Control Point Terms and Definitions

UPnP Cloud Capable Control Point (UCC-CP) is a control point (CP) that can interact with UCCDs, UCODs, and UCOSs directly. Note that a UCC-CP is not a UCCD.

Invited Device is a device from a Cloud Account that has received an invitation to be registered to a different Cloud Account.

1.1.4 Multi-User Chat Terms and Definitions

The following are from [XEP0045] and have been localized for UCA usage.

Affiliation - A long-lived association or connection with a Room; the possible Affiliations are "owner", "admin", "member", and "outcast" (naturally it is also possible to have no Affiliation); Affiliation is distinct from Role. An Affiliation lasts across a user’s visits to a Room.

MUC is the Multi-User Chat Protocol defined by [XEP0045].
**Role** - is a temporary position or privilege level within a **Room**, distinct from a user's long-lived **Affiliation** with the Room; the possible **Roles** are "moderator", "participant", and "visitor" (it is also possible to have no defined **Role**. A **Role** lasts only for the duration of an occupant's visit to a **Room**. Note that this **Role** is distinct from Role as defined by [DP].

**Room** is a virtual space that users (and device) figuratively enter in order to participate in real-time, text-based conferencing (and in the UCA case data sharing) with other users (UCC-CPs and UCCDs).

**Room ID** The localpart of a **Room JID**, which might be opaque and thus lack meaning for human users; contrast with **Room Name**.

**Room JID** The <Room@service> address of a **Room**.

**Room Name** A user-friendly, natural-language name for a **Room**, configured by the **Room** owner and presented in Service Discovery queries; contrast with **Room ID**.

**Room Nickname** The resourcepart of an Occupant **JID**; this is the "friendly name" by which an occupant is known in the **Room**.

### 1.2 Notation

For readability purposes this document uses fonts as in [UDA] to indicate specific protocol components or:

- Strings that are to be taken literally are enclosed in “double quotes”.
- Words that are emphasized are printed in *italic*.
- Keywords that are defined by the UPnP Working Committee are printed using the **forum** character style.
- Keywords that are defined by the UPnP Device Architecture are printed using the **arch** character style.
- Keywords that are defined specific to the UPnP Device Architecture Annex C are printed using **UCA** character style.
- Keywords that are defined specific to XMPP are printed using **XMPP** character style.
- Keywords that are defined by a vendor are printed in **Vendor** style.
- A double colon delimiter, "::", signifies a hierarchical parent-child (parent::child) relationship between the two objects separated by the double colon. This delimiter is used in multiple contexts, for example: Service::Action(), Action()::Argument, parentProperty::childProperty.
- Example text will be in **green courier new** with a shaded background (not necessarily gray).

Additionally, clarifying text is sometimes included. There are two general forms:

1) Extracts from relevant XMPP specification indicated by an XEP or RFC reference followed by text enclosed in a box (see below).

```
From XEP or RFC
Extracted relevant text from XEP or RFC goes here.
```

2) Implementation warnings shown as a "caution sign symbol" with explanatory text (see below).

---

1 Note: example text can contain whitespace and line feeds to improve readability.
1.3 The MUC Scenarios

In this document, two general scenarios are described along with their suggested implementation. In the first scenario, only UCCDs and UCC-CPs from the same Cloud Account (jeffrey@mycloud.org) interact in a Room; in the second scenario, UCCDs and UCC-CPs, from a different, second account (mary@theircloud.org), are added to the Room. In both cases, the basic XMPP MUC mechanisms will remain the same.

In the single user scenario the basic MUC call flow is as follows: 1) Discovery a MUC Service, 2) Create a Room, 3) Configure the Room, 4) Send invitations to the UCCD (or UCC-CPs) needed in the Room, 5) Accept the invitations (with PASSNORD), 6) Conduct UPnP sessions within the Room via participant-to-participant <iq> stanzas, 7) send events to the Room via <message> stanzas of type "groupchat", 8) Tear-down (or Destroy) the Room if the session is over and not intended to be a persistent experience.

In the multi-user scenario, the main difference is that during the invitation steps, a UCC-CP of the other user is invited to the Room and then given additional privileges. This allows that UCC-CP to then invite its own UCCDs to the Room.

The suggested implementations will also take a conservative approach towards security and privacy. This is achieved primarily by limiting Room privileges and device nicknaming. The two scenarios are illustrated at a high level in Figure 1-1 and 1-2 respectively. For illustration purposes it is assumed that the MUC Service is co-located with the UCS.
Implementing MUC for UPnP Cloud

Figure 1-1: Single User MUC Scenario (jeffrey@mycloud.org)
Figure 1-2: Multi User MUC Scenario (jeffrey@mycloud.org, mary@theircloud.org)
2 The MUC Protocol

MUC is an add-on Service usually supported on the UCS. It can be discovered via XMPP Service Discovery [XEP-0030] as an `<identity>` response with a category of "conference", type of "chat", and with an associated feature var of "http://jabber.org/protocol/muc". A typical response element has the form shown below.

```
<identity category="conference" type="chat" name="ChatRoom"/>
<feature var="http://jabber.org/protocol/muc"/>
```

It can be used to create MUC Rooms which are somewhat analogous, from a UPnP perspective to a localized network where access can be controlled and both unicast and multicast type messaging are supported and, from an XMPP perspective, as an limited Roster of UCCDs and UCC-CPs authorized for the specific usage of the Room.

2.1 MUC Rooms

MUC Rooms can be configured to have many layers of privacy and security. They can be:

- "hidden" or "public",
- "members-only" or "open",
- "password-protected" or "unsecured",
- "temporary" or "persistent"
- "moderated" or "unmoderated"
- "semi-anonymous" or "non-anonymous"

Rooms have a Bare JID identity or Room ID which will reside on the domainpart supporting the MUC Service. For example, the jeffrey Cloud Account, could be located at ChatRoom@conference.mycloud.org, as Bare JID: virtual-family-Room@conference.mycloud.org, while occupants of the Room are identified as Full JIDs attached to the Room, for example:

```
virtual-family-Room@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:randomstring
```

2.1.1 MUC Room Creation

A Room can be created by simply sending a `<presence>` stanza to the MUC Service with Room Name (localpart) and Occupant Name (resourcepart) of the form shown below.

```
<presence to="RoomNamelocalpart@MUCdomainpart/OccupantName(resourcepart)"
<x xmlns="http://jabber.org/protocol/muc"/>
</presence>
```

The MUC protocol requires that the Room Name and Occupant Name (aka NickName) also be unique to the MUC Service instant. Therefore, it is suggested that they both contain some randomization or uniqueness component plus a human readable component (in the case of UCC-CPs and UCCDs the UPnP schema urn plus the device or control point type string). The design principle behind the naming suggestion is to keep some degree of autonomy between the Full JID of the UCC-CP or UCCD and its Room persona.

---

2 The conference.upnpcloud.org JID is discovered initially with #item requests to the UCS.
In the example below, Jeffrey's UCC-CP (jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8) requests creation of a MUC Room with Name (localpart):

```
virtual-family-Room-4hQjleTZyx
```

and semi-autonomous Occupant Name with resourcepart of:

```
urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd.
```

UCC-CP jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8 Sends:

```
<presence
to="virtual-family-Room-4hQjleTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
xmlns="http://jabber.org/protocol/muc">
  <x xmlns="http://jabber.org/protocol/muc#user">
    <item affiliation="owner" role="moderator" jid="jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
      xmlns="http://jabber.org/protocol/muc#user" />
    <status code="110" />
  </x>
</presence>
```

Upon successful creation of the Room, a <presence> stanza is received from the newly created Room with an <item> and one or more <status> elements. The response is of the form shown below.

```
<presence
from="RoomName@MUCdomainpart/OccupantName@resourcepart"
to="FullJID@MUCdomainpart/FullJID@resourcepart"
xmlns="http://jabber.org/protocol/muc#user">
  <item affiliation="owner" role="moderator" jid="FullJID@MUCdomainpart/FullJID@resourcepart" />
  <status code="110" />
  <status code="XXX" />
</presence>
```

The <item> element indicates the JID of the Room Occupant (UCC-CP), its Affiliation (#owner) and Role (#moderator); the <status> element indicates code 110 - "Room Created". See [XEP0045] section 5 for more details on Affiliations and Roles and section 15.6.2 for the registered <status> codes.

The specific response for the example is:

```
<presence
from="virtual-family-Room-4hQjleTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
xml:lang="en"
to="jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8"
xmlns="http://jabber.org/protocol/muc#user">
  <item affiliation="owner" role="moderator" jid="jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd" />
  <status code="110" />
</presence>
```
2.1.2 Configuring a MUC Room

When a Room is created it is created with a default configuration. The default configuration can be determined by the Room owner by sending an <iq> stanza of type "get" to the Room Bare JID with the <query> element. The stanza form is shown below.

```
<iq type="get"
  id="vendor defined value"
  to="RoomNamelocalpart@MUCdomainpart">
  <query xmlns="http://jabber.org/protocol/muc"/>
</iq>
```

In response, the MUC Service will return a FORM indicating the configuration features supported and the current and or allowed values. The response stanza will be in the form shown below.

```
<x xmlns="jabber:x:data"
  type="form"
  title="Configuration of Room RoomNamelocalpart@MUCdomainpart">
  <field type="hidden"
    var="FORM_TYPE">
    <value>http://jabber.org/protocol/muc#Roomconfig</value>
  </field>
  <field type="text-single"
    label="Room title"
    var="muc#Roomconfig_Roomname">
    <value/>
  </field>
  <field type="list-single"
    label="Maximum Number of Occupants"
    var="muc#Roomconfig_maxusers">
    <option label="10">
      <value>10</value>
    </option>
    <option label="20">
      <value>20</value>
    </option>
    <option label="50">
      <value>50</value>
    </option>
  </field>
</x>
```

The configuration of the Room can be modified by sending an <iq> stanza of type "set" to the Room Bare JID. It is suggested that the MUC support the configuration of the following parameters with the suggested default values.
### Table 2-1: Room Configuration and Suggested Defaults

<table>
<thead>
<tr>
<th>&lt;Field&gt;::&lt;var&gt;</th>
<th>Suggested Default Value</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>muc#Roomconfig_Roomname</td>
<td>-</td>
<td>Vendor/User defined according to use case</td>
</tr>
<tr>
<td>muc#Roomconfig_Roomdesc</td>
<td>-</td>
<td>Vendor/User defined according to use case</td>
</tr>
<tr>
<td>muc#Roomconfig_changesubject</td>
<td>0</td>
<td>Owner sets &lt;Subject&gt; according to use case or leaves blank</td>
</tr>
<tr>
<td>muc#Roomconfig_allowinvites</td>
<td>0</td>
<td>Moderators only</td>
</tr>
<tr>
<td>muc#Roomconfig_allowpm</td>
<td>anyone</td>
<td>allows direct &lt;iq&gt; and &lt;message&gt; exchanges between Room participants, this is needed for UPnP UCA description and control</td>
</tr>
<tr>
<td>muc#Roomconfig_presencebroadcast</td>
<td>participant</td>
<td>Enables discovery as described in UCA</td>
</tr>
<tr>
<td>muc#Roomconfig_getmemberlist</td>
<td>participant</td>
<td>allows UCC-CP or UCCD to query Room occupants if state is lost</td>
</tr>
<tr>
<td>muc#Roomconfig_publicRoom</td>
<td>0</td>
<td>Use case dependent, allows Room to be discovered but not necessarily entered by any user connected to the MUC service</td>
</tr>
<tr>
<td>muc#Roomconfig_persistentRoom</td>
<td>1</td>
<td>Use case dependent. Avoids having to reconfigure Room but requires clean up if Room no longer needed</td>
</tr>
<tr>
<td>muc#Roomconfig_moderatedRoom</td>
<td>0</td>
<td>Reflects typical UPnP use cases, ensures all UCC-CPs and UCCDs in the Room can communicate by not requiring Voice to send information</td>
</tr>
<tr>
<td>muc#Roomconfig_membersonly</td>
<td>1</td>
<td>Use case dependent, needed for Password protected Room</td>
</tr>
<tr>
<td>muc#Roomconfig_passwordprotectedRoom</td>
<td>1</td>
<td>Makes Room secure by default</td>
</tr>
<tr>
<td>muc#Roomconfig_Roomsecret</td>
<td>PASSWORD</td>
<td>Vendor/User defined</td>
</tr>
<tr>
<td>muc#Roomconfig_whois</td>
<td>moderator</td>
<td>Defines who can discover real JIDs</td>
</tr>
<tr>
<td>muc#maxhistoryfetch</td>
<td>0</td>
<td>Retrieval of history generally not needed, would generate extra traffic</td>
</tr>
<tr>
<td>muc#Roomconfig_Roomadmins</td>
<td>-</td>
<td>use case dependent but typically no additional admins are needed</td>
</tr>
<tr>
<td>muc#Roomconfig_Roomowners</td>
<td>-</td>
<td>use case dependent but typically no additional owners are needed</td>
</tr>
</tbody>
</table>

Continuing the example, the Room owner sends a configuration request to the Room Bare JID to make the Room more secure, as well as, persistent; this includes a request to make the Room Password protected with a PASSWORD of "imapassword". The example request follows.
Implementing MUC for UPnP Cloud

UCC-CP jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8
Sends:

<iq id="setconfig1"
to="virtual-family-Room-4hQjleTzyx@conference.mycloud.org"
type="set">
  <query xmlns="http://jabber.org/protocol/muc#owner">
    <x xmlns="jabber:x:data" type='submit'>
      <field var="FORM_TYPE">
        <value>http://jabber.org/protocol/muc#Roomconfig</value>
      </field>
      <field var="muc#Roomconfig_publicRoom">
        <value>0</value>
      </field>
      <field var="muc#Roomconfig_persistentRoom">
        <value>1</value>
      </field>
      <field var="muc#Roomconfig_membersonly">
        <value>1</value>
      </field>
      <field var="muc#Roomconfig_passwordprotectedRoom">
        <value>1</value>
      </field>
      <field var="muc#maxhistoryfetch">
        <value>0</value>
      </field>
      <field var="muc#Roomconfig_Roomsecret">
        <value>imapassword</value>
      </field>
    </x>
  </query>
</iq>

Upon success, the response is as follows:

<iq from="virtual-family-Room-4hQjleTzyx@conference.mycloud.org"
type="result"
to="jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8"
id="setconfig1">
  <query xmlns="http://jabber.org/protocol/muc#owner"/>
</iq>

Note that the response is from the Room JID to the Full JID of the UCC-CP. Note also that when multiple configuration fields are sent in a single stanza that an "error" will not distinguish which field failed, therefore it is probably a sound strategy to send the request in a more granular fashion.

To avoid frequent resetting of the Room occupants and loss of Roles, implementers are encouraged to make Room presence as long lived as possible but also minimize "ghost" occupants.

2.1.3 MUC Room Invitations and Acceptance

Once the Room has been configured, the next step is to populate the Room with participants, that is, other UCCDs (and perhaps UCC-CPs). The preferred method is for the UCC-CP (owner) to send a Room invitation to the participant. The invitation will occur in two parts: 1) a <message> stanza invitation request from the owner to the Room, and 2) a modified <message> stanza invitation from the Room to the requested participant. The form of the first part is shown below.

<message>
Implementing MUC for UPnP Cloud

Note that the message type needs to be "normal" or undefined. The second part of the invitation is an augmentation, by the Room, of the original invite with the PASSWORD of the Room added in a <password> element and, optionally, some additional information provided in a <body> element. The form of the second part is shown below.

The invitation is accepted by the participant (UCC-CP or UCCD) by sending a <presence> stanza to the Room JID with the <password> element. The stanza form is show below.

An example invitation sequence follows.

Jeffrey's UCC-CP with MUC admin Role sends an invite to a Room JID requesting a known MediaRenderer belonging to jeffrey@mycloud.org.

UCC-CP jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8 Sends:

```
<message
  id="invite1"
  type="normal"
```
The Room modifies the request to include the Room PASSWORD and forwards it to the MediaRenderer UCCD; a human readable <body> element is also added composed autonomously by the Room.

The MediaRenderer UCCD accepts the invitation by sending a <presence> stanza to the Room JID with the provided PASSWORD. Depending on the use case, the request can be vetted by a human user (via a User Interface) prior to accepting the invite. The UCCD uses a shortened JID to indicate its UCC DeviceType, the User component of the UCCD is hidden from Room participants and known only by the "owner/moderator".

UCC-CP jeffrey@mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:e70e9d0e-d9eb-4748-b163-636a323e7950 Sends:

The UCCD (MediaServer) receives <presence> from the UCC-CP and itself (with status) from the Room.

Implementing MUC for UPnP Cloud
The UCC-CP receives `<presence>` from the UCCD (MediaServer) with status for the Room as well.

Note that the received presence is from the Room Occupant JIDs. To complete the initial scenario Jeffrey's UCC-CP also successfully invites a MediaRenderer to virtual-family-Room-4hQj1eTzyx@conference.mycloud.org.

The Room virtual-family-Room-4hQj1eTzyx@conference.mycloud.org now contains the semi-anonymous UPnP participants:
Implementing MUC for UPnP Cloud

- **UCC-CP**: urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd
- **UCCD**: urn:schemas-upnp-org:device:MediaRenderer:3:mPEPPiqIH4
- **UCCD**: urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4

which correspond, respectively, to their **Full JID** versions:

- jeffrey@mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ad93e8f5-634b-4123-80ca-225886a5c0e8
- jeffrey@mycloud.org/urn:schemas-upnp-org:device:MediaRenderer:3:88509d0e-e8f5-80ca-4123-225886a50ee7
- jeffrey@mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:e70e9d0e-d9eb-4748-b163-636a323e7950

2.2 The MUC Session

Once the desired devices are members of the **Room** normal UPnP Cloud sharing can occur, that is, device interaction involving UPnP UCA description, eventing, and control. Since the **Room** has been configured to support **muc#Roomconfig_allowpm** with "anyone", **<iq>** and **<message>** stanzas can be exchanged between the participants in a unicast fashion (although they will be relayed through the **Room**), as well as, a multicast fashion using **<message>** stanzas of **type** "groupchat"

2.2.1 Description

DDD and SCPD descriptions are exchanged via **Room JID** to **Room JID** **<iq>** stanzas. The format is the same as described in [UDA] 2.0 section C.6.6 with the exception of the addressing. The **<iq>** stanza is of the form shown below.

```
<iq id="vendor defined value"
   to="RoomName@MUCdomainname/OccupantName/resourcepart of UCCD"
   from="RoomName@MUCdomainname/OccupantName/resourcepart of UCC-CP"
   type="get">
   <query xmlns="urn:schemas-upnp-org:cloud-1-0"
      type="description"
      name="value of uuid"/>
</iq>
```

The control point from the continuing example requests the MediaServer:4 DDD.

**UCC-CP** virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/
urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd

**Sends:**

```
<iq id="get-MediaServer-ddd"
   to="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4"
   type="get">
   <query xmlns="urn:schemas-upnp-org:cloud-1-0"
      type="description"
      name="uuid:e70e9d0e-d9eb-4748-b163-636a323e7950"/>
</iq>
```

Note that the **<iq>** stanza is relayed through the **Room**.

The UCCD sends the response as described in the form below.

```
<iq id="vendor defined value"
   from="RoomName@MUCdomainname/OccupantName/resourcepart of UCCD"
   to="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4"
   type="get">
   <query xmlns="urn:schemas-upnp-org:cloud-1-0"
      type="description"
      name="uuid:e70e9d0e-d9eb-4748-b163-636a323e7950"/>
</iq>
```
Continuing the previous example, Jeffrey's MediaServer UCCD sends the following response.

UCCD virtual-family-Room-4hQjleTzyx@conference.mycloud.org/
urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4
Sends:

```xml
<iq id="get-MediaServer-ddd"
  from="virtual-family-Room-4hQjleTzyx@conference.mycloud.org/
  urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4"
  to="virtual-family-Room-4hQjleTzyx@conference.mycloud.org/
 urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
  type="result">
  <query xmlns="urn:schemas-upnp-org:cloud-1-0"
    type="described"
    name="uuid:e70e9d0e-d9eb-4748-b163-636a323e7950"/>
  <root xmlns="urn:schemas-upnp-org:device"
    configId="2">
    <specVersion>
      <major>1</major>
      ...
    </specVersion>
    <device>
      <deviceType>urn:schemas-upnp-org:device:MediaServer:4<deviceType>
      ...
      <iconList>
      ...
    </iconList>
    <serviceList>
      ...
    </serviceList>
  </device>
</root>
<scpd xmlns="urn:schemas-upnp-org:service-1-0"
  configId="2">
  <specVersion>
    <major>4</major>
    ...
  </specVersion>
  <actionList>
    <action>
    ...
  </actionList>
</scpd>
...
<scpd xmlns="urn:schemas-upnp-org:service-1-0"
  configId="2">
  <specVersion>
    <major>3</major>
    ...
  </specVersion>
```

Implementing MUC for UPnP Cloud
2.2.2 Control

Control in the MUC session requires an addressing modification as described for MUC Description and is otherwise the same as described in [UDA] 2.0 section C.8. That is the UCC-CP sends an <iq> request to the UCCD Room JID with the form shown below.

```xml
<iq id='vendor defined value'
to='RoomName:localpart@MUC:domainpart/OccupantName:resourcepart of UCCD'
from='RoomName:localpart@MUC:domainpart/OccupantName:resourcepart of UCC-CP'
type='set'>
  <s:Envelope
    xmlns:s='http://schemas.xmlsoap.org/soap/envelope'/
    s:encodingStyle='http://schemas.xmlsoap.org/soap/encoding'/>
  <s:Header mustUnderstand='1'>
    <uc xmlns='urn:schemas-upnp-org:cloud-1-0' serviceId='serviceId'/>
  </s:Header>
  <s:Body>
    <u:actionName
      xmlns:u='urn:schemas-upnp-org:service:serviceType:1'>
      <argumentName in arg value</argumentName>
    </u:actionName>
  </s:Body>
</s:Envelope>
```

The reply is identical to the "result" and "error" response described in [UDA] 2.0 section C.8 with the exception of modified addressing; that is, the response is sent to the UCC-CP Room JID of the UCC-CP. An example MUC based Control <iq> Request and Response is shown below.

**UCC-CP virtual-family:Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd** Sends:

```xml
<iq
to='virtual-family:Room-4hQj1eTZyx@conference.mycloud.org/
  urn:schemas-upnp-org:device:MediaRenderer:3:mPPPPqIH4'
id='cp-1-soap-action-1'
type='set'>
  <s:Envelope
    xmlns:s='http://schemas.xmlsoap.org/soap/envelope'/
    s:encodingStyle='http://schemas.xmlsoap.org/soap/encoding'/>
  <s:Header mustUnderstand='1'>
    <uc xmlns='urn:schemas-upnp-org:cloud-1-0'
        serviceId='RenderingControl'/>
  </s:Header>
  <s:Body>
    <u:SetVolume
      xmlns:u='urn:schemas-upnp-org:service:RenderingControl:3'>
      <DesiredVolume>20</DesiredVolume>
    </u:SetVolume>
  </s:Body>
</s:Envelope>
```
Implementing MUC for UPnP Cloud

The UCCD responds with a SOAP action or SOAP error response as defined in [UDA] 2.0 section C.8.

In the response below the Action is successful.

UCCD virtual-family-Room-4hQjleTzYx@conference.mycloud.org/
urn:schemas-upnp-org:device:MediaRenderer:3:mPEPPIqIH4
Sends:

<iq
from="virtual-family-Room-4hQjleTzYx@conference.mycloud.org/
urn:schemas-upnp-org:device:MediaRenderer:3:mPEPPIqIH4"
to="virtual-family-Room-4hQjleTzYx@conference.mycloud.org/
urn:schemas-upnp-org:device:MediaRenderer:3:mPEPPIqIH4"
id="cp-1-soap-action-1"
type="result">
<s:Envelope
xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
<s:Header mustUnderstand="1">
<uc xmlns="urn:schemas-upnp-org:cloud-1-0"
  serviceId="RenderingControl"/>
</s:Header>
<s:Body>
<u:SetVolume xmlns:u="urn:schemas-upnp-org:
  service:RenderingControl:3">
<DesiredVolume>20</DesiredVolume>
</u:SetVolume>
</s:Body>
</s:Envelope>
</iq>

2.2.3 Eventing

UPnP UCA eventing is based on XMPP PubSub, however, since the Room can serve as a convenient PubSub alternative, UCCD events are sent directly to the Room within a <message> stanza of type "groupchat". The advantage of eventing this way being three fold: 1) PubSub authorization to "other" user's PubSub is not needed, 2) events within the Room are likely to have less jitter in their delivery and 3) the implementation is slightly less complex. On the other hand, a slight disadvantage is that there could be more unwanted event traffic for some UCC-CPs.

The UCCD waits until it has received an acknowledgement from its PubSub with <item> id so that it can supply the value in the event message to the Room. This is to allow UCC-CPs that have already subscribed to the even via regular PubSub to easily filter duplicates. The MUC event is of the form shown below.

<message
id="Vendor defined value"
type="groupchat"
to="RoomName@localpart@MUCdomainpart"
from="RoomName@localpart@MUCdomainpart/OccupantName@resourcepart of UCCD"
<event xmlns="http://jabber.org/protocol/pubsub#event"/>
<items node="name of event node the UCCD published its UCA event to">n
<item id="unique id assigned by PubSubName and copied here before sending to the Room">
<e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">
<e:property>
<variableName>new value</variableName>
</e:property>
</e:propertyset>
</item>
Note that the `<event>` element is opaque when included in a message body and will need to be caught by interested UCC-CPs.

In the example below the UCCD MediaServer:4 events a `SystemUpdateID` [CDS4] state variable and it is sent to the MUC `Room`.

**UCCD virtual-family-Room-4hQj1eTZYx@conference.mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4**

**Sends:**

```xml
<message id="WEnCONy2g1rJQ9SN0TWN" to="virtual-family-Room-4hQj1eTZYx@conference.mycloud.org" type="groupchat">
  <e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">
    <e:property>
      <SystemUpdateID>2716658</SystemUpdateID>
    </e:property>
  </e:propertyset>
</message>
```

The event is sent to each occupant of the Room. Note that there is no need to subscribe to events for MUC sessions. The `<message>` sent to the UCC-CP is shown below.

**Room virtual-family-Room-4hQj1eTZYx@conference.mycloud.org**

**Sends:**

```xml
<message id="WEnCONy2g1rJQ9SN0TWN"
  from="virtual-family-Room-4hQj1eTZYx@conference.mycloud.org/urn:schemas-upnp-org:cloud:1-0:ControlPoint:1:ZugTTaInPd"
  to="virtual-family-Room-4hQj1eTZYx@conference.mycloud.org/urn:schemas-upnp-org:device:MediaServer:4:RFH163jJt4"
  type="groupchat">
  <e:propertyset xmlns:e="urn:schemas-upnp-org:event-1-0">
    <e:property>
      <SystemUpdateID>2716658</SystemUpdateID>
    </e:property>
  </e:propertyset>
</message>
```

2.3 MUC with Other Users

The strategy suggested when devices from a different account are to be added to a MUC session is to first invite a UCC-CP from the other user's Cloud Account, say mary@theircloud.org, make that user a moderator and then allow their UCC-CP to invite other devices to the Room. Note that the trust among UCCDs and UCC-CPs is established outside of the MUC Session. If Jeffrey does not trust Mary to invite only desired devices then an out-of-band mechanism for Jeffrey to invite Mary's devices to the Room directly could be implemented.

Note that the moderator Role is granted to the OccupantNameResource in the Room and does not necessarily have to be a UCC-CP and in fact could well be a separate application with knowledge of Mary's UCC-CPs and UCCDs. This allows the scenario where Jeffrey sends the invite to Mary's Bare JID which is well-known and easier to remember by a human user (does not require knowing the randomized part or UUID), she accepts the invite and then sends Jeffrey knowledge of her OccupantNameResource via normal chat, whereby Jeffrey escalates the corresponding participants Role to moderator.

For the example, Jeffrey's UCC-CP sends an invite similar to the one in 2.1.3 to Mary's Control point at:
Mary's UCC-CP accepts as:

```
virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:e8f5ad93-634b-4123-c0e8-86a5225880ca.
```

Jeffrey's UCC-CP (the Room's owner) then escalates Mary's UCC-CP to the moderator Role via an `<iq>` stanza to the Room JID. The stanza is of the form shown below.

```
<iq
  from="RoomName@localpart@MUCdomainpart"
  id="vendor defined value"
  to="RoomName@MUCdomainpart"
  type="set">
  <query xmlns="http://jabber.org/protocol/muc#admin">
    <item nick="urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd" role="moderator">
      <reason>Mary can you share your devices?</reason>
    </item>
  </query>
</iq>
```

Upon success the owner UCC-CP is informed with an `<iq>` response of type "result" and the new Role of Mary's UCC-CP is sent to all Room occupants with an updated `<presence>` response. An example sequence is shown below.

```
UCC-CP virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd Sends:
```

```
<iq
  from="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
  id="make-mary-moderator"
  to="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org"
  type="set">
  <query xmlns="http://jabber.org/protocol/muc#admin">
    <item nick="urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd" role="moderator">
      <reason>Mary can you share your devices?</reason>
    </item>
  </query>
</iq>
```

And receives from the MUC Service the response `<iq>` stanza

```
<iq
  from="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org"
  id="make-mary-moderator"
  to="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
  type="result">
</iq>
```

All Room members receive updated `<presence>`, the response to Jeffrey's UCC-CP is shown below.

```
<presence
  from="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org"
  to="virtual-family-Room-4hQj1eTZyx@conference.mycloud.org/urn:schemas-upnp-org:cloud-1-0:ControlPoint:1:ZugTTaInPd"
  xmlns="http://jabber.org/protocol/muc#user">\n</presence>
```
The Room now contains two moderator UCC-CPs both with privileges to invite other UCCDs and UCC-CPs to the Room. Note that Jeffrey's UCC-CP is still the only UCC-CP with owner and admin privileges. Mary can now invite her UCCDs as describing in section 2.1.3. It needs to be noted that Mary might need to add the Room JID to her Roster before the invitation is sent from the Room if non-Roster stanzas are blocked.

3 Other Considerations

3.1 Room Maintenance
Currently a subscription Service to a MUC Room does not exist, therefore if participants are disconnected from the Room they will need to re-join by re-sending <presence> as described in 2.1.3. Also, only Affiliations are maintained across re-joins so privileges associated with Roles, such as moderator, would have to be re-established by the owner if still needed.

3.1.2 Room Moderation
The MUC Service defines several capabilities for moderating and managing MUC RoomS. In general descriptive capabilities, such as defining a Room subject are left to the implementer. Also, the level of management capability for the owner UCC-CP is implementation dependent, however, the owner UCC-CP at a minimum will probably need to be able to kick a participant UCC-CP or UCCD out of the Room and perhaps Ban them.

3.1.3 Room Teardown
Only the owner of a Room can destroy it. If the Room is no longer needed then it can be removed from the MUC Service by sending an <iq> stanza of the form below.

Upon success all participants are informed with a <presence> stanza of type "unavailable" from their Room JID to their Full JID that includes the <x> element including the <destroy> element. The owner can include an alternate Room for continuing the use case, as well as a <reason> element to describe why the Room is being destroyed. This is helpful when the same participants are desired in an expanded or combined new use case. The stanza is of the form shown below.

Implementing MUC for UPnP Cloud
3.1.4 Roles and Affiliations

This section provides information from [XEP0045] for the convenience of the implementer. See the XEP for specific details.

As previously described UCCDs will be Participants and UCC-CPs (or applications) can be either Participants or Moderators. Entries in Table 3-1 and Table 3-2 that correspond to Privileges or Roles not suggested in the Room configuration are grayed out.

Table 3-1: Roles and Associated Privileges

<table>
<thead>
<tr>
<th>Privilege</th>
<th>None</th>
<th>Visitor</th>
<th>Participant</th>
<th>Moderator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is this Role Required?</td>
<td>N/A</td>
<td>NO</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Present in Room</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Receive Messages</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Receive Occupant Presence</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Broadcast Presence to All Occupants</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Change Availability Status</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Change Room Nickname</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Send Private Messages</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Invite Other Users</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Send Messages to All</td>
<td>No</td>
<td>No**</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Modify Subject</td>
<td>No</td>
<td>No*</td>
<td>Yes*</td>
<td>Yes</td>
</tr>
<tr>
<td>Kick Participants and Visitors</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Grant Voice</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Revoke Voice</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes***</td>
</tr>
</tbody>
</table>

* Default; configuration settings MAY modify this privilege.
** An implementation MAY grant voice by default to visitors in unmoderated Rooms.
*** A moderator MUST NOT be able to revoke voice privileges from an admin or owner.

Table 3-2: Affiliations and Associated Privileges

<table>
<thead>
<tr>
<th>Privilege</th>
<th>Outcast</th>
<th>None</th>
<th>Member</th>
<th>Admin</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enter Open Room</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Register with Open Room</td>
<td>No</td>
<td>Yes</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>Retrieve Member List</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Enter Members-Only Room</td>
<td>No</td>
<td>No</td>
<td>Yes*</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ban Members and Unaffiliated Users</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Edit Member List</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Assign and Remove Moderator Role</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes**</td>
<td>Yes**</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>------</td>
<td>------</td>
<td>------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>Edit Admin List</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Edit Owner List</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Change Room Configuration</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Destroy Room</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* As a default, an unaffiliated user enters a moderated Room as a visitor, and enters an open Room as a participant. A member enters a Room as a participant. An admin or owner enters a Room as a moderator.

** As noted, a moderator SHOULD NOT be allowed to revoke moderation privileges from someone with a higher affiliation than themselves (i.e., an unaffiliated moderator SHOULD NOT be allowed to revoke moderation privileges from an admin or an owner, and an admin SHOULD NOT be allowed to revoke moderation privileges from an owner).

4 Summary of Suggested Capabilities

Table 4-1 provides a short summary of MUC stanza support needed for UCC-CPs and UCCDs to realize the above described MUC experience. The assumption is that the UCS implements MUC according to [XEP0045] and that the configuration parameters described in section 2.1.2 are implemented.

Table 4-1: MUC Stanza Summary for UCC-CP and UCCD

<table>
<thead>
<tr>
<th>XMPP Stanza</th>
<th>UCC-CP MUC Capabilities</th>
<th>UCCD MUC Capabilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;iq&gt; discovery MUC service</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;presence&gt; Generate and Examine Room Create stanzas</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;iq&gt; Fetch Room Configuration stanzas</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;iq&gt; Generate Room Configuration stanzas</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;message&gt; Generate Room invitation</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;presence&gt; Accept Room invitation</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&lt;iq&gt; send to Room Occupant for description and control</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&lt;iq&gt; receive and respond from Room Occupant for description and control</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>&lt;message&gt; send event to Room with PubSub ID</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;message&gt; receive and recognize event from Room and PubSub ID</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>&lt;iq&gt; send Room destroy</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

X - indicates support that could be implemented by a separate application from a UCC-CP and UCCD.

5 References

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.


