



# Cisco TelePresence Management Suite Extension Booking API

## Product Programming Reference Guide

---

Version 6.0

D13566.11

May 2011

---

# Contents

<b>Contents</b> .....	<b>2</b>
<b>Document Revision History</b> .....	<b>3</b>
<b>Introduction</b> .....	<b>4</b>
Document structure .....	4
Intended audience .....	4
Functional overview.....	4
Booking principles .....	4
<b>Entities in Cisco TMS</b> .....	<b>6</b>
System entity .....	6
Conference entity .....	6
User entity.....	6
<b>GUI pattern</b> .....	<b>7</b>
System information.....	7
Availability information.....	7
Booking management.....	7
<b>Synchronization pattern</b> .....	<b>8</b>
Importing.....	8
Synchronizing .....	8
Booking.....	8
<b>API reference</b> .....	<b>9</b>
API version .....	9
<b>Remote setup API reference</b> .....	<b>10</b>
TMSSystem object .....	10
TMS user object .....	11
Functions/Methods .....	12
<b>Booking API reference</b> .....	<b>15</b>
Conference object .....	15
Functions/Methods .....	21
<b>Code examples</b> .....	<b>25</b>
<b>Related documents</b> .....	<b>30</b>
<b>Disclaimers and notices</b> .....	<b>31</b>

## Document Revision History

- |        |  |
|--------|--|
| Rev 10 | November 2010. Cisco rebranding. Updated with Cisco template. Added new recording feature information (GetRecordingAliases, GetConferenceforUser)  |
| Rev 11 | March 2011. Added SaveConference function for new or existing conference series. New method GetRecurrentConferenceById. New SOAP exception error handling procedure for meetings created in the past |

# Introduction

The Cisco TelePresence Management Suite Extension Booking API (Cisco TMSBA) gives developers access to Cisco TelePresence Management Suite (Cisco TMS) booking functionality. This API is employed by Cisco TelePresence in its Microsoft Exchange and IBM Lotus Domino integrations, and provides the same feature set as the Cisco TMS Scheduler user interface.

## Document structure

This document includes the following parts:

- ▶ **Entities in Cisco TMS** - describes the objects and entities utilized by the Cisco TMSBA.
- ▶ **Usage patterns** - describes how an external GUI, booking database, or entire booking system can interact with the Cisco TMS booking system. The Microsoft Exchange and IBM Lotus Domino integrations work against the API in this manner.
- ▶ **API Reference** – describes the functions and objects available in the Cisco TMSBA.

## Intended audience

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 functionality/interoperability. Such booking systems will be referred to as external booking systems.

## Functional overview

There are four main features of the API:

- ▶ **Importing:** Importing and/or displaying Cisco TMS resources in an external application. See the [remote setup API section](#). This import process can be automated or initiated via a GUI.
- ▶ **Synchronizing:** Synchronizing resources booked in Cisco TMS with resources managed by an external booking system. (Information exchange: **TMS -> External Booking System**).
- ▶ **Booking:** Forwards booking requests made by an external booking system to TMS and reserve the resources there. (Information exchange: **External Booking System -> TMS**)
- ▶ **Availability:** Read and display bookings and reservations stored in the Cisco TMS reservation database.

It is not possible to forward booking requests made in Cisco TMS to an external booking system. The Cisco TMS database must control resource bookings for the API to function.

---

**Note:** The API cannot be used for system management, call management or other features beyond booking.

---

## Booking principles

By using the Cisco TMSBA, users can book video resources from their own booking application.

Systems can be read from Cisco TMS using the remote service API of the Cisco TMSBA to import and/or display them in the booking application.

One type of API usage is to make a front-end GUI utilizing the booking capabilities of Cisco TMS. By importing systems using the remote service API, the booking application can show availability information about systems in Cisco TMS. This lets users book conferences through the external application using the booking API.

Another scenario is a separate booking system with its own reservation database. When a system/resource is booked from this application, the external booking system contacts Cisco TMS to check availability via the Cisco TMSBA. If available, the system/resource is then booked in the third party application as well.

In some cases, the third party application will use a service to contact Cisco TMS that runs under a service account, so the user's credentials will not be used for booking the meeting. It is important that the service account has the necessary privileges to book a meeting on behalf of others, if ownership of the meeting is to be retained by the correct user. The remote setup API includes a method for creating such an account, and also verification mechanisms to make sure the required credentials are present during creation of the service account.

It is possible to list and book endpoints and rooms from the API, but Cisco TMS will allocate the needed network resources, e.g. if a user books 5 endpoints, this is sent to Cisco TMS. Cisco TMS will then determine if network resources like a MCU are required, and automatically reserve these resources.

To make sure that meetings booked in Cisco TMS are exported to the third party application, synchronization is used to update the database of the third party application from the Cisco TMS database. The synchronization mechanism should be run fairly often (e.g. every 10 minutes) to track the latest Cisco TMS bookings, and import them into the third party application. To reduce the delay of meeting imports from Cisco TMS to the third party application, the service account can be configured to send e-mail notifications every time a meeting is created/deleted or changed from Cisco TMS. When an e-mail notification is received, the third party application can immediately start synchronization, reducing the time it will take for the meeting to be imported.

# Entities in Cisco TMS

## System entity

A Cisco TMS System Entity is an entity used to describe an item that can be booked. In the Cisco TMS user interface, Cisco TMS system entities are seen as systems and rooms (e.g. the entities viewable in the System Navigator). Neither phone book entries nor web conference servers are systems.

In Cisco TMS each systems has a unique identifier or ID. This ID is visible in the user interface as of Cisco TMS 9.5. Cisco TMS allows a single system to be located in multiple folders, however the underlying system entity (and ID) will be equal for all instances of the system in Cisco TMS.

The table in the database that contains systems is the objSystem table. It is not recommended to update this table manually; however reading information does not cause issues.

## Conference entity

A Cisco TMS Conference Entity is an entity that describes a reservation in Cisco TMS (Conferences in Cisco TMS are also known as Bookings). All conferences in Cisco TMS must make at least one reservation of a Cisco TMS System Entity. For example, it is not possible to create a conference that contains only phone book entries. Cisco TMS will at the time the conference is saved add the required MCU reservations (the Cisco TMS System Entity) to allow the call to complete.

One limitation is when reserving web conference resources. Due to Web Conference resources not being Cisco TMS System Entities, it is not possible to only reserve a web conference; you will also need to include at least one Cisco TMS system entity reservation.

A conference is stored in the database in the ScheduledCall table. Each conference has a unique identifier (ID). The Cisco TMS System Entity reservation, dial-in slots, phone book entries etc. are stored in the ScheduledParticipant table. This table is coupled to the ScheduledCall table with the foreign key the ScheduledCall.Id from ScheduledParticipant.ScheduledCallId.

A conference ID can be seen in the Cisco TMS user interface under **Booking -> List Conferences -> Id column**.

## User entity

The Cisco TMS User Entity holds information about Cisco TMS users such as name, time zone, and e-mail.

## GUI pattern

The 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.
- ▶ **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.
- ▶ **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**)

### System information

Use the `GetSystems` or `GetSystemsForUser` function to get a list of available systems in Cisco TMS. This function returns a list of `TMSSystem` objects, which includes 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

Use the `GetConferences` to get all Cisco TMS reservations between two specified dates; `GetConferencesForSystem` is used to get system availability information; `GetRecurrentConferenceById` is used to get to get a conference, including any exceptions. This information can be used, For example, by an external application to display an availability calendar. If filtering of availability information is required, use the `Conference` object. `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

Use the `GetDefaultConference` functions to get `Conference` objects with Cisco TMS defined default values for `Conference` properties.

Use `GetConferenceById`, `GetConferenceIdByExternalId` or `GetRecurrentConferenceById` functions to retrieve already saved conferences.

To save changes to a conference, edit the properties on the `Conference` and use the 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.

To delete a conference use the `DeleteConferenceById` function. Conference participants will be disconnected if the conference is deleted while it is active or connected.

To add recording to a conference, use the `GetRecordingAliases` function to get information about a user's recording aliases and use this information to add recording participant(s) to the conference.

## Synchronization pattern

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

- ▶ **Importing:** Importing resources from Cisco TMS into an external application. The API can automate importing systems from Cisco TMS into the third party application, or this can be user-initiated via a GUI.
- ▶ **Synchronizing:** Synchronizing resources booked in Cisco TMS with resources available in an external booking system. An external system can keep track of booking transaction on the Cisco TMS server, and synchronize itself with booking made using Cisco TMS. (Information exchange: **TMS -> External Booking System**). This part is not applicable for external GUI front ends that do not have their own reservation database.
- ▶ **Booking:** Booking resource booked in an external booking system in Cisco TMS. This part of the API allows you to forward booking requests from an external booking system to Cisco TMS, and reserve the resources there. (Information exchange: **External Booking System -> TMS**)
- ▶ **Availability:** Reads bookings in Cisco TMS. The API allows you to display Cisco TMS reservations from Cisco TMS internal reservation database.

### Importing

Use the `GetSystems` or `GetSystemsForUser` function to get a list of available systems in Cisco TMS. This function returns a list 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 only return the systems the user has booking privileges for. If the external application controls system access, use `GetSystems` and filter the systems in the application.

### Synchronizing

Use the `GetTransactionsSince` function to get a list of transactions by the transaction ID (all conferences have a transaction ID property). The list of transaction contains the transaction type (New, Update, and Delete) and an associated `ConferenceId`. 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

Use the `GetDefaultConference` functions to get conference objects with Cisco TMS defined default values for conference properties.

Use `GetConferenceById`, `GetConferenceIdByExternalId` or `GetRecurrentConferenceById` functions to retrieve saved conferences.

To save changes to a conference, edit 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.

To delete a conference use the `DeleteConferenceById` function. Conference participants will be disconnected if the conference is deleted while it is active or connected.

## API reference

The Cisco TMSBA provides a Web Services API that interface 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.

The WSDL file for the Cisco TMS Remote setup API is located at:

<http://127.0.0.1/tms/external/booking/remotesetup/remotesetupservice.asmx>

The WSDL file for the Cisco TMSBA is located at:

<http://127.0.0.1/tms/external/Booking/BookingService.asmx>

---

**Note:** Exchange 127.0.0.1 with the name of the web-server Cisco TMS is installed on.

---

Microsoft Visual Studio .NET users can reference the API by selecting **Project -> Add Web Reference**, or enter the URLs above. For network load balancing, using the clusters virtual IP-address or DNS-name for this task is recommended. This also allows fail over for the API.

To use the Cisco TMSBA you will need one Application Integration License for each server using the API. Please contact your Cisco reseller/partner for more information.

To import from Cisco TMS or book meetings through the API, requires authentication with the API. To be able to book meetings using the API requires at least Misc Booking rights.

---

**Note:** 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), so you may need to allow for Basic Authentication on the /TMS/external/booking virtual directory (this can be done using the Internet Information Services manager). Anonymous authentication is not recommended. If you choose to do so, the IUSR\_<machinename> needs to be given Book on behalf of permissions in Cisco TMS.

---

## API version

The Cisco TMSBA has gone through several versions, and backwards compatibility was design priority. To utilize the full potential of the API, the API version needs to be specified in the headers when the functions of the API are called.

---

**Note:** Setting a number greater than the latest API version will break compatibilities when using revisions of the API.

---

## ExternalAPIVersionSoapHeader

Each call made to the Cisco TMSBA should include a 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.

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

# Remote setup API reference

## TMSSystem object

The TMSSystem object contains information about a system in Cisco TMS. This object is used to read information from Cisco TMS; 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, e.g. like the name of the system.

The XML below describes the TMSSystem object. Following the XML is a description of the elements and what information each element it contains.

```
<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>
```

---

Note that all fields are not required, so the output might contain less system information than the object can hold.

---

## TMSSystem

SystemId	The ID of the system in Cisco TMS. Use this to refer to the associated system in Cisco TMS from your application. Forexample, 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 like number of chairs in the meeting room the system is located.
SystemType	The type of system, e.g. Cisco TelePresence 1000MXP
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.
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 offset.

## SystemCategory

SystemCategory	An enumeration value of what category of a system this is. For example, endpoint.
----------------	---

## 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.
--------------	--

## 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>
  <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>

```

## User

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 if 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.

## Functions/Methods

### 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 method requires Cisco TMS Site Administrator privileges.

This function is typically used during uninstall procedures.

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

### 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 (as a hidden user not in normal user lists). The user is added to the Site Administrator Group. The user is configured to receive e-mail event scheduling notifications for all creation/update/deletions of bookings. This method is used during installation to create a separate user for the booking API.

The current user must be a Cisco TMS Site Administrator, along with being a local computer Administrator in order for the method to complete. The e-mail scheduling event notifications are typically used for updating the external booking system with changes done on the Cisco TMS server.

This function is typically used during install/setup procedures.

Input	
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.

Returns the user name of the created user (NT4 domain/username style).

## GetSystemByld

This function returns information about a specific system. If the system is not found [this causes an error](#).

Input	
TMSSystemId	System ID as given in Cisco TMS.

Returns a *TMSSystem* object.

## 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 set-up of resources in the external booking system to connect resources in Cisco TMS with resources in the external booking system.

Input	
None	

Returns 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 the 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.

Input	
None	

Returns an array of *TMSSystem* objects.

## IsAlive

This is used to check the connection to the web-services of Cisco TMS.

Typically used during installation to check the URL to this web-service.

Input	
None	

Returns a boolean value *True/False*. *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.

---

**Input**

None	
------	--

Returns an array of User objects.

**IsLocalAdmin**

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

This is typically used during installation to check if the user installing the integration has sufficient access to Active Directory. This method should return True in order for the GenerateConferenceAPIUser method to succeed.

---

**Input**

None	
------	--

Returns a boolean value True/False. True if the user is a local admin user.

**IsTMSServiceUser**

This function is used to check if 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 if the user installing the integration has enough access towards the Cisco TMS server.

---

**Input**

None	
------	--

Returns a boolean value True/False. True if user is a Cisco TMS service user.

**IsTMSSiteAdmin**

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

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

---

**Input**

None	
------	--

Returns a boolean value True/False. True if the user is a Cisco TMS Site Administrator.

# Booking API reference

## Conference object

Use this object to read and write conference properties like Start Time, End Time, Conference Title, Conference Password etc. Also, use this object for conference call related values like Bandwidth, Picture mode, Encryption mode etc.

All conference resources (video participants, audio participants, phone book participant, external participants etc.) are held in this object, together with the call route for connecting the resources.

You also define the conference type:

- ▶ Automatic call launch, which will connect the added participant at conference start time and disconnect them again at conference end time.
- ▶ Manual call launch, which asks the conference master participant to connect the call at conference start time.
- ▶ Reservation Only, which only reserves the participants for the conference duration.
- ▶ Conference data can be saved/updated, and handled by Cisco TMS using the SaveConference function described below.

The XML document below describes the Conference object. Following the XML is a description of the elements and what format input values require.

```

<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>
  </RecurrencePattern>
  <FirstOccurrenceRecInstanceIdUTC>string</FirstOccurrenceRecInstanceIdUTC>
  <PatternInstances>int</PatternInstances>
  <Exceptions>
    <RecurrenceException xsi:nil="true" />
  </Exceptions>
</Conference>

```

```

    <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</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</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 Telephone <- 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 Telephone <- 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>

```

## Conference

Attribute	Read/Write	Notes
Conferenceld	r/w - optional, if not specified -1 is assumed	Initially set to -1 to state to the saveconference method that the conference needs to be created. If set to a value greater than 0, the existing conference with the given ID is replaced with the conference specified.
Title	r/w - optional	If none specified the default name as defined in the Administrator Tools Page in Cisco TMS is used
StartTimeUTC	r/w - required	The start time of the conference in UTC format. See <a href="http://www.w3.org/TR/NOTE-datetime">http://www.w3.org/TR/NOTE-datetime</a> for more information. Only UTC times are supported (e.g. ending in a Z). Example: 1975-06-01T23:32:11Z.
EndTimeUTC	r/w - required	The end time of the conference in UTC format. See <a href="http://www.w3.org/TR/NOTE-datetime">http://www.w3.org/TR/NOTE-datetime</a> for more information. Only UTC times are supported (e.g. ending in a Z). Example: 1975-06-01T23:32:11Z.
RecurrenceInstanceldUTC	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. If the string contains 'deleted', it is a meeting that has been deleted from a series of recurring meetings.
FirstOccurrenceRecInstanceldUTC	r - only used when getting conference from Cisco TMS	Gives the start date of first instance of the meeting according to the recurrence pattern. If this is different from starttimeutc, the first meeting of the series is an exception to the recurrence pattern.
RecurrencePattern	r/w - optional	Sets the recurrence patterns for recurrent meetings. This is not valid if you call the 'saveconferencerecinstance' method.
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 acluser table of the Cisco TMS database.
OwnerUserName	w - optional	The username of the person booking the conference. This is used by Cisco TMS to lookup the ownerid 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/OwnerLastName/OwnerEmailAddress	w - optional	The first and last name of the owner of the conference. This is used by Cisco TMS to look-up the ownerid in the Cisco TMS database. If no owner is specified the user authenticated to the webservice is used. If ownerusername or ownerid is used instead of this field, those fields will be

		used instead of this field.
ConferenceType	r/w - optional, if not specified Default is assumed	<p>Can be set to one of the values:</p> <p><b>Reservation Only</b> – Cisco TMS reserves the resources, but will not set-up the call.</p> <p><b>Automatic Call Launch</b> – Cisco TMS will reserve the resources, and at the start time of the conference, connect the participant.</p> <p><b>Manual Call Launch</b> – Cisco TMS will reserve the resource, and wait for the VC Master (first TAA system in the Participant List) to select 'Connect'. These calls can also be connected using the Cisco TMS Conference Control Center interface.</p> <p><b>Default</b> – Use the conference type/reservation type that is defined in the Administrator Tools Page in Cisco TMS as the default type.</p>
Bandwidth (Discontinued)		This item is for backwards compatibility, and is no longer to be used. Use isdnbandwidth and ipbandwidth instead 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. Valid values are: Continuous Presence, Enhanced CP, Voice Switched and Default. If Default selected the default conference picture mode as defined in the Administrator Tools Page in Cisco TMS is used.
Encrypted	r/w - optional – if not specified, Default is assumed	The encryption mode for the conference. Valid values are: Yes, No, If Possible and Default. If Default is selected, the default encryption mode is defined in the Cisco TMS Administrator Tools Page.
DataConference	r/w - optional – if not specified, No is assumed	If data conference should be added to the conference. Valid values are: Yes, No and If Possible.
ShowExtendOption	r/w - optional, if not specified, Default is assumed	Set this value to allow the VC Master (the first TAA endpoint in the participant list) is to get a message to extend the conference, when the conference is close to ending. If Default is specified, the default Show Extend Option defined in the Administrator Tools Page in Cisco TMS is used.
Password	r/w - optional, if not specified, Cisco TMS may set a password if Cisco TMS is set to automatically generate passwords for new conferences	The password for the conference participants must enter to join the call.
BillingCode	r/w - optional, if not specified, blank is assumed	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.
ISDNRestrict	r/w - option, if not specified, No is assumed	If the ISDN channels should be restricted (e.g. Use 54k and not 64k)
ConferenceInfoText	r – only used when getting conference from Cisco TMS	Information on how the conference is to connect. Call Route, etc.
UserMessageText	r/w – optional – blank if not specified	A user definable text/description of the conference.
ExternalSourceId/ExternalPrimaryKey	r/w – optional – blank if not specified	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.

Participants	r/w, required	List of conference participants. When calling getdefaultconference, the participant list will be empty.
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 – if not specified, Default is assumed	The ISDN bandwidth of the conference
IPBandwidth	r/w - optional – if not specified, Default is assumed	The IP bandwidth of the conference

## Participant

Attribute	Read/Write	Notes
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, dial-in can be given the value 'Placeholder for John Doe', whereas 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=AreaCode, BN=Basenumber. 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: <b>TMS – A TMS System Entity.</b> When this is specified, the ParticipantId must be the Cisco TMS System Entity ID as given in Cisco TMS. <b>IP Video &lt;- or ISDN Video &lt;- - An IP/ISDN video dial-in.</b> If this is specified, you may give the participant a name using the NameOrNumber field. Cisco TMS will automatically give the participant and ID (less than 0) <b>IP Tel &lt;- or Telephone &lt;- - An IP/ISDN audio dial-in.</b> If this is specified, you may give the participant a name using the NameOrNumber field. Cisco TMS will automatically give the participant and ID (less than 0) <b>IP Video -&gt; or ISDN Video -&gt; – An IP/ISDN video dial-out site.</b> If this is specified, you must give Cisco TMS the number to use in the NameOrNumber field (Formats: ISDN: +1 (555) 1231234, H323 IP E.164: 12312321, H323 IP Address: 10.0.0.10). <b>IP Tel -&gt; or Telephone -&gt; – An IP/ISDN audio dial-out site.</b> If this is specified, you must give Cisco TMS the number to use in the NameOrNumber field (Formats: ISDN: +1 (555) 1231234, H323 IP E.164: 12312321, H323 IP Address:

		<p>10.0.0.10). Call will be placed using 64kbps/54kbps depending on restrict.</p> <p><b>TMS Master Participant</b> – Defines 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 TMS Master Participant per conference. A Cisco TMS System that will be the master of the conference.</p> <p><b>User – Not used/supported</b></p>
--	--	---

## RecurrencePattern

Attribute	Read/Write	Notes
FrequencyType	r/w required	The frequency of the recurrence rule. Legal values are: Daily, DailyWeekly, Weekly, Monthly, Yearly, Secondly, Minutely, Hourly, Default.
Interval	r/w required	Every X day/week/month as selected by FrequencyType
DaysOfWeek		<p>Days of week if FrequencyType is Weekly or the X mon/tue/weekend/day etc.</p> <ul style="list-style-type: none"> <li>• For every X mon-sun, only set the day.</li> <li>• For every day, set all days.</li> <li>• For every weekday, set mon-fri</li> <li>• For every weekend day, set sat &amp; sun</li> </ul>
FirstDayOfWeek		First day of week. Used to split DaysOfWeek[] in “every X week” weekly patterns. Default 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 month, -1 means last day of month. The allowed days must be defined in DaysOfWeek. 0 means monthly day of the meeting (every X day of a month).
PatternEndType		End type: by number of occurrences, by date, or never (not supported)
PatternEndDateUTC		In the case where PatternEndType is by date, this gives the end date of the recurrence pattern.
FirstOccurrenceRecInstanceUTC		Gives the original start time of the meeting of this occurrence. For example, if the occurrence is to start at 12am a day, and this particular instance of the recurring meeting is an exception (that the meeting time has been moved), this string gives the original start time of the meeting according to the recurrence pattern. If the meeting is not an exception to the recurrence pattern, this time will be the same as the start time of the meeting.
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. Supported using the GetRecurrentConferenceById and SaveConference functions. To get a conference with all its exceptions, use the GetRecurrentConferenceById. To update a conference with exceptions, use the SaveConference function. The exceptions should be provided in the RecurrencePattern.Exceptions array before saving the conference. As an alternative, use GetConferenceIdByExternalId with RecInstanceUTC (UTC string that points to the UTC day of the instance) to get conference id for the instance, and use

	SaveConferenceReclInstance to save this exception.
--	--

## ISDNBandwidth

Attribute	Read/Write	Notes
Bandwidth		The ISDN bandwidth will be used when dialing the conference participants, and also used when creating the conference. Note Max is not supported at this time. 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		Bandwidth: The IP bandwidth will be used when dialing the conference participants, and also used when creating the conference. Note Max is not supported at this time. 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.

## Functions/Methods

### DeleteConferenceById

Deletes a conference with the given Conferenceld (as defined in Cisco TMS). If the conference does not exist, [this causes an error](#). If the conference is part of a recurring series, the whole series will be deleted.

#### Input

Conferenceld	The Conferenceld of the conference to delete.
--------------	---

Returns nothing

### DeleteConferenceReclInstanceById

Deletes an occurrence of a recurring conference with the given Conferenceld (as defined in Cisco TMS). If the conference does not exist, [this causes an error](#). This function is typically used when deleting a single meeting in a recurring series.

#### Input

Conferenceld	The Conferenceld of the conference to delete.
--------------	---

Returns nothing

### EndConferenceById

Ends a conference with the given Conferenceld (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 method. If the conference is deleted or has not started yet, [this causes an error](#).

This function is typically used to end a running conference from a third party front-end GUI.

#### Input

Conferenceld	The Conferenceld of the conference to delete.
--------------	---

Returns nothing

### GetConferenceById

This function gets information about a particular conference. If the conference does not exist, [this causes an error](#).

**Input**

Conferenceld	The ID of the conference (Based on Cisco TMS IDs)
--------------	---

Returns a Conference object based on the Conferenceld.

**GetRecordingAliases****Input**

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

Returns 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.

**GetConferenceForUser**

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

**Input**

UserName	The TMS user to get bookings for. If no user name is provided (empty string), the logged in user is used
StartTime	The start date of bookings
EndTime	The end date of bookings
ConferenceStatus	An enumeration of what type of conferences that will be fetched from TMS. (All, AllExceptDeleted, Pending, Ongoing, Finished, PendingAndOngoingm MeetingRequest, Rejected, Finished or Deleted).

Returns an array with Conference objects.

**GetConferenceldByExternalId**

Returns a Conferenceld (as defined in Cisco TMS) given an ExternalSourceld and ExternalConferenceld. This function is used to look up conference that have been updated in the external source, and that must be updated in Cisco TMS. The ExternalSourceld and the ExternalPrimaryKey fields must have been provided in the initial SaveConference call.

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.

**Input**

ExternalSourceld	Unique identifier of the external source (i.e. server IP-address).
ExternalConferenceld	Unique identifier of the conference within the external source (e.g. primary key in database).

Returns a Conferenceld, as defined in Cisco TMS.

**GetRecurrentConferenceById**

Returns a Conference object with the given Conferenceld. If the conference does not exist, an exception is thrown. 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.

**Input**

Conferenceld	The Id of the conference (based on TMS Ids)
--------------	---

Returns a Conference object based on the Conferenceld.

## GetConferencesForSystem

This function returns all conferences for a list of systems between two dates.

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

This function 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.

Input	
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 what type of conferences that will be fetched from Cisco TMS. (All, AllExceptDeleted, Pending, Ongoing, Finished, PendingAndOngoingm MeetingRequest, Rejected, Finished or Deleted)

Returns an array with Conference objects.

## GetDefaultConference

Creates a default conference object 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.

Input	
None	

Returns a Conference object using the default values defined in Cisco TMS. The start time of the conference is set to the current time.

## GetTransactionsSince

Returns an Array of Transactions since the CurrentTransactionId.

This method 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.

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

Returns an Array of Transaction, giving the changes done since CurrentTransactionId

## SaveConference

Saves a conference in Cisco TMS. If conferenceld is not set, a new conference is created and saved. If the conferenceld is set, the existing conference is updated. If no conference with the given Conferenceld exists, [this causes an error](#).

This method will fail if any of the participants are already booked in the same time period or if a call route is to be made, but no call route could be found.

If this method is performed on a recurring conference, the complete series is affected.

Input	
Conference	The Conference object to be created/updated

Returns a Conference object updated with actual values saved in Cisco TMS.

---

**Note:** If an exception is thrown, you will be given a reason in the exception message. 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> as of TMS9.5 or `c:\tmsdebug\log-web.txt` for older versions)

---

### SaveConferenceReclInstance

Saves an instance of a recurring conference in Cisco TMS. Similar to SaveConference except that this is used to modify an occurrence of a series of recurring conferences.

This function is typically used when updating a single instance of a series of recurring conferences.

---

#### Input

Conference	The Conference object to be created/updated
------------	---

*Returns 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 method if the recurrence pattern of the Conference object does not support the recurrence model in the external application.

---

#### Input

Conference	An array of conference objects.
oneTransaction	True if they 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 method.

*Returns as array of Conference objects updated with actual values saved in Cisco TMS.*

## Code examples

### Visual Studio .NET using C# and Web-References

To use the Cisco TMSBA in Visual Studio .NET, you need to add a Web Reference to your project (**Project -> Add Web Reference**), specify the URL to your Cisco TMS server:

http://127.0.0.1/tms/external/Booking/BookingService.asmx for the booking API

---

**Note:** Exchange 127.0.0.1 with the name of the web-server Cisco TMS is installed on

---

For the remote setup API:

http://localhost/tms/external/booking/remotesetup/remotesetupservice.asmx.

You will be required to authenticate against the web-services to create the reference.

### 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 6
if (remoteSetupService.ExternalAPIVersionSoapHeaderValue == null)
    remoteSetupService.ExternalAPIVersionSoapHeaderValue = new
RemoteSetupService.ExternalAPIVersionSoapHeader();
remoteSetupService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 5;

// 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

---

**Note:** When using the API as a web-reference, the ParticipantsTypes for "IP Video <-", "ISDN Video ->" etc are created as enumerations called IPTel, IPTel1, etc. The values with an ending 1 are the dial-out, whereas without the ending 1 are dial-ins.

---

The code snippet below show 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 6
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
    bookingService.ExternalAPIVersionSoapHeaderValue = new
BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 5;

// 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;

// 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 shows 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 6
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
    bookingService.ExternalAPIVersionSoapHeaderValue = new
    BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 5;
// 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
RecordingDevice[] 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);

```

### Error handling example

The code show how to handle errors generated from API calls. If the Cisco TMS server is operational with the proper licenses, the errors are caused by sending wrong parameters to the API, like doing bookings in the past, or trying to get systems in Cisco TMS using the wrong ID.

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.

When an exception is caught, it is generally an indication that the client call must be changed before it is resent.

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

```
// 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 6
if (bookingService.ExternalAPIVersionSoapHeaderValue == null)
    bookingService.ExternalAPIVersionSoapHeaderValue = new
BookingService.ExternalAPIVersionSoapHeader();
bookingService.ExternalAPIVersionSoapHeaderValue.ClientVersionIn = 5;

// 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 the 12th of December, 2000
DateTime startTime = new DateTime(2000, 12, 12, 10,00,00);
DateTime endTime = new DateTime(2000, 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.

---

**Note:** The server will return a HTTP error with error code 500 for the SoapExceptions. If the HTTP error code 401 is received, this is an authorization/authentication error, and means that the user's credentials supplied is not authorized to access the server.

---

## Related documents

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

No.	Document reference or location	Name
1	<a href="#">D1356609</a> . Dated May 2009	TANDBERG Management Suite 3rd Party Booking API Version 5.0
2		
3		
4		
5		

## Disclaimers and notices

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

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

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

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

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

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

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