Cisco TelePresence Conductor API XC2.0

Product Programming Reference Guide

XC2.0

D14948.02

December 2012

Contents

Introduction **4**
Current Conductor API clients 4
Remote Management XML RPC API **5**
Authentication 5
XML-RPC return values 6
conference.destroy 6
Input parameters 6
Returned data 6
conference.enumerate 6
Input parameters 6
Returned data 7
conference.modify 7
Input parameters 7
Returned data 7
device.network.query 7
Input parameters 7
Returned data 8
device.query 8
Input parameters 8
Returned data 8
factory.conferencecreate 8
Input parameters 8
Returned data 9
factory.health.query 9
Returned data 9
feedbackReceiver.extended 10
Feedback Events 11
feedbackReceiver.extended.configure 12
feedbackReceiver.extended.query 13
feedbackReceiver.extended.remove 13
Participant 14
participant.add 14
participant.diagnostics 14
participant.disconnect 15
participant.enumerate 15
participant.message 16
participant.modify 17
Fault codes 18
REST APIs **21**
System information 22
REST API and Clusters 22
Reading records 22
Reading all records 22
Reading some records 22
Creating records 23
Updating records 23
Deleting records 23

Deleting some records 23
Deleting all records 23
Deleting implicitly indexed tables 23
POST data format 24
Result format 24
Code examples for accessing the JSON/REST API 25
Linux curl (JSON results from public API) 25
Linux curl (XML results from public API) 25
Linux wget (JSON results from a public API): 25
python: 25
Discovering the version of the TelePresence Conductor 26
Other APIs 27
SNMP 27
Syslog log messages 27
Event log messages with a severity of "Info" 28
Event log messages with a severity of "Debug" 30
Event log messages with a severity of "Warning" 31
Event log messages with a severity of "Error" 35
API performance and security 38
Current API performance limits 38
API performance considerations 38
Security considerations 39
Appendix A: System limits 40
Conductor performance/API goals and limits 40
Monitoring/Management API performance goals 40
Getting help 41

Introduction

This document describes the set of application programming interfaces (APIs) that is available on the Cisco TelePresence Conductor version XC2.0.

There are plans to update the APIs in future releases to better provide a generic API independent of the conference bridge or other resources that the TelePresence Conductor may be managing. These documented APIs are available in version XC2.0, but may change in future releases.

Many of the XC2.0 APIs have a large focus on the Cisco TelePresence MCU. If the conference bridge handling the conference is a Cisco TelePresence Server, the amount of command / status data supported may be less than indicated.

Current Conductor API clients

Current clients of the Conductor API include:

- Cisco TelePresence Management Suite (conference control center and scheduled calls)
- Cisco Unified Communications Manager (ad hoc conferencing)
- Prime Collaboration Manager

Remote Management XML RPC API

The TelePresence Conductor XML RPC API is loosely based on the existing Cisco TelePresence MCU remote management API.

Some calls are directly processed by the TelePresence Conductor, other calls are composed into messages suitable for underlying TelePresence MCUs or TelePresence Servers and forwarded to the appropriate conference bridge.

XML-RPC API calls are exposed under the **`/RPC2`** path, for example **`https://yourconductor/RPC2`**.

The following API calls are supported:

- conference.destroy
- conference.enumerate
- conference.modify
- device.network.query
- device.query
- factory.conferencecreate
- factory.health.query
- feedbackReceiver.extended.configure
- feedbackReceiver.extended.query
- feedbackReceiver.extended.remove
- participant.add
- participant.diagnostics
- participant.disconnect
- participant.enumerate
- participant.message
- participant.modify

Authentication

Note that systems which use XML RPC over HTTP send authentication over plain text. We therefore recommend using HTTPS instead of HTTP wherever possible.

The controlling application must authenticate itself to the TelePresence Conductor. Also, because the interface is stateless, every call must contain the authentication parameters.

Parameter	Type	Description
`authenticationUser`	string	Name of a user with sufficient privilege for the operation being performed. The name is case sensitive.
`authenticationPassword`	string	The password that corresponds with the given **`authenticationUser`**.

We recommend setting up an administrator account which supports API access and not web access.

XML-RPC return values

The current TelePresence Conductor API is based on the Cisco TelePresence MCU API. Many XML-RPC methods invoked on the TelePresence Conductor are simply proxied to the appropriate TelePresence MCU(s) or TelePresence Server(s) and the responses from the conference bridges are aggregated (if required) and then returned to the client of the TelePresence Conductor API with minimal modification.

The API may respond to requests with empty data structures when the data is not available. Your application must check whether the response includes the expected information, and if not, gracefully handle that situation.

conference.destroy

This call disconnects all participants in the conference and ends the conference. It is used to end ad hoc conferences.

Input parameters

Required parameters

Parameters	Type	Description
`conferenceName`	string	The name of the conference that should end.

Optional parameters

None

Returned data

Parameters	Type	Description
`status`	string	If successful it will display "operation successful".

conference.enumerate

This call returns all active conferences. It works by sending **`conference.enumerate`** messages to all conference bridges configured on the TelePresence Conductor and amalgamating their responses. Only conference data for primary conferences are returned. Cascade specific data will not be returned, however the returned conference struct will be flagged as a cascaded conference.

This call may be resource intensive and may result in multiple messages being sent to the conference bridges.

Input parameters

Required parameters

None

Optional parameters

None

Returned data

Parameters	Type	Description
`conferences`	array	Array of conference structs as returned by the TelePresence MCU API. TelePresence Conductor will add the following parameters to the structs:
→ `factoryTemplateType`	string	The type of template: *meet*, *lesson* or *unknown*. The type is *unknown*, if the configuration has changed since this conference was started.
→ `isCascaded`	boolean	*true* if the conference is cascaded, *false* otherwise.
→ `factoryConferenceId`	string	Internal conference id used by the TelePresence Conductor
Other parameters as documented in the Cisco TelePresence MCU API.		

conference.modify

This call modifies the settings of an existing conference.

Input parameters

Required inputs

Parameters	Type	Description
`conferenceName`	string	The name of the conference, used to identify which conference to modify.

Optional inputs

Parameters	Type	Description
As documented in the Cisco TelePresence MCU API.		

Returned data

Parameters	Type	Description
`status`	string	"operation successful"

device.network.query

This call returns network information about the device.

Input parameters

Required inputs

None

Optional inputs

None

Returned data

Parameters	Type	Description
portA	struct	A structure that contains configuration information for Ethernet port A on the device.
→ **macAddress**	string	Returns the MAC address of the device.

device.query

This call returns high level status information about the TelePresence Conductor.

Input parameters

Required inputs

None

Optional inputs

None

Returned data

Parameters	Type	Description
currentTime	dateTime.iso8601	The system's local current time.
serial	string	The serial number of the device.
softwareVersion	string	The version number of the software running on the device.
buildVersion	string	The build version of the software running on the device.
model	string	Cisco TelePresence Conductor
apiVersion	string	The version number of the API implemented by this device.
totalVideoPorts	integer	The total number of video ports on the device. This includes ports for disabled TelePresence MCUs, but does not include any ports for TelePresence Servers.

factory.conferencecreate

This call creates a new conference via the TelePresence Conductor.

Input parameters

Required parameters

Parameters	Type	Description
conferenceAlias	string	The dial string for the new conference to create.

Optional parameters

Parameters	Type	Description
`factoryMinDurationMinutes`	integer	The minimum time a conference will live even with no participants. Assumed to be 5 if absent.
`factoryOverridePIN`	string (< 32 chars)	If present, this is the string of numeric digits that participants with chairperson privileges need to enter to join the conference. It overrides the PIN defined on the conference template. For a meeting-type conference this PIN is the PIN used for meeting participants.
`factoryOverrideGuestPIN`	string (< 32 chars)	If present, this is the string of numeric digits that participants with guest privileges need to enter to join the conference. It overrides the guest PIN defined on the conference template.

Returned data

Parameters	Type	Description
`status`	string	"operation successful"
`factory_conference_id`	string	Conference UUID supplied by the TelePresence Conductor - for tracking messages

factory.health.query

This call returns system status information.

Note: clients should avoid calling factory.health.query more frequently than every 5 minutes. Because of performance implications TelePresence Conductor implements a caching layer, which will not be updated more frequently than every 5 minutes.

Returned data

Parameters	Type	Description
`fan_1`	string	The current speed of fan 1 (in RPM and updated approximately once per 5 minutes). Returns N/A if using a VM Conductor.
`fan_2`	string	The current speed of fan 2 (in RPM and updated approximately once per 5 minutes). Returns N/A if using a VM Conductor.
`fan_3`	string	The current speed of fan 3 (in RPM and updated approximately once per 5 minutes). Returns N/A if using a VM Conductor.
`committed_as`	string	The amount of memory that would be used if all the memory that has been allocated were to be used. The committed memory is a sum of all of the memory which has been allocated by processes, even if it has not been used by them as of yet. This is useful if one needs to guarantee that processes will not fail due to lack of memory once that memory has been successfully allocated.
`cpu_load_1`	integer	The average CPU load taken over a 1 minute period.
`cpu_load_5`	integer	The average CPU load taken over a 5 minute period.
`cpu_load_15`	integer	The average CPU load taken over a 15 minute period.

Note that the values reported for **cpu_load_1**, **cpu_load_5** and **cpu_load_15** are Linux CPU loads, as reported in the Linux "uptime" command.

TelePresence Conductor has the following amounts of RAM (to compare to the value of **committed_as**):

- on VM systems (systems that report N/A in the **fan_1** to **fan_3** values) RAM is 6G
- on appliance systems (systems that report anything other than N/A in the **fan_1** to **fan_3** values) RAM is 4G

feedbackReceiver.extended

The API allows you to register an application as a feedback receiver. This means that the application does not have to constantly poll the TelePresence Conductor if it wants to monitor activity.

The device publishes events when they occur, it will send XML-RPC messages to your application's interface when the events occur.

There is a limit of 20 feedback receivers in total.

The call **feedbackReceiver.extended** allows clients to register for extended feedback information for specific event notification with pertinent information. This API closely mirrors the existing TelePresence MCU/TelePresence Server feedback APIs. The notification is via the XML-RPC call **eventNotification**.

Example notification message:

```
<?xml version="1.0" encoding="UTF-8"?>
<methodCall>
  <methodName>eventNotification</methodName>
  <params>
    <param>
      <value>
        <struct>
          <member>
            <name>sourceIdentifier</name>
              <value>
                <string>id1</string>
              </value>
          </member>
          <member>
            <name>seqn</name>
              <value>
                <int>0</int>
              </value>
          </member>
          <member>
            <name>events</name>
            <value>
              <array>
                <data>
                  <value>
                    <struct>
                      <member>
                        <name>name</name>
                        <value>
                          <string>conferenceCreate</string>
                        </value>
```

```
                    </member>
                    <member>
                      <name>args</name>
                      <value>
                        <struct>
                          <member>
                            <name>factory_conference_id</name>
                            <value>
                              <string>a7062510-5625-11e1-bdb0-0010f31a4434</string>
                            </value>
                          </member>
                          <member>
                            <name>request_id</name>
                            <value>
                              <string>a7049626-5625-11e1-ad66-0010f31a4434</string>
                            </value>
                          </member>
                        </struct>
                      </value>
                    </member>
                  </struct>
                </value>
              </data>
            </array>
          </value>
        </member>
      </struct>
    </value>
  </param>
 </params>
</methodCall>
```

Feedback Events

Feedback events may be sent either from TelePresence Conductor or directly from the underlying conference bridge to the client.

Below is a list of the supported feedback events. Multiple events can be bundled into one XML-RPC notification message and are returned in an array of parameters in the associated **`args`** struct. The parameters documented below can be returned optionally.

Name	Parameters
`conferenceCreate`	factory_conference_id, request_id
`conferenceDestroyed`	factory_conference_id
`conferenceJoined`	factory_conference_id, unauthenticated_source_alias, participant_role, request_id Note that the data type for the parameter unauthenticated_source_alias has changed from a string to an array.
`joinCreateRequestReceived`	request_id
`cascadeCreated`	factory_conference_id, request_id
`cascadeDestroyed`	factory_conference_id

Name	Parameters
partitionRecovery	
joinCreateRequestFailed	request_id, factory_conference_id
applicationStarted	
alarmsChanged	
conferenceCreate	conference_name

feedbackReceiver.extended.configure

This call configures a feedback receiver on the TelePresence Conductor, which will report particular events that occur on the TelePresence Conductor to the system that is making the request.

Input parameters

Required parameters

Parameters	Type	Description
receiverURI	string	The fully-qualified URI identifying protocol and remote device, for example **http://10.2.134.40:5050/RPC2** Valid protocol types are currently “http” and “https”. If no port number override is specified, then 80 or 443 (respectively) will be used. If a **receiverIndex** has been specified and this parameter is absent or set to the empty string, then the feedback receiver at that index will effectively be de-configured and receive no further notifications.
sourceIdentifier	string	Will be returned in feedback messages (event notifications and **feedbackReceiver.extended.query** messages)

Optional parameters

Parameters	Type	Description
receiverIndex	integer	Which “slot” this receiver should use: If absent assumed to be 1. If <0, then find any available slot (preferred). ReceiverIndex must be in the range 1 – 20 (inclusive)
subscribedEvents	array	An array of strings of event names to subscribe to, where the strings are the names of the notification events. If this parameter is absent, then the receiver will be set up to receive all notifications.

Returned data

Parameters	Type	Description
receiverIndex	integer	Which “slot” has been configured.

Note that if a client tries to configure a feedback receiver using a URI of an existing feedback receiver the call will use the **receiverIndex** of the existing feedbackReceiver.

feedbackReceiver.extended.query

This call returns information about all the feedback receivers that have been configured on the particular TelePresence Conductor.

Input parameters

Required parameters

None

Optional parameters

None

Returned data

If there are no feedback receivers to enumerate, then **`feedbackReceiver.extended.query`** returns an empty array.

Parameters	Type	Description
`receivers`	array	The array of receivers containing the following:
→ **`receiverURI`**	string	Fully-qualified URI identifying protocol and remote device, for example **`http://10.2.134.40:5050/RPC2`**
→ **`sourceIdentifier`**	string	A string that is returned in feedback notification events to identify the originator of the notification event. This is provided when configuring the feedback receiver.
→ **`index`**	integer	The index number of the receiver

feedbackReceiver.extended.remove

This call removes the specified feedback receiver from the TelePresence Conductor.

Input parameters

Required parameters

Parameters	Type	Description
`receiverIndex`	integer	The index returned by a **`feedbackReceiver.extended.configure`** request

Optional parameters

None

Returned data

None

Participant

For methods that perform functions on participants in a conference, the unique identifier is **participantName**. Where multiple participants in a conference have the same **participantName**, the functions are performed on all the participants with that **participantName**.

- **participant.disconnect**, for example, will disconnect all participants with the **participantName** specified on the selected conference.
- **participant.modify** will modify all participants with the **participantName** specified on the selected conference.
- **participant.message** will send the message to all participants with the **participantName** specified on the selected conference.
- **participant.diagnostics** and **participant.mcu** will return the result of doing the function on one of those participants with the same name, the selection of which is completely arbitrary.

participant.add

This call adds a participant to a conference. The participant will end up as a guest (a "participant" in a meeting or a "guest" in a lecture) on the conference, unless **addAsGuest** parameter is specified and set to *False*.

The participant.add request is forwarded to the appropriate conference bridge. The TelePresence Conductor always enforces the input parameter **participantType** to be *ad_hoc*.

Input parameters

Required parameters

Parameters	Type	Description
conferenceName	string	The name of the conference to add the participant to.
participantName	string	The unique name of the participant to add.

Optional parameters

For conferences hosted on a TelePresence MCU the details are documented in the Cisco TelePresence MCU API. For conferences hosted on a TelePresence Server, the TelePresence Conductor will endeavor to provide similar information.

Returned data

None

participant.diagnostics

This call returns diagnostic information about a given participant.

Input parameters

Required parameters

Parameters	Type	Description
`conferenceName`	string	The name of the conference.
`participantName`	string	The unique name of a participant.
`participantProtocol`	string	The protocol: *h323* , *sip* or *vnc*
`participantType`	string	The type of the participant: *by_address*, *by_name*, or *ad_hoc*

Optional parameters

None

Returned data

For conferences hosted on a TelePresence MCU the returned data is documented in the Cisco TelePresence MCU API. For conferences hosted on a TelePresence Server, the TelePresence Conductor will endeavor to provide similar information.

participant.disconnect

This call causes the connection to the specified participant to be torn down, if such a connection exists.

Input parameters

Required parameters

Parameters	Type	Description
`conferenceName`	string	The name of the conference.
`participantName`	string	The unique name of a participant.
`participantProtocol`	string	The protocol: *h323* , *sip* or *vnc*
`participantType`	string	The type of the participant: *by_address*, *by_name*, or *ad_hoc*

Optional parameters

None

Returned data

TelePresence Conductor will proxy back the return parameters of the `participant.disconnect` call from the conference bridge.

participant.enumerate

This call returns data about all participants in active conferences.

The call works by amalgamating responses for `participant.enumerate` calls to all configured conference bridges.

Input parameters

Required parameters

None

Optional parameters

Parameters	Type	Description
`factoryConferenceIds`	array	An array of `factoryConferenceId` strings. If present, only participants belonging to these conferences will be returned. If absent, details for all participants of all conferences will be returned.

Returned data

Parameters	Type	Description
`participants`	array	An array of participant structs as returned by the conference bridge API.
→ `mcuIPAddress`	string	IP address of the conference device holding the participant. Added by TelePresence Conductor.
→ `factoryConferenceId`	string	Internal conference id used to identify which conference the participant belongs to.
Additional parameters contained in a `participant` struct, as documented in the Cisco TelePresence MCU API.		

Note that the return information about all participants in all conferences (participant information structures for approximately 2,400 participants) might be a bit unwieldy.

participant.message

This call is used to send a message to a participant in their video stream.

Input parameters

Required parameters

Parameters	Type	Description
`conferenceName`	string	The name of the conference.
`participantName`	string	The unique name of a participant.
`message`	string	The string of up to 255 characters to send to the participant.

Optional parameters

Parameters	Type	Description
`participantProtocol`	string	The protocol: *h323* , *sip* or *vnc*
`participantType`	string	The type of the participant: *by_address*, *by_name*, or *ad_hoc*

Returned data

Parameters	Type	Description
`status`	string	"operation successful"

participant.modify

This call modifies the active state of a participant in a conference.

Input parameters

Required parameters

Parameters	Type	Description
`conferenceName`	string	The name of the conference.
`participantName`	string	The unique name of a participant.
`participantProtocol`	string	The protocol: *h323* , *sip* or *vnc*
`participantType`	string	The type of the participant: *by_address*, *by_name*, or *ad_hoc*

Optional parameters

Parameters	Type	Description
displayNameOverrideStatus	boolean	*true* if the endpoint uses the **displayNameOverrideValue** text to identify itself to other participants.
displayNameOverrideValue	string	This value overrides the participant's display name if **displayNameOverrideStatus** is *true*.
cpLayout	string	This sets the initial conference view layout for the video sent to the participant.
layoutControlEnabled	boolean	Defines whether the endpoint's participant will have control over the layout.
audioRxMuted	boolean	*true* means that audio from this participant will not be heard by other conference participants.
audioRxGainMode	string	*none*, *automatic*, or *fixed*
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.
videoTxWidescreen	boolean	If *true*, the TelePresence MCU sends video in a form suitable for a widescreen 16:9 display to this participant.
autoDisconnect	boolean	*true* allows the device to automatically disconnect the endpoint, and all remaining endpoints that have this property, when none of the remaining endpoints require manual disconnection. *false* means this endpoint requires manual disconnection.
suppressDtmf	string	Controls the muting of DTMF tones. One of *fecc*, *always*, *never*, or *default*
dtmfSequence	string	A string of characters that will be converted to DTMF signals, allowing the device to navigate through audio menus. The sequence may contain 0-9, *, #, and ,. The comma becomes a two second pause.
audioTxMuted	string	*true* if audio is not being transmitted to this participant.
borderWidth	integer	Controls the width of the outer border of a participant's layout. *0* is no border.

Returned data

TelePresence Conductor will proxy back the return parameters of the **participant.modify** call from the conference bridge.

Fault codes

The TelePresence Conductor returns a fault code when it encounters a problem with processing an XMLRPC request.

The following table lists the fault codes that may be returned by the TelePresence Conductor and their most common interpretations.

Fault Code	Description
1	**method not supported**. This method is not supported on this device.
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 device 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	**authorization failed**. The requested operation is not permitted on this device.
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 method invocation was invalid. Only values returned by the device should be used in enumerate methods.
17	**port reservation failure**. This is in the case that reservedAudioPorts or reservedVideoPorts value is set too high, and the device cannot support this.
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 method. 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 device.
25	**new port limit lower than currently active**.
26	**floor control not enabled for this conference**.
27	**no such template**. The specified template wasn't found.
30	**unsupported bit rate**. A call tried to set a bit rate that the device does not support.
31	**template name in use**. This occurs when trying to create or rename a template to have the same name as an existing template.
32	**too many templates**. This occurs when trying to create a new template after the limit of 100 templates has been reached.
36	**required value missing**. The call has omitted a value that the TelePresence MCU requires to make the change requested by the call.

Fault Code	Description
42	**port conflict**. The call attempts to set a port number that is already in use by another service.
43	**route already exists**. The call attempts to add a route that has the same **destination** and **prefixLength** as a route that already exists on the TelePresence Conductor.
44	**route rejected**. The call attempts to add a route to a forbidden subnet
45	**too many routes**. The call can not add the route because doing so would exceed the allowed number of routes.
46	**no such route**. The TelePresence Conductor has no record of a route that has the provided **routeId**.
48	**IP address overflows prefix length**. The call attempts to make a route **destination** more specific than the range defined by the **prefixLength**.
49	**operation would disable active interface**.
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 - parameter_name".
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 - parameter_name".
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 - parameter_ name".
104	**mismatched parameters**. The call provides related parameters that, when considered together, are not expected/supported.
201	**operation failed**. This is a generic fault for when an operation does not succeed as required.

REST APIs

TelePresence Conductor supports the following REST APIs:

- https://yourconductor/status - summary information about the version and status (active/inactive) of the TelePresence Conductor system.
- https://yourconductor/systemunit.xml - summary information about the system version.
- https://yourconductor/api/external/ - the preferred REST API interface.

The api/external REST API exposes tables consisting of one or more records - each with one or more fields. Most tables use a uuid as the primary key.

The following tables are supported:

- configuration/conferencefactory/pools
- configuration/conferencefactory/servicepreferences
- configuration/time
- status/cluster
- status/clusterpeer
- status/networkinterface
- status/conferencefactory/mcustatus
- status/conferencefactory/mcucallsignallingloadstatus

Access to any undocumented resource or any resource not under https://yourconductor/status, https://yourconductor/systemunit.xml or https://yourconductor/api/external/ is unsupported - and access may be modified or completely withdrawn without notice in a future release.

The two main groups of information are:

- Status information (read-only) : https://yourconductor/api/external/status
- Configuration (read-write) : https://yourconductor/api/external/configuration

Current status resources available include:

- https://yourconductor/api/external/status/conferencefactory/primaryconferences - summary information about currently running conferences
- https://yourconductor/api/external/status/alarm - information about all alarms, whether they're raised or lowered, and an English language translation of the descriptions of the alarms.
- https://yourconductor/api/external/status/system - detailed information about the system software version

The following configuration resources are available:

- https://yourconductor/api/external/configuration/dnsserver - DNS server configuration
- https://yourconductor/api/external/configuration/ntpserver - NTP server configuration
- https://yourconductor/api/external/configuration/snmp - SNMP server configuration
- https://yourconductor/api/external/configuration/dns - DNS server configuration
- https://yourconductor/api/external/configuration/conferencefactory/mcuinfo - used by TMS to gain information about TelePresence MCU and TelePresence Server systems known to TelePresence Conductor

- https://yourconductor/api/external/configuration/conferencefactory/mcuaddress - also used by TMS to gain information about TelePresence MCU and TelePresence Server systems known to TelePresence Conductor

The REST API returns results in XML format and permits the restriction of results to the addressed peer's result set by use of the peer=local query string parameter (e.g. https://yourconductor/api/external/status/networkinterface?peer=local)

System information

The version of the TelePresence Conductor software that is running, can be obtained using **`systemunit.xml`**.

The following is example XML for **`systemunit.xml`**:

```
<SystemUnit>
        <Name>TestConductor</Name>
        <Software>
                <Version>XC2.0</Version>
        </Software>
</SystemUnit>
```

REST API and Clusters

Some tables, containing global configuration or status information applicable to all peers in a cluster, are shared by all members of a cluster.

Other tables, containing system specific configuration or status information, contain a sub-table per cluster peer. All peers (conceptually) have access to all tables (including the system-specific tables of other peers) although by convention one cluster peer will never modify another cluster peer's system specific table.

Reading records

Reading all records

GET from **`http://yourconductor/api/external/<basepath>/`**

Reading some records

GET from **`http://yourconductor/api/external/<basepath>/<key>/<value>`**

where:

- **`<key>`** is the column name. The column must be indexed
- **`<value>`** is the value to match.

This works with explicitly indexed tables only.

To restrict the result set to a single peer, include **`peer=<IP>`** in the query string, where **`<IP>`** is the IP address of the peer to retrieve results for. This IP address must match the corresponding cluster alternate IP

for the node. As a special case, substitute "local" in place of the IP address to retrieve results for the local peer only.

Pagination of results may be achieved by specifying an offset and limit in the query string. Offset is 0-based (i.e. to obtain the first record, provide a query string of "offset=0&limit=1"). For example, to obtain the second 10 results, the query string would contain "offset=10&limit=10". This may be used in conjunction with restricting results to a single peer.

When paginating, it is possible to sort by columns other than the uuid. The sort column is specified by the **`sortby`** query parameter. This takes the column name as its value. For example, to sort by field3, specify "**`sortby=field3`**".

By default, the sort order is ascending. This may be specified explicitly using the **`sortdirection`** query parameter. The value of the **`sortdirection`** parameter is either "ascending" or "descending".

Creating records

POST to **`http://yourconductor/api/<basepath>/`**

(see also note on POST format below)

Updating records

POST to **`http://yourconductor/api/<basepath>/<key>/<value>`**

where

- **`<key>`** is the column name. The column must be indexed
- **`<value>`** is the value to match

(see also note on POST format below)

Deleting records

Deleting some records

DELETE to **`http://yourconductor/api/<basepath>/<key>/<value>`**

where

- **`<key>`** is the column name. The column must be indexed
- **`<value>`** is the value to match

Deleting all records

DELETE to **`http://yourconductor/api/<basepath>/`**

Deleting implicitly indexed tables

DELETE to **`http://yourconductor/api/<basepath>/`**

POST data format

Data sent to the REST API by the client must be application/x-www-form-urlencoded :

- data ::= kvpair *['&' kvpair]
- kvpair ::= key '=' value
- key, value ::= URL encoded string

Omitted fields will be defaulted.

Assuming a table definition like this:

Field Name	Type	Default Value	Constraints
uuid	string		
field1	integer	1	Minimum: 1, Maximum: 5
field2	integer	13	
field3	string		

then, sending valid POST data for a table with three fields:

- uuid=12345678-0000-0000-0000-123456789012
- field1=2
- field2=13
- field3=foo bar

would result in the following record with some fields set to their default values:

{ "uuid": "12345678-0000-0000-0000-123456789012", "field1":2, "field2":13, field3: "foo bar" }

Note that, in normal usage, the UUID is not specified in the POST data.

Result format

Data returned from the REST API is either JSON or XML encoded.

The result format may be controlled either through use of an Accept header in the request, or by including a query string with **`format={json|xml}`**. The use of an Accept header is preferred. If no result format control data is provided by the client, JSON will be returned. JSON is generally more compact and often faster to parse than XML. For JSON, the results of GET requests will be returned as follows:

```
[
{ "peer": <IP>,
"num\_recs": 123,
"records": [ <Record>, ... ]
},
...
]
```

where:

- **<IP>** is the IP address of the peer, as a string,
- **<Record>** is a JSON object representing a record.

There may be multiple peer descriptors in the results: one per peer in the cluster.

The **num_recs** field contains the total number of results that matched the request on a peer. The total number of results across the cluster may be calculated by summing the peers' **num_recs** fields. If it was impossible to compute the number of matching records, **num_recs** will have a value of -1.

There may be fewer than **num_recs** results in the records field. This will be the case when the request has been limited for pagination.

The result of a POST request is a list of records affected by the request. Such a JSON response would look like the following:

```
[ <Record>, ... ]
```

where **<Record>** is a JSON object representing a record.

Code examples for accessing the JSON/REST API

All access to the REST API requires authentication.

TelePresence Conductor uses HTTPS with standards-based basic HTTP authentication to restrict access to the API.

Currently, the TelePresence Conductor supports only a single username ("admin") and password - shared with the main TelePresence Conductor web UI. This most definitely will change in a future release (to allow for API-only accounts) - so all systems integrating against the TelePresence Conductor *must* allow both username and password credentials to be configurable. Do not assume that there will always be an "admin" account.

Reading values from the REST API is easy. The examples below assume the existence of a user named "admin" with a password of "xxx":

Linux curl (JSON results from public API)

```
curl --user admin:xxx https://yourconductor/api/external/status/alarm
```

Linux curl (XML results from public API)

```
curl --user admin:xxx  -H "Accept: application/xml"
https://yourconductor/api/external/status/alarm
```

Linux wget (JSON results from a public API):

```
wget --no-check-certificate --http-user admin --http-password xxx
https://yourconductor/api/external/status/alarm
```

python:

```
import urllib2

theurl = 'https://yourconductor/api/external/status/alarm'

username = 'admin'

password = 'xxx'
```

```
passman = urllib2.HTTPPasswordMgrWithDefaultRealm()
passman.add_password(None, theurl, username, password)
authhandler = urllib2.HTTPBasicAuthHandler(passman)
opener = urllib2.build_opener(authhandler)
urllib2.install_opener(opener)
pagehandle = urllib2.urlopen(theurl)
pagehandle.read()
```

Writing (creating and/or updating) values via the REST API is also easy. However, beware that most TelePresence Conductor API resources should be treated as "read only".

Discovering the version of the TelePresence Conductor

Before accessing the TelePresence Conductor REST or XML-RPC API, external systems should check the version of the TelePresence Conductor software by calling **`device.query`** in order to adjust their behavior (if need be) to be appropriate to the version of TelePresence Conductor they're accessing.

Other APIs

SNMP

The TelePresence Conductor has limited support for SNMP. To view the details:

1. enable SNMP (for example, by selecting *v2c* on the **System > SNMP** page)
2. enter the command **`snmpwalk`** from a Linux workstation to "explore":
   `snmpwalk -c public -v2c localcf`

Syslog log messages

Listed below are all the messages that can appear in the TelePresence Conductor's event logs, and appear in the remote syslog feed if enabled on the TelePresence Conductor, together with their parameters.

Event log messages with a severity of "Info"

Message	Parameters
Successful login into the interactive debugging console.	Username
A conference has been deleted.	Conference_bridge_conference_name, Conference_name, Conference_bridge_UUID, Conference_bridge_address, Conference_unique_identifier
Cannot allocate conference bridge resource.	Tag, Destination-alias, Source_protocol, Source_registered_alias,Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address,Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_unique_identifier
The maximum number of participants for this role has been reached.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_unique_identifier
A request to join a conference was successfully processed.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_bridge_UUID, Conference bridge uuid, Conference_bridge_address, Conference_bridge_type, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Process_time, Request_unique_identifier, Known_multiscreen_endpoint
Auto-dialed participant request has been sent to the conference bridge.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester (VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Auto-dial_participant_match, Auto-dial_participant_rule, Auto-dial_participant_address, Auto-dial_participant_keep_alive, Auto-dial_participant_protocol, Auto-dial_participant_role
Atempting to add an auto-dialed participant a conference.	Auto-dialed_participant_name, Conference_name, Conference_template_name
A conference bridge that was previously unusable is now active.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference bridge type, Detail, Conference_name, Conference_bridge_conference_name, Status_detail
An attempt has been made to access the interactive debugging console.	Username

Message	Parameters
An incoming call request has been rejected because the conference is not present.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases,Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Reason, Conference_name, Conference_bridge_conference_name,Conference_unique_identifier, Conference_bridge_participant
Recovered from a network partition; service is resumed.	
A conference has been marked for deletion.	Conference_bridge_conference_name, Conference_name, Conference_bridge_UUID, Conference_bridge_address, Conference unique identifier
A management request has been received.	Command, Conference_name, Conference_bridge_conference_name, Participant_name, Participant_type, Participant_protocol, Requester_(VCS/Unified_CM/client)_address)
A conference has been successfully created on the conference bridge and is ready to receive participants.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases,Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Process_time, Request_unique_identifier
A conference's cascade has been marked for deletion.	Conference_bridge_conference_name, Conference_name, Conference_bridge_UUID, Conference_bridge_address, Sub-conference UUID, Conference unique identifier
The TelePresence Conductor application has been stopped.	Detail
The clustering system partition status has been set.	Old_partition_state, New_partition_state
Request rejected. The conference alias has an invalid role for the conference type.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_alias, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Conference_bridge_UUID, Conference bridge uuid, Conference_bridge_address,Conference bridge type, Status_good, Config_good, Reason, Conference_name, Conference_bridge_conference_name, Conference_template_name, Participant_role, Conference_unique_identifier
The TelePresence Conductor application has started.	
A request has been received from a client for a participant to create or join a conference.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Request_unique_identifier
A conference duration has expired. Conductor will now terminate the conference.	Conference_name, Conference_bridge_UUID

Message	Parameters
A conference has been successfully cascaded to an additional MCU.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases,Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_bridge_UUID, Conference bridge uuid, Conference_bridge_address, Conference bridge type, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Request_unique_identifier, H323_cascade_call_routing
A conference's cascade has been deleted.	Conference_bridge_conference_name, Conference_name, Conference_bridge_UUID, Conference_bridge_address, Sub-conference UUID, Conference_unique_identifier

Event log messages with a severity of "Debug"

Message	Parameters
A participant's allocated resource has been reduced.	Participant_name, Resource_change
The TelePresence Conductor application has been stopped.	Detail
A participant's allocated resource has been increased.	Participant_name, Resource_change

Event log messages with a severity of "Warning"

Message	Parameters
An error occurred while communicating externally.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Conference_name, Conference_bridge_conference_name, Command
Due to a network partition conferences may not have enough reserved resources.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type
A CPL 'reject' has been returned to a client. Reasons could include: no matching alias, no conference template, or no dial plan prefix configured.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Detail, Request_unique_identifier, Conference_bridge_participant
An auto-dialed participant has failed to be added to a conference. This is because H.323 participants are not supported in Unified CM deployments.	Auto-dialed_participant_name, Conference_name, Conference_template_name
A Unified CM location has been configured with an unknown IP address.	Unified_CM_location, IP_address
Auto-dialed participant address cannot be resolved.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases,Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Conference_template_UUID, Conference_name,Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Auto-dial_participant_match, Auto-dial_participant_rule, Auto-dial_participant_address, Auto-dial_participant_keep_alive, Auto-dial_participant_protocol, Auto-dial_participant_role
Auto-dialed participant request has been sent to the conference bridge.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Auto-dial_participant_match, Auto-dial_participant_rule, Auto-dial_participant_address, Auto-dial_participant_keep_alive, Auto-dial_participant_protocol, Auto-dial_participant_role
More chairpersons than expected found on conference.	Conference_name, Conference_bridge_conference_name, Chairperson_count, Reserved_chairperson_participants
Conference bridge pool resource usage is approaching full capacity.	Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID

Message	Parameters
Bad regex match configuration.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail
Application watchdog has detected a jump in system time. Check your NTP settings.	Last_modified, Current_time
Management request 'participant.add' failed. This is because only SIP participants are supported in Unified CM deployments.	Participant_name, Conference_name, Conference_template_name
A conference has been destroyed because it is missing from an MCU.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name
A conference bridge that was previously active is now unusable.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference bridge uuid, Conference_bridge_address, Conference bridge type, Detail, Conference_name, Conference_bridge_conference_name
A CPL request with no alias specified, or with an alias that does not use UTF-8, has been received from a client.	
Changes to vital conference bridge configuration have been detected. All conferences on this conference bridge have been deleted from the database.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conferences
Request rejected. The conference exists but is not responding in a timely fashion.	Tag, Destination-alias,Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Status_good, Config_good, Reason, Conference_name, Conference_bridge_conference_name, Timeout
Unable to process request because the requested service was not found.	
A conference or a cascade could not be recovered.	Conference_name, Conference_bridge_conference_name
Keep conference alive will be ignored for auto-dialed participant.	Conference_template_name, Auto-dialed_participant_name
An error occurred while communicating externally, retrying.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Conference_name, Conference_bridge_conference_name, Command

Message	Parameters
Bad match regex substitution configuration when trying to create a canonical conference name.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail
A multiscreen endpoint is dialing into a single-screen conference.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases,Requester_(VCS/Unified_CM/client)_address
An attempt to create a conference or a cascade was unsuccessful.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_bridge_UUID, Conference bridge uuid, Conference_bridge_address, Conference_bridge_type, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Request_unique_identifier
Cannot resolve conference bridge hostname.	Conference_bridge_address
Not enough conference bridge resource to handle request.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_unique_identifier
Conference bridge pool resource usage has reached full capacity.	Conference bridge pool UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
A conference has been destroyed because its conference bridge is not configured.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name
A conference bridge is reporting zero available resource.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type
Adding an auto-dialed participant to a conference bridge failed.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Auto-dial_participant_match, Auto-dial_participant_rule, Auto-dial_participant_address, Auto-dial_participant_keep_alive, Auto-dial_participant_protocol, Auto-dial_participant_role
A Unified CM location has been configured without an ad hoc IP address.	Unified_CM_location
An API request could not be processed.	Command, Conference_name, Conference_bridge_conference_name, Participant_name, Participant_type, Participant_protocol, Detail, Xmlrpc_parameters

Message	Parameters
A cascade link has been lost. The system will attempt to rebuild it.	Conference_name,Conference_bridge_conference_name, Primary_bridge_UUID, Primary_bridge_address,Cascade_bridge_UUID,Cascade_bridge_address
Deleting from a conference bridge a conference unknown to this peer.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail,Conference_name,Conference_bridge_conference_name
A conference bridge is unusable because it is not in a running state.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type
A Unified CM location with a conference type of 'ad hoc' has been configured with a lecture-type ad hoc conference template.	Unified_CM_location, Conference_template_name
Incoming call request rejected.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Reason, Conference_name, Conference_bridge_conference_name, Conference_unique_identifier, Known_multiscreen_endpoint
Request rejected. The conference exists but is stopping.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Status_good, Config_good, Reason, Conference_name, Conference_bridge_conference_name
A Unified CM location with an ad hoc IP address is missing an ad hoc conference template.	Unified_CM_location
A conference has been removed from the database because it is missing from a conference bridge.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name, Conference_part
A conference was not recovered because the conference template was not found.	Conference_template_UUID, Conference_name, Conference_bridge_uuid
A Unified CM location has been configured without a rendezvous IP address.	Unified_CM_location
A template has the same number of, or more, auto-dialed and reserved participants associated with it than the maximum number of participants allowed for that template.	Conference_template_name
Failure to log into the interactive debugging console	Username

Event log messages with a severity of "Error"

Message	Parameters
A TelePresence MCU pool is erroneously in a TelePresence Server Service Preference.	Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
A prohibited JSON key has been provided via the conference template.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier
Cannot create a conference, because this version of conference bridge requires both Chair PIN and Guest PIN to be set when using PINs.	Conference_template_name, Conference_bridge_UUID, Conference_name, Conference_bridge_conference_name, Conference_bridge_address
Bad conference name.	Call_policy_prefix, Match
A conference has been destroyed because it is missing from a MCU.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name
A conference bridge is missing a dial plan prefix. This is causing conferences to fail.	Conference_bridge_name
Unable to find parent record.	Table, Record, Field, Parent_uuid
An attempt to reduce the resource allocated to a participant has failed.	Participant_name, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name
An attempt to create a conference or a cascade was unsuccessful.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier, Request_unique_identifier
Letter case inconsistency found in a field.	Table, Record, Field
Request rejected. The TelePresence Conductor service is currently unavailable.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Status_good, Config_good, Reason, Conference_name, Conference_bridge_conference_name

Message	Parameters
Invalid JSON has been provided via the conference template.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Incoming_alias_match, Conference_name_rule, Detail, Conference_alias_UUID, Conference_alias_name, Conference_alias_description, Participant_type, Participant_role, Conference_template_UUID, Conference_name, Conference_bridge_conference_name, Conference_template_name, Conference_unique_identifier
Application watchdog has performed a number of system restarts but the application has not responded. Manual intervention is required.	
An incorrect role has been used for the conference.	Table, Record, Field, Participant_role, Parent_uuid, Conference_type, Allowed_roles
An auto-dialed participant references advanced template parameters that do not match the conference bridge type for the template.	Auto-dialed_participant_name, Conference_template_name
An attempt to increase the resource allocated to a participant has failed.	Participant_name, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_name, Conference_bridge_conference_name
A pool has a conference bridge type that is different from its associated Service Preference.	Service_Preference_uuid, Service_Preference_name, Service_Preference_conference_bridge_type, Service_Preference_pool_uuids, Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
TelePresence Conductor application is not responding. Restarting system.	
An active conference bridge pool does not contain any conference bridges.	Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
Incoming call request rejected.	Tag, Destination-alias, Source_protocol, Source_registered_alias, Source_address, Unauthenticated_source_aliases, Requester_(VCS/Unified_CM/client)_address, Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Detail, Reason, Conference_name, Conference_bridge_conference_name, Conference_unique_identifier, Known_multiscreen_endpoint
The TelePresence Conductor application has been stopped.	Detail
A conference bridge has a conference bridge type that is different from the conference bridge type of the pool to which it belongs.	Conference_bridge_UUID, Conference_bridge_uuid, Conference_bridge_address, Conference_bridge_type, Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
An unlinked record was found.	Parent_table, Child_table, Child_reference, Child_entry_UUID

Message	Parameters
A conference template has been configured with primary advanced parameters that do not match the conference bridge type for the template.	Conference_template_name
Unexpected role for conference type.	Table, Record, Field, Participant_role, Conference_type
Invalid release key detected.	
A conference bridge does not belong to a conference bridge pool.	Conference_bridge_name
A conference bridge does not have a Unified CM location of type 'rendezvous' or 'both'.	Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
A TelePresence Server pool is erroneously in a TelePresence MCU Service Preference.	Conference_bridge_pool_UUID, Conference_bridge_pool_name, Conference_bridge_pool_type, Service_Preference_ID
Conflict between conference alias and conference bridge dial plan prefix.	Match, Conference_bridge_prefix
A Unified CM location without a trunk IP address is being referenced by a conference bridge pool.	Unified_CM_location_name
A lecture-type conference template is not referenced by a 'lecture' or 'guest' alias.	Conference_template_name, Missing_alias_role
An error occurred while communicating with the internal database.	Communication_type
A conference template has been configured with cascade advanced parameters that do not match the conference bridge type for the template.	Conference_template_name

API performance and security

Current API performance limits

The currently supported API performance limits are:

Feature	Limit
Maximum number of API clients	<= 4
Maximum number of API connections per client	<= 1
Maximum number of concurrent API client connections (across all clients, including TMS, PCM and your application)	<= 4

API performance considerations

Accessing the REST API of the TelePresence Conductor often requires the TelePresence Conductor to lock access to the database and wait for database replication to complete in order to ensure that a coherent response is returned. This could adversely affect performance of an entire TelePresence Conductor cluster. Accessing the XML-RPC API of the TelePresence Conductor may cause the TelePresence Conductor to contact the conference bridge(s) - thus causing load on the conference bridge as well as on the TelePresence Conductor.

Repetitive or high volume accesses to the TelePresence Conductor API may therefore adversely affect the call handling performance of the TelePresence Conductor and the conference bridges managed by the TelePresence Conductor - and therefore designs, which require this, should be avoided.

- Accessing the REST or XML RPC API incurs a performance penalty on the TelePresence Conductor. If the TelePresence Conductor is under heavy load, API responses may be delayed.
- Many XML RPC API invocations are simply passed on to one or more underlying conference bridges- thus causing additional load and performance penalties on the conference bridges themselves. Other calls involve TelePresence Conductor accessing its own database, incurring cluster-wide REST access penalties.
- Some types of conference bridges have a very limited capacity for processing simultaneous XML RPC requests - so TelePresence Conductor is forced to serialize XML RPC requests for those conference bridges in order to avoid overloading them. This can result in slow response times even if the TelePresence Conductor itself is not under heavy load.
- Finally, the CPU time cost of doing the handshakes required for establishment of an HTTPS connection to TelePresence Conductor is quite high.

So, to ensure good performance:

- Please remember that various systems other than yours (such as Cisco TMS and Cisco Prime Collaboration Manager) may also be simultaneously accessing the TelePresence Conductor API
- Please limit the number of concurrent TCP connections you make, as the TelePresence Conductor can handle no more than four concurrent API connections without degrading system performance.
- Please remember that the TelePresence Conductor may be under heavy call handling load. Because call handling may be just as important (or even more important) than handling the API request, use API requests sparingly.

- Please avoid performing large amounts of background polling (using REST or XML-RPC) for information that is not needed
- Please avoid making multiple concurrent API requests to the TelePresence Conductor - wait for your last request to complete before making the next one.
- In cases where a client will be making multiple REST and/or XML-RPC requests to TelePresence Conductor in quick succession, please make use of HTTP 1.1 connection keep-alives, to allow multiple requests to be sequentially handled using a single connection to TelePresence Conductor. This will avoid the considerable expense of dropping/re-establishing an HTTPS connection for every request.

The best architecture for a client of the TelePresence Conductor API 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 from being impaired by unmanageable numbers of API requests.

Security considerations

The TelePresence Conductor API is accessible only over HTTPS. Production deployments of TelePresence Conductor should install a valid certificate, signed by an appropriate certificate authority. Clients of the TelePresence Conductor API should then (optionally) allow the administrator to configure the clients such that the client service checks the validity of the certificate of the TelePresence Conductor that they are connecting to (and thus protect the overall service from man-in-the-middle attacks).

Be aware: if any TelePresence MCU or TelePresence Server configured on the TelePresence Conductor is configured to use HTTP rather than HTTPS, then the TelePresence Conductor will transmit sensitive information in the clear to the XML-RPC API of the conference bridge(s). In security sensitive deployments it is important for solution security to also enable HTTPS on the conference bridges and to configure the TelePresence Conductor to communicate to the conference bridges over HTTPS - even though the TelePresence Conductor API is HTTPS only.

Appendix A: System limits

Conductor performance/API goals and limits

Feature	Limit
Conference aliases	<= 1,000
Conference templates	<= 1,000
Auto-dialed participants	<= 5,000
Concurrent conferences	<= 1,000
Conference bridges	<= 30
Total conference bridge ports	<= 2,400
Maximum participants per conference	<= 2,342
Conference create events per second	<= 2
Conference join events per second	<= 8

Monitoring/Management API performance goals

Feature	Limit
Monitored conferences	<= 40
Minimum conference poll period	>= 5 s
Maximum poll requests per second	<= 8
Mute requests per second	<= 10

The above limits are not enforced - but should not be exceeded in normal usage. If your monitoring/management system can safely be designed to make poll requests less frequently than the maximum supported rate(s) then that is better for overall system performance. We aim to test to 150% of each limit to ensure we have enough performance "headroom".

We strongly suggest that such performance/capacity testing is appropriate for customers of the Conductor API too - as it might be that the burden placed on Conductor by external API clients is significant.

In practice, this means you should arrange to test the impact of your system running against a heavily loaded Conductor - and verify that the presence of your system does not prevent Conductor from meeting its performance targets.

Getting help

If you experience any problems when configuring or using Cisco TelePresence Conductor, see the documentation available on the Cisco support web site at http://www.cisco.com/en/US/products/ps11775/tsd_products_support_series_home.html. Here you will also be able to get the most up to date version of software.

If you need to raise a case with TAC, make sure that you have the following information available as you raise the case:

- Identifying information for your product, such as model number, firmware version, and software version (where applicable).
- Your contact email address or telephone number.
- A full description of the problem.
  - What you did, what you expected to happen and what actually happened.
  - An architectural diagram of the setup of the devices involved.

To view a list of Cisco TelePresence products that are no longer being sold and might not be supported, visit http://www.cisco.com/en/US/products/prod_end_of_life.html and scroll down to the TelePresence section.