RMX 1000 XML API Documentation

Document Identification	
Document Name	RMX 1000 XML API Documentation
Document Number	
Product Name	RMX 1000
Product Version	1.1
Document Author	Wang Jianjun / He Minli

Revision #	**Author**	**Modification**
1 -20080624	Amily He	Initial version

CONTENT

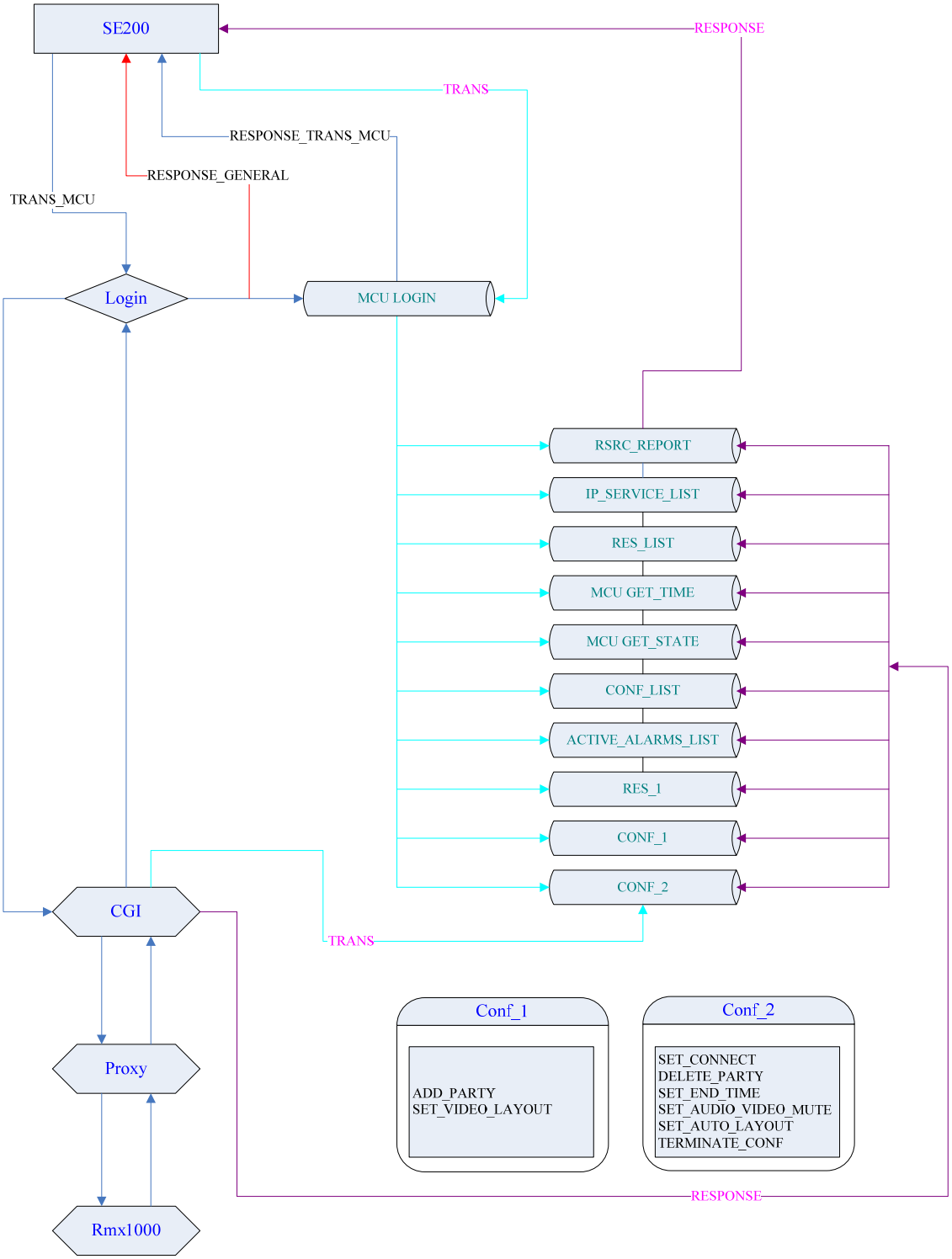
Introduction

SE200 can be used to manage RMX2000 & MGC device, RMX1000 is a new product of RMX series. In order to support SE200 integration management too, RMX1000 XML API is developed.

The XML API of RMX1000 is a subset of RMX2000 API, it's same in working style, for more information please reference RMX2000 API documents.

This guide is to clarify that the details about the SE200 and RMX1000 use XML message to login the MCU, conference create and manage operations.

API Flow Chart



XML Message

Types of Schema

The schemas in RMX 1000 XML API include two categories with different prefix: prefix trans and prefix response.

Transaction Schemas (Prefix trans)
The transaction schemas are used to retrieve the data of objects in the MCU (get requests), or to act on objects in the MCU (set requests). For example, the schema trans_res1 defines the XML format to start, start repeated, login start and update reservations on the MCU.

The transaction schemas often contain other types of schemas, such as object schemas and common schemas. For example, the trans_res1 schema includes the obj_reservation schema, because it has to include the reservation data when it is sent to start a reservation.

The transaction schemas have a consistent pattern that demonstrates object oriented methodology, as follows:

1. The root element is the schema name, which is similar to an MGC API class name.
2. The next element is the trans_common_params element, which is similar to a base class. This element mainly describes the MCU on which the action is to operate, and the synchronization method to be used
3. The next element is the ACTION element.
4. After this there are details of the individual actions that the transaction can make, which are similar to the MGC API class functions. For example, the action in trans_res1 used in RMX1000 XML is START (to start or reserve a conference). There is a description of the parameters that must be specified in order to initiate the action.

Response Schemas (Prefix response_trans)
The response schemas describe the XML format returned by the XML API for the sent transactions.

The response schemas have a consistent pattern, as follows:

1. The root element is the schema name.
2. The next element is the return status for the transaction.
3. The next element is the ACTION element.
4. After this is a choice of the actions that this response answers, corresponding to the actions in the trans_xxx schema. Under each action there is a description of the returned data.

In general, each trans_xxx schema has a corresponding response_trans_xxx schema. However, although due to internal performance considerations the trans_res and trans_conf schemas

were each split into two schemas (trans_res1 and trans_res2, and trans_conf1 and trans_conf2, respectively), there is only one response schema for each pair of transactions, (response_trans_res and response_trans_conf, respectively).
Each response schema contains a status which is a numeric value, and usually also a description.

Schemas Used to Log in to the MCU and to Create and Manage Conferences

The following schemas are used to log in to the MCU and to create and manage conferences:

1. **Function: get alarms list**

   Message:

   <TRANS_ACTIVE_ALARMS_LIST>

   …

   < ACTIVE_ALARMS_LIST >

   Description: Used to retrieve details of system alerts.

   Response:

   <RESPONSE_TRANS_ACTIVE_ALARMS_LIST>

   Description: Indicates and contains the requested information.

2. **Function: get On Going conference details**

   Message:

   < TRANS_CONF_2 >

   …

   <ID>

   Description: Used to manage On Going conferences.

   Response:

   < RESPONSE_TRANS_CONF >

   Description: Indicates that the requested action was to retrieve the details of a specified On Going Conference, and contains the conference details.

   Depending on the value you specify in the OBJ_TOKEN element, you will receive full conference data, or only object members that were changed since the previous GET request.

3. **Function: get On Going conference list**

   Message:

   < TRANS_CONF_LIST >

   …

   < CONF_SUMMARY_LS >

   Description: Used to retrieve a list of On Going conferences from MCU.

   Response:

   < RESPONSE_TRANS_CONF_LIST >

   Description: The RESPONSE_TRANS_CONF_LIST element contains the response to the TRANS_CONF_LIST schema, which is used to retrieve a list of On Going conferences.

4. Function: get IP network services details

Message:

< TRANS_IP_SERVICE_LIST >

…

< IP_SERVICE_LIST >

Description: Used to retrieve details of IP network services.

Response:

< RESPONSE_TRANS_IP_SERVICE_LIST >

Description: The RESPONSE_TRANS_IP_SERVICE_LIST element contains the response to the trans_ip_service_list schema, which used to retrieve details of IP Network Services.

5. Function: log in MCU

Message:

< TRANS_MCU >

…

< LOGIN >

Description: Used to connect a user to the MCU (SE200 login to RMX1000)

Response:

< RESPONSE_TRANS_MCU >

Description: Indicates that the requested action is to log in the MCU, and return the login parameters.

6. Function: get MCU details

Message:

< TRANS_MCU >

…

< GET_STATE >

Description: Used to retrieve information about the MCU.

Response:

< RESPONSE_TRANS_MCU >

Description: Indicates and contains the requested information.

7. Function: get MCU time

Message:

< TRANS_MCU >

…

< GET_TIME >

Description: Used to retrieve the current time and date from the MCU.

Response:

< RESPONSE_TRANS_MCU >

Description: Indicates and contains the requested information.

8. Function: new reservation conference

Message:

< TRANS_RES_1 >

…

< RESERVATION >

Description: Used to setup a new reservation conference.

Response:

< RESPONSE_TRANS_RES >

Description: Indicates and contains the reservation conference details.

9. Function: get meeting room reservation list

Message:

< TRANS_RES_LIST >

…

< GET_MEETING_ROOM_LIST >

Description: Used to retrieve a list of Meeting Room Reservation summaries.

Response:

< RESPONSE_TRANS_RES_LIST >

Description: Indicates and contains the meeting room reservation summary information and requested information. Depending on the value of the OBJ_TOKEN element, the GET_MEETING_ROOM_LIST element will either retrieve all the Meeting Room reservation summaries, or only those reservations which are new or modified since the last time this request was sent.

10. Function: get profile list

Message:

< TRANS_RES_LIST >

…

< GET_PROFILE_LIST >

Description: Used to retrieve a list of Profile summaries.

Response:

< RESPONSE_TRANS_RES_LIST >

Description: Indicates and contains the profile summary information and the requested information.

Depending on the value of the OBJ_TOKEN element, the GET_PROFILE_LIST element will either retrieve all the Profile summaries, or only those Profiles which are new or modified since the last time this request was sent.

11. Function: get resource details

Message:

< TRANS_RSRC_REPORT >

…

< GET_CARMEL_REPORT >

Description: Used to retrieve resource reports and to set the resource allocation

method.

Response:

< RESPONSE_TRANS_RSRC_REPORT >

Description: The RESPONSE_TRANS_RSRC_REPORT element contains the response to the TRANS_RSRC_REPORT schema, which is used to retrieve resource reports and to set the resource allocation method.

12. Function: participant connects or disconnects to conference

Message:

< TRANS_CONF_2 >

…

< SET_CONNECT >

Description: Used to connect a specific participant to a conference, or disconnects a specific participant from a conference.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to connect a participant to a conference, or disconnect a participant from a conference.

13. Function: delete participant from conference

Message:

< TRANS_CONF_2>

…

< DELETE_PARTY >

Description: Used to remove a specified participant from the conference.

Deleting a participant removes the participant's definition from the conference.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to remove a specified participant from a conference.

14. Function: set end time of conference

Message:

< TRANS_CONF_2 >

…

< SET_END_TIME >

Description: Used to set the end time of an On Going conference.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to set the end time of the On Going conference.

15. Function: audio video mute

Message:

< TRANS_CONF_2 >

…

< SET_AUDIO_VIDEO_MUTE >

Description: Used to mute or unmute the audio and/or video signal of a specified participant. While the audio and/or video is muted, the participant receives audio/video signal from other participants, but does not transmit audio/video to them.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to mute or unmute the audio and/or video signal of a specified participant.

16. Function: auto layout setting

Message:

< TRANS_CONF_2 >

…

< SET_AUTO_LAYOUT >

Description: Used to set Auto Layout activated or inactivated for the conference.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to activate or de-activate the Auto Layout feature.

17. Function: add participant to conference

Message:

< TRANS_CONF_1 >

…

< ADD_PARTY >

Description: Used to add a participant to an on going conference.

If the participant's connection properties allows the MCU to initiate a connection, the MCU will attempt to automatically connect the added participant to the conference.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to add a participant to the conference.

18. Function: video layout setting

Message:

< TRANS_CONF_1 >

…

< SET_VIDEO_LAYOUT >

Description: Used to set the video layout of an On Going conference.

You can use this element to change the number of cells in the layout and to force each cell.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to update the video layout for the conference.

19. Function:

Message:

< TRANS_CONF_2 >

…

< TERMINATE_CONF >

Description: Used to terminate a conference.

Specify the conference ID in the ID element.

Response:

< RESPONSE_TRANS_CONF >

Description: Indicates that the requested action was to terminate the conference.

The analyze of the XML message for RMX1000 XML API

Logging in to the MCU

The first thing you have to do to before sending any transactions to the MCU is to log in to the MCU. This is done according to the TRANS_MCU schema with action LOGIN.
The following example shows the login XML sent to the RMX1000 MCU, and the corresponding response XML.

```
<TRANS_MCU>
    <TRANS_COMMON_PARAMS>
        <MCU_TOKEN>-1</MCU_TOKEN>
        <MCU_USER_TOKEN>-1</MCU_USER_TOKEN>
    </TRANS_COMMON_PARAMS>
    <ACTION>
        <LOGIN>
            <MCU_IP>
                <IP>172.21.100.19</IP>
                <LISTEN_PORT>80</LISTEN_PORT>
            </MCU_IP>
            <USER_NAME>POLYCOM</USER_NAME>
            <PASSWORD>POLYCOM</PASSWORD>
            <STATION_NAME>se200</STATION_NAME>
        </LOGIN>
    </ACTION>
</TRANS_MCU>
```

The response will be according to the RESPONSE_TRANS_MCU schema:

```
<RESPONSE_TRANS_MCU>
    <RETURN_STATUS>
        <ID>0</ID>
        <DESCRIPTION>Status OK</DESCRIPTION>
        <YOUR_TOKEN1>0</YOUR_TOKEN1>
        <YOUR_TOKEN2>0</YOUR_TOKEN2>
        <MESSAGE_ID>0</MESSAGE_ID>
        <DESCRIPTION_EX/>
```

```
    </RETURN_STATUS>
    <ACTION>
        <LOGIN>
            <MCU_TOKEN>11</MCU_TOKEN>
            <MCU_USER_TOKEN>11</MCU_USER_TOKEN>
            <VERSION_LIST>
                <MCU_VERSION>
                    <MAIN>2</MAIN>
                    <MAJOR>0</MAJOR>
                    <MINOR>0</MINOR>
                </MCU_VERSION>
            </VERSION_LIST>
    <!-- administrator/operator/moderator/attendant/recording_user/recording_administrator -->
            <AUTHORIZATION_GROUP>administrator</AUTHORIZATION_GROUP>
            <!-- internal use, it should be set '1000' -->
            <API_NUMBER>1000</API_NUMBER>
<!--
mgc_100/mgc_50/mgc_25/mgc_25_recorder/mgc_100_plus/mgc_50_plus/mgc_25_plus/Rmx_1000/
Rmx_2000 -->
            <!-- internal use, it should be set 'Rmx_1000' -->
            <PRODUCT_TYPE>Rmx_1000</PRODUCT_TYPE>
            <!-- it should be set, default value is '0' -->
            <HTTP_PORT>0</HTTP_PORT>
        </LOGIN>
    </ACTION>
</RESPONSE_TRANS_MCU>
```

Retrieving Conference List

This example shows the XML sent to retrieve a list of On Going conferences, and the response that will be received.

```
<TRANS_CONF_LIST>
    <TRANS_COMMON_PARAMS>
        <MCU_TOKEN>14</MCU_TOKEN>
        <MCU_USER_TOKEN>14</MCU_USER_TOKEN>
        <SYNC/>
        <MESSAGE_ID>0</MESSAGE_ID>
    </TRANS_COMMON_PARAMS>
    <ACTION>
```

```
        <GET_LS>
            <OBJ_TOKEN>-1</OBJ_TOKEN>
        </GET_LS>
    </ACTION>
</TRANS_CONF_LIST>
```

The response will be according to the TRANS_CONF_LIST schema:

```
<RESPONSE_TRANS_CONF_LIST>
    <RETURN_STATUS>
        <ID>0</ID>
        <DESCRIPTION>Status OK</DESCRIPTION>
        <YOUR_TOKEN1>0</YOUR_TOKEN1>
        <YOUR_TOKEN2>0</YOUR_TOKEN2>
        <MESSAGE_ID>0</MESSAGE_ID>
        <DESCRIPTION_EX/>
    </RETURN_STATUS>
    <ACTION>
        <GET_LS>
            <CONF_SUMMARY_LS>
                <OBJ_TOKEN>15</OBJ_TOKEN>
                <!-- if CONF_SUMMARY contents changed, its value must be changed  -->
                <CHANGED>true</CHANGED>
                <CONF_SUMMARY>
                <!-- conf_info.DISP_NAME -->
                <NAME>POLYCOM_Mar_5_2008_10:55</NAME>
                    <ID>101</ID> <!-- conf_info.GUID -->
                    <!-- update/none/new -->
                    <CONF_CHANGE>new</CONF_CHANGE>
                    <CONF_STATUS><!-- conf_info.STATUS -->
                        <CONF_OK>false</CONF_OK>
                        <CONF_EMPTY>true</CONF_EMPTY>
                        <SINGLE_PARTY>false</SINGLE_PARTY>
                        <NOT_FULL>true</NOT_FULL>
                        <RESOURCES_DEFICIENCY>false</RESOURCES_DEFICIENCY>
                        <BAD_RESOURCES>false</BAD_RESOURCES>
                        <PROBLEM_PARTY>false</PROBLEM_PARTY>
                    </CONF_STATUS>
                    <!-- conf_info.START_TIME -->
                    <START_TIME>2008-03-05T10:51:39</START_TIME>
                    <!-- conf_info.END_TIME -->
                    <END_TIME>2008-03-05T12:51:39</END_TIME>
                    <!-- conf_info.IS_CONF_LOCK -->
                    <LOCK>false</LOCK>
```

```
        <!-- if conf_info.VIDEO_MODE is 'LECTURE' then LECTURE_MODE is 'true', else 'false' -->
                    <LECTURE_CONF>false</LECTURE_CONF>
                    <!-- the number of participants -->
                    <NUM_PARTIES>0</NUM_PARTIES>
                    <!-- the number of endpoints connected to the conference -->
                    <NUM_CONNECTED_PARTIES/>
                    <!-- conf_info.CONF_PASSWD -->
                    <ENTRY_PASSWORD/>
                    <!-- conf_info.CHAIR_PASSWD -->
                    <PASSWORD/>
                    <!-- conf_info.E164 -->
                    <NUMERIC_ID>9604</NUMERIC_ID>
                <!-- if conf_info.LAYOUT_MODE is '-1' then AUTO_LAYOUT is 'true', else 'false' -->
                    <AUTO_LAYOUT>true</AUTO_LAYOUT>
                    <!-- conf_info.END_TIME minus conf_info.START_TIME -->
                    <DURATION>
                        <HOUR>2</HOUR>
                        <MINUTE>0</MINUTE>
                        <SECOND>0</SECOND>
                    </DURATION>
                    <!-- conf_profile_info.IS_ENCRYPTION-->
                    <ENCRYPTION>false</ENCRYPTION>
                    <!-- conf_info.RECORD_STATUS -->
                    <RECORDING_STATUS>stop</RECORDING_STATUS>
                    <TRANSFER_RATE/> <!-- conf_info.LINE_RATE -->
                    <!-- conf_info.DISP_NAME -->
                    <DISPLAY_NAME>POLYCOM_Mar_5_2008_10:55</DISPLAY_NAME>
                </CONF_SUMMARY>
            <!-- used by comparison, it may be including 'ID' which belongs to conference ID -->
                <DELETED_CONF_LIST/>
            </CONF_SUMMARY_LS>
        </GET_LS>
    </ACTION>
</RESPONSE_TRANS_CONF_LIST>
```

Retrieving the Meeting Room list

The following is an XML example for retrieving the Meeting Room list, and the response that will be received.

```
<TRANS_RES_LIST>
    <TRANS_COMMON_PARAMS>
        <MCU_TOKEN>11</MCU_TOKEN>
        <MCU_USER_TOKEN>11</MCU_USER_TOKEN>
    </TRANS_COMMON_PARAMS>
    <ACTION>
        <GET_MEETING_ROOM_LIST>
            <OBJ_TOKEN>-1</OBJ_TOKEN>
        </GET_MEETING_ROOM_LIST>
    </ACTION>
</TRANS_RES_LIST>
```

The response will be according to the `TRANS_RES_LIST` schema:

```
<RESPONSE_TRANS_RES_LIST>
    <RETURN_STATUS>
        <ID>0</ID>
        <DESCRIPTION>Status OK</DESCRIPTION>
        <YOUR_TOKEN1>0</YOUR_TOKEN1>
        <YOUR_TOKEN2>0</YOUR_TOKEN2>
        <MESSAGE_ID>0</MESSAGE_ID>
        <DESCRIPTION_EX/>
    </RETURN_STATUS>
    <ACTION>
        <GET_MEETING_ROOM_LIST>
            <MEETING_ROOM_SUMMARY_LS>
                <OBJ_TOKEN>13</OBJ_TOKEN>
            <!-- if MEETING_ROOM_SUMMARY contents changed, its value must be changed  -->
                <CHANGED>true</CHANGED>
                <DELETED_RES_LIST/> <!-- to be deleted res list, be made of 'ID' -->
                <MEETING_ROOM_SUMMARY>
                <!-- meeting_room_info.DISP_NAME -->
                <NAME>POLYCOM_Mar_5_2008_10:1</NAME>
                    <!-- meeting_room_info.GUID -->
                    <ID>1</ID>
                    <RES_CHANGE>new</RES_CHANGE> <!-- new/update/none -->
                    <DURATION> <!-- meeting_room_info.DURATION -->
                        <HOUR>2</HOUR>
```

```
                        <MINUTE>0</MINUTE>
                        <SECOND>0</SECOND>
                    </DURATION>
                    <!-- meeting_room_info.CONF_PASSWD -->
                    <ENTRY_PASSWORD>1234</ENTRY_PASSWORD>
                    <!-- meeting_room_info.CHAIR_PASSWD -->
                    <PASSWORD>1111</PASSWORD>
                    <!-- meeting_room_info.E164 -->
                    <NUMERIC_ID>2324</NUMERIC_ID>
                    <!-- meeting_room_info.PROFILE_GUID -->
                    <AD_HOC_PROFILE_ID>1</AD_HOC_PROFILE_ID>
                    <!-- meeting_room_info.DISP_NAME -->
                    <DISPLAY_NAME>POLYCOM_Mar_5_2008_10:1</DISPLAY_NAME>
                </MEETING_ROOM_SUMMARY>
            </MEETING_ROOM_SUMMARY_LS>
        </GET_MEETING_ROOM_LIST>
    </ACTION>
</RESPONSE_TRANS_RES_LIST>
```

Setting the end time

The following is an example for setting the end time of an On Going conference, and the response that will be received.

```
<TRANS_CONF_2>
    <TRANS_COMMON_PARAMS>
        <MCU_TOKEN>24</MCU_TOKEN>
        <MCU_USER_TOKEN>24</MCU_USER_TOKEN>
        <SYNC/>
    </TRANS_COMMON_PARAMS>
    <ACTION>
        <SET_END_TIME>
            <ID>437</ID> <!-- conference id -->
            <!-- conference ending time -->
            <END_TIME>2008-01-16T11:46:39</END_TIME>
        </SET_END_TIME>
    </ACTION>
</TRANS_CONF_2>
```

The response will be according to the TRANS_CONF_2 schema:

```
<RESPONSE_TRANS_CONF>
    <RETURN_STATUS>
        <ID>0</ID>
        <DESCRIPTION>Status OK</DESCRIPTION>
        <YOUR_TOKEN1>0</YOUR_TOKEN1>
        <YOUR_TOKEN2>0</YOUR_TOKEN2>
        <MESSAGE_ID>0</MESSAGE_ID>
        <DESCRIPTION_EX/>
    </RETURN_STATUS>
    <ACTION>
        <!-- this is only action -->
        <SET_END_TIME/>
    </ACTION>
</RESPONSE_TRANS_CONF>
```

Audio Muting a Participant

This example shows the XML that must be sent in order to audio or video mute a party according to the TRANS_CONF_2 schema, and the response that will be received.

```
<TRANS_CONF_2>
    <TRANS_COMMON_PARAMS>
        <MCU_TOKEN>24</MCU_TOKEN>
        <MCU_USER_TOKEN>24</MCU_USER_TOKEN>
        <SYNC/>
    </TRANS_COMMON_PARAMS>
    <ACTION>
        <SET_AUDIO_VIDEO_MUTE>
            <ID>436</ID> <!-- conference id -->
            <!-- 'true' is audio mute action, 'false' is audio unmute action -->
            <AUDIO_MUTE>true</AUDIO_MUTE>
            <!-- 'true' is video mute action, 'false' is video unmute action -->
            <VIDEO_MUTE>false</VIDEO_MUTE>
            <PARTY_ID>0</PARTY_ID>
        </SET_AUDIO_VIDEO_MUTE>
    </ACTION>
</TRANS_CONF_2>
```

The response will be according to the TRANS_CONF_2 schema:

```
<RESPONSE_TRANS_CONF>
    <RETURN_STATUS>
        <ID>0</ID>
        <DESCRIPTION>Status OK</DESCRIPTION>
        <YOUR_TOKEN1>0</YOUR_TOKEN1>
        <YOUR_TOKEN2>0</YOUR_TOKEN2>
        <MESSAGE_ID>0</MESSAGE_ID>
        <DESCRIPTION_EX/>
    </RETURN_STATUS>
    <ACTION>
        <!-- this is only action -->
        <SET_AUDIO_VIDEO_MUTE/>
    </ACTION>
</RESPONSE_TRANS_CONF>
```

Reference Documents

1. MGC_XML_API_VER_8_0_ebook_ReleaseA.chm
2. MGC_XML_Tracer_V7-5_User's_Guide_ReleaseA .pdf
3. Copies of all of the RMX 1000 XML message (located in the **RMX 1000 XML Message** folder).