



Cisco TelePresence MCU Remote Management API Reference Guide

D14626.04

December 2010

Contents

API version history	4
Version 2.7	4
Version 2.6	5
Introduction	6
HTTP and HTTPS	6
Clustering	6
Consider API overhead when writing applications	6
XML-RPC protocol overview	8
Message flow	8
Authentication	8
Example command message	8
Unicode support	10
HTTP headers	10
XML header	10
Common message elements	10
Authentication	10
Participant identification parameters	11
Enumerate functions	11
Filtering	12
API reference	14
addressBookEntry.enumerate	16
auditlog.delete	18
auditlog.query	19
autoAttendant.destroy	19
autoAttendant.enumerate	19
autoAttendant.status	20
cdrlog.delete	20
cdrlog.query	21
conference.create	21
Deprecated parameters	25
conference.destroy	25
conference.end	25
conference.enumerate	25
conference.floor.modify	30
conference.floor.query	30
conference.modify	31
Deprecated parameters	33
conference.paneplacement.modify	34
conference.paneplacement.query	35
conference.status	36
conference.streaming.modify	36
conference.streaming.query	36
conferenceme.query	38
device.health.query	39
device.network.query	39
device.query	42

device.restartlog.query	43
gatekeeper.query	43
gateway.enumerate.....	44
participant.add.....	45
Deprecated parameters	47
participant.connect	48
participant.diagnostics.....	48
participant.disconnect	50
participant.enumerate	51
Deprecated parameters	58
participant.fecc	58
participant.message	58
participant.modify	59
Deprecated parameters	61
participant.move	61
participant.remove.....	62
participant.status	62
sip.query	63
template.modify.....	63
template.status	66
template.enumerate	66
Deprecated calls	67
participant.enumerate	68
Related information sources	71
system.xml	71
Fault codes.....	73
Participant disconnect reasons	75
Conference layouts.....	78
Family layouts	78
Specific layouts	78
Linking conferences across MCUs.....	80
Example message 1 - creating conference "linked1" on "mcu1"	81
Example message 2 - creating conference "linked2" on "mcu2"	81
Example message 3 - calling into "linked2" from "linked1"	82
Example message 4 - setting the new "linked2" participant to use a full screen view layout.....	82
Message responses	84
Revision numbers.....	85
Discovering record removal	86
Dead records.....	86
HTTP keep-alives	87
References	88

API version history

The latest version of the Cisco TelePresence MCU Remote Management API is version 2.7. The following Cisco TelePresence MCU products support this version:

- ▶ MCU 4200 Series
- ▶ MCU 4500 Series
- ▶ MCU MSE 8420
- ▶ MCU MSE 8510

Version 2.7

Version 2.7 introduced the following changes:

XML-RPC request	Parameter	Addition/Deprecated /Removed
conference.create conference.modify conference.enumerate conference.status	suppressDtmfEx	Additional parameter
	layoutControlEx	Additional parameter
	dtmfMuteControl	Additional parameter
	automaticLectureModeEnabled	Additional parameter
	automaticLectureModeTimeout	Additional parameter
	encryptionRequired	Additional parameter
	contentContribution	Additional parameter
	contentTransmitResolutions	Additional parameter
	suppressDtmf	Removed parameter
layoutControlEnabled	Deprecated parameter	
participant.add participant.modify participant.enumerate participant.status	suppressDtmfEx	Additional parameter
	layoutControlEx	Additional parameter
	suppressDTMF	Removed parameter

	layoutControlEnabled	Deprecated parameter
participant.enumerate	lecturer	Additional parameter
	enumerateFilter	Additional parameter
template.modify		New call
template.status		New call
template.enumerate		New call

Version 2.6

Version 2.6 introduced the following changes:

XML-RPC request	Parameter	Addition/Deprecated
auditlog.delete		Addition
auditlog.query		Addition
autoAttendant.destroy		Addition
autoAttendant.status		Addition
cdrlog.delete		Addition
cdrlog.query		Addition
conference.status		Addition
device.query	totalStreamingAndContentPorts	Addition
participant.status		Addition

Introduction

This reference guide contains the specification of the Cisco TelePresence MCU Remote Management API, by which it is possible to control the following Cisco TelePresence products:

- ▶ MCU 4200 Series
- ▶ MCU 4500 Series
- ▶ MCU MSE 8420
- ▶ MCU MSE 8510

This is accomplished via messages sent using the **XML-RPC** protocol. XML-RPC is a simple protocol for remote procedure calling using HTTP as the transport and XML as the encoding. It is designed to be as simple as possible, whilst allowing complex data structures to be transmitted, processed and returned. XML-RPC has no platform or software dependence and was chosen over SOAP because of its simplicity.

The interface is stateless. Currently, there is no mechanism for the MCU to call back the controlling application and therefore the controlling application must poll the MCU for status, as required. A future enhancement *may* provide a mechanism for signaling MCU status changes to the controlling application.

In this implementation of XML-RPC all parameters and return values are part of a <struct> and are all explicitly named. For example, the device.query call returns the current time value as a structure member named currentTime rather than as a single value of type <dateTime.iso8601>.

Note: Unless otherwise stated, assume string length is a maximum of 32 characters.

For further details of XML-RPC refer to the [specification](#) [1].

HTTP and HTTPS

MCUs expect to receive HTTP communication over TCP/IP connections to port 80. The HTTP messages should be “POST”s to the URL “/RPC2”.

HTTPS (a secure, encrypted version of HTTP) is supported on all MCU products from software version 2.3 and later.

By default, HTTPS is provided on TCP port 443, although MCUs can be configured to receive HTTP and HTTPS connections on non-standard TCP port numbers if required.

The MCUs implement HTTP/1.1 as defined by RFC 2616 [2].

Clustering

From Version 4.1 of the MCU, it is possible to configure MCU blades in a cluster in order to increase the number of HD conference participants. One MCU acts as a master controlling between up to two slave MCUs.

Consider API overhead when writing applications

Every API command that your application sends incurs a processing overhead within the device’s own application. The exact amount of overhead varies widely with the command type and the parameters sent. It is important to bear this in mind when designing your application’s architecture and software. If the device receives a high number of API commands every second, its overall performance could be seriously impaired – in the same way that it would if several users accessed it from the web interface simultaneously.

For this reason, the best architecture is a single server running the API application and sending commands to the device. If multiple users need to use the application simultaneously, provide a web interface on that server or write a client that communicates with the server. The server would then manage the clients' requests and send API commands directly to the device. Implement some form of control in the API application on your server to prevent the device being overloaded with API commands. This provides much more control than having the clients send API commands directly and will prevent the device's performance being impaired by unmanageable numbers of API requests.

XML-RPC protocol overview

Message flow

An application external to the MCU can create and manage conferences by sending command messages to the MCU. For each command sent (provided the message is correctly formatted according to the [XML-RPC spec](#)), the MCU responds with a message indicating success or failure. The response message may also contain any data that was requested.

Authentication

In order to manage the MCU, the controlling application must authenticate itself as a user with relevant privileges. Accordingly, each command message contains a user name and password; see the section *Common message elements* below for details of the format. It is worth noting that authentication information is sent using plain text and should only be sent over a trusted network.

Example command message

Command messages are sent in XML format. For example, the following message schedules a conference on an MCU to begin at 10:45 on 18 February 2005 and last for one hour:

```
POST /RPC2 HTTP/1.1
User-Agent: Frontier/5.1.2 (WinNT)
Host: 10.2.1.100
Content-Type: text/xml
Content-length: 713

<?xml version="1.0"?>
<methodCall>
<methodName>conference.create</methodName>
<params>
<param>
<value>
<struct>
<member>
<name>authenticationUser</name>
<value>
<string>api_test</string>
</value>
</member>
<member>
<name>authenticationPassword</name>
<value>
<string>123456</string>
</value>
</member>
<member>
<name>conferenceName</name>
<value>
<string>Meeting 1</string>
</value>
</member>
<member>
<name>startTime</name>
```

```

<value>
<dateTime.iso8601>20050218T10:45:00</dateTime.iso8601>
</value>
</member>
<member>
<name>durationSeconds</name>
<value>
<int>3600</int>
</value>
</member>
</struct>
</value>
</param>
</params>
</methodCall>

```

If the command was successful, the MCU sends a success response. For example, in response to a successful `conference.create` message, the MCU returns:

```

HTTP/1.1 200 OK
Connection: close
Content-Type: text/xml
Content-Length: 240

<?xml version="1.0"?>
<methodResponse>
<params>
<param>
<value>
<struct>
<member>
<name>status</name>
<value>
<string>operation successful</string>
</value>
</member>
</struct>
</value>
</param>
</params>
</methodResponse>

```

If the command fails, the MCU sends a fault response. For example, in response to a `conference.create` message where the conference name is not unique, the MCU returns:

```

HTTP/1.1 200 OK
Connection: close
Content-Type: text/xml
Content-Length: 411

<?xml version="1.0"?>
<methodResponse>
<fault>
  <value>
    <struct>
      <member>
        <name>faultCode</name>

```

```

    <value>
      <int>2</int>
    </value>
  </member>
  <member>
    <name>faultString</name>
    <value>
      <string>duplicate conference name</string>
    </value>
  </member>
</struct>
</value>
</fault>
</methodResponse>

```

The complete list of command messages, their required and optional parameters, and the expected responses are detailed in the sections below. The possible [fault codes](#) are listed later in this document.

Unicode support

Parameters in this version of the API can be in ASCII text or unicode (UTF-8). In order to distinguish between these encodings, any of several methods can be used. If no method is present, ASCII is assumed.

HTTP headers

There are two different ways of specifying unicode in the HTTP headers; either using "Accept-Encoding: utf-8", or modifying the Content-Type header to read "Content-Type: text/xml; charset=utf-8".

XML header

The `<?xml>` tag is required at the top of each XML file. This API will accept an additional encoding parameter with value UTF-8 for this tag, i.e. `<?xml version="1.0" encoding="UTF-8"?>`.

Common message elements

Authentication

All messages must contain a user name and password as follows:

Parameter	Type	Comments
authenticationUser	String	Name of a user with sufficient privilege for the operation being performed. The name is case sensitive.
authenticationPassword	String	The corresponding user's password. This parameter is ignored if the user has no password set. Note that this differs from the web interface where a blank password must be blank.

Participant identification parameters

The following parameters appear in the majority of conference control messages, and identify a specific participant on which operations are to be performed. The use of “MCU” below refers to any device which acts as a video conferencing server.

Parameter	Type	Comments
participantName	String	This is an “internal” name, and therefore is not necessarily related to any name configured on an endpoint. Within the scope of a particular conference or auto attendant, the combination of “participantType”, “participantProtocol” and “participantName” is always unique.
participantProtocol	String	Used in conjunction with “participantName” to uniquely identify a participant within a conference. Typically, these parameters should be treated as opaque values, but the current possibilities are: - for “participantProtocol”: h323 – an endpoint using the H.323 protocol vnc – a VNC connection (e.g. remote desktop) sip – an endpoint using the SIP protocol - for “participantType”: ad_hoc – this participant called into the MCU or was dialed out via the web interface and is not in the MCU’s endpoint list by_address – fully-specified participant added through the API by_name – MCU-configured endpoint irrespective of whether the endpoint dialed in or the MCU dialed out API-created participants in scheduled conferences (i.e. those originated by “participant.add”) will be of type by_address (unless they are added explicitly as temporary ad_hoc participants.)
participantType	String	
conferenceName	String	Unique conference name – the conference name space is shared between API-generated conferences and all other ad hoc and scheduled MCU conferences.
autoAttendantUniqueld	String	If the participant in question is connected to an auto attendant rather than a conference, this field contains a unique identifier for that auto attendant.

When modifying or querying parameters for a specific endpoint, participantName, participantProtocol and participantType parameters are supplied, along with either a conferenceName or an autoAttendantUniqueld. The only safe way to find these values is to use the fields returned from participant.enumerate.

Enumerate functions

Due to the potential for a very large number of responses, all enumerate functions return an enumerateID response. This contains a value which should be passed to subsequent calls of the same enumerate function in order to retrieve the remainder of the values.

The use of this parameter is as follows:

1. The client computer sends an enumerate call with any necessary parameters (e.g. operationScope) and no enumerateID parameter.
2. The MCU returns with an array containing the requested data, and possibly a new enumerateID.
3. If there is an enumerateID, the client will call the enumerate method again, with any parameters that are required or desired, and an enumerateID parameter containing the ID returned by the device from the previous call. This should be repeated while the MCU continues to provide new enumerateID values in responses.
4. After all data is returned, the MCU will reply with all remaining results, but no enumerateID.

This method should only be called using enumerateID values as provided by the MCU.

Filtering

Enumerate methods contain an optional enumerateFilter parameter, which can be used to restrict the responses to the enumerate call. The valid expressions depend on the method to which they are applied, but the syntax is the same for all enumerate functions: the section in this document for each call provides a list of valid filters for that call.

To use the filters, the expression is evaluated, with any method or expression symbols evaluated for the given entity being enumerated (e.g. if enumerating conferences, the *active* expression will evaluate to true if the conference is active, and false otherwise). If the result of evaluating the filter is true, the entity is returned to the client. If the expression evaluates to false, the entity in question is not returned to the client and the next entity (if any) is considered. As an example, if the expression (*active && scheduled*) is used when enumerating conferences, the returned conferences will be only those which are both active and scheduled.

Filters can consist of atomic expressions, joined together with operators, and brackets in the traditional manner. Whitespace is ignored. Methods are valid, and any parameters are in a comma separated list in brackets following the function name, for example `function(expression1, expression2).`

From a Boolean perspective, the integer 0 is **false**, and all other numbers are **true**.

Integer values can be expressed using any string of valid digits, optionally prefixed by 0x for hexadecimal.

0t for decimal and 0z for binary. If no prefix is specified, decimal is assumed.

The following binary operators are valid, in order of priority (lowest priority first):

Operator	Description
	Boolean or
&&	Boolean and
	Bitwise or
^	Bitwise exclusive or
&	Bitwise and
==	Equality
!=	Inequality
<	Less than
<=	Less than or equal
>=	Greater than or equal

Operator	Description
>	Greater than
<<	Bitwise left shift
>>	Bitwise right shift
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulo

There are also the following unary operators, all of which bind tighter than any binary operator.

Operator	Description
-	Unary minus
+	Unary plus
!	Logical negation
~	Bitwise negation

An example filter would be $!(expression_1 \ \&\& \ expression_2)$.

API reference

This section is a reference guide to each of the MCU API calls. For each API call the following information is provided where applicable:

- ▶ Description of the call's function
- ▶ Parameters
- ▶ List of responses
- ▶ Response data types
- ▶ Structure formats
- ▶ Additional information that will assist in using the API call
- ▶ Parameters deprecated from previous versions

The table below lists all currently supported MCU API calls. Click on the call to go to the page containing a complete description of the call.

The *Slave support* column indicates whether the call is supported on slave MCUs in clustered configurations. Refer to the online for more information on clustering.

API call	Page	Slave support
addressBookEntry.enumerate	16	N
auditlog.delete	18	Y
auditlog.query	19	Y
autoAttendant.destroy	19	N
autoAttendant.enumerate	19	N
autoAttendant.status	20	N
cdrlog.delete	20	N
cdrlog.query	21	N
conference.create	21	N
conference.destroy	25	N
conference.end	25	N
conference.enumerate	25	N
conference.floor.modify	30	N
conference.floor.query	30	N
conference.modify	31	N
conference.paneplacement.modify	34	N

API call	Page	Slave support
conference.paneplacement.query	35	N
conference.status	36	N
conference.streaming.modify	36	N
conference.streaming.query	36	N
conferenceme.query	38	N
device.health.query	39	Y
device.network.query	39	Y
device.query	42	Y
device.restartlog.query	43	Y
gatekeeper.query	43	N
gateway.enumerate	44	N
participant.add	45	N
participant.connect	48	N
participant.diagnostics	48	N
participant.disconnect	50	N
participant.enumerate	51	N
participant.fecc	58	N
participant.message	58	N
participant.modify	59	N
participant.move	61	N
participant.remove	62	N
participant.status	62	N
sip.query	63	N
template.modify	63	N
template.status	66	N
template.enumerate	66	N

addressBookEntry.enumerate

Enumerates configured endpoints on the MCU.

Parameter	Type	Comments
Optional parameters		
enumerateID	String	The value returned by the last enumeration call. If it is omitted, a new enumeration is started.

This call returns the following:

Response	Type	Comments
addressBookEntries	Array	See below for details.
Optional parameters		
enumerateID	String	The value that should be used in the next call to get the next set of data. If this is omitted, no further data is available from the MCU.

The array "addressBookEntries" contains structures with the following fields:

Field	Type	Comments
Name	String	The configuration's name.
address	String	The participant's E.164 directory number, hostname or IP address.
protocol	String	One of h323, sip or vnc.
gatewayName	String	Present for H.323 endpoints which are configured to use a gateway. This name corresponds to the name of a gateway returned via gateway.enumerate.
gatewayAddress	String	Present for H.323 endpoints which are configured to use a gateway. This is the address of the gateway this endpoint is configured to use.
useSIPRegistrar	Boolean	Whether this endpoint is configured to use a SIP registrar when being called.
password	String	The password for VNC endpoints.

Field	Type	Comments
portNumber	Integer	The port number for VNC endpoints.
callInParams	Array	See below for details.
conferencingParameters	Array	See below for details.

The array “callInParams” contains structures with the following fields. This is used to match incoming participants to endpoint configurations. For a positive match a participant must match fields which have values. Blank fields are not considered in the comparison.

Field	Type	Comments
name	String	Endpoint name.
address	String	IP address.
e164	String	E.164 number.

The array conferencingParameters contains structures with the following fields:

Field	Type	Comments
useDefaultMotionSharpness	Boolean	If true, this endpoint will use box-wide default motion sharpness settings.
minFrameRateMotionSharpness	Integer	Only present if “useDefaultMotionSharpness” is false. Specifies the minimum frame rate for this endpoint.
useDefaultVideoTransmitResolutions	Boolean	If true , this endpoint will use box-wide default video transmit resolutions.
videoTransmitResolutions	String	One of: allowAll, 4to3Only, 4to3WidescreenOverride or 16to9Only.
maxMediaTxBitRate	Integer	Max media transmit bit rate.
maxMediaRxBitRate	Integer	Max media receive bit rate.
defaultLayout	String	Refer to later in this document for a list of layouts.
layoutControlDefault	Boolean	If true, this endpoint will use box-wide layout control settings.
layoutControlEnabled	Boolean	Only present if “layoutControlDefault” is false. Indicates whether the participant will have control over their layout.

Field	Type	Comments
h239ContributionDefault	Boolean	If true, this endpoint will use box-wide H.239 contribution settings.
h239ContributionEnabled	Boolean	Only present if "h239ContributionDefault" is false. Specifies whether the endpoint will be able contribute H.239.
initialAudioMuted	Boolean	Whether this participant would initially have their audio muted.
initialVideoMuted	Boolean	Whether this participant would initially have their video muted.
autoDisconnect	Boolean	When a participant disconnects from a conference and only participants who have autoDisconnect set to true remain, all those participants are disconnected.
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.

auditlog.delete

Deletes stored events from the CDR log.

Parameter	Type	Comments
deleteIndex	Integer	You can delete logs in chunks of 400 entries. To delete entries, you can enter the value returned after a auditlog.query call in deleteableIndex. This will delete all complete chunks (400 entries) below this value, leaving the residuals. Alternatively, you can delete fewer entries by entering a number below the value of deleteableIndex. This will delete all complete chunks (400 entries) below the entered number, leaving any residuals.

Stored audit events up to and including the indicated deleteIndex will be permanently deleted.

auditlog.query

This call queries the audit log for . It takes no parameters.

The response returns the following.

Response	Type	Comments
firstIndex	Integer	The index of the oldest stored audit event.
deleteableIndex	Integer	The index of the most recent deletable audit event.
numEvents	Integer	The total number of events stored.
percentageCapacity	Integer	The percentage of total available capacity used by the audit log.

autoAttendant.destroy

This call destroys auto attendants.

Parameter	Type	Comments
autoAttendantUniqueID	String	Identifier for the auto attendant to be destroyed.

autoAttendant.enumerate

Parameter	Type	Comments
Optional parameters		
enumerateID	String	The value returned by the last enumeration call, if omitted, a new enumeration is started.

The call has no valid enumerate filter expressions.

This call returns:

Response	Type	Comments
autoAttendants	Array	See below for details.
Optional responses		
enumerateID	String	The value which should be used in the next call to get the next set of data. If this is omitted, then no further data is available from the MCU.

The array “autoAttendants” contains structures with the following fields:

Response	Type	Comments
autoAttendantUniqueID	String	A unique identifier for the auto attendant
autoAttendantConfiguredName	String	If this participant is connected to an auto attendant, this field holds the name of that auto attendant: the value will change as the user navigates through an MCU's configured menu structure.
startTime	dateTime.iso8601	The time at which the auto attendant was created.

autoAttendant.status

This call returns a struct as described in [autoAttendant.enumerate](#) above, containing information about the auto attendant indicated.

A fault code of “no such conference” is returned if there is no auto attendant with the given identifier.

Parameter	Type	Comments
autoAttendantUniqueID	String	Identifier for the auto attendant from which information is required.

cdrlog.delete

Deletes stored events from the CDR log.

Parameter	Type	Comments
deleteIndex	Integer	You can delete logs in chunks of 400. To delete logs, you can enter the value returned by deleteableIndex. This will delete all complete chunks (400 logs) below this amount, leaving the residuals. Alternatively, you can delete less than this amount by picking a number below the value of deleteableIndex. This will delete all complete chunks (400 logs) below that number, leaving any residuals.

Stored audit events up to and including the indicated deleteIndex will be permanently deleted.

cdrlog.query

This call queries for statistics about the CDR log.

This call takes no parameters. The response returns the following:

Field	Type	Comments
firstIndex	Integer	The index of the oldest stored CDR event.
deleteableIndex	Integer	The index of the most recent deletable CDR event.
numEvents	Integer	Total number of events stored.
percentageCapacity	Integer	The percentage of total available capacity used by the CDR log.

conference.create

This call creates a new conference on the MCU. Conferences created through the management API will appear in the list of conferences accessible via the web interface, and vice versa.

Parameter	Type	Comments
conferenceName	String (<32 chars)	Name of the conference to be created. The conference name must be unique.
private	Boolean	Determines the visibility of this conference. This parameter corresponds to the "Visibility" setting on the web UI, which can have the value Public or Private.
joinAudioMuted	Boolean	Audio mute on join.
joinVideoMuted	Boolean	Video mute on join.
enforceMaximumAudioPorts	Boolean	Assumed to be true if absent. These can be set to false in order to specify no limit on the number of audio/video ports.
enforceMaximumVideoPorts	Boolean	
Optional parameters		
numericId	String (<32 chars)	Numeric identifier of the conference.
registerWithGatekeeper	Boolean	Registers the conference's "numericId" with the gatekeeper.
registerWithSIPRegistrar	Boolean	Registers this conference with the SIP registrar.

Parameter	Type	Comments
startTime	dateTime.iso8601	If you do not specify a time, the conference starts immediately.
durationSeconds	Integer	The length of each repeating conference instance, in seconds. If this parameter is absent, or set to "0", the conference is permanent.
pin	String (< 32 chars)	If present, this is the string of numeric digits that people need to enter to join the conference.
description	String (< 32 chars)	
multicastStreamingEnabled	Boolean	
unicastStreamingEnabled	Boolean	
h239Enabled	Boolean	
maximumAudioPorts	Integer	These fields set the limit on the number of audio (audio only) and video (video + audio) ports for the conference. The "reserved" values are for port reservation mode, whereas the "maximum" figures apply to non-reserved mode (and will be absent if no limits have been configured).
maximumVideoPorts	Integer	
reservedAudioPorts	Integer	
reservedVideoPorts	Integer	
repetition	String	One of: none , daily , weekly , everyTwoWeeks or monthly .
weekDay	String	Must be present if repetition is monthly . One of monday , tuesday , wednesday , thursday , friday , saturday or sunday . Note that if repetition is not weekly or everyTwoWeeks, the "weekDays" parameter should be used.
whichWeek	String	Must be present if repetition is monthly . One of: first (the first X of the month, where X is the day specified by weekday), second, third, fourth, or last (i.e. last X of the month).

Parameter	Type	Comments
weekDays	String	Must be present if repetition is weekly or everyTwoWeeks. A comma separated string of weekdays (i.e. any of monday, tuesday, wednesday, thursday, friday, saturday or sunday), e.g. monday,wednesday,friday.
terminationType	String	One of: noTermination afterNRepeats endOnGivenDate .
terminationDate	dateTime.iso8601	If terminationType is endOnGivenDate , this is the day that the conference repetition will end on.
numberOfRepeats	Integer	If terminationType is afterNRepeats , this is the number of repeats to end after.
customLayoutEnabled	Boolean	true if the layout is enabled, false otherwise.
layoutControlEx	String	Controls whether and how view layout can be controlled. One of: disabled: Layout control is disabled. feccOnly: Layout can be controlled via FECC. feccAndDtmf: Layout can be controlled via FECC or DTMF if FECC is unavailable.
newParticipantsCustomLayout	Boolean	true if new participants use the custom layout, false otherwise. Only valid if customLayoutEnabled is true.
customLayout	Integer	The video layout seen by the participant. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts.
chairControl	String	The chair control setting for the conference. This can be none , floorControlOnly or chairAndFloorControl - these values correspond to the web interface "Floor and chair control" setting values of "Do not allow floor or chair control", "Allow floor control only" and "Allow floor and chair control" respectively. If not specified, the chair control setting for the new conference will be "Allow floor control only".
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available. always : DTMF tones are always muted never : DTMF tones are never muted

Parameter	Type	Comments
dtmfMuteControl	Boolean	Controls whether the user can mute audio using *6. Either: <code>true</code> : participant can mute audio using *6 <code>false</code> : participant cannot mute audio using *6.
automaticLectureModeEnabled	Boolean	Automatic lecture mode shows the speaker full screen. Either: <code>true</code> : automatic lecture mode enabled. <code>false</code> : automatic lecture mode disabled. If automatic lecture mode is enabled, the <code>automaticLectureModeTimeout</code> parameter is required.
automaticLectureModeTimeout	Integer	Length of time in seconds that a speaker must be talking for them to appear in full screen mode. The parameter has a range of 0 to 60 seconds. A setting of 0 seconds will cause a new speaker to appear in full screen immediately.
encryptionRequired	Boolean	The encryption setting for this conference, if you have the encryption feature key enabled. Either: <code>true</code> : Encryption is required for this conference. <code>false</code> : Encryption is optional for this conference.
contentContribution	Boolean	Whether, by default, endpoints are permitted to contribute the content channel for a conference through the mechanism of opening a content video channel. Either: <code>true</code> : Content contribution is enabled. <code>false</code> : Content contribution is disabled.
contentTransmitResolutions	String	The resolution for the content channel that will be transmitted to endpoints in this conference. One of: <code>4to3Only</code> : The MCU will encode the content and transmit it in a resolution of ratio 4:3 only. <code>16to9Only</code> : the MCU will encode the content and transmit it in a resolution of ratio 16:9 only. <code>allowAll</code> : the MCU will decide on the most optimal resolution depending on information about capabilities sent by the endpoints in the conference.

Deprecated parameters

Parameter	Type	Comments
conferenceID	String (< 32 chars)	Deprecated alternative for numericId.
endTime	DateTime.iso8601	If you do not specify an end time, then the conference will be permanent (until it is explicitly deleted). Application code should use "durationSeconds" instead.
layoutControlEnabled	Boolean	Layout control via FECC/DTMF. Deprecated by layoutControlEx.

conference.destroy

This call destroys a conference on the MCU. It is removed from the list of conferences. (Compare with `conference.end` below.) A conference can be destroyed at any time; that is, before the conference has begun, during the conference or after the conference has ended. Destroyed conferences are removed entirely from the system; this includes all future repetitions of the conference.

Parameter	Type	Comments
conferenceName	String	Name of the conference to be destroyed.

conference.end

This call ends a conference on the MCU. A conference remains in the list of conferences even after the conference has ended — until `conference.destroy` (above) is called. In particular, this can be used to end an instance of a conference without deleting all future repetitions.

Parameter	Type	Comments
conferenceName	String	Name of the conference to be ended.

conference.enumerate

Returns some or all conferences scheduled, running or completed on the MCU.

Parameter	Type	Comments
enumerateID (optional)	String	The value returned by the last enumeration call. If it is omitted, a new enumeration is started.
enumerateFilter	String	A filter expression.

Valid expressions within the enumerate filter are as follows:

Expression	Type	Comments
active	Boolean	True if the conference is active.
completed	Boolean	True if the conference has finished.
scheduled	Boolean	True if the conference is a scheduled conference (regardless of if it has been completed or not).

This call returns:

Response	Type	Comments
enumerateID (optional)	String	The value which should be used in the next call to get the next set of data. If this is omitted, no further data is available from the MCU.
conferences	Array	See below for details.
joinAudioMuted	Boolean	Audio mute on join.
joinVideoMuted	Boolean	Video mute on join.
layoutControlEnabled	Boolean	Layout control via FECC/DTMF.

The array "conferences" contains structures with the following fields:

Field	Type	Comments
conferenceName	String	
conferenceType	String	One of: scheduled or ad_hoc.
uniqueId	Integer	An ID unique among all scheduled and ad hoc conferences, each instantiation of a scheduled conference will have the same uniqueId.
conferenceActive	Boolean	Indicates whether conference is currently active.
description	String	Extra user-specified information about the conference
pin	String	The security PIN.
guestPin	String	Security PIN for a guest.
numericId	String	

Field	Type	Comments
guestNumericId	String	
registerWithGatekeeper	Boolean	
registerWithSIPRegistrar	Boolean	Registers this conference with the SIP registrar.
multicastStreamingEnabled	Boolean	
unicastStreamingEnabled	Boolean	
h239Enabled	Boolean	
h239Important	Boolean	Whether the H.239 channel is set to be important.
locked	Boolean	Whether the conference is locked or unlocked.
maximumAudioPorts	Integer	These fields set the limit on the number of audio (audio only) and video (video + audio) ports for the conference. The “reserved” values are for port reservation mode, whereas the “maximum” figures apply to non-reserved mode (and will be absent if no limits have been configured).
maximumVideoPorts	Integer	
reservedAudioPorts	Integer	
reservedVideoPorts	Integer	If the value of the reservedAudioPorts parameter exceeds the total number of available audio ports, the MCU will reserve all available audio ports and reserve the remainder as video ports. For example, if the MCU has 20 video and 20 audio only ports and a request is made to reserve 30 audio only ports, the MCU will reserve 20 audio only ports and 10 video ports.
customLayoutEnabled	Boolean	True if a custom layout has been enabled for this conference.
customLayout (optional)	Integer	The index of the custom layout. This is only present if the custom layout is enabled. See <i>Conference layouts</i> on page 78 for a list of index values.
newParticipantsCustomLayout	Boolean	True if new participants will use the conference custom layout.
private	Boolean	True if this conference is a private conference.

Field	Type	Comments
chairControl	String	The chair control setting for this conference. See the chairControl description in conference.create for an explanation of the values this parameter can take.
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available always : DTMF tones are always muted never : DTMF tones are never muted
layoutControlEx	String	Controls whether and how view layout can be controlled. One of: disabled : Layout control is disabled. feccOnly : Layout can be controlled via FECC. feccAndDtmf : Layout can be controlled via FECC or DTMF if FECC is unavailable.
dtmfMuteControl	Boolean	Controls whether the user can mute audio using *6. Either: true : participant can mute audio using *6 false : participant cannot mute audio using *6.
automaticLectureModeEnabled	Boolean	Automatic lecture mode show the speaker full screen. Either: true : automatic lecture enabled. false : automatic lecture mode disabled. If automatic lecture mode is enabled, the <code>automaticLectureModeTimeout</code> parameter is required.
automaticLectureModeTimeout	Integer	Length of time in seconds that a speaker must be talking for them to appear in full screen mode. The parameter has a range of 0 to 60 seconds. A setting of 0 seconds will cause a new speaker to appear in full screen immediately.

Field	Type	Comments
encryptionRequired	Boolean	The encryption setting for this conference, if you have the encryption feature key enabled. Either: <code>true</code> : Encryption is required for this conference. <code>false</code> : Encryption is optional for this conference.
contentContribution	Boolean	Whether, by default, endpoints are permitted to contribute the content channel for a conference through the mechanism of opening a content video channel. Either: <code>true</code> : Content contribution is enabled. <code>false</code> : Content contribution is disabled.

The following timing fields will be present for scheduled conferences only

startTime	dateTime.iso8601	The time at which the conference started at or will start at.
durationSeconds	Integer	How long each repeating instance of the conference should last for. If absent, the conference is permanent.
repetition(optional)	String	One of <code>none</code> , <code>daily</code> , <code>weekly</code> , <code>everyTwoWeeks</code> or <code>monthly</code> .
weekDay(optional)	String	Present if repetition is <code>monthly</code> . One of <code>monday</code> , <code>tuesday</code> , <code>wednesday</code> , <code>thursday</code> , <code>friday</code> , <code>saturday</code> or <code>sunday</code> .
whichWeek (optional)	String	Present if repetition is <code>monthly</code> . One of: <code>first</code> (the first X of the month, where X is the day specified by weekday), <code>second</code> , <code>third</code> , <code>fourth</code> , or <code>last</code> (i.e. last X of the month).
weekDays (optional)	String	A comma separated string of weekdays (i.e. any of <code>monday</code> , <code>tuesday</code> , <code>wednesday</code> , <code>thursday</code> , <code>friday</code> , <code>saturday</code> or <code>sunday</code>), e.g. <code>monday,wednesday,friday</code> . This field is present when repetition is <code>weekly</code> or <code>everyTwoWeeks</code> .
terminationType(optional)	String	One of: <code>noTermination</code> , <code>afterNRepeats</code> or <code>endOnGivenDate</code> .
terminationDate(optional)	dateTime.iso8601	If terminationType is <code>endOnGivenDay</code> , this is the day that the conference repetition will end on.

The following timing values will be present for active conferences only

Field	Type	Comments
activeStartTime	dateTime.iso8601	If the conference is currently active, these fields show the time span of the current activation. If the conference is permanent then "activeEndTime" will be absent.
activeEndTime	dateTime.iso8601	
activeConferenceId	string	A unique ID for the active instance of this conference; this conference will have this ID even if, for example, the conference is renamed while active, but each scheduled instance of this conference will have a different activeConferenceId.

conference.floor.modify

This call modifies the status of the conference floor control.

Response	Type	Comments
conferenceName	String	The name of the conference to affect.
floorStatus	String	'inactive' if no floor status is enabled, 'active' if an endpoint has floor control, or 'assign' if the chair or API had assigned floor control.
Optional parameters		
floorParticipant	Struct	If floorStatus is not 'inactive' then the participant identification structure will contain participantName, participantType, and participantProtocol.

conference.floor.query

This call queries the status of the conference floor control.

Response	Type	Comments
enabled	Boolean	Whether box wide and conference settings allow floor control to be assigned.
floorStatus	String	'inactive' if no floor status is enabled, 'active' if an endpoint has floor control, or assigned if the chair or API had assigned floor control.
Optional parameters		
floorParticipant	Struct	If floorStatus is not 'inactive' then the participant identification structure will contain participantName, participantType, and participantProtocol.

conference.modify

This call modifies the settings of an existing conference. Conferences created through the management API will appear in the list of conferences accessible via the web interface. Therefore, the API can be used to modify conferences scheduled via the web interface, and vice versa.

Parameter	Type	Comments
conferenceName	String (< 32 chars)	Name of the conference to modify.
newConferenceName (optional)	String (< 32 chars)	If present, the conference will be renamed to specified value.
Optional parameters		
numericId	String (< 32 chars)	Fields as per conference.create described above. These fields can only be used for conferences which are not of type ad_hoc.
pin	String	
registerWithGatekeeper	Boolean	
registerWithSIPRegistrar	Boolean	
startTime	dateTime.iso8601	
durationSeconds	Integer	
Description	String	
multicastStreamingEnabled	Boolean	
unicastStreamingEnabled	Boolean	
h239Enabled	Boolean	
private	Boolean	
reservedVideoPorts	Integer	
reservedAudioPorts	Integer	
maximumVideoPorts	Integer	
maximumAudioPorts	Integer	
repetition	String	
weekDay	String	
whichWeek	String	

Parameter	Type	Comments
weekDays	String	
terminationType	String	
terminationDate	dateTime.iso8601	
numberOfRepeats	Integer	
h239Important	Boolean	Sets the H.239 channel to be important.
locked	Boolean	Locks or unlocks the conference.
customLayoutEnabled	Boolean	
layoutControlEx	String	
newParticipantsCustomLayout	Boolean	Fields, as for the conference.create call above.
customLayout	Integer	
chairControl	String	
enforceMaximumAudioPorts	Boolean	
enforceMaximumVideoPorts	Boolean	Assumed to be true if absent. These can be set to false in order to specify no limit on the number of audio/video ports.
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available always : DTMF tones are always muted never : DTMF tones are never muted
dtmfMuteControl	Boolean	Controls whether the user can mute audio using *6. Either: true : participant can mute audio using *6 false : participant cannot mute audio using *6.
automaticLectureModeEnabled	Boolean	Automatic lecture mode shows the speaker full screen. Either: true : automatic lecture enabled. false : automatic lecture mode disabled. If automatic lecture mode is enabled, the

Parameter	Type	Comments
automaticLectureModeTimeout	Integer	Length of time in seconds that a speaker must be talking for them to appear in full screen mode. The parameter has a range of 0 to 60 seconds. A setting of 0 seconds will cause a new speaker to appear in full screen immediately.
encryptionRequired	Boolean	The encryption setting for this conference, if you have the encryption feature key enabled. Either: <code>true</code> : Encryption is required for this conference. <code>false</code> : Encryption is optional for this conference.
contentContribution	Boolean	Whether, by default, endpoints are permitted to contribute the content channel for a conference through the mechanism of opening a content video channel. Either: <code>true</code> : Content contribution is enabled. <code>false</code> : Content contribution is disabled.
contentTransmitResolutions	String	The resolution for the content channel that will be transmitted to endpoints in this conference. One of: <code>4to3Only</code> : The MCU will encode the content and transmit it in a resolution of ratio 4:3 only. <code>16to9Only</code> : the MCU will encode the content and transmit it in a resolution of ratio 16:9 only. <code>allowAll</code> : the MCU will decide on the most optimal resolution depending on information about capabilities sent by the endpoints in the conference.

Deprecated parameters

Parameter	Type	Comments
oldConferenceName	String (< 32 chars)	Deprecated conference renaming scheme – new code should use <code>conferenceName</code> and <code>newConferenceName</code> as above.
conferenceName	String (< 32 chars)	
conferenceID	String	The unique ID for the conference.

Parameter	Type	Comments
endTime	dateTime.iso8601	If you do not specify an end time, then the conference will be permanent (until it is explicitly deleted). Application code should use “durationSeconds” instead.
layoutControlEnabled	Boolean	Layout control via FECC/DTMF. Deprecated by layoutControlEx.

conference.paneplacement.modify

Modifies the pane placement configuration of a particular conference.

Parameter	Type	Comments
conferenceName	String	Name of the conference to be modified.
Optional parameters		
enabled	Boolean	true to enable pane placement, false to disable.
panes	Array	See below for details.

The panes array contains structures which define a specific pane and its contents. If a pane index is not present in the array, then that pane will remain unchanged. Participant identification is as returned in participant.enumerate.

Field	Type	Comments
index	Integer	The index of the pane to be changed.
type	String	Any one of: <ul style="list-style-type: none"> default - the default behaviour blank - a blank window loudest - the current loudest speaker rolling – shows a sequence of conference participants, changing according to the configured rolling interval h239 - the h239 content channel participant - a participant as described in the three optional fields.

Optional parameters		
participantType	String	Participant identification. Only required if type is participant, these identify a specific participant.
participantProtocol	String	
participantName	String	

Because not all panes are guaranteed to be changed, this call returns the following structure:

Response	Type	Comments
panesModified	Integer	The number of panes successfully modified. This will be the number of elements in the panes array on complete success, and zero if there is no panes array.

conference.paneplacement.query

Queries the current pane placement configuration.

Parameter	Type	Comments
conferenceName	String	The name of the conference to be queried.

This returns a struct containing the following response fields:

Response	Type	Comments
enabled	Boolean	true if pane placement is enabled and in use; false otherwise.

Optional parameters		
panes (optional)	Array	This is only present if enabled above is true. The struct definition is as below.

The panes array contains structures with the following format:

Field	Type	Comments
type	String	Any one of: default - the default behaviour blank - a blank window loudest - the current loudest speaker rolling – shows a sequence of conference participants, changing according to the configured rolling interval h239 - the h239 content channel participant - a participant as identified below.
index	Integer	The index of this pane.

Optional parameters		
participantType	String	Participant identification. Only present if this pane contains a specific participant.
participantProtocol	String	
participantName	String	

conference.status

Returns information about a named conference on the MCU.

Parameter	Type	Comments
conferenceName	String	Name of the conference for which information is required.

This call returns a struct as described in [conference.enumerate](#) above, containing information about the conference indicated.

A fault code of “no such conference” is returned if there is no conference with the given name.

conference.streaming.modify

Modifies the current layout for streaming viewers for a conference.

Parameter	Type	Comments
conferenceName	String	Name of the conference whose layout is to be modified.
Optional parameters		
cpLayout	String	The video layout seen by the participant. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts..
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
focusType	String	One of: voiceActivated, h239 or participant.
focusParticipant	Struct	A participant identification structure (i.e. with conferenceName, participantName, participantProtocol and participantType members). Should only be present if focusType is participant.

conference.streaming.query

Returns details on the current state of streaming viewers for a conference.

Parameter	Type	Comments
conferenceName	String	Name of the conference from which streaming information is required.

This will return a structure with the following fields:

Response	Type	Comments
unicastViewers	Integer	The number of unicast streaming viewers.
multicastViewers	Integer	The number of multicast streaming viewers.
audioRTCPReceiverReports	Integer	The number of RTCP receiver reports for the audio streams seen by the MCU.
audioRTCPSenderReports	Integer	The number of RTCP sender reports for the audio streams seen by the MCU.
audioRTCPOther	Integer	The number of other RTCP packets seen for the audio streams.
audioRTCPPacketsSent	Integer	The number of RTCP packets send by the MCU.
videoRTCPReceiverReports	Integer	As for the audio equivalents.
videoRTCPSenderReports	Integer	
videoRTCPOther	Integer	
videoRTCPPacketsSent	Integer	
currentLayout	Integer	The actual layout in use for the video stream being sent by the MCU to streaming viewers. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts.
layoutSource	String	One of family<x>, conferenceCustom, or participantCustom, and describes the reason for the current layout.
focusType	String	One of: <code>participant</code> , <code>voiceActivated</code> or <code>h239</code> .
focusParticipant	Struct	A participant identification structure (i.e. with <code>conferenceName</code> , <code>participantName</code> , <code>participantProtocol</code> and <code>participantType</code> members). Should only be present if <code>focusType</code> is <code>participant</code> .
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
Optional parameters		
audioStreams (optional)	Array	An array of stream structs (defined below). These

Response	Type	Comments
videoStreams (optional)	Array	are only present if there are any streams of either type currently in use.

The stream structures used in the audioStreams/videoStreams responses above have the following fields:

Field	Type	Comments
codec	String	The codec in use, or <code>other</code> for undefined codecs.
count	Integer	The number of users of this codec.
Optional parameters		
bitRate	Integer	The bit rate of this stream in bits/second. This is only present for video streams with a defined codec.
width	Integer	The maximum width and height of this stream. Only present for defined video streams
height	Integer	

This call will return a fault code of "no such conference" if there is no *active* conference with the given name, regardless of the presence a configured but inactive conference of that name.

conferenceme.query

Queries for information about ConferenceMe.

Response	Type	Comments
enabled	Boolean	True if ConferenceMe is enabled and in use; false otherwise. (If no feature key is installed this will be false).
mediaOverTcp	Boolean	Allow fall-back to media via TCP. (If no feature key is installed this will be false).
maxBitRateFromMCU	Number	Maximum bandwidth from the MCU. (If no feature key is installed this will be 0).
maxBitRateToMCU	Number	Maximum bandwidth to the MCU. (If no feature key is installed this will be 0).

device.health.query

Returns the current status of the MCU, such as health monitors and CPU load.

Response	Type	Comments
cpuLoad	Integer	The CPU load, as a percentage.
mediaLoad	Integer	Loads for the media processors (total, and split between audio and video) as percentage values.
audioLoad	Integer	
videoLoad	Integer	
fanStatus	String	One of ok, outOfSpec or critical.
fanStatusWorst	String	
temperatureStatus	String	
temperatureStatusWorst	String	
rtcBatteryStatus	String	
rtcBatteryStatusWorst	String	
voltagesStatus	String	
voltagesStatusWorst	String	
operationalStatus	String	One of active, shuttingDown or shutDown.

device.network.query

This call queries the MCU for information about port configuration. The call takes no parameters. The response returns the following:

Parameter	Type	Comments
portA	Array (see below)	Contains the configuration and status for port A.
portB	Array (see below)	Contains the configuration and status for port B.

The format for the two structures above is:

Field	Type	Comments
enabled	Boolean	true if the port is enabled, otherwise false.
linkStatus	Boolean	true if the link is up, false if the link is down.
Speed	Integer	One of 10, 100 or 1000, in Mbps.
fullDuplex	Boolean	true if full duplex enabled, false if half.
macAddress	String	A 12 character string, no separators.
packetsSent	Integer	Stats from the web interface. It is worth noting that all these values are 32 bit signed integers, and thus may wrap.
packetsReceived	Integer	
multicastPacketsSent	Integer	
multicastPacketsReceived	Integer	
bytesSent	Integer	
bytesReceived	Integer	
queueDrops	Integer	
collisions	Integer	
transmitErrors	Integer	
receiveErrors	Integer	
bytesSent64	String	64 bit versions of the above stats, using a string rather than an integer.
bytesReceived64	String	
Optional parameters		
hostName	String	The host name of this port.
dhcp	Boolean	true if configured by DHCP, otherwise false .
ipAddress	String	a.b.c.d format.
subnetMask	String	a.b.c.d format.

Field	Type	Comments
defaultGateway	String	a.b.c.d format.
domainName	String	The domain name of this port.
nameServer	String	a.b.c.d format.
nameServerSecondary	String	a.b.c.d format.

All fields above marked optional will only be returned if the interface has been enabled and has been configured.

device.query

Queries for information about the MCU device. There are no parameters passed with this call. The call response returns the following:

Parameter	Type	Comments
currentTime	dateTime.iso8601	The system's current time (UTC).
restartTime	dateTime.iso8601	The date and time at which the system was last restarted.
serial	String	The serial number of the MCU.
softwareVersion	String	The software version of the running software.
buildVersion	String	The build version of the running software.
model	String	The model of this MCU, e.g. "..... MCU 4210".
apiVersion	String	The version number of the API implemented by this MCU.
activatedFeatures	Array	Currently, only contains the string "feature" with a short description of the feature.
totalVideoPorts	Integer	The total number of video ports on the MCU.
totalAudioOnlyPorts	Integer	The total number of additional audio-only ports on the MCU.
totalStreamingAndContentPorts	Integer	The total number of streaming and content ports on the MCU. Only provided if non-zero.
portReservationMode	String	enabled or disabled, depending on the Media Port Reservation configuration setting. Only present on MCU products.
maxVideoResolution	String	One of <code>cif</code> or <code>4cif</code> .
videoPortAllocation	Array	Array of structs containing a type element and a count element. The type element can be one of <code>sd</code> , <code>hd</code> , or <code>hd1080p</code> . The count element is an integer indicating the number of video ports allocated.

device.restartlog.query

Returns the restart log - also known as the system log on the web interface.

Response	Type	Comments
Log	Array	Contains the restart log in structures as described below.

The "Log" array consists of structures which contain the following fields.

Field	Type	Comments
Time	dateTime.iso8601	The time of the last reboot.
Reason	String	The reason for the reboot (one of unknown, User requested shutdown or User requested upgrade).

gatekeeper.query

Retrieves the gatekeeper settings and current status for an MCU. Takes no parameters.

Response	Type	Comments
gatekeeperUsage	String	One of disabled, enabled or required.

Optional parameters – only present if gatekeeperUsage is not disabled.

address	String	The address of the gatekeeper.
dnsStatus	String	The status of the DNS resolution: one of inProgress, resolved or failed.
ip	String	If "dnsStatus" is resolved. Contains the IP address of the gatekeeper.
activeRegistrations	Integer	The number of active registrations.
pendingRegistrations	Integer	The number of registrations in progress.
registrationPrefix	String	The registration prefix used by the MCU.
h323ID	String	The h323 ID used by the MCU.
mcuServicePrefix	String	The service prefix used by the MCU.
scheduledConferenceIDRegistration	String	The value enabled or disabled; corresponds to web interface "ID registration for scheduled conferences" option.

Response	Type	Comments
h323IDStatus	String	The current status of the ID/service prefix registration process. One of idle, registering, registered, deregistering, pendingReregistration, waitingRetry, noID or idTooLong.
mcuServicePrefixStatus	String	

gateway.enumerate

Enumerates configured H.323 gateways on an MCU.

Parameter	Type	Comments
Optional parameters		
enumerateID	String	The value returned by the last enumeration call. If it is omitted, a new enumeration is started.

This call returns:

Response	Type	Comments
gateways	Array	See below for details.
Optional parameters		
enumerateID	String	The value which should be used in the next call to get the next set of data. If this is omitted, no further data is available from the MCU.

The array "gateways" contains structures with the following fields:

Field	Type	Comments
name	String	The name of the configured gateway.
address	String	The gateway's E.164 directory number, hostname or IP address.
conferencingParameters	Array	See below for details.

The array conferencingParameters contains structures with the following fields:

Field	Type	Comments
useDefaultMotionSharpness	Boolean	If <code>true</code> , this endpoint will use box-wide default motion sharpness settings.
minFrameRateMotionSharpness	Integer	Only present if "useDefaultMotionSharpness" is false. Specifies the minimum frame rate for this endpoint.

Field	Type	Comments
maxMediaTxBitRate	Integer	Max media transmit bit rate.
maxMediaRxBitRate	Integer	Max media receive bit rate.

participant.add

Adds a participant to a conference. All participants in a conference must have a “participantName” that is unique to the conference but it need not be unique across all conferences.

Participants can be added before or during a conference. A participant which is added at any time via the API will be added to the configured list of participants, and thus will be called at the start of the conference by the MCU for any conference which has any sort of repetition; to avoid this, a participant must be removed directly using participant.remove.

Note: If a participantName matches the name of an endpoint in the list of configured endpoints (go to **Endpoints** in the web interface) the two are treated as unrelated. This is because in the web interface named, configured, endpoints have the participantType value by_name, whereas API participants are of type by_address.

Parameter	Type	Comments
conferenceName	String	The name of the conference to which to add the participant.
participantName	String	The name of the participant to be added. This must be a unique value, i.e. not the same as any existing participant. Note that for ad_hoc participants, this is not used, but must be present otherwise.
Optional parameters		
participantProtocol	String	If present, must be h323, sip or vnc – these are the only protocols that the API can currently use.
participantType	String	If present, must be by_address or ad_hoc. Note that if this conference is an ad_hoc conference, this value should also be ad_hoc. Ad hoc participants can only be added to active conferences.
address	string (< 32 chars)	The participant's E.164 directory number, hostname or IP address.
gatewayAddress	string (< 32 chars)	IP address or hostname of an H.323 gateway.
useSIPRegistrar	Boolean	Whether to use a registrar in making this a call. (Ignored if the protocol is not sip)
transportProtocol	String	One of: default, tcp, udp or tls. (Ignored if the protocol is not SIP).

Parameter	Type	Comments
password	String	The password for VNC endpoints.
deferConnection	Boolean	If true, don't call out to this participant immediately, but wait for a "participant.connect" command.
All of the following parameters are optional, and control the conferencing behaviour of the MCU with respect to the endpoint in question; for example, the maximum resolution of the video streams used, or whether the participant is able to control their conference view layout.		
maxBitRateToMCU	Integer	The maximum bit rate to the MCU specified as kbit/s.
maxBitRateFromMCU	Integer	The maximum bit rate from the MCU specified as kbit/s.
motionSharpnessTradeoff	String	One of <code>default</code> (to use the global default setting), <code>preferMotion</code> , <code>preferSharpness</code> and <code>balanced</code> .
displayNameOverrideStatus	Boolean	If true, use the specified "displayNameOverrideValue" text as the participant's display name during the conference.
displayNameOverrideValue	string (< 32 chars)	Value to use as the participant's display name (if "displayNameOverrideStatus" set to true).
cpLayout	String	This sets the initial conference view layout for the video sent to this participant. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts.
layoutControlEx	String	Controls whether and how view layout can be controlled. One of: <code>disabled</code> : Layout control is disabled. <code>feccOnly</code> : Layout can be controlled via FECC. <code>feccAndDtmf</code> : Layout can be controlled via FECC or DTMF if FECC is unavailable.
audioRxMuted	Boolean	1 (true) means that audio from this participant will not be heard by other conference participants.
audioRxGainMode	String	One of: <code>none</code> – no extra gain applied <code>automatic</code> – automatic gain control applied <code>fixed</code> – fixed number of dBs of gain applied.
audioRxGainMillidB	Integer	If audio gain mode is fixed, this is the number of decibels of gain applied, multiplied by 1000, and can be a negative value.
videoRxMuted	Boolean	true means that video from this participant will not be seen by other conference participants.

Parameter	Type	Comments
videoTxWidescreen	Boolean	If true, the video sent to this participant will be in a form suitable for a widescreen (16:9) display.
videoTxMaxResolution	String	One of: cif, 4cif or max.
videoRxMaxResolution	String	Same as above.
autoConnect	Boolean	If this is true and a participant whose E.164, DNS, or IP address matches this participant's address dials into the MCU, it will be moved directly to this conference. In order to stop the MCU dialing out to the participant, as the conference starts, use deferConnection.
autoDisconnect	Boolean	When set to true, the participant will be disconnected from the conference if another participant disconnects and only participants configured to be automatically disconnected remain in the conference.
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
dtmfSequence	String	This sets the DTMF sequence used for dialing with an audio bridge.
linkType	String	Currently, must be one of: cascadeSlaveToMaster or default. (Currently this is only taken note of if participantType is by_address).
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available always : DTMF tones are always muted never : DTMF tones are never muted default : The setting specified for this parameter when the conference is created. All new participants take this setting.

Deprecated parameters

Parameter	Type	Comments
layoutControlEnabled	Boolean	Controls whether this participant is able to change the conference view layout that they see; 1 (true) means that the participant can change the layout using FECC or DTMF, 0 (false) means that they cannot. Deprecated in favour of layoutControlEx.

participant.connect

Used primarily for API-configured participants with deferConnection set to TRUE, but can also be used to reconnect disconnected participants.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	

participant.diagnostics

Returns diagnostic information about a given participant.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	

The call response returns the following:

Response	Type	Comments
videoTxFrameRate	Integer	
videoRxFrameRate	Integer	
videoRxFramesReceived	Integer	
videoTxChannelBitRate	Integer	
videoTxSelectedBitRate	Integer	
videoTxActualBitRate	Integer	

Response	Type	Comments
videoTxBitRateLimitReason	String	One of: notLimited, viewedSize, quality, aggregateBandwidth, flowControl or endpointLimitation.
videoRxChannelBitRate	Integer	
videoRxSelectedBitRate	Integer	
videoRxActualBitRate	Integer	
videoRxBitRateLimitReason	String	One of: notLimited, viewedSize, quality, aggregateBandwidth, flowControl or endpointLimitation.
videoTxWidth	Integer	
videoTxHeight	Integer	
videoTxInterlaced	Boolean	
videoRxWidth	Integer	
videoRxHeight	Integer	
videoRxInterlaced	Boolean	
videoTxReportedLost	Integer	
videoRxCodec	String	
videoRxJitter	Integer	
audioTxReportedLost	Integer	
videoTxSent	Integer	
audioRxLost	Integer	
audioRxReceived	Integer	
videoTxCodec	String	
videoRxFramesReceivedWithErrors	String	
audioTxSent	Integer	

Response	Type	Comments
videoRxReceived	Integer	
videoRxLost	Integer	

participant.disconnect

This call causes the MCU to tear down its connection to the specified participant, if such a connection exists. This is different from `participant.remove` above because:

- ▶ In the case of configured participants, it does not remove the configuration (thus allowing later re-connection with `participant.connect`).
- ▶ In the case of ad hoc participants, it does not remove the record of the previous connection.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	

participant.enumerate

Returns data about participants in conferences on the MCU. Several calls may be required to receive data about all participants; see the notes on enumerateID below.

Note: The participant.enumerate call has deprecated behaviour if there is no operationScope parameter in the call. See *participant.enumerate* on page 68.

Parameter	Type	Comments
operationScope	array of strings	This should contain none, either or both of currentState or configuredState. If currentState is present, the active configuration of each participant is returned by the MCU in the currentState structure. If configuredState is present, the stored configuration is returned in the configuredState structure
Optional parameters		
enumerateFilter	String	An enumerate filter string.
enumerateID	String	The value returned by the last enumeration call. If this parameter is omitted, a new enumeration is started.

Valid expressions within the enumerate filter are as follows:

Expression	Type	Comments
connected	Boolean	True if the participant is currently connected to a conference.
disconnected	Boolean	True if the participant has been connected to a conference, but is now disconnected.
connecting	Boolean	True if the scheduled participant is in the process of connecting.

Note: A participant that has not yet connected to a conference (for example, they have deferred connection specified) is neither connected nor disconnected.

This call response returns the following:

Response	Type	Comments
participants	Array	See below for details.
Optional responses		
enumerateID	String	The value which should be used in the next call to get the next set of data. If this is not present, there is no further data available from the MCU.

The array “participants” contains the following structures:

Field	Type	Comments
participantName	string	Participant identification as described above.
participantProtocol	String	
participantType	String	
conferenceName	String	
autoAttendantUniqueld	String	
connectionUniqueld	Integer	Corresponds to the uniqueld returned by a conference or autoattendant.
Optional parameters		
currentState	Array	The current state of the participant as used by the MCU. Details of this struct are given below. This is only present if requested in the operationScope
configuredState	Array	The stored configuration of the participant, if any. Details of this struct are given below. This is only present if requested in the operationScope

If the endpoint is not configured, the configuredState structure is empty; otherwise the configuredState structure contains the following entries:

Field	Type	Comments
address	String	The address used to connect to the remote endpoint in question. Only returned when the address is known.
gatewayAddress	String	
useSipRegistrar	Boolean	Whether to use a registrar in making this a call.
transportProtocol	String	One of: default, tcp, udp or tls.
password	String	The password for vnc endpoints.
deferConnection	Boolean	true if this participant's connection is being deferred.
displayNameOverrideStatus	Boolean	Indicates whether the displayName value is the result of being overridden.
maxBitRateToMCU	Integer	As for “participant.add”; in kbps.

Field	Type	Comments
maxBitRateFromMCU	Integer	
motionSharpnessTradeoff	String	One of <code>default</code> (if set to the global default setting), <code>preferMotion</code> , <code>preferSharpness</code> and <code>balanced</code> .
audioRxMuted	Boolean	
audioRxGainMode	String	One of <code>fixed</code> , <code>none</code> or <code>automatic</code> .
audioRxGainMillidB	Integer	Only returned if <code>audioRxGainMode</code> is <code>fixed</code> .
videoRxMuted	Boolean	
videoTxWidescreen	Boolean	
layoutControlEx	Boolean	Whether and how view layout can be controlled. One of: <code>disabled</code> : Layout control is disabled. <code>feccOnly</code> : Layout can be controlled via FECC. <code>feccAndDtmf</code> : Layout can be controlled via FECC or DTMF if FECC is unavailable
cpLayout	String	The layout configured for this participant. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts.
autoConnect	Boolean	Whether or not participants matching this address should be automatically connected to the conference.
autoDisconnect	Boolean	Whether or not the participant should be automatically disconnected from the conference when all other participants disconnect.
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
linkType	String	One of: <code>cascadeSlaveToMaster</code> or <code>default</code> .
dtmfSequence	String	This sets the DTMF sequence used for dialing with an audio bridge.

Field	Type	Comments
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available always : DTMF tones are always muted never : DTMF tones are never muted default : The setting specified for this parameter when the conference is created. All new participants take this setting.

The `currentState` structure contains the following responses, if present.

Field	Type	Comments
address	String	The address used to connect to the remote endpoint in question. Only returned when the address is known (i.e. the participant was or is to be called out by the MCU, or the participant is an ad_hoc participant calling in which provided an address).
gatewayAddress	String	
ipAddress	String	The IP address to which the MCU is connected for this endpoint - this will usually be the endpoint itself, but may be a gatekeeper or gateway. Only present for active participants.
displayName	string	The name used by the endpoint to identify itself. This may be different to the <code>participantName</code> . Only available after the participant has connected.
displayNameOverrideStatus	Boolean	Indicates whether the <code>displayName</code> value is the result of being overridden.
maxBitRateToMCU	Integer	As for "participant.add"; in kbps.
maxBitRateFromMCU	Integer	
motionSharpnessTradeoff	String	One of <code>default</code> (if set to the global default setting), <code>preferMotion</code> , <code>preferSharpness</code> and <code>balanced</code> .
callState	String	One of: <code>dormant</code> , <code>alerting</code> , <code>connected</code> or <code>disconnected</code> . When <code>dormant</code> , the <code>callDirection</code> field is not returned.
connectTime	dateTime.iso8601	Only returned after the participant is connected. This value is always present if the call state is <code>connected</code> . It may or may not be defined for participants in the <code>disconnected</code> state, depending on whether they were ever connected.

Field	Type	Comments
disconnectTime	dateTime.iso8601	Only returned after the participant has disconnected.
disconnectReason	String	Only returned after the participant has disconnected; this contains one of the disconnect reason strings given in section 8.
connectPending	Boolean	true if sending a "participant.connect" command for this participant will cause either the initial connection to that endpoint (in the event that it was configured with "deferConnection" set) or a re-connection to that endpoint (in the event that it has disconnected).
audioRxCodec	String	
audioRxLost	Integer	
audioRxReceived	Integer	
audioTxCodec	String	
audioTxReportedLost	Integer	
audioTxSent	Integer	
audioRxMuted	Boolean	
audioRxGainMode	String	One of fixed , none or automatic .
audioRxGainMillidB	Integer	Only present if the audioRxGainMode is fixed .
audioTxMuted	Boolean	true if audio is not being transmitted to this participant.
videoRxCodec	String	
videoRxLost	Integer	
videoRxReceived	Integer	
videoTxCodec	String	
videoTxReportedLost	Integer	
videoTxSent	Integer	

Field	Type	Comments
videoRxMuted	Boolean	
videoTxWidescreen	Boolean	
autoDisconnect	Boolean	
important	Boolean	
activeSpeaker	Boolean	true if this participant is currently the active speaker in the conference.
lecturer	Boolean	Whether this participant is the lecturer or not. Either true: This participant is the lecturer. false: This participant is not the lecturer.
layoutControlEx	Boolean	Whether and how view layout can be controlled. One of: disabled: Layout control is disabled. feccOnly: Layout can be controlled via FECC. feccAndDtmf: Layout can be controlled via FECC or DTMF if FECC is unavailable
activeConferenceId	String	The active conference ID of the current conference – see conference.enumerate for details of this field. This field is only present if the participant is currently in an active conference.
currentLayout	Integer	The actual layout in use for the video stream being sent by the MCU to this participant. This parameter will not be present if the participant is in an auto attendant rather than a conference, or if the MCU is not currently transmitting video to the participant in question. Refer to <i>Conference layouts</i> on page 78 for the full list of available layouts..
layoutSource	String	This will be one of <code>family<x></code> , <code>conferenceCustom</code> , or <code>participantCustom</code> , and describes the reason for the current layout. This parameter is only present if the <code>currentLayout</code> parameter is also present, i.e. if the participant is in an active conference.
callDirection	String	Either <code>incoming</code> or <code>outgoing</code> . When the <code>callState</code> field is dormant, the <code>callDirection</code> field is not returned.
previewURL	String	The location of the preview image; this is not a complete URL, and requires a prefix of

Field	Type	Comments
		http://<hostname> (where hostname is the hostname of this MCU) before it is used.
focusType	String	One of: participant, voiceActivated or h239.
focusParticipant	Struct	Only present if focusType is <code>participant</code> . This structure contains participant identification members (i.e. conferenceName, participantName, participantType and participantProtocol).
callIdentifier	base64	The H.323 id of this caller.
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
autoAttendantConfiguredName	String	If this participant is connected to an auto attendant, this field holds the name of that auto attendant: the value will change as the user navigates through an MCU's configured menu structure.
mediaEncryption	String	One of the following: encrypted : all media channels to and from this endpoint are encrypted unencrypted : all media channels to and from this endpoint are unencrypted mixed : some channels are encrypted and some not unknown : none of the above; this may occur when a participant has very recently connected and no media channels have been established yet
audioRxEnergyMillidB	Integer	The measured energy of a participant's audio sent to the MCU. Typically this will be a negative value in the range -30000 (-30dB for very quiet) and 0 (very loud).
audioRxMutedRemotely	Boolean	Whether this endpoint is muted remotely.
packetLossWarning	Boolean	This will be true if any packet loss has been seen within the last 15 seconds.
packetLossCritical	Boolean	This will be true if any packet loss above a certain level (5%) is seen within the last five seconds.

Note: This participant information is returned for all participants added to the conference using the `participant.add` call, even after they have disconnected. However, this information is only returned for other participants (i.e. those added via the web interface or those who dialled into the conference) whilst they are connected but not after they have disconnected.

Deprecated parameters

Parameter	Type	Comments
layoutControlEnabled	Boolean	Deprecated by layoutControlEx

participant.fecc

Controls far end camera control (FECC).

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	
direction	String	One of: up, down, left, right, zoomIn, zoomOut, focusIn, focusOut.

participant.message

Puts a message on the display of a given participant.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	
message	String (length <256)	The string to send to the participant.

Optional parameters

verticalPosition	String	Specifies where to show the message: "top", "middle" or "bottom". Message is always horizontally centered, and omitting this parameter is equivalent to choosing "middle".
------------------	--------	--

Parameter	Type	Comments
durationSeconds	Integer	The length of time, in seconds, to display the message. This defaults to 30 seconds.

participant.modify

Depending on the operationScope parameter below, this call modifies the configuration of a participant (configuredState), or the active state of a participant in a conference (activeState).

For example, if the parameter layoutControlEnabled is included in a call to participant.modify, then the effect of the call will depend on operation scope as follows:

- ▶ If operationScope is **activeState**, the active participant's ability to control their layout will immediately change, but the configured value will remain unchanged, so that if they were to reconnect later, the state of layoutControlEnabled would revert back to how it is in the configuration.
- ▶ If operationScope is **configuredState**, the participant's current ability to control their layout will be unaffected, but their configuration will be changed so that in future occurrences of the conference (or when the participant is reconnected) they will have the newly configured state.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	
operationScope	String	One of activeState or configuredState. This parameter specifies the scope of the changes to be made, be they to the configured state of an endpoint, or to the active state of a participant in a conference.
address	String	All these parameters are optional. They should not be present if operationScope is activeState. These parameters override the configured values.
gatewayAddress	String	
useSipRegistrar	Boolean	
transportProtocol	String	
password	String	
deferConnection	Boolean	
autoConnect	Boolean	

Parameter	Type	Comments
linkType	String	
maxBitRateToMCU	Integer	
maxBitRateFromMCU	Integer	
motionSharpnessTradeoff (optional)	String	One of <code>default</code> (if set to the global default setting), <code>preferMotion</code> , <code>preferSharpness</code> and <code>balanced</code> . Optional, and valid for both <code>configuredState</code> and <code>activeState</code> scopes.
displayNameOverrideStatus	Boolean	
displayNameOverrideValue	String	
cpLayout	String	
audioRxMuted	Boolean	
audioRxGainMode	String	
audioRxGainMillidB	Integer	
videoRxMuted	Boolean	
videoTxWidescreen	Boolean	
autoDisconnect	Boolean	
dtmfSequence	String	This sets the DTMF sequence used for dialing with an audio bridge.
Optional parameters		
important	Boolean	This setting should not be present if <code>operationScope</code> is "configuredState". Specifies whether this participant is important.
audioTxMuted	Boolean	This setting should not be present if <code>operationScope</code> is "configuredState".
borderWidth	Integer	0 (no border), or 1, 2, or 3 for +1/+2/+3.
focusType	String	One of: <code>voiceActivated</code> , <code>h239</code> or <code>participant</code> .
focusParticipant	Struct	A participant identification structure (i.e. with <code>conferenceName</code> , <code>participantName</code> , <code>participantProtocol</code> and <code>participantType</code> members). Should only be present if <code>focusType</code> is <code>participant</code> .

Parameter	Type	Comments
suppressDtmfEx	String	Controls the muting of DTMF tones. One of: fecc : DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available always : DTMF tones are always muted default : The setting specified for this parameter when the conference is created. All new participants take this setting.
layoutControlEx	String	Controls whether and how view layout can be controlled. One of: disabled : Layout control is disabled. feccOnly : Layout can be controlled via FECC. feccAndDtmf : Layout can be controlled via FECC or DTMF if FECC is unavailable.

If there is no operationScope parameter, the MCU will attempt to change both active and configured states. This is deprecated behaviour, and should not be relied upon.

Deprecated parameters

Parameter	Type	Comments
layoutControlEnabled	Boolean	Controls whether this participant is able to change the conference view layout that they see; 1 (true) means that the participant can change the layout using FECC or DTMF, 0 (false) means that they cannot.

participant.move

Moves a participant from one conference to another.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	
newConferenceName	String	The name of the conference to move the participant to.

This will only move an active participant. Even if this participant is preconfigured, the configuration is unchanged.

A fault code of "no such participant" is returned when the participant isn't found, "too many participants" when the conference has reached its limit and "operation failed" for other move failures such as moving an unencrypted participant into a conference which requires encryption.

participant.remove

Removes a participant from the database of configured participants, and also removes this participant from any conferences. It will also remove all records of this participant's presence in a conference.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	

participant.status

Returns information about an individual participant on the MCU.

Parameter	Type	Comments
conferenceName	String	Participant identification as described above.
autoAttendantUniqueld	String	
participantName	String	
participantProtocol	String	
participantType	String	
operationScope	Array of strings	This should contain none, either, or both of currentState or configuredState. If currentState is present, the active configuration of each participant is returned by the MCU in the currentState structure. If configuredState is present, the stored configuration is returned in the configuredState structure.

This call returns a struct as described in [participant.enumerate](#) above, containing information about the participant indicated.

A fault code of "no such participant" is returned if the participant does not exist.

sip.query

Retrieves information on SIP configuration for an MCU. Takes no parameters.

Response	Type	Comments
configuredRegistrar	String	The currently configured SIP registrar address. This corresponds to the “SIP registrar address” on the Settings > SIP web page, and will be an empty string value if there is no currently configured SIP registrar.
configuredProxy	String	The currently configured SIP proxy address. This corresponds to the “SIP proxy address” on the Settings > SIP web page, and will be an empty string value if there is no currently configured SIP proxy.
conferenceRegistration	String	This value is only present if the MCU being queried is an MCU. It will be enabled if the MCU is configured to register conferences’ numeric IDs with the configured SIP registrar, and disabled if not. The enabled value corresponds to “SIP registration settings” being set to “Allow conference registration” on the Settings > SIP page.
registrarContactURI	String	The URI used to register.
registrarContactDomain	String	The domain portion of the URI used to register.

template.modify

This call modifies the settings for conference templates (top level and ad hoc). The value for a setting of “default” is the same as specified for the equivalent parameter in the top level template.

Parameter	Type	Comments
templateNumber	Integer	Identifies the template to be modified. Either: 0: Top level template. 1: Ad hoc template.
registerWithGatekeeper	String	Registers the conference’s “numericId” with the gatekeeper. True, false, or default.
registerWithSIPRegistrar	String	Registers this conference with the SIP registrar. True, false, or default.
private	String	Determines the visibility of this conference. This parameter corresponds to the “Visibility” setting on the web UI. Either true (private), false (public), or default.

Parameter	Type	Comments
streaming	String	Specifies the type of streaming to be used on the conference. One of: none unicast multicast unicastAndMulticast default.
h239Enabled	String	Enables the H.239 protocol. One of: true: H.239 enabled. false: H.239 disabled default.
contentContribution	String	Whether, by default, endpoints are permitted to contribute the content channel for a conference through the mechanism of opening a content video channel. Equivalent to content contribution from endpoints on the web interface. Either: true: Content contribution is enabled. false: Content contribution is disabled default.
contentTransmitResolutions	String	The resolution for the content channel that will be transmitted to endpoints in this conference. Equivalent to transmitted content resolutions on the web interface. One of: 4to3only: The MCU will encode the content and transmit it in a resolution of ratio 4:3 16to9only: The MCU will encode the content and transmit it in a resolution of ratio 16:9. allowAll: The MCU will decide on the most optimal resolution depending on information about capabilities sent by the endpoints in the conference. default For ad hoc conferences, transmitted content resolution is controlled by the ad hoc conference template.
joinAudioMuted	String	Participant joins the conference without audio. One of: true: Audio is muted on joining the conference. false: Audio is not muted on joining the conference default.
joinVideoMuted	String	Participant joins the conference without video. One of: true: Video is disabled on joining the conference. false: Video is enabled on joining the conference default.

Parameter	Type	Comments
layoutControlEx	String	<p>Prevents or permits conference participants changing their view layout or focused participant using Far-end Camera Control (FECC) or DTMF tones. Equivalent to Layout control via FECC / DTMF on the web interface. One of:</p> <p>disabled: In this conference, participants will not be allowed to change their view layout using either FECC or DTMF, unless you have overridden this setting in an endpoint's individual configuration.</p> <p>feccOnly: In this conference, participants will be only be allowed to change their view layout using FECC, unless you have overridden this setting in an endpoint's individual configuration.</p> <p>feccAndDtmf: In this conference, participants will be allowed to change their view layout using FECC. If FECC is not available, this participant will be able to use DTMF</p> <p>default</p>
dtmfMuteControl	String	<p>Controls whether the user can mute audio using *6. Either:</p> <p>true: participant can mute audio using *6</p> <p>false: participant cannot mute audio using *6</p> <p>default.</p>
encryptionRequired	String	<p>The encryption setting for this conference, if you have the encryption feature key enabled. Equivalent to the setting "Encryption" on the web interface. One of:</p> <p>true: Encryption required.</p> <p>false: Encryption optional</p> <p>default</p>
suppressDtmfEx	String	<p>Controls the muting of DTMF tones. One of:</p> <p>fecc: DTMF tones are muted when DTMF is being used to control layout but far end camera control (FECC) is not available</p> <p>always: DTMF tones are always muted</p> <p>never: DTMF tones are never muted</p> <p>default</p> <p>The value "fecc" is equivalent to "When used for MCU control" on the web interface setting.</p>

Parameter	Type	Comments
automaticLectureModeEnabled	String	Automatic lecture mode gives shows the speaker full screen. Equivalent to automatic lecture mode on the web interface. Either: true: automatic lecture enabled. false: automatic lecture mode disabled. default If automatic lecture mode is enabled, the automaticLectureModeTimeout parameter is required.
automaticLectureModeTimeout	Integer	Length of time in seconds that a speaker must be talking for them to appear in full screen mode. The parameter has a range of 0 to 60 seconds. A setting of 0 seconds will cause a new speaker to appear in full screen immediately.
chairControl	String	The chair control setting for the conference. This can be: none floorControlOnly chairAndFloorControl default These values correspond to the web interface "Floor and chair control" setting values of "Do not allow floor or chair control", "Allow floor control only" and "Allow floor and chair control" respectively. If not specified, the chair control setting for the new conference will be "Allow floor control only".

template.status

The `template,status` call returns a structure containing all the settings for the selected template.

Parameter	Type	Comments
templateNumber	String	Selects the required template to query the status. Either: 0: Top level template. 1: Ad hoc template.

template.enumerate

The `template.enumerate` function returns an array of the template status structures showing the settings of all templates. The call does not take any parameters.

Deprecated calls

The calls listed below were supported in software version 1.0 of the MCU 4200 Series Management API but have since been superseded.

API call	Superseded by
conference.participant.add	participant.add
conference.participant.modify	participant.modify
conference.participant.remove	participant.remove
conference.query	conference.enumerate participant.enumerate
participant.enumerate	While this call is not itself deprecated, there is deprecated behaviour if there is no operationScope parameter. See the description on page 68 for details.
system.query	conference.enumerate device.query

participant.enumerate

While this call is not itself deprecated, there is deprecated behaviour if there is no operationScope parameter. In this case the MCU will return a participant structure with the following members:

Response	Type	Comments
participantName	String	Participant identification as described above
participantProtocol	String	
participantType	String	
conferenceName	String	
autoAttendantUniquelId	String	
address	String	The address used to connect to the remote endpoint in question. Only returned when the address is known, or if the participant is configured via the API (which requires the address to be specified when added).
gatewayAddress	String	
deferConnection	Boolean	
displayName	String	The name used by the endpoint to identify itself. This may be different to the participantName. Only available after the participant has connected.
displayNameOverrideStatus	Boolean	Indicates whether the displayName value is the result of being overridden.
maxBitRateToMCU	Integer	As for "participant.add"; in kbps.
maxBitRateFromMCU	Integer	
callState	String	One of: dormant, alerting, connected or disconnected.
connectTime	dateTime .iso8601	Only returned after the participant is connected. This value is always present if the call state is connected; it may or may not be defined for participants in the disconnected call state, depending on whether they were ever connected.
disconnectTime	dateTime .iso8601	Only returned after the participant has disconnected.
disconnectReason	String	Only returned after the participant has disconnected. See <i>Participant disconnect reasons</i> on page 75.

Response	Type	Comments
connectPending	Boolean	TRUE if a "participant.connect" command is required for this participant. This parameter will cause either the initial connection to that endpoint (in the event that it was configured with deferConnection set) or a re-connection to that endpoint (in the event that it has disconnected).
audioRxCodec	String	
audioRxLost	Integer	
audioRxReceived	Integer	
audioTxCodec	String	
audioTxReportedLost	Integer	
audioTxSent	Integer	
audioRxMuted	Boolean	
audioRxGainMode	String	
audioRxGainMillidB	Integer	
videoRxCodec	String	
videoRxLost	Integer	
videoRxReceived	Integer	
videoTxCodec	String	
videoTxReportedLost	Integer	
videoTxSent	Integer	
videoRxMuted	Boolean	
videoTxWidescreen	Boolean	
important	Boolean	
activeSpeaker	Boolean	true if this participant is currently the active speaker in the conference.
layoutControlEnabled	Boolean	

Response	Type	Comments
cpLayout	String	The configured layout behavior for this participant. See the <i>Conference layouts</i> section on page 78. This parameter will be present only for participants configured via the API.
currentLayout	Integer	Actual layout in use for the video stream being sent by the MCU to this participant. This parameter will not be present if the participant is in an auto attendant rather than a conference, nor if the MCU is not currently transmitting video to the participant in question. The values for this are those described in the <i>Conference layouts</i> section on page 78.
callDirection	String	Either incoming or outgoing.

Related information sources

system.xml

While not strictly part of the XML-RPC API, some information can be retrieved from the system.xml file. This can be downloaded via HTTP as the file system.xml in the root of the unit, for example, <http://TestMCU/system.xml>.

The fields included in the system.xml file depend on whether the file is from a master or slave MCU. This is indicated in the table below. An example system.xml file is as follows:

```
<system>
  <manufacturer>Cisco</manufacturer>
  <model>MCU 4210</model>
  <serial>MRV1001SM0004B8</serial>
  <softwareVersion>1.4(1.2)</softwareVersion>
  <buildVersion>6.6(1.2)</buildVersion>
  <hostName>TestMCU</hostName>
  <totalVideoPorts>20</totalVideoPorts>
  <totalAudioOnlyPorts>20</totalAudioOnlyPorts>
  <totalStreamingAndContentPorts>20</totalStreamingAndContentPorts>
  <videoPortAllocation>
    <sd>40</sd>
  </videoPortAllocation>
  <portReservationMode>disabled</portReservationMode>
  <maxVideoResolution>cif</maxVideoResolution>
  <uptimeSeconds>2345</uptimeSeconds>
  <clusterType>master</clusterType>
</system>
```

The fields in the file are described in the table below.

Field	Comments
manufacturer	The manufacturer of this MCU..
model	The model of this particular MCU, e.g. MCU 4210.
serial	The serial number of this MCU.
softwareVersion	The software version currently running.
buildVersion	The build version of the software currently running.
hostName	The host name of the system.
totalVideoPorts	The total number of video ports on the MCU. In an MCU cluster, a slave returns a value of 0.
totalAudioOnlyPorts	The total number of audio only ports on the MCU. In an MCU cluster, a slave returns a value of 0.
totalStreamingAndContentPorts	The total number of ports allocated for streaming content. This value is only returned if it is non-zero.

Field	Comments
videoPortAllocation	The number of ports allocated to the selected video resolution. This field has the subfields <sd>, <hd>, and <hd1080p>. Not returned on cluster slaves.
portReservationMode	"enabled" or "disabled", depending on the Media Port Reservation configuration setting. Master only.
maxVideoResolution	The maximum video resolution for the MCU; either "cif" or "max" if larger video is enabled. Master only.
uptimeSeconds	The number of seconds since the MCU was booted.
clusterType	The type of MCU if part of a cluster. Either "master", "slave", or "not set".

Fault codes

The MCU has a series of fault codes which are returned when a fault occurs during the processing of an XML-RPC request. While individual call descriptions above give some indication of which faults may occur, below is a description of all possible fault codes used within this specification and the most common interpretation.

Fault Code	Description
1	Call not supported. This call is not supported on this MCU.
2	Duplicate conference name. A conference name was specified, but is already in use.
3	Duplicate participant name. A participant name was specified, but is already in use.
4	No such conference or auto attendant. The conference or auto attendant identification given does not match any conference or auto attendant.
5	No such participant. The participant identification given does not match any participants.
6	Too many conferences. The MCU has reached the limit of the number of conferences that can be configured.
7	Too many participants. There are already too many participants configured and no more can be created.
8	No conference name or auto attendant id supplied. A conference name or auto attendant identifier was required, but was not present.
9	No participant name supplied. A participant name is required but was not present.
10	No participant address supplied. A participant address is required but was not present.
11	Invalid start time specified. A conference start time is not valid.
12	Invalid end time specified. A conference end time is not valid.
13	Invalid PIN specified. A PIN specified is not a valid series of digits.
14	Unauthorised. The requested operation is not permitted on this MCU.
15	Insufficient privileges. The specified user id and password combination is not valid for the attempted operation.
16	Invalid enumerateID value. An enumerate ID passed to an enumerate call invocation was invalid. Only values returned by the MCU should be used in enumerate calls.
17	Port reservation failure. This is in the case that reservedAudioPorts or reservedVideoPorts value is set too high, and the MCU cannot support this.

Fault Code	Description
18	Duplicate numeric ID. A numeric ID was given, but this ID is already in use.
19	Unsupported protocol. A protocol was used which does not correspond to any valid protocol for this call. In particular, this is used for participant identification where an invalid protocol is specified.
20	Unsupported participant type. A participant type was used which does not correspond to any participant type known to the MCU.
21	No such folder. A folder identifier was present, but does not refer to a valid folder.
22	No such recording. A recording identifier was present, but does not refer to a valid recording.
23	No changes requested. This is given when a method for changing something correctly identifies an object, but no changes to that object are specified.
24	No such port. This is returned when an ISDN port is given as a parameter which does not exist on an ISDN gateway.
101	Missing parameter. This is given when a required parameter is absent. The parameter in question is given in the fault string in the format "missing parameter - <i>parameter_name</i> ".
102	Invalid parameter. This is given when a parameter was successfully parsed, is of the correct type, but falls outside the valid values; for example an integer is too high or a string value for a protocol contains an invalid protocol. The parameter in question is given in the fault string in the format "invalid parameter - <i>parameter_name</i> ".
103	Malformed parameter. This is given when a parameter of the correct name is present, but cannot be read for some reason; for example the parameter is supposed to be an integer, but is given as a string. The parameter in question is given in the fault string in the format "malformed parameter - <i>parameter_name</i> ".
201	Operation failed. This is a generic fault for when an operation does not succeed as required.

Participant disconnect reasons

These are the possible values of the “disconnectReason” field in participant information responses:

Value	Description
authenticationFailed	VNC authentication failed. Check username and password
Busy	The endpoint is in another call
capabilityNegotiationError	Unable to negotiate a common capability set between endpoint and MCU. For example there is no video codec that both sides support
destinationUnreachable	The destination endpoint could not be reached or did not respond
disconnectAll	The MCU disconnected all calls. This occurs at the end of a scheduled conference or a user initiates a disconnect all from the web interface
dnsFailed	A DNS lookup has failed. This can occur when dialling by DNS name
failedToConnectToServer	Unable to connect to VNC server. This can be due to a network problem or if a VNC server is not listening on the specified host
gatekeeperEnded	The gatekeeper ended the call
gatekeeperError	The gatekeeper refused to let the call complete or did not respond
gatekeeperForced	The gatekeeper forced the call to disconnect. For example the end call option was selected on the gatekeeper
gatekeeperRequiredButAbsent	No gatekeeper has been configured but MCU settings require that one be present
h225DecodeError	Error decoding incoming H.225 message. For example the MCU was unable to decode the incoming H.225 message
h225ProtocolError	There has been a H.225 protocol error. For example the endpoint has sent an invalid H.225 message to the MCU
h225SocketError	There has been a error establishing a TCP connection to the H.225 socket on the endpoint. For example there is no route to the desired IP address
h245DecodeError	Error decoding incoming H.245 message. For example the MCU was unable to decode the incoming H.245 message
h245ProtocolError	There has been a H.225 protocol error. For example the endpoint has sent an invalid H.255 message to the MCU

Value	Description
h245SocketError	There has been a error establishing a TCP connection to the H.245 socket on the endpoint. For example the endpoint is not listening on the H.245 port it had previously specified
incompatibleVncVersion	VNC version is incompatible with MCU. Check knowledge base for details of supported versions
localGatekeeperRefused	The local gatekeeper refused the call. This maybe because the destination is not registered to the gatekeeper, for example when dialling direct by IP address
localTeardown	The MCU disconnected the call
messageQueueOverflow	An excess of information in the message buffer has caused it to run out of space and overflow
Moved	The endpoint has moved to a different conference
networkError	There has been an unspecified network error
noAnswer	The endpoint started ringing but the call was not accepted by the user
noGatekeeperForDN	No gatekeeper has been found for dialed number. This can occur when attempting a call to an invalid E164 number
portAllocationExceeded	The number of available ports (both audio and video) on the MCU has been exceeded
protocolError	There has been an unspecified protocol error
q931DecodeError	Error decoding incoming Q.931 message. For example the MCU was unable to decode the incoming Q.931 message
q931ProtocolError	There has been a Q.931 protocol error. For example the endpoint has sent an invalid Q.931 message to the MCU
Rejected	The endpoint chose to reject an incoming call instead of answering
rejectedImmediately	The endpoint rejected the call without ringing
remoteGatekeeperRefused	The remote gatekeeper refused the call. This maybe because the MCU is not registered to the the gatekeeper required by the endpoint
remoteGatekeeperUnreachable	The remote gatekeeper did not respond to the endpoint that the MCU was trying to call
remoteGatewayResources	The remote gateway has insufficient resources to let the call complete. For example the call is being routed to an ISDN gateway with insufficient channels to allow the call to complete

Value	Description
remoteTearDown	The endpoint disconnected the call
serviceUnavailable	The requested service is unavailable. This directly corresponds to an H.323 or SIP message received from the far end to indicate that the call is unable to proceed. The far end could have made this decision for any one of a number of reasons, including lack of resource availability or a call routing policy that prevents the MCU from calling the destination number
Timeout	Could not establish call due to network timeout
Unspecified	This is a "catch all" reason used when no extra information can be provided
unspecifiedError	This is a "catch all" reason used when no extra information can be provided
videoPortAllocationExceeded	The number of available video ports on the MCU has been exceeded

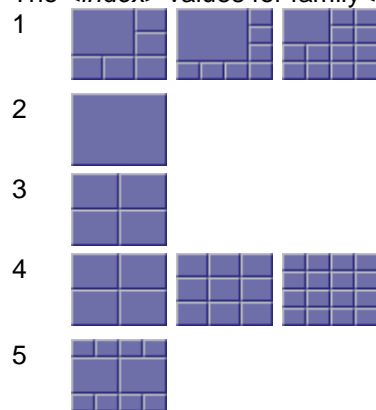
Conference layouts

Some API calls allow a particular layout to be specified for video sent to that participant via the `cpLayout`, `currentLayout`, `customLayout` parameters. These parameter can take the following values:

- ▶ **default** - use the MCU's default view family
- ▶ **family<index>** - use the specified layout family (see below)
- ▶ **layout<index>** - use a specific layout (see below)
- ▶ **conferenceCustom** - use the conference custom layout

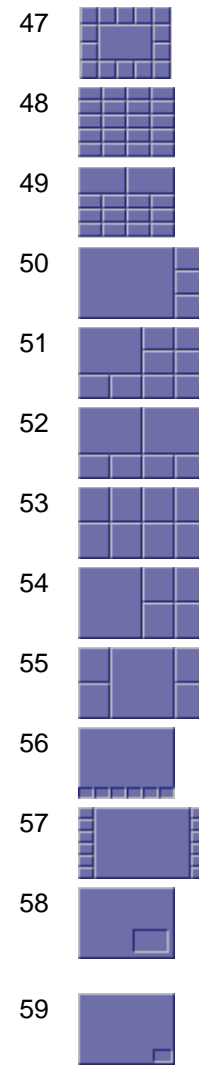
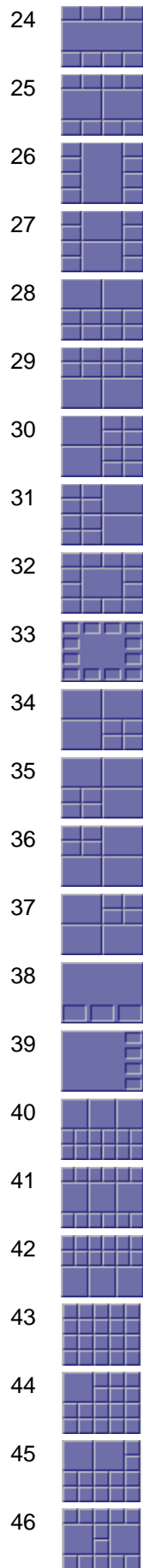
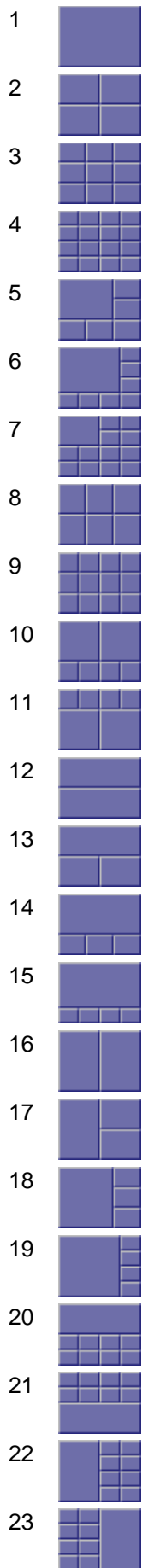
Family layouts

The `<index>` values for `family<index>` correspond to the pane arrangements shown below.



Specific layouts

The `<index>` values for `layout<index>` correspond to the pane arrangements shown in the pages following



Linking conferences across MCUs

For the purposes of this description, two conferences are said to be "linked" if there is a bi-directional H.323 connection between them and each MCU is sending a video channel to the other, showing the active speaker full screen. The audio communicated between the MCUs will be the usual mix of active speakers. For clarification, the linked conferences are given different names ("linked1" and "linked2") in the explanation, but they can have the same name.

The first step is to set up the two conferences. It is important to ensure that the conferences have a numeric id set (the "conferenceID" field in "conference.create"), because, without this configured field, it is not possible to call in directly to a conference. In this example both conferences are given a numeric id, though strictly it is only necessary on the *target* MCU (i.e. the one that is called rather than the one calling).

In this specific example, "linked1" is set up on "mcu1" and "linked2" set up on "mcu2". The creation of "linked1" is shown in **example message 1**, and it is configured with numeric id "1234"; the creation of "linked2" is shown in **example message 2**, and this conference is given the numeric id "5678".

Next, a participant needs to be added to the "linked1" conference and connected to "linked2" on the target MCU. The most reliable way to accomplish this, which does not rely on the target MCU's gatekeeper usage, is to call from "mcu1" into the target conference using "mcu2" as a gateway and the target conference's numeric id as the remote address. The participant addition is shown in **example message 3** - as well as the address and gateway. It also configures the view layout to be full screen (by setting "cpLayout" to "layout1") to make sure that just the active speaker from "linked1" is sent to "linked2".

The final step is slightly more complex — it involves modifying the new "linked2" participant on "mcu2" which was the result of the call from "mcu1". The modification required is to change the view layout setting (for the video sent from "linked2" to "linked1") to full screen so that a view of the "linked2" active speaker is sent.

The complication here is that the "linked2" participant in question is not a participant created via the API, and so the API does not know the name in advance. Therefore, it is necessary to:

- ▶ poll membership of "linked2" after the connection from "linked1" has been made
- ▶ identify the participant corresponding to the call
- ▶ use its name in a "participant.modify" call to set the view layout

The simplest way to identify the participant is to look for an absence of the "address" field in a "conference.query" response: for incoming, non-API, connections this will not be present. **Example message 4** shows such a "participant.modify" call; in this case the participant name needed was "1_Cisco MCU 4210".

Example message 1 - creating conference "linked1" on "mcu1"

```
<?xml version="1.0"?>
<methodCall>
  <methodName>conference.create</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>authenticationUser</name>
            <value>
              <string>admin</string>
            </value>
          </member>
          <member>
            <name>conferenceName</name>
            <value>
              <string>linked1</string>
            </value>
          </member>
          <member>
            <name>conferenceID</name>
            <value>
              <string>1234</string>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

Example message 2 - creating conference "linked2" on "mcu2"

```
<?xml version="1.0"?>
<methodCall>
  <methodName>conference.create</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>authenticationUser</name>
            <value>
              <string>admin</string>
            </value>
          </member>
          <member>
            <name>conferenceName</name>
            <value>
              <string>linked2</string>
            </value>
          </member>
          <member>
            <name>conferenceID</name>
            <value>
              <string>5678</string>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

Example message 3 - calling into "linked2" from "linked1"

```
<?xml version="1.0"?>
<methodCall>
  <methodName>participant.add</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>authenticationUser</name>
            <value>
              <string>admin</string>
            </value>
          </member>
          <member>
            <name>conferenceName</name>
            <value>
              <string>linked1</string>
            </value>
          </member>
          <member>
            <name>participantName</name>
            <value>
              <string>remote_mcu</string>
            </value>
          </member>
          <member>
            <name>address</name>
            <value>
              <string>5678</string>
            </value>
          </member>
          <member>
            <name>gatewayAddress</name>
            <value>
              <string>10.2.1.27</string>
            </value>
          </member>
          <member>
            <name>cpLayout</name>
            <value>
              <string>layout1</string>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodCall>
```

Example message 4 - setting the new "linked2" participant to use a full screen view layout

```
<?xml version="1.0"?>
<methodCall>
  <methodName>participant.modify</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>authenticationUser</name>
            <value>
              <string>admin</string>
            </value>
          </member>
          <member>
            <name>conferenceName</name>
            <value>
```

```
        <string>linked2</string>
      </value>
    </member>
    <member>
      <name>participantName</name>
      <value>
        <string>1_Cisco MCU 4210</string>
      </value>
    </member>
    <member>
      <name>operationScope</name>
      <value>
        <string>active</string>
      </value>
    </member>
    <member>
      <name>cpLayout</name>
      <value>
        <string>layout1</string>
      </value>
    </member>
  </struct>
</value>
</param>
</params>
</methodCall>
```

Message responses

The response to each of the above method invocations should be the same normal success indication:

```
<?xml version="1.0"?>
<methodResponse>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>status</name>
            <value>
              <string>operation successful</string>
            </value>
          </member>
        </struct>
      </value>
    </param>
  </params>
</methodResponse>
```

Revision numbers

Note: This feature is available from API version 2.4 onwards. An application can determine the API version supported by an MCU from the *apiVersion* value returned in the response to a *device.query* request.

In order to reduce the size of responses when querying the MCU, the following methods support a “revision number” system:

- ▶ participant.enumerate
- ▶ conference.enumerate
- ▶ autoAttendant.enumerate

When the MCU responds to a call that supports revision numbers, it returns an extra integer field called “currentRevision”. The following is an example of such a field:

```
<member>
<name>currentRevision</name>
<value>
<int>18</int>
</value>
</member>
```

The revision number is a monotonically increasing value that increases every time any query is made on the MCU via the API. In order to reduce the size of subsequent query responses, the parameter “lastRevision” may be passed in as part of a request; for example:

```
<member>
<name>lastRevision</name>
<value>
<int>18</int>
</value>
</member>
```

This indicates to the MCU that only records that have changed since this revision number should be returned. For example, in *participant.enumerate*, if a *lastRevision* parameter is provided, then the enumeration response will only include participants that have changed since this revision.

Note that when using revision numbers with *enumerate* methods, the same *lastRevision* parameter should be used for each stage of the enumeration, even though a greater *currentRevision* parameter will be returned at each stage. Not doing so could result in records which have changed not being returned. Likewise having completed an enumeration, the only *currentRevision* parameter which should be stored is the one that was returned with the first stage of the enumeration. This is the revision number that should be used as the *lastRevision* parameter next time an enumeration cycle is started.

Using the same revision numbers throughout enumerations is necessary to ensure that all records that have changed are reported, but it does mean that a record may be reported more than once occasionally when there has only been one change to it.

Discovering record removal

The problem with only returning records in responses when they have changed is that if a record is removed there is no way for the client to distinguish between it being removed and just not having changed.

There are two solutions to this problem; the first is the `listAll` parameter and the second is described in the next section. A client may periodically include the `listAll` Boolean parameter set to `TRUE` to indicate that the MCU should return every record available. Enumeration limits still apply so multiple calls using the standard enumeration protocol may be required. This allows a client to resynchronize to the MCU, because it can safely assume that any record not returned by this request (or series of requests, in the case of enumerations) is no longer a record on the MCU.

For example, any participants not returned by `participant.enumerate` when `listAll` is set to `true` can be assumed to have been removed from the MCU.

The `listAll` parameter can still be used in conjunction with the `lastRevision` parameter: doing so means that every record will be returned, but records that have not changed since the specified revision may have many members removed from their substructures. Substructures that have had members removed in this way will contain a field named `“changed”` instead, which will be set to `false` indicating that there are no changes to the data in this substructure since the specified revision number.

Dead records

The second approach to the record removal problem is the `dead` parameter. The MCU will maintain a cache of records that have been removed and are in no sense considered active; a `“dead”` record will never be returned if revision numbers are not being used or if the `listAll` parameter is set to `true` (e.g. a previous participant record is still not considered a dead record because it would be returned by a normal `participant.enumerate` request).

A dead record will be returned by a call supporting revision numbers if the `lastRevision` parameter designates a revision at which the record was not yet dead. The returned record will contain only the fields necessary for its identification and an extra field `“dead”`, which will be set to `true` to indicate that this record should no longer be considered to be present on the MCU.

These dead records are only cached on the MCU for a few minutes; therefore a client should not rely on them unless it is doing very regular polling. When using less frequent polling using the `listAll` parameter described above is more appropriate.

HTTP keep-alives

Note: This feature is available from API version 2.4 onwards.

A method of reducing the amount of TCP traffic when polling the MCU via the API is to use HTTP keep-alives. This method can be used with other products that also support the API, such as the Cisco TelePresence VCR and ISDN Gateway.

Any client which supports HTTP keep-alives may include the following line in the HTTP header of an API request:

```
Connection: Keep-Alive
```

This indicates to the product that the client supports HTTP keep-alives. The MCU *may* then choose to not close the TCP connection after returning its response to the request. If the connection will be closed, the MCU returns the following line in the HTTP header of its response:

```
Connection: close
```

The absence of this line indicates that the MCU will keep the TCP connection open and that the client may use the same connection for a subsequent request.

The MCU will not allow a connection to be kept alive if:

- ▶ the current connection has already serviced a set number of requests
- ▶ the current connection has already been open for a certain amount of time
- ▶ there are already more than a certain number of connections in a “kept alive” state

These restrictions are in place to limit the resources associated with kept-alive connections. If a connection is terminated for either of the first two reasons, the client will probably find that the connection is back in a keep-alive state following the next request.

The client should never assume a connection will be kept alive.

Also note that, even after a response not containing the “connection: close” header, the connection will still be closed if no further requests are made within one minute. If requests from the client are likely to be this far apart then there is little to be gained by using HTTP keep-alives.

References

The following table lists documents and web sites referenced in this document. All product documentation can be found on our [web site](#).

Title	Link
XML-RPC	http://www.xmlrpc.com/
Hypertext Transfer Protocol (HTTP/1.1)	http://www.faqs.org/rfcs/rfc2616.html

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco Logo are trademarks of Cisco Systems, Inc. and/or its affiliates in the U.S. and other countries. A listing of Cisco's trademarks can be found at www.cisco.com/go/trademarks. Third party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1005R)

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© December 2010 Cisco Systems, Inc. All rights reserved.