Cisco TelePresence Management Suite Extension Booking API

Programming Reference Guide

API version 9
(Cisco TMS 13.2)

D13566 12

May 2012

Contents

Introduction

Cisco TelePresence Management Suite Extension Booking API (Cisco TMSBA) gives developers access to Cisco TelePresence Management Suite (Cisco TMS) booking functionality. The API is also used by the Cisco TMS extensions for Microsoft Exchange and IBM Lotus Notes.

This document details the objects and entities used by Cisco TMSBA, includes references of the functions and objects available in the booking and remote setup APIs, and usage patterns suggesting how an external user interface, booking database, or booking system may interact with Cisco TMS using the APIs.

The target audience for this document is developers seeking to implement a data/audio/video conferencing booking solution that is not supported by Cisco TMS directly, or where existing Cisco TMS features do not provide the necessary interoperability. Such booking systems will be referred to as external booking systems in this document.

API overview

In this chapter:

- Functional overview of Cisco TMSBA and its booking principles.
- Cisco TMS entities explained.
- API versioning explained.

Functional overview

Cisco TMSBA makes it possible to let users book resources from custom booking applications (in this document referred to as "clients"), integrating the custom booking application with the Cisco TMS reservation database.

Using Cisco TMSBA to interact with Cisco TMS is the only safe and supported way of integrating third party or custom booking applications with Cisco TMS. The Cisco TMSBA is a versioned and stable interface into Cisco TMS, where backward compatibility is guaranteed in new releases.

Main features

Cisco TMSBA has three main features:

- Importing and/or displaying Cisco TMS-managed resources in an external client or booking system.
- Reading and displaying system availability information and bookings from the reservation database of Cisco TMS. This information can be used for replicating bookings from Cisco TMS to the external booking system.
- Forwarding booking requests made by an external booking system to Cisco TMS. Forwarding a booking request to Cisco TMS lets the client utilize the routing logic of Cisco TMS, which determines whether network resources like an Multipoint Control Unit (MCU) are required, and automatically reserves these resources. The client therefore does not need to worry about infrastructure resources such as gateways and MCU port availability.

Booking ownership

The client can choose whether to authenticate with Cisco TMS as a service account or as the user actually making the booking. If authenticating as a service account, the client must explicitly specify the actual owner of the booking when performing the **`SaveConference`** function. If choosing this option, it is important that the service account is a member of a group having *Book on behalf of* permissions in Cisco TMS.

The **`IsTMSBookOnBehalfUser`** in the remote setup API can be used to assess whether a service account has *Book on behalf of* permissions. The **`GenerateConferenceAPIUser`** function can be used for creating a service account. See Remote setup API reference.

Conference routing

When booking through the Cisco TMSBA, clients book endpoints only. Network infrastructure products, such as gateways and Multipoint Control Units (MCUs) are automatically added by Cisco TMS if needed. The settings under **Administrative Tools > Conference Settings > Advanced Conference Options** in Cisco TMS are used when routing conferences created through Cisco TMSBA.

Replication

Cisco TMSBA supports two-way replication between the client and Cisco TMS. If the client maintains its own copy of the reservation database, it must periodically ask Cisco TMS for changes.

In Cisco TMS, all changes to bookings are tracked in the Cisco TMS (tmsng) database. The data in this table is available in Cisco TMSBA through the `GetTransactionsSince` function. This function gives the client a list of recent changes on the Cisco TMS side. This function should be run fairly often (such as every five minutes) so that the client and Cisco TMS reservation databases are in a consistent state.

Users in Cisco TMS have an "Exchange Integration Service Account" attribute. If this attribute is set to "yes", Cisco TMS sends an email message to the user every time a booking is created, changed, or deleted. By monitoring the mailbox of an "Exchange Integration Service Account" for new email, clients can be prompted to run the `GetTransactionsSince` function.

However, this should not be the only mechanism triggering replication from the client side, as email may be delayed or fail to reach the mailbox of the "Exchange Integration Service Account". Clients should thus always call the `GetTransactionsSince` function periodically as a fallback mechanism.

Cisco TMS entities

System

A system entity is any item that can be booked. Note that systems provisioned using Cisco TelePresence Management Suite Provisioning Extension or Cisco TMS Agent Legacy cannot be booked and are therefore not considered systems in this context.

There are two types of system entities: systems that are properly managed by Cisco TMS (and thus receive phonebooks, generate CDRs and so on), and systems added to Cisco TMS as the "Room" and "Equipment" types. Other types of entries, such as phonebook entries or provision directory users, are not system entities in Cisco TMS.

Each system entity has its own **TMS System Id**, which is used to uniquely identify the system in Cisco TMS. A subset of this system data is available in Cisco TMSBA through the `GetSystemById` function.

In the tmsng database, systems are stored in the **objSystem** table. System settings (such as SIP URIs, gatekeeper addresses, and software versions) are stored in the **field_SystemField** table.

Conference

A conference entity, or booking, is a reservation in Cisco TMS. Each conference has a unique conference ID.

Conference and their attributes (ID, start time, end time, title, and so on) are stored in the tmsng database in the **ScheduledCall** table. Phonebook entries, systems, and external participants that are part of a conference are stored in the ScheduledParticipant table. The Cisco TMSBA functions that interact with conferences, such as `GetConferenceById`, get and set data to and from both tables..

User

The Cisco TMS user entity holds information about Cisco TMS users, such as their first name, last name, username, and email address. All users have a unique user ID.

This information is stored in the **AclUser** table in the tmsng database

API versioning

Cisco TMSBA versioning is designed to provide a backwards compatible API to clients. This means that applications written for an older version of the API will keep working when upgrading to Cisco TMS with a newer version of the API.

Each SOAP message sent to the API by the client must announce which version it complies with in a header. The API will filter what is returned to the client based on this announced version.

ExternalAPIVersionSoapHeader

Each call made to the Cisco TMSBA must include the following header specifying the version of the API. The value specified in ClientVersionIn is used by the API to determine the output from the function. The XML below describes the ExternalAPIVersionSoapHeader object that is common for all calls to the API.

Do not set a number greater than the latest version of the API, as this may break compatibility when using later revisions.

```
<ExternalAPIVersionSoapHeader
xmlns="http://tandberg.net/2004/02/tms/external/booking/">
    <ClientVersionIn>int</ClientVersionIn>
    <ClientIdentifierIn>string</ClientIdentifierIn>
    <ClientLatestNamespaceIn>string</ClientLatestNamespaceIn>
    <NewServiceURL>string</NewServiceURL>
</ExternalAPIVersionSoapHeader>
```

If no version number is set, version 0 is the default.

Version history

Version 0 of the API corresponds to version 9 of Cisco TMS. Not all versions of Cisco TMS include changes to the API. Below is an overview of Cisco TMS versions that included a new version of Cisco TMSBA.

API version	Cisco TMS version
9	13.2
8	13.1.2
7	13.1
6	13.0.1
5	12.6
4	11.9
3	11.0
2	
1	
0	9

Versioning example

When a change to the API is needed, the WSDL must be updated. The WSDL is the contract towards the API clients and provides information covering all methods and data types, including values, that the Booking API supports.

When, for example, support for an additional bandwidth value is added to the API, the WSDL changes, and the version number is increased. If this new bandwidth value is added in version 9 of the API, every client declaring that they are using version 9 or later will get this new value returned from Cisco TMSBA.

Clients that announce compliance to versions earlier than 9 will *not* get the new value returned. Instead , the API returns the special value *default*. The API allows all incoming values, but filters values going out. A client declaring a version lower than 9, that then enumerates a conference with the new bandwidth value, will get the special value *default* returned.

If the client uses this conference and updates the conference with bandwidth set to default, no change to the bandwidth value is made, as the API knows this special value means no changes should be made.

This filtering of values that older clients do not understand makes the API backwards compatible; older clients built on older WSDLs can still use an API with a higher version.

Using the APIs for remote setup and booking

In this chapter:

- Usage requirements for the APIs.
- Setting up your environment to let your development tools access and work with the APIs.
- Suggestions for GUI pattern and Replication pattern to use when implementing the APIs.
- Known limitations of the APIs.

Usage requirements

This section describes the licensing, permission, and authentication requirements for using the APIs.

Licensing

One of the following licenses must be available to make full use of Cisco TelePresence Management Suite Extension Booking API:

- One Cisco TMS Application Integration License for each server using the API.
- One Cisco TelePresence Management Suite Extension Booking API license per 25 registered systems.

The following API functions require one of the above licenses:

- GetRecordingAliases
- GetTransactionsSince
- SaveConference
- SaveConferenceRecInstance
- SaveConferences

The remaining API features do not require special licensing for use. Contact your Cisco reseller/partner for more licensing information.

Booking rights

Importing from Cisco TMS and booking meetings through the API requires authentication with Cisco TMS. All users who will book meetings using the API must be members of a group whose permissions include *Booking*. Users who book on behalf of others also require *Book on behalf of* permissions.

Permissions in Cisco TMS are set on a group level. To modify the permissions set for a group:

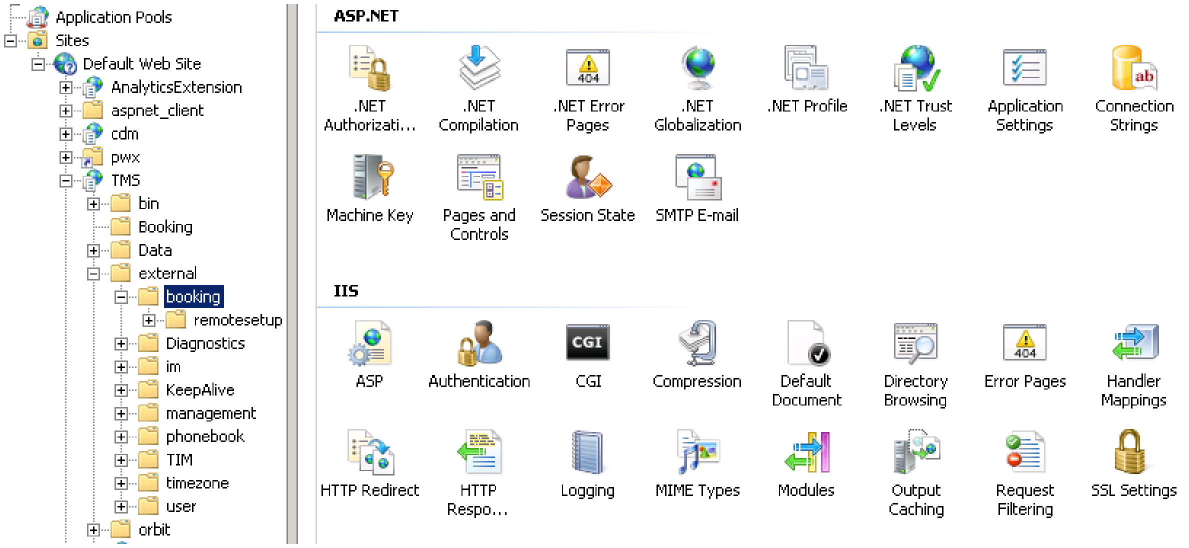
1. Go to **Administrative Tools > User Administration > Groups**.
2. Hover over the desired group, click the drop-down button and select **Set permissions**.
3. In the **Booking** section, under **Misc**, check *Booking* and, if appropriate, *Book on behalf of*.
4. Click **Save**.

NTLM authentication

On a default Cisco TMS installation, any API requires the use of Windows Challenge Response or NTLM authentication.

Not all environments support this authentication mechanism (non-Windows based environments), and you therefore may need to allow for Basic Authentication:

1. Open Internet Information Services manager on the Cisco TMS server.
2. Expand **Sites > Default Web Site** and browse to the **/TMS/external/booking** virtual directory.



3. In the **IIS** section, double-click on **Authentication**.
4. Right-click on **Basic Authentication** and select **Enable**.

Allowing Anonymous Authentication is not recommended and requires the **IUSR_<machinename>** to be given *Book on behalf of* permissions in Cisco TMS.

Setting up your environment

Cisco TMSBA provides a Web Services API that interfaces with the Cisco TMS booking engine. Web Services allows for simple integration into most common language and programming environments.

See your development tool reference for information on how to build implementation stubs to help speed the development of applications that use Web Services.

API locations

- The WSDL file for the Cisco TMS remote setup API is located at: **http://[Cisco TMS server]/tms/external/booking/remotesetup/remotesetupservice.asmx**.
- The WDSL file for Cisco TMSBA is located at: **http://[Cisco TMS server]/tms/external/booking/bookingservice.asmx**.

Microsoft Visual Studio .NET users can reference the APIs by selecting **Project > Add Web Reference**, or entering the URLs above. You will be required to authenticate through web services to create the reference.

In a network load balancing scenario, use the virtual IP address or DNS name of the cluster for this task to allow failover for the API.

GUI pattern

Cisco TMSBA can supply data to the front-end GUI of an external booking application. There are three information types.

System information

Information on Cisco TMS resources can be exported to an external application. By using the remote setup API, data on systems in Cisco TMS can be exported to a front-end GUI and used to display system entities available in Cisco TMS.

Get a list of available systems in Cisco TMS by using either:

- **`GetSystems`**
- **`GetSystemsForUser`**

These functions return lists of TMSSystem objects, that include information such as the ID of the system, to show in the front-end GUI. **`GetSystems`** will return all systems in Cisco TMS, while **`GetSystemsForUser`** will only return the systems the user has booking privileges for. If the external GUI application controls system access, use **`GetSystems`** and filter the systems in the application.

Availability information

Information on the availability Cisco TMS entities can be exported to an external application. Reservations in the Cisco TMS internal reservation database can be displayed and bookings filtering by users.

- Get all Cisco TMS reservations between two specified dates by using **`GetConferencesForUser`**.
- Get system availability information by using **`GetConferencesForSystems`**.
- Get a specific conference, including any exceptions if it is a recurring conference, by using **`GetRecurrentConferenceById`**. This information can be used, for example, by an external application to display an availability calendar.
- Filter availability information by using the Conference object.
- The Remote Setup API function **`GetUsers`** returns all users registered in Cisco TMS. The output of this function can be used to display a drop-down list of all users in Cisco TMS, or show conferences booked by a specific person.

Booking management

The API allows you to forward booking requests from an external booking system to Cisco TMS, and reserve resources in Cisco TMS. (Information exchange: **External Booking System > TMS**).

- Get Conference objects with default values for Conference properties defined in Cisco TMS by using the **`GetDefaultConference`** function.
- Retrieve existing conferences by using **`GetConferenceById`**, **`GetConferenceIdByExternalId`** or **`GetRecurrentConferenceById`**.

- Save changes to a conference by editing the conference properties and using the function **`SaveConference`**. This will save the conference to Cisco TMS if the properties validate. If not, an exception will be raised.
- Delete a conference by using **`DeleteConferenceById`**. Beware that conference participants will be disconnected if the conference is deleted while it is active or connected.
- Add recording to a conference by using **`GetRecordingAliases`** to get information about a user's recording aliases and use this information to add recording participant(s) to the conference.

Replication pattern

The APIs can be used in conjunction with external booking applications that have their own reservation database. There are three main components.

Import

The API can automate importing systems from Cisco TMS into a third-party application, or this can be initiated by the user via a GUI.

Get a list of available systems in Cisco TMS by using either of the following from the remote setup API:

- **`GetSystems`**
- **`GetSystemsForUser`**

These functions return lists of TMSSystem objects, and information such as the ID of the system, for use by a third party application. GetSystems will return all systems in Cisco TMS, while GetSystemsForUser will return only the systems the user has booking privileges for. If the external application controls system access, use GetSystems and filter the systems in the application.

Replication

External booking systems can keep track of booking transactions on the Cisco TMS server, and replicate bookings made using Cisco TMS. This part does not apply to external GUI front ends that do not have their own reservation database.

Get a list of transactions listed by transaction ID by using the GetTransactionsSince function. All conferences have a transaction ID property

The list of transactions contains:

- The transaction type (*New*, *Update*, and *Delete*)
- An associated conference ID.

Use **`GetConferenceById`** to get an updated Conference object – and update the conference with the external source. The current transaction ID should then be updated to the last conference’s TransactionId.

Booking

The API allows you to forward booking requests from an external booking system to Cisco TMS, and reserve the resources there.

- Get Conference objects with default values for Conference properties defined in Cisco TMS by using **`GetDefaultConference`**.
- Retrieve saved conferences by using one of the following functions:
  - **`GetConferenceById`**
  - **`GetConferenceIdByExternalId`**
  - **`GetRecurrentConferenceById`**

    - **`GetConferencesForUser`**
    - **`GetConferencesForSystems`**

- Save changes to a conference, by editing the conference properties and use function **`SaveConference`**. This will save the conference to Cisco TMS if the validation of the properties is OK. If not, an exception will be raised.
- Delete a conference by using the **`DeleteConferenceById`** function. Note that conference participants will be disconnected if the conference is deleted while it is active or connected.

Availability

The API allows you to display reservations from theCisco TMS internal reservation database.

Known limitations

No support for setup and teardown buffers

Using Cisco TMSBA is not currently compatible with the use of setup/teardown buffers in Cisco TMS scheduling.

Unused values

The following values are present in the API, but not in use:

Function	**Attribute**	**Value**
RecurrencePattern	PatternEndType	Never
RecurrencePattern	FrequencyType	■ *Secondly* ■ *Minutely* ■ *Hourly* ■ *Yearly*
Participant	ParticipantCallType	User
ISDNBandwith	Bandwidth	Max
ISDNBandwith	IPBandwidth	Max

Remote setup API reference

The remote setup API accommodates the setup of users and systems prior to using the Booking API.

In this chapter:

- The TMSSystem object
- The TMS user object
- Remote Setup API functions

TMSSystem object

The TMSSystem object contains information about a system in Cisco TMS. This object is used to read information, as the remote setup API does not support updating system information in Cisco TMS.

Use this object to import the required information into the third party application. The **SystemId** is required to connect the application entity with the system in Cisco TMS. In addition other information can be imported and shown for informative purposes, for example the name of the system.

TMSSystem

SystemId	The ID of the system in Cisco TMS. Use this to refer to the associated system in Cisco TMS from your application. For example, when booking a conference, insert the IDs of the chosen systems into the Conference object.
SystemName	The name of the system in Cisco TMS. Use this to display the name of the system in your application.
Contact	The system contact associated with the system in Cisco TMS.
Manufacturer	The manufacturer of the system. For example, Cisco.
Description	A textual description stored in Cisco TMS. This file can contain information such as the number of chairs in the meeting room where the system is located.
SystemType	The type of system. For example, Cisco TelePresence System EX90.
QNumber	The IP or DNS address of the system.
ISDNNumber	The ISDN location where the system is located.
Location	The ISDN number of the system.
NetworkAddress	The fully qualified ISDN number of the system. A fully qualified ISDN number always includes the country code and area code. This is not implemented.
WebInterfaceURL	The http address of the web server of the system.
SIPUri	The SIP URI of the system.
H323Id	The H.323 ID of the system.
E164Alias	The E.164 alias of the system.
TimeZone	The time zone where the system is located.
SystemCategory	The system category.
SystemStatus	The status of the system.

TimeZone

TimezoneName	The name of the time zone.
StartTimeDTS	The start date of daylight saving time.
EndTimeDTS	The end date of daylight saving time.
GMTOffset	The GMT (UTC) offset.

SystemCategory

SystemCategory	An enumeration value of what category of system this is. The available options are: ▪ *Endpoint* ▪ *Equipment* ▪ *Room* ▪ *Recording*

SystemStatus

SystemStatus	An enumeration with the status of the system when this function is call. Note that the status of the system can change frequently. The possible values are: ▪ *Alive* ▪ *Idle* ▪ *InCall* ▪ *NoResponse* ▪ *Unknown*

TMSSystem object XML

The XML below describes the TMSSystem object. Following the XML is a description of the elements and what information each element it contains. Note that as all fields are not required, the output may contain less system information than the object can hold.

```
<TMSSystem>
    <SystemId>long</SystemId>
    <SystemName>string</SystemName>
    <Contact>string</Contact>
    <Manufacturer>string</Manufacturer>
    <Description>string</Description>
    <SystemType>string</SystemType>
    <NetworkAddress>string</NetworkAddress>
    <Location>string</Location>
    <ISDNNumber>string</ISDNNumber>
```

```
<QNumber>string</QNumber>
<WebInterfaceURL>string</WebInterfaceURL>
<SIPUri>string</SIPUri>
<H323Id>string</H323Id>
<E164Alias>string</E164Alias>
<TimeZone>
    <TimezoneName>string</TimezoneName>
    <StartTimeDTS>string</StartTimeDTS>
    <EndTimeDTS>string</EndTimeDTS>
    <GMTOffset>string</GMTOffset>
</TimeZone>
<SystemCategory>
    <systemCategory>Endpoint or Equipment or Room or Recording</systemCategory>
</SystemCategory>
<SystemStatus>
    <SystemStatus>Alive or Idle or InCall or NoResponse or Unknown</SystemStatus>
</SystemStatus></TMSSystem>
```

TMS user object

The Cisco TMS user object contains information about Cisco TMS users. Use this object to access information about users in Cisco TMS. The XML document below describes the User object. Following the XML is a description of the elements and what information each element it contains.

User

Attribute	Description
DisplayName	The display name of the user.
EmailAddress	The e-mail address of the user.
FirstName	The first name of the user.
LastName	The last name of the user.
UserName	The Windows login name of the user.
IsHiddenUser	Boolean value used to represent whether this is a normal user (True), or a service account (False) that normally should not be displayed in a list of users.
TimeZone	The time zone where the user is located. Uses the same TimeZone object as TMSSystem.

TMS user object XML

```
<User>
    <DisplayName>string</DisplayName>
    <EmailAddress>string</EmailAddress>
    <FirstName>string</FirstName>
    <LastName>string</LastName>
    <UserName>string</UserName>
    <IsHiddenUser>boolean</IsHiddenUser>
    <TimeZone>
        <TimezoneName>string</TimezoneName>
        <StartTimeDTS>string</StartTimeDTS>
        <EndTimeDTS>string</EndTimeDTS>
        <GMTOffset>string</GMTOffset>
    </TimeZone>

</User>
```

Remote Setup API functions

This reference section describes all the available functions of the remote setup API.

DisableConferenceAPIUser

This function is used to disable a ConferenceAPI user. E-mail notifications for the user are disabled, and the user is removed from all groups in Cisco TMS except the Users group. This is done to keep references valid. Executing this function requires Cisco TMS Site Administrator privileges.

This function is typically used during uninstall procedures.

Supported parameters:

`userName`	The full username of the user to delete in NT4 style (domain\username).

Returned data: None.

GenerateConferenceAPIUser

This function generates a Cisco TMS Booking API account in the default user container on the Cisco TMS server, including registering the user in Cisco TMS. It is typically used during installation/setup procedures.

The user will be:

- Hidden from normal user lists.
- Added to the Site Administrator Group.
- Configured to receive e-mail event scheduling notifications for all creation/update/deletions of bookings. These are typically used for updating the external booking system with changes done on the Cisco TMS server.

In order for the function to complete, the current user must be:

- A Cisco TMS Site Administrator
- A local computer administrator

Supported parameters:

`userNameBase`	The base portion of the user name. If a user with the name already exists a numeric postfix is added (e.g. tms-confuser ==> tms-confuser1).
`encPassword`	A base64 encoded password that is to be used for the newly created user.
`emailAddress`	The email address of the user.
`sendNotifications`	If the user should receive scheduling notifications.

Returned data: The username of the created user (NT4 domain/username style).

GetSystemById

This function returns information about a specific system. If the system is not found, this causes an error.

Supported parameters:

`TMSSystemId`	System ID as given in Cisco TMS.

Returned data: A TMSSystem object.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

GetSystems

This function returns all endpoints and rooms registered in Cisco TMS. Note that network systems, such as a Cisco TelePresence MCU, are not returned since they are normally not booked by the users, but are added to the conference by Cisco TMS if required.

Typically used during setup of resources in the external booking system to connect resources in Cisco TMS with resources in the external booking system.

Supported parameters: None

Returned data: An array of TMSSystem objects.

GetSystemsForUser

This function returns all endpoints and rooms that can be booked by the current user, the account credentials are used to communicate with Cisco TMSBA.

Note that network systems, such as a Cisco TelePresence MCU, are not returned since they are normally not booked by the users, but are added to the conference by Cisco TMS if required.

Typically used in the external booking system to list Cisco TMS resources in external booking system.

Supported parameters: None.

Returned data: An array of TMSSystem objects.

IsAlive

This function checks the connection to the web services of Cisco TMS.

It is typically used during installation to check the URL to this web service.

Supported parameters: None

Returned data: A boolean value true/false, which is *true* if the connection works.

GetUsers

This function returns all users registered in Cisco TMS.

This function is typically used in the front-end GUI to provide a list of Cisco TMS users, and can filter output from the Cisco TMSBA based on users from this output.

Supported parameters: None.

Returned data: An array of User objects.

IsLocalAdmin

This function checks whether the current user can create local or Active Directory accounts in the default user container on the Cisco TMS server.

This is typically used during installation to check whether the user installing the integration has sufficient access to Active Directory. This function must return *True* in order for the GenerateConferenceAPIUser function to succeed.

Supported parameters: None.

Returned data: A boolean value true/false, which is *true* if the user is a local administrator.

IsTMSServiceUser

This function is used to check whether the current user is flagged as an Exchange Integration user and has access to book on behalf of other users.

This is typically used during installation to check whether the user installing the integration has sufficient access permissions for the Cisco TMS server.

Supported parameters: None.

Returned data: A boolean value true/false, which is *true* if the user is a Cisco TMS service user.

IsTMSSiteAdmin

This function checks whether the current user is a member of the Cisco TMS Site Administrators group.

This is typically used during installation to check whether the user installing the integration has enough access towards the Cisco TMS server. This function should return True in order for the `GenerateConferenceAPIUser` function to succeed.

Supported parameters: None.

Returned data: A boolean value *true*/*false*, which is *true* if the user is a Cisco TMS Site Administrator.

Booking API reference

The booking API lets you schedule conferences in Cisco TMS using a third-party client, and replicate existing bookings between the two. This chapter is a reference to the conference object and to all available functions and parameters.

In this chapter:

- Conference object described and explained
- Booking API functions

Conference object

This object can be used to read and write

- Conference properties such as **Start Time**, **End Time**, **Conference Title**, and **Conference Password**.
- Conference call-related values such as **Bandwidth**, **Picture mode**, and **Encryption mode**.

All conference resources, including video participants, audio participants, phone book participants, external participants and so on, are held in this object, together with the call route for connecting the resources. You also use the Conference object to define the conference type, that is, how the conference should be connected.

Conference data can be saved/updated, and handled by Cisco TMS using the **`SaveConference`** function described below.

Conference

Attribute	Read/Write	Description
ConferenceId	r/w - optional	Initially set to -1 to state to the `SaveConference` function that the conference needs to be created. If set to a value greater than 0, the existing conference with the given ID is updated with the conference specified.
Title	r/w - optional	If no title is specified, the Cisco TMS default will be used.
StartTimeUTC	r/w - required	The start and end times of the conference in UTC format. See http://www.w3.org/TR/NOTE-datetime for more information. Only UTC times, ending with Z, are supported. Example: 1975-06-0123:32:11Z.
EndTimeUTC	r/w - required	
RecurrenceInstanceIdUTC	r - only used when getting conference from Cisco TMS	Gives the start date of the instance of the meeting according to the recurrence pattern. If this is different from **StartTimeUTC**, the meeting is an exception to the recurrence pattern.
RecurrenceInstanceType	r - only used when getting conference from Cisco TMS	If this string contains the value 'modify' it means that the particular meeting is an exception to a recurrence pattern. It the string contains "deleted", it is a meeting that has been deleted from a series of recurring meetings.
FirstOccurrenceRecInstanceIdUTC	r - only used when getting conference from Cisco TMS	Gives the start date of first instance of the meeting according to the recurrence pattern.
RecurrencePattern	r/w – optional	Sets the recurrence patterns for recurrent meetings. This is not valid if you call the SaveConferenceRecInstance function.
OwnerId	r/w - optional	The user ID of the owner of the conference. This is typically not set by the external booking API, but by Cisco TMS. If no owner is specified, the user authenticated to the webservice is used. IDs of users existing in Cisco TMS can be found in the cluster table of the Cisco TMS database.
OwnerUserName	w - optional	The username of the person booking the conference, used for looking up the owner ID in the Cisco TMS database. If no owner is specified, the user authenticated to the webservice is used. If **OwnerId** is specified, it will be used instead of this field.
OwnerFirstName	w - optional	The username, first and last name, and email address of the person booking the conference. This information is used for looking up the owner id in the Cisco TMS database. If no owner is specified, the user authenticated to the webservice is used. If OwnerUserName or OwnerId are specified, those fields will be used instead.
OwnerLastName	w - optional	
OwnerEmailAddress	w - optional	

Attribute	Read/Write	Description
ConferenceType	r/w - optional	Setting determining how the conference will be launched. The valid values are: ▪ *Automatic Call Launch*, connect all participants at conference start time and disconnect them again at conference end time. ▪ *One Button to Push*, allows for OBTP call setup on supported systems. ▪ *Manual Call Launch*, the conference master participant will be asked to connect the call at conference start time. ▪ *Reservation Only*, reserve the participants for the conference duration. ▪ *Ad-Hoc conference*, ▪ *Default*, keep the conference type specified on initial booking of the conference. If unspecified, the default conference type configured in Cisco TMS will be used.
Bandwidth (Discontinued)	—	This item has been deprecated and is included for backwards compatibility only. Use **ISDNBandwidth** and **IPBandwidth** to control the conference bandwidth.
PictureMode	r/w - optional – if not specified, Default is assumed	The picture mode/conference layout to use for the conference. The valid values are: ▪ *Continuous Presence*, ▪ *Enhanced CP*, ▪ *Voice Switched*, ▪ *Default*, keep the picture mode specified on initial booking of the conference. If unspecified, the default conference type configured in Cisco TMS will be used.
Encrypted	r/w - optional – if not specified, Default is assumed	The encryption mode for the conference. The valid values are: ▪ *Yes* ▪ *No* ▪ *If Possible* ▪ *Default*, keep the picture mode specified on initial booking of the conference. If unspecified, the default encryption setting configured in Cisco TMS will be used.
DataConference	r/w - optional	Whether a data conference should be added to the conference. The valid values are: ▪ *Yes* ▪ *No* (default) ▪ *If Possible* ▪ *Default*, keep the data conference setting specified on initial booking of the conference.

Attribute	Read/Write	Description
ShowExtendOption	r/w - optional, if not specified, Default is assumed	Set this value to specify extend option behavior when the scheduled conference is close to ending. - *Yes*—prompt the VC Master (the first endpoint in the participant list) to extend the conference - *No*—do not offer to extend conference - *Automatic Best Effort*—automatically extend conference - *Default*—keep the Show Extend Option setting specified on initial booking of the conference. If unspecified, the default Show Extend Option defined in the **Administrator Tools** Page in Cisco TMS is used.
Password	r/w - optional	The password for the conference participants must enter to join the call. Auto-generated password may be added if Cisco TMS is set up to create them.
BillingCode	r/w - optional	The billing code to use for the conference. If Cisco TMS requires billing codes, this field must be specified and will be validated against the list of billing codes in Cisco TMS. If no match is found, the conference will not be created and the API will throw an "Invalid billing code" exception. This setting is blank by default.
ISDNRestrict	r/w - option	Whether the ISDN channels should be restricted (for example use 54k and not 64k). The default is *No*.
ConferenceInfoText	r – only used when getting conference from Cisco TMS.	Information on how the conference will connect, including routing information. Based on Cisco TMS templates.
ConferenceInfoTextHTML	r – only used when getting conference from Cisco TMS.	Information in HTML markup on how the conference will connect, including routing information. Based on Cisco TMS templates.
UserMessageText	r/w – optional	A user definable text/description of the conference. Default is blank.
ExternalSourceId, **ExternalPrimaryKey**	r/w – optional	A user definable external source and ID, this is used to synchronize the Cisco TMS database with the external sources database. If Cisco TMS is given a value for these fields, Cisco TMS will return the value for all instances of the same conference. Default is blank.
Participants	r/w, required	List of conference participants. When calling `GetDefaultConference`, the participant list will be empty.

Attribute	Read/Write	Description
RecordedConferenceUri	r – only used when getting conference from Cisco TMS	If the conference is recorded, this is the URI of the conference recording.
WebConferencePresenterUri	r – only used when getting conference from Cisco TMS	If the conference includes a web conference, this is the URI of the presenters web conference.
WebConferenceAttendeeUri	r – only used when getting conference from Cisco TMS	If the conference includes a web conference, this is the URI of the attendee web conference. If this field and **WebConferencePresenterUri** is the same, they will both include the same information.
ISDNBandwidth	r/w - optional	The ISDN and IP bandwidths of the conference. If unspecified, the default ISDN bandwidth configured in Cisco TMS will be used. The valid values are: ▪ *1b/64kbps* ▪ *2b/128kbps* ▪ *3b/192kbps* ▪ *4b/256kbps* ▪ *5b/320kbps* ▪ *6b/384kbps* ▪ *8b/512kbps* ▪ *12b/768kbps* ▪ *18b/1152kbps* ▪ *23b/1472kbps* ▪ *30b/1920kbps* ▪ *32b/2048kbps* ▪ *48b/3072kbps* ▪ *64b/4096kbps* ▪ *Max*, use as much bandwidth as available and necessary. ▪ *Default*, keep the bandwidth specified on initial booking of the conference.
IPBandwidth	r/w - optional	

Participant

Attribute	Read/Write	Description
ParticipantId	r/w – optional	For Cisco TMS System Entities, this value must be the SystemId of the system. For external participants this value may be set, but is not required. If not set for external participants, Cisco TMS will create an ID with an integer less than 0.
NameOrNumber	r/w – optional	For external participants, the participant name for dial-ins, or the fully qualified number to dial for dial-outs. For example: ■ A dial-in can be given the value `Placeholder for John Doe`. ■ An ISDN dial-out would be given the value `+1 (555) 1231234`. This value is required for external dial-out participants, and must be the fully qualified number to dial. Fully qualified numbers are of the format +CC (AC) BN where: ■ CC=Country Code ■ AC=Area Code ■ BN=Base Number If the country does not use Area Codes, that element can be omitted completely, and the format would be +CC BN.
ParticipantCallType	r/w – required	The participant type. Valid values are: ■ *TMS*, a TMS System Entity. When this is specified, the **ParticipantId** must be the Cisco TMS System Entity ID as given in Cisco TMS. ■ *IP Video* <- *or ISDN Video* <-, an IP/ISDN video dial-in. If this is specified, you may give the participant a name using the **NameOrNumber** field. Cisco TMS will automatically give the participant an ID (less than 0). ■ *IP Tel* <- *or Telephone* <-, an IP/ISDN audio dial-in. If this is specified, you may give the participant a name using the **NameOrNumber** field. Cisco TMS will automatically give the participant an ID (less than 0). ■ *IP Video* -> *or ISDN Video* ->, an IP/ISDN video dial-out site. Specifying this value requires providing Cisco TMS with the number to use in the **NameOrNumber** field. (Formats: ISDN: `+1 (555) 1231234`, H.323 IP E.164: `12312321`, H323 IP Address: `10.0.0.10`). ■ *IP Tel* -> *or Telephone* ->, an IP/ISDN audio dial-out site. If this is specified, you must give Cisco TMS the number to use in the NameOrNumber field (Formats: ISDN: `+1 (555) 1231234`, H.323 IP E.164: `12312321`, H.323 IP Address: `10.0.0.10`). Call will be placed using 64kbps/54kbps depending on restrictions. ■ *TMS Master Participant*, the conference master. When this entity is specified, the **ParticipantId** must be the Cisco TMS System Entity ID as given in Cisco TMS. It is only possible to specify a single Master Participant per conference, which must be a Cisco TMS System. *User* is not currently supported by the API.

RecurrencePattern

Attribute	Read/Write	Description
FrequencyType	r/w required	The frequency of the recurrence rule. Legal values are: ■ *Daily* ■ *DailyWeekday* ■ *Weekly* ■ *Monthly* ■ *Default* Available, but unsupported values are: ■ *Secondly* ■ *Minutely* ■ *Hourly* ■ *Yearly*
Interval	r/w required	Every X day/week/month as selected by **FrequencyType**
DaysOfWeek		Days of week if relevant for your **FrequencyType**.Valid values are full names of days, capitalized. ■ Set the day if the meeting is to occur on one fixed day of the week. ■ Set all days for a meeting that occurs every day. ■ Set all weekdays for a meeting that occurs every weekday. ■ Set *Saturday* and *Sunday* for a meeting that occurs only on weekends.
FirstDayOfWeek		First day of week. Used to split **DaysOfWeek** into "every X weeks" weekly patterns. The default value is *Sunday*.
BySetPosition		Relative position of the instance in a pattern. For example, in a monthly pattern, a value of 2 means second day of the month, -1 means last day of the month. The allowed days must be defined in **DaysOfWeek**. 0 means every X day of the month.
PatternEndType		End type: ■ by number of occurrences ■ by date ■ never (not currently supported)
PatternEndDateUTC		In the case where **PatternEndType** is by date, this gives the end date of the recurrence pattern.

Attribute	Read/Write	Description
FirstOccurrenceRecInstanceIdUTC		Gives the original start time of the first occurrence of this meeting. If the meeting is not an exception to the recurrence pattern, this time will be the same as the start time of the meeting. If the meeting time for the occurrence has been modified, this string gives the original start time according to the recurrence pattern. This field is only populated by the API when **PatternEndType** is by number of occurrences.
PatternInstances		In the case where **PatternEndType** is by number of instances, defines the number of instances to generate from the pattern.
Exceptions		Exceptions to the pattern are supported using the **GetRecurrentConferenceById** and **SaveConference** functions. To get a conference with all its exceptions, use **GetRecurrentConferenceById**. To update a conference with exceptions, use the **SaveConference** function providing the exceptions in the **RecurrencePattern.Exceptions** array before saving the conference. As an alternative: 1. Use **GetConferenceIdByExternalId** with **RecInstanceIdUTC** (UTC string that points to the UTC day of the instance) to get conference id for the instance. 2. Use **SaveConferenceRecInstance** to save this exception.

ISDNBandwidth

Attribute	Read/Write	Description
Bandwidth		The ISDN bandwidth that will be used when dialing the conference participants and when creating the conference. Note that *Max* is not currently supported. Example value **`3b/193kbps`**. If *Default* is selected, the value is set to the default conference ISDN bandwidth as defined in the **Administrator Tools** page in Cisco TMS.
IPBandwidth		The IP bandwidth that will be used when dialing the conference participants and when creating the conference. Note that *Max* is not currently supported. Example value **`3b/193kbps`**. If *Default* is selected, the value is set to the default conference IP bandwidth as defined in the **Administrator Tools** page in Cisco TMS.

Conference object XML

The XML document below describes the Conference object.

```
<Conference>
       <ConferenceId>int</ConferenceId>
       <Title>string</Title>
       <StartTimeUTC>string</StartTimeUTC>
       <EndTimeUTC>string</EndTimeUTC>
       <RecurrenceInstanceIdUTC>string</RecurrenceInstanceIdUTC>
       <RecurrenceInstanceType>string</RecurrenceInstanceType>
       <FirstOccurrenceRecInstanceIdUTC>string</FirstOccurrenceRecInstanceIdUTC>
       <RecurrencePattern>
               <FrequencyType>Daily or DailyWeekday or Weekly or Monthly or Yearly or Secondly or
Minutely or Hourly or Default</FrequencyType>
               <Interval>int</Interval>
               <DaysOfWeek>
               <DayOfWeek>Sunday or Monday or Tuesday or Wednesday or Thursday or Friday or
Saturday</DayOfWeek>
               <DayOfWeek>Sunday or Monday or Tuesday or Wednesday or Thursday or Friday or
Saturday</DayOfWeek>
               </DaysOfWeek>
               <FirstDayOfWeek>Sunday or Monday or Tuesday or Wednesday or Thursday or Friday or
Saturday</FirstDayOfWeek>
               <BySetPosition>int</BySetPosition>
               <PatternEndType>EndByDate or EndByInstances or EndNever or Default</PatternEndType>
               <PatternEndDateUTC>string</PatternEndDateUTC>
               <FirstOccurrenceRecInstanceIdUTC>string</FirstOccurrenceRecInstanceIdUTC>
               <PatternInstances>int</PatternInstances>
               <Exceptions>
                       <RecurrenceException xsi:nil="true" /><RecurrenceException xsi:nil="true" />
               </Exceptions>
       </RecurrencePattern>
       <OwnerId>long</OwnerId>
       <OwnerUserName>string</OwnerUserName>
       <OwnerFirstName>string</OwnerFirstName>
       <OwnerLastName>string</OwnerLastName>
       <OwnerEmailAddress>string</OwnerEmailAddress>
       <ConferenceType>Reservation Only or Automatic Call Launch or Manual Call Launch or
Default or Ad-Hoc conference or One Button To Push</ConferenceType>
       <Bandwidth>1b/64kbps or 2b/128kbps or 3b/192kbps or 4b/256kbps or 5b/320kbps or
6b/384kbps or 8b/512kbps or 12b/768kbps or 18b/1152kbps or 23b/1472kbps or
30b/1920kbps or 32b/2048kbps or 48b/3072kbps or 64b/4096kbps or Max or
Default</Bandwidth>
       <PictureMode>Continuous Presence or Enhanced CP or Voice Switched or
Default</PictureMode>
       <Encrypted>Yes or No or If Possible or Default</Encrypted>
       <DataConference>Yes or No or If Possible or Default</DataConference>
       <ShowExtendOption>Yes or No or Default or AutomaticBestEffort</ShowExtendOption>
       <Password>string</Password>
       <BillingCode>string</BillingCode>
       <ISDNRestrict>boolean</ISDNRestrict>
       <ConferenceInfoText>string</ConferenceInfoText>
       <UserMessageText>string</UserMessageText>
       <ExternalSourceId>string</ExternalSourceId>
       <ExternalPrimaryKey>string</ExternalPrimaryKey>
       <Participants>
               <Participant>
                       <ParticipantId>int</ParticipantId>
                       <NameOrNumber>string</NameOrNumber>
                       <ParticipantCallType>TMS or IP Video <- or IP Tel <- or ISDN Video <- or Telepho
<- or IP Video -> or IP Tel -> or ISDN Video -> or Telephone -> or Directory or User
or SIP <- or SIP -> or 3G <- or 3G -> or TMS Master Participant</ParticipantCallType>
```

```
               </Participant>
               <Participant>
                       <ParticipantId>int</ParticipantId>
                       <NameOrNumber>string</NameOrNumber>
                       <ParticipantCallType>TMS or IP Video <- or IP Tel <- or ISDN Video <- or Telepho
<- or IP Video -> or IP Tel -> or ISDN Video -> or Telephone -> or Directory or User
or SIP <- or SIP -> or 3G <- or 3G -> or TMS Master Participant</ParticipantCallType>
               </Participant>
       </Participants>
       <RecordedConferenceUri>string</RecordedConferenceUri>
       <WebConferencePresenterUri>string</WebConferencePresenterUri>
       <WebConferenceAttendeeUri>string</WebConferenceAttendeeUri>
       <ISDNBandwidth>
               <Bandwidth>1b/64kbps or 2b/128kbps or 3b/192kbps or 4b/256kbps or 5b/320kbps or
6b/384kbps or 8b/512kbps or 12b/768kbps or 18b/1152kbps or 23b/1472kbps or
30b/1920kbps or 32b/2048kbps or 48b/3072kbps or 64b/4096kbps or Max or
Default</Bandwidth>
       </ISDNBandwidth>
       <IPBandwidth>
               <Bandwidth>1b/64kbps or 2b/128kbps or 3b/192kbps or 4b/256kbps or 5b/320kbps or
6b/384kbps or 8b/512kbps or 12b/768kbps or 18b/1152kbps or 23b/1472kbps or
30b/1920kbps or 32b/2048kbps or 48b/3072kbps or 64b/4096kbps or Max or
Default</Bandwidth>
       </IPBandwidth>
</Conference>
```

Booking API functions

This reference section describes all the available functions of the booking API.

DeleteConferenceById

Deletes a conference with the given ConferenceId (as defined in Cisco TMS). If the conference is part of a recurring series, the whole series will be deleted.

Supported parameters:

`ConferenceId`	The ConferenceId of the conference to delete.

Returned data: Nothing.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

DeleteConferenceRecInstanceById

Deletes an occurrence of a recurring conference with the given ConferenceId (as defined in Cisco TMS). This function is typically used when deleting a single meeting in a recurring series.

Supported parameters:

`ConferenceId`	The ConferenceId of the conference to delete.

Returned data: Nothing.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

EndConferenceById

Ends an ongoing conference with the given **ConferenceId** (as defined in Cisco TMS). The conference will be set to *Finished*, and the end time will be set to the time of execution of the function. This function is typically used to end a running conference from a third party front-end GUI. The function will fail with an error if the conference has not yet started.

Supported parameters:

`ConferenceId`	The ConferenceId of the conference to delete.

Returned data: Nothing.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

GetConferenceById

Get the available information about a particular conference.

Supported parameters:

`ConferenceId`	The ID of the conference (Based on Cisco TMS IDs)

Returned data: A Conference object based on the `ConferenceId`.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

GetRecordingAliases

Supported parameters:

`UserName`	The user to retrieve recording alias for. If no UserName is provided (empty string), the logged in user will be used.

Returned data: An array of RecordingDevice, where the key is the string representation of a recording device name, or a recording cluster name. The value is an array of AliasInfo for that particular recording device/cluster, holding an AliasId (string) and a SystemId (int) . The AliasId and SystemId can be used to add a recording participant to a conference.

GetConferencesForUser

This function returns all conferences owned by a particular user between two dates.

Supported parameters:

`UserName`	The Cisco TMS user to get bookings for. If no user name is provided (empty string), the logged in user is used.
`StartTime`	The start date and time of bookings.
`EndTime`	The end date and time of bookings.
`ConferenceStatus`	An enumeration of what type of conferences that will be fetched from Cisco TMS. The available types are: ▪ *All* ▪ *AllExceptDeleted* ▪ *Pending* ▪ *Ongoing* ▪ *PendingAndOngoing* ▪ *MeetingRequest* ▪ *Rejected* ▪ *Finished* ▪ *Deleted*

Returned data: An array with Conference objects.

GetConferenceIdByExternalId

This function is used to look up a conference that has been updated in the external source, and that must be updated in Cisco TMS. The **`ExternalSourceId`** and **`ExternalPrimaryKey`** fields must have been provided with the initial performance of the SaveConference function.

This function is typically used when information about a conference reserved in the external application is needed. First this function is call to get the corresponding conference is Cisco TMS. The GetConferenceById is used to get information about the conference from Cisco TMS.

Supported parameters:

`ExternalSourceId`	Unique identifier of the external source (server IP address).
`ExternalConferenceId`	Unique identifier of the conference within the external source (primary key in database).
`RecurrenceIdUTC`	Identifies an instance in a series of conferences. UTC-formatted datetime string.

Returned data: A ConferenceId, as defined in Cisco TMS.

GetRecurrentConferenceById

Returns a Conference object with the given ConferenceId. If the conference is a recurrent conference, existing exceptions to the recurrent series are returned in the RecurrencePattern Exceptions array of the returned Conference object.

Supported parameters:

`ConferenceId`	The ID of the conference, based on TMS IDs.

Returned data: A Conference object based on the ConferenceId.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

GetConferencesForSystems

This function returns all conferences for a list of systems between two dates. It is typically used to build a display of resource availability information in external application for a specific system when the external application does not store its own resource availability information.

The function should be used with caution. If a large number of conferences are booked between the two dates in Cisco TMS, it will take a long time to process the result of this function.

Supported parameters:

`SystemIds`	An array of IDs of the systems, based on Cisco TMS IDs.

`StartDate`	The start date of bookings
`EndDate`	The end date of bookings
`ConferenceStatus`	An enumeration of the type of conferences that will be fetched from Cisco TMS. The available types are: ■ *All* ■ *AllExceptDeleted* ■ *Pending* ■ *Ongoing* ■ *PendingAndOngoing* ■ *MeetingRequest* (Conference has been requested by a user without booking rights, needs approval.) ■ *Finished* ■ *Deleted*

Returned data: An array with Conference objects.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

GetDefaultConference

Creates a default conference object with ID equals -1 based on the conference settings specified in Cisco TMS.

This function is typically used as a basis for new meetings, where all that is needed is to define the start and end time, along with the participants in the conference.

Supported parameters: None

Returned data: A Conference object using the default values defined in Cisco TMS.

- The start time of the conference is set to the current time.
- The end time is set to the start time + **Default Scheduled Call Duration (in minutes)** as configured under **Administrative Tools > Configuration > Conference Settings**.

GetTransactionsSince

This function is used to get a list of conference creations, updated and deletions that must be performed in order to keep a mirrored conference database synchronized. The transaction identified as **CurrentTransactionId** will not be included in the array.

Supported parameters:

`CurrentTransactionId`	The transaction ID of the last committed transaction of the last synchronization.

Returned data: An array of transactions since the **CurrentTransactionId**.

If the provided ID does not exist, this will cause an error. See the section Error codes and error handling.

SaveConference

This function saves a conference in Cisco TMS.

- If **ConferenceId** is not set, a new conference is created and saved.
- If **ConferenceId** is set, the existing conference is updated. If the provided ID is incorrect, an error is caused. See the section Error codes and error handling.

The function will fail in both of the following scenarios:

- One or more of the participants are already booked in the same time period.
- A call route is to be made, but no call route could be found.

If this function is performed on a recurring conference, the entire series is affected.

Supported parameters:

`Conference`	The Conference object to be created/updated.

Returned data: A Conference object updated with actual values saved in Cisco TMS.

SaveConferenceRecInstance

Saves an instance of a recurring conference in Cisco TMS. Similar to SaveConference.

Supported parameters:

`Conference`	The Conference object to be created/updated

Returned data: A Conference object updated with actual values saved in Cisco TMS.

SaveConferences

Saves a list of conferences to Cisco TMS, with the option to save either all or none depending on availability information.

Use this function if the recurrence pattern of the Conference object does not support the recurrence model in the external application.

Supported parameters:

`Conference`	An array of conference objects.
`oneTransaction`	*True* if the objects should be booked as one transaction, meaning that either all or none of the meetings will be booked depending on the free/busy information. Currently only *true* is supported for this function.

Returned data: An array of Conference objects updated with actual values saved in Cisco TMS.

Error codes and error handling

In this chapter:

Error codes

Error	Code	Description
LICENSE	-2147219503	Client attempts to book more systems than there are licenses for.
DATABASE_DOWN	-2147217499	There is a problem with the Cisco TMS database.
MEETINGNOTFOUND	-2147208994	Client tries to access (get or update) a conference that does not exist.
SYSTEMNOTFOUND	-2147205791	Client tries to access (get or update) a system/participant that does not exist in Cisco TMS.
SYSTEM_ALREADY_BOOKED	-2147202587	Client tries to schedule a participant that has already been scheduled.
MEETINGNOTACTIVE	-2147189684	Client tries to end a conference that is not active.
MEETINGISDELETED	-2147186451	Client tries to end a conference that has been deleted.
MEETINGACTIVE	-2147183217	Client tries to modify the start time of a conference that is already active.
CANNOTBOOKINTHEPAST	-2147179982	Client tries to book a conference with an end time that is in the past.

If an exception is thrown, the exception message will contain the reason. If you get an Unspecified Exception/Unspecified Error, this usually means that there is a syntax flaw in the conference sent to the SaveConference function. In such a case, an error description would be given in the Cisco TMS log files (**http://<tms-server address>/tms/data/logs/tmsdebug/log-web.txt**)

Error handling example

If the Cisco TMS server is operational with the proper licenses, errors are caused by sending the wrong parameters to the API, such as trying to create a booking in the past, or trying to get systems, users, or conferences from Cisco TMS using the wrong ID. When an exception is caught, it is generally an indication that the client call must be changed before it is sent again.

Exceptions

All errors generated from the API are SoapExceptions, hence each time a save operation is performed against the API, the code should handle exceptions of type SoapException.

The message field of the exception will contain a string with a description of what went wrong. In many cases, displaying this information to the user will be helpful.

Code example

The code shows how to handle errors generated from API calls.

```
// Specify username and password to authenticate to service.
// (Can also be done in web.config)
NetworkCredential credentials = new NetworkCredential("xxx", "yyy", "ZZZ");

BookingService bookingService = new BookingService();
bookingService.Credentials = credentials;

// Set API to use version 9
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
bookingService.ExternalAPIVersionSoapHeaderValue = new
BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 9;

// Get a default conference object, where most common values are set
// (using default values specified in TMS)
Conference conference = bookingService.GetDefaultConference();

// Create an array of participants
Participant[] participants = new Participant[2];

// Create the elements of the array (the actual participants)
participants[0] = new Participant();
participants[0].ParticipantCallType = ParticipantType.IPVideo1; // Dial-out video
participants[0].NameOrNumber = "10.47.8.170";

participants[1] = new Participant();
participants[1].ParticipantCallType = ParticipantType.IPVideo1; // Dial-out video
participants[1].NameOrNumber = "10.47.8.171";

// Add the participants to the conference.
conference.Participants = participants;

// Set start date to December 12, 2012
DateTime startTime = new DateTime(2012, 12, 12, 10,00,00);
DateTime endTime = new DateTime(2012, 12, 12, 11,00,00);

// Save the conference, saving the returned conference (where all values are now
specified)
try
{
conference = bookingService.SaveConference(conference);
}
catch (SoapException e)
{
Console.WriteLine(e.Message);
}
```

Running this code will output the message: You cannot book a conference in the past.

HTTP error 401

The server will normally return the HTTP error code 500 Internal Server Error for the SoapExceptions.

If the HTTP error code 401 Unauthorized is received, the user credentials supplied were not authorized to access the server.

Code examples

This chapter provides examples of how to apply the remote setup API and the booking API in your development.

Code examples are provided in Java and C#.

In this chapter:

- Remote setup API example
- Booking with and without recording participant.

Remote setup API example

The code snippet below shows how to loop through all systems in Cisco TMS, and display information about each system.

```
// Specify username and password to authenticate to service.
// (Can also be done in web.config)
NetworkCredential credentials = new NetworkCredential("xxx", "yyy", "ZZZ");

RemoteSetupService remoteSetupService = new RemoteSetupService();
remoteSetupService.Credentials = credentials;

// Set API to use version 9
if (remoteSetupService.ExternalAPIVersionSoapHeaderValue == null)
remoteSetupService.ExternalAPIVersionSoapHeaderValue = new
RemoteSetupService.ExternalAPIVersionSoapHeader();
remoteSetupService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 9;

// Get all systems from TMS
TMSSystem[] tmsSystems = remoteSetupService.GetSystems();

// Loop through the systems and output information about each system
foreach (TMSSystem tmsSystem in tmsSystems)
{
Console.Out.WriteLine("SystemId: " + tmsSystem.SystemId + " System Name:" +
tmsSystem.SystemName);
}
```

Booking API example

When using the API as a web reference, the ParticipantsTypes for "IP Video", "ISDN Video" and so on are created as enumerations called **`IPTel`**, **`IPTel1`**, and so on. Values ending in 1 are dial-out, values not ending in 1 are dial-ins.

The code snippet below shows how to create a conference to two external participants (specified by IP address). A Cisco TelePresence MCU is required for this call to be saved.

```
// Specify username and password to authenticate to service.
// (Can also be done in web.config)
NetworkCredential credentials = new NetworkCredential("xxx", "yyy", "ZZZ");

BookingService bookingService = new BookingService();
bookingService.Credentials = credentials;

// Set API to use version 9
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
bookingService.ExternalAPIVersionSoapHeaderValue = new
BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 9;

// Get a default conference object, where most common values are set
// (using default values specified in Cisco TMS)
Conference conference = bookingService.GetDefaultConference();

// Create an array of participants
Participant[] participants = new Participant[2];

// Create the elements of the array (the actual participants)
participants[0] = new Participant();
participants[0].ParticipantCallType = ParticipantType.IPVideo1; // Dial-out video
participants[0].NameOrNumber = "10.47.8.170";

participants[1] = new Participant();
participants[1].ParticipantCallType = ParticipantType.IPVideo1; // Dial-out video
participants[1].NameOrNumber = "10.47.8.171";

// Add the participants to the conference.
conference.Participants = participants;

// Save the conference, saving the returned conference (where all values are now
specified)
conference = bookingService.SaveConference(conference);

// Output information about the conference.
Console.Out.WriteLine(conference.ConferenceInfoText);
Console.Out.WriteLine(conference.UserMessageText);
Console.Out.WriteLine(conference.ConferenceId);
```

Booking API example with a recording participant

The code snippet below show how to create a conference with two participants. One of the participants is a recording participant, the other a video system registered in TMS.

```
// Specify username and password to authenticate to service.
// (Can also be done in web.config)
NetworkCredential credentials = new NetworkCredential("xxx", "yyy", "ZZZ");
BookingService bookingService = new BookingService();
bookingService.Credentials = credentials;
// Set API to use version 9
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
bookingService.ExternalAPIVersionSoapHeaderValue = new
BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 9;
// Get a default conference object, where most common values are set
// (using default values specified in TMS)
Conference conference = bookingService.GetDefaultConference();
// Create an array of participants
Participant[] participants = new Participant[2];
// Create the elements of the array (the actual participants)
participants[0] = new Participant();
participants[0].ParticipantCallType = ParticipantType.IPVideo1; // Dial-out video
  participants[0].NameOrNumber = "10.47.8.170";
// get the recording aliases from the service
RecordingDevicd[] recordingDevicesWithAliases =
bookingService.GetRecordingAliases("");
Participant tcs = new Participant();
if (recordingDevicesWithAliases != null && recordingDevicesWithAliases.Count() > 0)
{
    // use the first recording device in the array
    var recordingAlias = recordingDevicesWithAliases.First();
    if (recordingAlias.Value != null && recordingAlias.Value.Count() > 0)
    {
// use the first alias found on the first recording device
AliasInfo aliasInfo = recordingAlias.Value.First();
tcs.ParticipantCallType = ParticipantType.TMS;
tcs.ParticipantId = aliasInfo.SystemId;
tcs.NameOrNumber = aliasInfo.AliasId;
    }
}
participants[1] = tcs;
conference.Participants = participants;
// Save the conference, saving the returned conference (where all values are now
specified)
conference = bookingService.SaveConference(conference);
// Output information about the conference.
Console.Out.WriteLine(conference.ConferenceInfoText);
Console.Out.WriteLine(conference.UserMessageText);
Console.Out.WriteLine(conference.ConferenceId);
```

Bibliography

All product documentation for the latest versions of Cisco TMS can be found at http://www.cisco.com/en/US/products/ps11338/tsd_products_support_series_home.html.

Title	Reference	Link
SOAP Version 1.2		http://www.w3.org/TR/soap12/
Web Services Description Language (WSDL) 1.1		http://www.w3.org/TR/wsdl
WSDL 1.1 Binding Extension for SOAP 1.2		http://www.w3.org/Submission/wsdl11soap12
Date and Time Formats		http://www.w3.org/TR/NOTE-datetime

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.