

# **TANDBERG Management Suite 3rd Party Booking API**

**Software version 9.x**

---

TANDBERG

D13566 Rev 02

# Table of Contents

<b>1</b>	<b>INTRODUCTION .....</b>	<b>2</b>
1.1	BASIC ENTITIES .....	2
1.1.1	<i>TMS System Entity</i> .....	2
1.1.2	<i>TMS Conference Entity</i> .....	3
<b>2</b>	<b>USAGE PATTERN.....</b>	<b>4</b>
2.1	SYNCHRONIZING .....	4
2.2	BOOKING .....	4
<b>3</b>	<b>API REFERENCE.....</b>	<b>5</b>
3.1	REFERENCE.....	5
3.1.1	<i>Conference Object</i> .....	5
3.1.2	<i>GetDefaultConference</i> .....	10
3.1.3	<i>SaveConference</i> .....	10
3.1.4	<i>DeleteConferenceById</i> .....	10
3.1.5	<i>GetDefaultConference</i> .....	10
3.1.6	<i>GetConferenceIdByExternalId</i> .....	10
3.1.7	<i>GetTransactionsSince</i> .....	11
3.1.8	<i>GetConferenceById</i> .....	11
<b>4</b>	<b>CODE EXAMPLES.....</b>	<b>12</b>
4.1	VISUAL STUDIO .NET USING C# AND WEB-REFERENCES .....	12
4.2	VBSSCRIPT/ASP USING MSSOAP.SOAPCLIENT .....	13

# 1 Introduction

The TANDBERG Management Suite (TMS) 3<sup>rd</sup> Party Booking API is an API that gives developers access to the booking functionality in TMS. The interface is used by TANDBERG in its Microsoft Exchange and IBM Lotus Domino implementations, and provides the same feature set as the user interface for the TANDBERG Scheduler.

The target audience for this document are developers that are required to implement a data/audio/video conferencing solution towards a booking interface not supported by TMS directly or where TMS does not provide the needed functionality, for example a company specific Enterprise Resource Planning system (ERP). Such booking systems will be referred to as external booking systems in the document.

There are two main usages of the API, which are often used in conjunction with one-another:

1. **Synchronizing:** Synchronizing resources booked in TMS with resources available in an external booking system. Using this part of the API allows an external system to keep track of booking transaction on the TMS server, and to synchronize itself with booking done through TMS. (Direction: TMS -> External Booking System).
2. **Booking:** Booking resource booked in an external booking system in TMS. Using this part of the API allows you to forward booking requests from an external booking system to TMS, and reserving the resources there. (Direction: External Booking System -> TMS)

The API requires that the master database for resource bookings is the TMS database. Therefore, it is not possible to forward booking requests made in TMS to an external booking system.

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

This document is divided into the following parts:

- Usage pattern – describes the way the Microsoft Exchange and IBM Lotus Domino integrations work against the API.
- API Reference – describes the functions and objects available in the 3<sup>rd</sup> Party Booking API.

## 1.1 Basic entities

### 1.1.1 TMS System Entity

A TMS System Entity is an entity used to describe an item that can be booked. In the TMS user interface TMS system entities are seen as systems and rooms (e.g. the entities viewable in the System Navigator). Phone Book Entries are not systems. Web Conference servers or instances are also not systems.

In TMS each systems has a unique identifier or id. This id is visible in the user interface as of TMS 9.5. It is important to note, that TMS allows a single system to be located in multiple folders, however the underling system entity (and id) will be equal for all instances of the system in 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 should not cause issues.

### **1.1.2 TMS Conference Entity**

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

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

Also note that the TMS System Entity reservation does not require to reserve the complete system, but can book parts of the system. These parts are known as Ports within TMS – and are often also referred to as Conferences (on the system. E.g. a TANDBERG 16+16 MCU has three *conferences*, and these can be reserved at the same time)

A conference is stored in the database in the ScheduledCall table. Each conference has a unique identified (id). The 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 TMS user interface under Booking -> All Meetings -> Id column.

## 2 Usage Pattern

This chapter will describe how the API is used in conjunction with the Microsoft Exchange integration component. It describes two main components:

1. **Synchronizing:** Synchronizing resources booked in TMS with resources available in an external booking system. Using this part of the API allows an external system to keep track of booking transaction on the TMS server, and to synchronize itself with booking done through TMS. (Direction: TMS -> External Booking System).
2. **Booking:** Booking resource booked in an external booking system in TMS. Using this part of the API allows you to forward booking requests from an external booking system to TMS, and reserving the resources there. (Direction: External Booking System -> TMS)

### 2.1 Synchronizing

Use the **GetTransactionsSince** function to get a list of transactions since a given 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 the **GetConferenceById** to get an updated Conference object – and update the conference within the external source.

The current transaction Id should then be updated to the last Conference's TransactionId.

### 2.2 Booking

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

Use **GetConferenceById** or **GetConferenceIdByExternalId** functions to retrieve already saved conferences.

To save changes to a conference, edit the properties on the Conference and use function **SaveConference**. This will save the conference to 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.

## 3 API Reference

The TMS 3<sup>rd</sup> Party Booking API provides a Web Services API towards the booking engine of TMS. Web Services allows for simple integration into most common languages and programming environments. See your development tool reference for information on how to build implementation stubs to help speed the development application that use Web Services.

The WDSL file for the TMS 3<sup>rd</sup> Party Booking API is located at:

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

If you are using Microsoft Visual Studio .Net, the simplest way to reference the API is to select Project -> Add Web Reference, then enter the URL above (exchanging 127.0.0.1 with the name of the web-server TMS is installed on). If you are using network load balancing, it is recommended to use the clusters virtual IP-address or DNS-name for this task, this will then also allow failover for the API.

To use the TMS 3<sup>rd</sup> Party Booking API you will need one Application Integration License per server using the API. Please contact TANDBERG for more information (<http://www.tandberg.net>)

To book meetings through the API, you must first authenticate yourself to the booking API. To be able to book meetings using the API, you need to at least have Misc Booking rights.

On a default TMS installation, the 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). It is not recommended to allow anonymous authentication – but if you choose to do so, the IUSR\_<machinename> needs to be given Misc Booking access in TMS.

### 3.1 Reference

#### 3.1.1 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 also held in this object, together with the call route for connecting the resources.

You also define the conference type in Conference;

- 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 ask the conference master participant to connect the call at conference start time.
- Reservation Only, which only reserves the added participants for the conference duration.

Conference data can be saved/updated, and handled by 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>
  <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</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</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</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</ParticipantCallType>
  </Participants>

```

```
</Participant>  
</Participants>  
</Conference>
```

#### Conference:

- ConferenceId (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 TMS is used)
- StartTimeUTC (r/w - required) - the start time of the conference in UTC format. See <http://www.w3.org/TR/NOTE-datetime> 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 <http://www.w3.org/TR/NOTE-datetime> for more information. Only UTC times are supported (e.g. ending in a Z). Example: 1975-06-01T23:32:11Z.
- OwnerId (r/w - optional) – the user Id of the owner of the conference. This is typically not set by external booking APIs, but by TMS. If no owner is specified the user authenticated to the WebService is used. IDs of users existing in TMS can be found in the aclUser table of the TMS database.
- OwnerUserName (w - optional) – the user name of the person booking the conference. This is used by TMS to look-up the OwnerId in the 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 TMS to look-up the OwnerId in the 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) – can be set to one of the values
  1. Reservation Only – TMS reserves the resources, but will not set-up the call.
  2. Automatic Call Launch – TMS will reserve the resources, and at the start time of the conference, connect the participant.
  3. Manual Call Launch – 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 TMS Conference Control Center interface.
  4. Default – Use the conference type/reservation type that is defined in the Administrator Tools Page in TMS as the default type.

- Bandwidth (r/w - optional – if not specified, Default is assumed) – The bandwidth that should be selected 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 bandwidth as defined in the Administrator Tools Page in TMS.
- 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 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 as defined in the Administrator Tools Page in TMS is assumed.
- 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 TMS is used.
- Password (r/w - optional, if not specified, TMS may set a password if TMS is set to automatically generate passwords for new conferences). The password for the conference participants must enter to actually join the call itself.
- BillingCode (r/w - optional, if not specified, blank is assumed). The billing code to use for the conference. If TMS requires billing codes, this field must be specified and will be validated against the list of billing codes in 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 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 TMS database with the external sources database. If TMS is given a value for these fields, 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.

Participant:

- ParticipantId – (r/w – optional) – For TMS System Entities, this value must be the SystemId of the system. For Phone Book Entries, this value must be the PhoneBookEntry Id from the gdrDirectoryEntry table in the TMS database. For external participants this value may be set, but is not required. If not set for external participants, TMS will create a 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. E.g. 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
  1. TMS – A TMS System Entity. When this is specified, the ParticipantId must be the TMS System Entity Id as given in TMS.
  2. 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. TMS will automatically give the participant and Id (less than 0)
  3. 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. TMS will automatically give the participant and Id (less than 0)
  4. IP Video -> or ISDN Video -> – An IP/ISDN video dial-out site. If this is specified, you must give 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).
  5. IP Tel -> or Telephone -> – An IP/ISDN audio dial-out site. If this is specified, you must give 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). Call will be placed using 64kbps/54kbps depending on restrict.
  6. Directory – A TMS Phone Book dial-out entry, where TMS should dial out to a given phone book entry. The ParticipantId must be set to the PhoneBookEntry Id as given from the gdrDirectoryEntry table in the TMS database.
  7. User – Not used/supported

### **3.1.2 GetDefaultConference**

Creates a default conference object based on the conference settings specified in 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 TMS. The start time of the conference is set to the current time.

### **3.1.3 SaveConference**

Saves a conference in TMS. If conferenceId is not set, a new conference is created and saved. If the conferenceId is set, the existing conference is updated. If no conference with the given ConferenceId exists, an exception is thrown.

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.

Input:

Conference - the Conference object to be created/updated

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

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 TMS-log files (<http://tmsrd/tms/data/logs/tmsdebug/log-web.txt> as of TMS9.5 or c:\tmsdebug\log-web.txt for older versions)

### **3.1.4 DeleteConferenceById**

Deletes a conference with the given ConferenceId (as defined in TMS). If the conference does not exist, an exception is thrown.

Input:

ConferenceId - the ConferenceId of the conference to delete.

### **3.1.5 GetDefaultConference**

Returns a default Conference object filled with default values as given in Administrative Tools -> Configuration -> Conference.

Returns a Conference object filled with the default settings for a conference.

### **3.1.6 GetConferenceIdByExternalId**

Returns a ConferenceId (as defined in TMS) given an ExternalSourceId and ExternalConferenceId. This function is used to lookup conference that have been updated in the external source, and that must be updated in TMS. The ExternalSourceId and the ExternalPrimaryKey fields must have been provided in the initial SaveConference call.

Input:

ExternalSourceId - Unique identifier of the external source (i.e. server IP-address).

ExternalConferenceId - Unique identifier of the conference within the external source (e.g. primary key in database).

Returns a ConferenceId, as defined in TMS. Can be used with e.g. GetConferenceById(...)

### **3.1.7 GetTransactionsSince**

Returns an Array of Transactions since the CurrentTransactionId. This method is used to get a list of conference creations, updated and deletes that must be performed in order to keep a mirrored conference database synchronized. The transaction with id 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

### **3.1.8 GetConferenceById**

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

Input:

ConferenceId - The Id of the conference (Based on TMS Ids)

## 4 Code examples

### 4.1 Visual Studio .Net using C# and Web-References

To use the 3<sup>rd</sup> Party Booking API in Visual Studio .Net, you need to add a Web Reference to your project (Project -> Add Web Reference), specify the URL to your TMS server: <http://127.0.0.1/tms/external/Booking/BookingService.asmx> - you will be required to authenticate against the web-service to create the reference. You will be asked to enter a name for the service, in the example below “TMSBooking” is used.

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). An 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;

// 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.IPVideol; // Dial-out video
participants[0].NameOrNumber = "10.47.8.170";

participants[1] = new Participant();
participants[1].ParticipantCallType = ParticipantType.IPVideol; // 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);
```

## 4.2 VBScript/ASP using MSSOAP.SoapClient

Note: Using the MSSOAP.SoapClient requires that you have a good understanding of XML documents and the MSXML2.DOMDocument objects. If you do not have a good understanding, it is highly recommended to use Microsoft Visual Studio .NET that allows you to add web-references to WebServices, and have objects automatically created.

The example below requires the APIs WSDL file to be located on c:\booking.wdsl, as the MSSOAP.SoapClient cannot access WSDL files via HTTP that require authentication. However the actual execution of commands requiring authentication is possible. This WSDL is located at: /tms/external/Booking/BookingService.asmx?WSDL

The example below shows how to create a conference with two dial-out participants. The sites are added as IP Video sites, dialling out using IP-addresses (10.0.0.45 and 10.0.0.55). This demo requires that you have an MCU available to be allowed to create the reservation.

```
<%
Option Explicit

Dim strServer
Dim strUserName
Dim strPassword

strServer = "tms-server"
strUserName = "domain\user" ' that has access to TMS server.
strPassword = "password"

Function MakeTextNode(objDocument, strElementName, strNodeText)
    Dim objElementNode
    Set objElementNode = objDocument.createElement(strElementName)

    objElementNode.appendChild(objDocument.createTextNode(strNodeText))

    Set MakeTextNode = objElementNode
End Function

dim objBookingService
Set objBookingService = Server.CreateObject("MSSOAP.SoapClient")

' Set Headers

objBookingService.ClientProperty("ServerHttpRequest") = True
' Load the WSDL locally, as the WSDL on the server needs a user name and password.
objBookingService.mssoapinit("c:\booking.wdsl")

' Set user name and password to access SOAP service.
objBookingService.ConnectorProperty("AuthUser") = strUserName
objBookingService.ConnectorProperty("AuthPassword") = strPassword

' Get a default conference.
Dim objConferenceNodeList
Set objConferenceNodeList = objBookingService.GetDefaultConference()

' Make an xml document out of the node list.
Dim objConferenceDOMDocument
Set objConferenceDOMDocument = Server.CreateObject("Msxml2.FreeThreadedDOMDocument")

Dim objConferenceNode
Set objConferenceNode = objConferenceDOMDocument.createElement("Conference")

objConferenceDOMDocument.appendChild(objConferenceNode)
```

```
Dim objNode
For Each objNode In objConferenceNodeList
    objConferenceNode.appendChild(objNode)
Next

' Add two participants to the conference
Dim objParticipantsNode
Set objParticipantsNode =
objConferenceNode.appendChild(objConferenceDOMDocument.createElement("Participants"))

Dim objParticipantNode
Set objParticipantNode =
objConferenceNode.appendChild(objConferenceDOMDocument.createElement("Participant"))

objParticipantNode.appendChild(MakeTextElementNode(objConferenceDOMDocument,
"ParticipantCallType", "IP Video ->"))
objParticipantNode.appendChild(MakeTextElementNode(objConferenceDOMDocument,
"NameOrNumber", "10.0.0.55"))

' Add the first IP Dial Out (->) participant
objParticipantsNode.appendChild(objParticipantNode)

Set objParticipantNode =
objConferenceNode.appendChild(objConferenceDOMDocument.createElement("Participant"))

objParticipantNode.appendChild(MakeTextElementNode(objConferenceDOMDocument,
"ParticipantCallType", "IP Video ->"))
objParticipantNode.appendChild(MakeTextElementNode(objConferenceDOMDocument,
"NameOrNumber", "10.0.0.45"))

' Add the second IP Dial Out (->) participant
objParticipantsNode.appendChild(objParticipantNode)

' Add the participants to the conference.
objConferenceNode.appendChild(objParticipantsNode)

' Save the conference, we get a node list of conference properties back, where
' participants may be resolved, missing settings have been set, etc.
Dim objNodeList
Set objNodeList = objConferenceDOMDocument.selectNodes("Conference/node()")

Set objConferenceNodeList = objBookingService.SaveConference(objNodeList)

%>
```