SW6000 User Manual

ESI External System Interface

User guide for the Shure SW6000 Conference Management System.

Version: 9.3 (2021)

Table of Contents

1 System applications and abbreviations 3

1.1 Introduction 3
1.2 Licensing 3
1.3 Abbreviations 3
1.4 CAA - Conference Administration Application 3
1.5 CUA - Conference User Application 3
1.6 CUI - Central Unit Interface 3
1.7 WSI - Web Service Interface 3
1.8 Web Service 3
1.9 Database 4

2 Web Service Interface (Live Data) 5

2.1 Introduction 5
2.2 CAA 5
2.2.1 CAA/Setup/Equipment/Web service 5
2.2.1.1 Third party streaming provider 6
2.3 CUA 6
2.3.1 Starting and Stopping streaming and archiving 6
2.4 CUI 7
2.4.1 Communication Status display 7
2.5 Solution overview and requirements 8
2.6 Initialization/restart/recovering 8
2.6.1 Web service methods 8
2.6.1.1 PrepareMeeting 9
2.6.1.2 MeetingStart and MeetingStop 11
2.6.1.3 SpeakerChange 12
2.6.1.4 VotingStart 13
2.6.1.5 VotingStop 14
2.6.1.6 VotingHide 16
2.6.1.7 IndividualVotingResult 16
2.6.1.8 StatusUpdateStart and StatusUpdateDone 18
2.6.1.9 StreamingStart 18
2.6.1.10 StreamingStop 18
2.6.1.11 AgendaUpdate 19
2.6.1.12 DelegateDetails 20
2.6.1.13 DelegateLogin 21
2.6.1.14 DelegateLogout 22
2.6.1.15 AlertStart 22
2.6.1.16 AlertStop 23
2.6.1.17 WebServiceStatus 23
2.6.1.18 RequestDisplayLanguage 24
2.6.1.19 RequestStreamingSupport 24
2.6.1.20 RequestVoteHideDelay 24
2.6.1.21 VerifyPassword 24
2.6.2 Handling Agenda changes 24
2.6.3 Handling Status request 25
2.6.4 Handling web service and SW6000 reboot/restart/breakdown 25
2.6.4.1 Starting up Web Service Interface (CUI) while Web Service is running 26
2.6.5 Handling Streaming Start/Stop 27

3 Meeting Import/Export 28

3.1 Introduction 28
3.2 CAA 28
3.2.1 CAA/Setup/Equipment/Meeting Import/Export 28
3.2.1.1 Meeting Import/Export Mode 28
3.2.1.2 Web Publish Mode 29
3.3 File Import/Export 30
3.3.1 Data format 30
3.3.1.1 Conference (Meeting) 31
3.3.1.2 Delegate (Participants) 31
3.3.1.3 Group 32
3.3.1.4 Agenda 32
3.3.1.5 VotingSession 32
3.3.1.6 VotingResult 34
3.3.2 File format XSD 34
3.3.3 Example XML file 37
3.4 Web Service 38
3.4.1 Web Service Functions 38
3.4.1.1 GetConferenceList 38
3.4.1.2 GetConference 40
3.4.1.3 SetConference 42

1 System applications and abbreviations

1.1 Introduction

The purpose of this document is to describe the requirements and information the SW6000 system offers to communicates with third part systems.

1.2 Licensing

The SW6000-ESI External System Interface is an add-on to the SW6000 Software and is licensed separately.

1.3 Abbreviations

- CAA – Conference Administration Application
- CUA – Conference User Application
- CUI – Central Unit Interface
- WSI - Web Service Interface – Interface implemented on the SW6000 system
- WS - Web Service – Web Service implemented on the server of the Streaming provider
- Database – SQL Server
- System environment: COM+ application and MSMQ

1.4 CAA - Conference Administration Application

The CAA is an application designed to organize and configure meetings. It's used by administrators prior to and during an event for creating and maintaining basis meeting data such as agendas, participant information, participant lists and voting settings.

1.5 CUA - Conference User Application

Used by Chairpersons, delegates, or other persons attending a meeting for viewing the current agenda, participant information and for starting/stopping meetings and managing microphones and speakers.

1.6 CUI - Central Unit Interface

The CUI application is the Central Unit Interface. As the name indicates it is managing the communication with the CCU (Central Unit).

The CUI is also interfacing with all client applications (CUA's, CAA's, CDA's and ECA), The CDA's and ECA are applications used for information sharing and outside the scope of this document.

The client applications will typically receive messages from the CCU through the CUI. And finally the CUI is used by client applications to broadcast messages to all other client applications.

1.7 WSI - Web Service Interface

The SW6000 has an interface through which it communicates with the web service that is implemented on the streaming provider server.

1.8 Web Service

Web services that through methods called by SW6000, communicates with the streaming solution.

1.9 Database

The SW6000 database includes:

- Central tables and columns
- Handles multiple languages
- ‘Enumerated types’

2 Web Service Interface (Live Data)

2.1 Introduction

The purpose of this chapter is to describe the requirements and information flow for the interface between a SW6000 system and a 3rd party web service that communicates with a Web Cast or Web Streaming system.

The web service interface delivers 'dynamic' live data during the meeting to a third party system like microphone events, agenda events or voting events and the data is delivered upon happening.

The web service interface cannot be used to import data to SW6000, or being used before or after the meeting for exchanging data with third part systems. For this purpose the 'Meeting Import/Export' facility in SW6000 must be used.

2.2 CAA

2.2.1 CAA/Setup/Equipment/Web service

Using the CAA application, the following parameters shall be setup for using Web Service calls:

- Web service address
- Web service administrator address
- Passwords for connection between WSI and WS
- Checkboxes to select, which additional information to be send to the web service. This is also used for making the present version of the web service compatibility with previous version of SW6000.

When checked the information selected is sent to the Web Service. Only information on active participants is sent.

The WSI will send a password specified in the CAA for verification in the WS.
The WS will send a password for verification in the WSI. A verification password is to be specified in the CAA.

Multiple web services can be specified.

This screen is used to setup the connection parameters for the Web Service Interface:

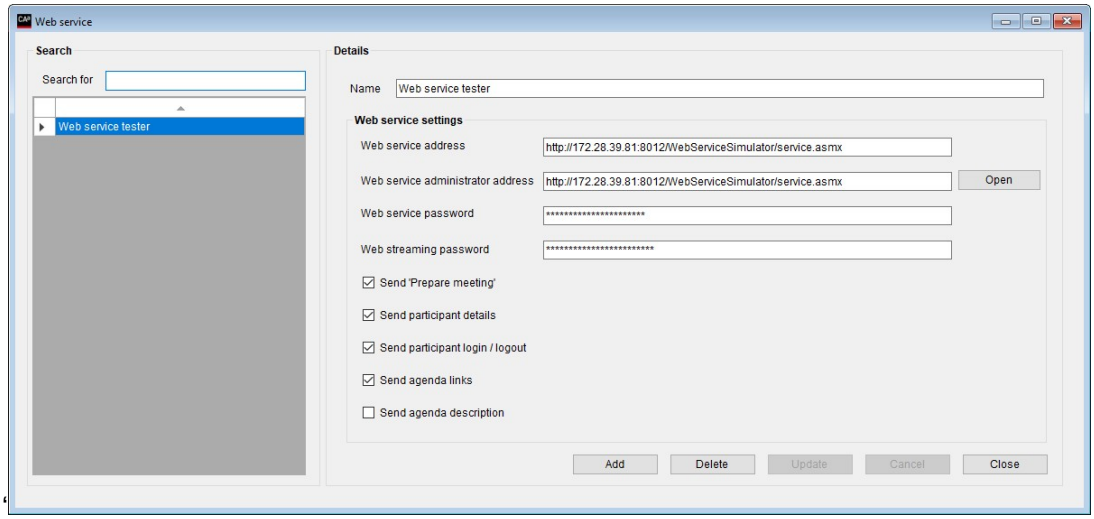


Figure 2.2-A

Web service address	Field for entering the web service address
Web service administrator address	Field for entering the web service administrator address for a third party streaming provider for configuring/controlling.
Web service password	Field for entering password for the web service interface to be verified by the web service
Web streaming password	Field for entering password for the web service to be verified by the web service interface
Send prepare meeting	Tick this option to send the prepare meeting information
Send participant details	Tick this option if Participant details (text and picture) shall be sent to the web service
Send participant login/logout	Tick this option if Participant login/logout information shall be sent to the web service
Send agenda links	Tick this option if agenda links shall be sent to the web service
Send agenda description	Tick this option if agenda description shall be sent to the web service
Open	Selecting open will open a configuration page if the web service is used together with a third party streaming provider.

2.2.1.1 Third party streaming provider

The webpage should be a minor administration interface that can be managed from the CAA and a status check showing what's going on. In this way the user doesn't have to run around between several computers to view streaming status, except for trouble shooting.

Following information may be implemented on the webpage:

1. Start/stop broadcast (streaming)
2. Start/stop archiving (this setting can be set in the administration interface in the Streaming software so the recording will start automatically when using start Broadcast)
3. Encoder status – showing the status of the encoder
4. Broadcast check (showing the output media stream so it is shown in a player in the CAA. This shows the user that something is going out from the encoder)
5. Edit and publish on demand
   Changing streaming mode

Changing between Broadcast/Archiving/BroadcastArchiving and displaying current setting may be possible in the Web Service Configuration screen,

Return values for selected streaming mode:

Broadcast	Return value:	Broadcasting
Archiving	Return value:	Archiving
BroadcastArchiving	Return value:	BroadcastArchiving

2.3 CUA

2.3.1 Starting and Stopping streaming and archiving

If the web service is used together with a third party streaming provider, Streaming Control functionality in the CUA is possible. Availability of the control buttons is included in the Meeting Role settings in the CAA.

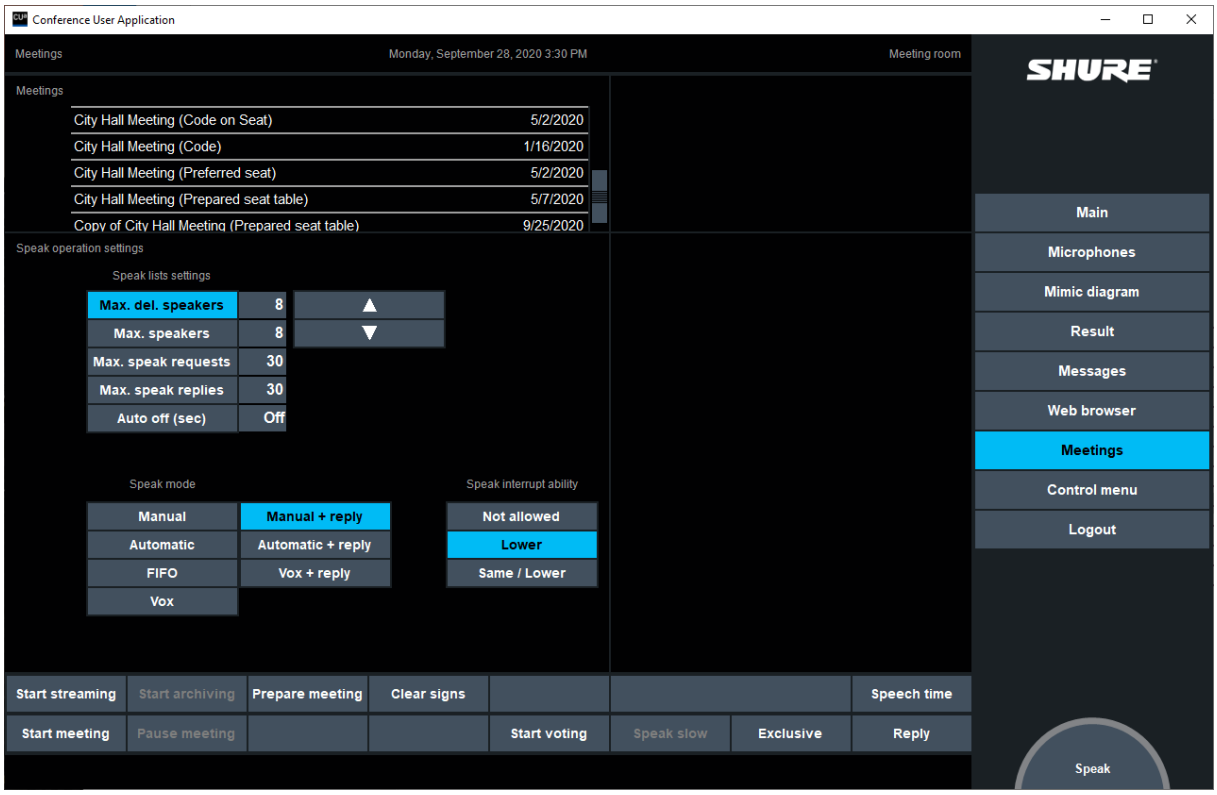


Figure 2.3-A

2 buttons are implemented. One for Starting the Broadcasting/Streaming and one for starting the Archiving/Recording.

Only buttons where the feature is supported by the Streaming solution (answered by the RequestStreamingSupport) is visible.

Buttons will change color from default to red when streaming/archiving is running. Button only enabled if Web Service is available else locked.

The following restrictions apply:

- Archiving is only possible when streaming (broadcast) is active.
- Archiving is only possible when a meeting is started.

Additionally if the user selects to stop a meeting (from CAA or CUA), then a stop archiving message is sent over the web service interface.

[Start streaming] Press this button to start/stop streaming of the active meeting. The button will blink red while the system activates the streaming and turn to [Stop Streaming] and remain red if the streaming is activated. The button is only available if the Web Service State in the CUI is ready and an external streaming solution is present.

[Start archiving] Press this button to start/stop archiving of the streamed meeting. The archiving can only be activated when the streaming has been activated. The button will blink red while the system activates the archiving and turn to [Stop Archiving] and remain red if the archiving is activated. The button is only available if the Web Service State in the CUI is 'Ready' and an external streaming solution is present.

2.4 CUI

2.4.1 Communication Status display

For debugging purpose, the CUI display contains a window showing the feedback from the WSI.

When sending pictures using the DelegateDetails method the picture is included in the XML format. The picture data is not send back to the CUI window, but is replaced with a <<picture>> tag instead.

2.5 Solution overview and requirements

The SW6000 system can deliver data to a streaming solution. The interface between those two systems is implemented using a web service interface, through a web service running on the streaming solution server.

CUI broadcast system information originating at the CU/CCU, or a client CUA/CAA – with data taken from the SW6000 database.

The WSI then posts the data to the Web Service (WS) on the Streaming solution server. The WS delivers the data sent through the WS to the Streaming solution.

The Video Streaming Solution may deliver a controllable web page, from where it's possible to configure the video streaming application. This webpage is embedded in the CAA application for access by an SW6000 administrator.

Handshaking and communication overview is handled between the WSI and WS.

The communication status between the WSI and WS is designed in a way so that minimal impact on the performance of the CUI is achieved.

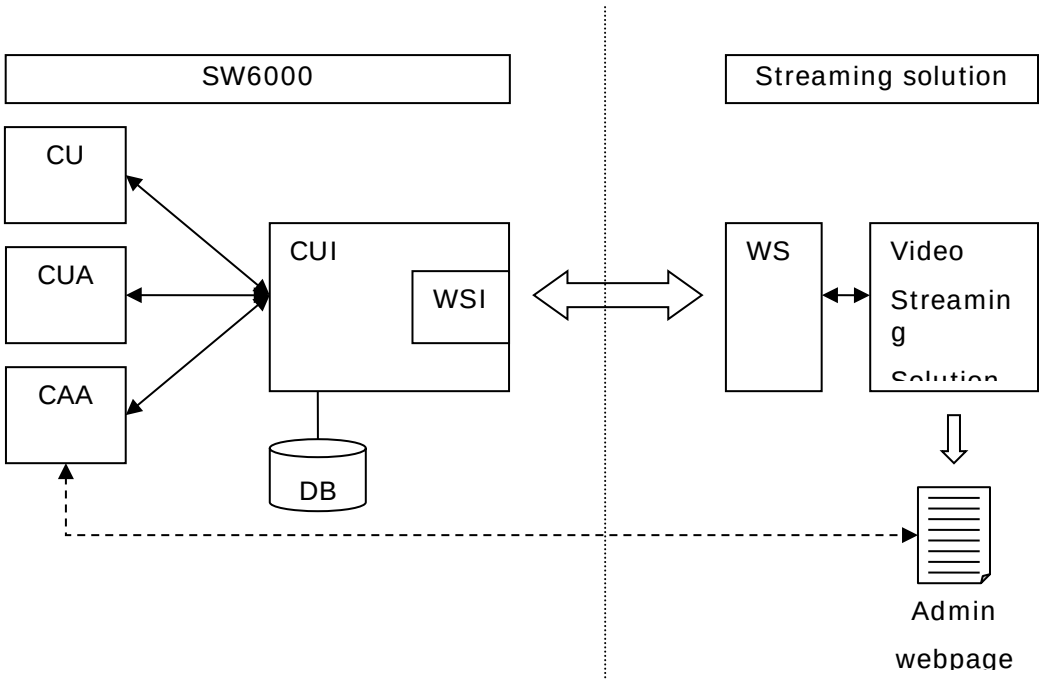


Figure 2.5-A

2.6 Initialization/restart/recovering

The WSI is responsible for keeping the WS updated with correct status of the SW6000 system. During normal operation changes to SW6000 status are transmitted through WS calls.

Communication from both WS and WSI is monitored by the other party in order to detect a failure situation – if the WS stops responding for some reason this will be evident to the WSI because of missing replies on monitoring calls. If a failure occurs – e.g. because of a restart of one of the computers it is ensured that the systems are able to resynchronize full status by going through an initialization sequence.Communication flow

2.6.1 Web service methods

The web service must support several events from the WSI.

When an event occurs on the SW6000 system, the WSI calls the appropriate Web Service method.

Methods to be implemented in the WS are described in the following subchapters.

Most Methods has xml strings as input parameters. These xml strings are output from datasets on the WSI and it is therefore very useful to read these xml strings into datasets in the WS using a standard xml parser for further data handling.

2.6.1.1 PrepareMeeting

Input parameters: ***strMeetingXML*** *[STRING]*

XML format:

```
<newdataset>
   <meeting>
      <system_id>Mødelokale 1<system_id>
      <password>Password1234<password>
      <title>Copenhagen Climate Conference 2009</title>
      <starttime>2009-11-07T12:09:25.7810767+01:00</starttime>
      <description>Discussions about meeting schedule</description>
      <public>1</public>
      </publishing>
      <searchtags>UN, United Nations, copenhagen climate meeting,
        UNFCCC, Subsidiary Body for Implementation, SBI, SBSTA,
        climate, kyoto</searchtags>
      <categories>Ungdomstinget</categories>
      <category_ids>5</category_ids>
      <web_id>1323</web_id>
   </meeting>
   <timestamp>
      <eventtime>2009-10-27T12:09:25.7810767+01:00</eventtime>
   </timestamp>
</newdataset>
```

Parameter description:

system_id	A unique string for identifying the system the meeting is being held on. I.e. this identifies the meeting room to the WebService making it possible for the WebService to identify multiple meeting rooms using the same WebService.
password	A password string used for identification with the system_id. This is the same password as is sent to the WebService in the VerifyPassword WebService call.
title	Title of the meeting
starttime	Date and time of planned meeting start. Will contain a date without a time part if no start time has been specified for the meeting.
description	Description of the meeting
public	Is meeting to be displayed publicly online (1) or is it private (0).
publishing	Publishing method for meeting recordings. Currently not used.
searchtags	Search keywords to be tagged on meeting recordings.
categories	Names of categories under which to publish meeting recordings. Category names are separated by a semicolon if multiple categories are present.

To avoid illegal category separation by user-typed semicolons in categories the following character replacements are performed on the individual category names:

: (colon) => :. (colon period)
; (semicolon) => :, (colon comma)

For example the following two category names:
- Salen: Evaluering
- Belysning; En diskussion

would be encoded as:
<categories>Salen:. Evaluering;Belysning:, En diskussion</categories>

The reverse replacements must be made in the receiving end.

category_ids	String of unique identifier integers for the categories named in the 'categories' tag. One ID must exist for each category named in the 'categories' tag. IDs are separated by a semicolon. The ID for a category is what defines the category – the name is just a humanly readable name related to that ID which is to be used when presented to a user (e.g. if presenting meetings by category in an external application). If an ID is specified and the accompanying category name is different from what has been previously sent for that category ID, the receiving system should update its name for the category to the new name.
web_id	Unique identification of the meeting in the Web Service. When creating a new meeting this field is not specified (not present or empty), but if updating data on a meeting already prepared once, this is the ID returned by the PrepareMeeting() WebService function when the meeting was initially prepared.
eventtime	When the event occurred

Return values:

A complex value indicating the result of preparing the meeting in the external system

```
<result>
 <status>Success</status>
 <web_id>3214</web_id>
 <message>Some error message – if applicable</message>
</result>
```

result	Container tag
status	Status of preparing the meeting. One of the following • Success. The meeting has been prepared in the external system. • Failure. The meeting could not be prepared in the external system. The <message> tag contains an error description. • Duplicate. The meeting has already been prepared. Meeting data has been updated to reflect the new data, so this is also a success indicator.

web_id	The ID of the meeting in the external system. This is what you'll pass as 'web_id' argument to calls which require a web_id argument.
message	An error message. Only valid when <status> is Failure.

Description: Called when a new meeting has been added to make the WebService aware of it before the meeting is referenced in other webservice calls. May also used to update meeting data for a meeting which the webservice has previously been made aware of – in this case a web_id field is contained in the data.

2.6.1.2 MeetingStart and MeetingStop

Input parameters: strMeetingTitleXML [STRING]

XML format:

```
<newdataset>
   <meeting>
      <title>Copenhagen Climate Conference 2009</title>
      <import_id>6066</import_id>
      <web_id>1323</web_id>
      <categories>Ungdomstinget</categories>
      <category_ids>5</category_ids>
   </meeting>
   <timestamp>
      <eventtime>2009-10-27T12:09:25.7810767+01:00</eventtime>
   </timestamp>
</newdataset>
```

Parameter description:

title	Title of the meeting
import_id	Unique number identifying an imported meeting
web_id	Unique identification of the meeting in the WebService. This is the ID returned by the PrepareMeeting() WebService function.
categories	Names of categories assigned to the meeting. See PrepareMeeting call (section 2.7.1.1) for more information on this tag.
category_ids	String of unique identifier integers for the categories named in the 'categories' tag. One ID will exist for each category named in the 'categories' tag. See PrepareMeeting call (section 2.7.1.1) for more information on this tag.
eventtime	When the event occurred

Return values: empty string [STRING]

Description: Called when meeting starts or stops. Note the dataset may be empty for the MeetingStop call – this is the case when MeetingStop is called as part of the startup sequence and no meeting is active. Note the MeetingStart and MeetingStop are always followed by an AgendaUpdate - if the MeetingStart event is called and the meeting has an Agenda the AgendaUpdate call will contain the meeting agenda, if meeting has no agenda or MeetingStop is called the subsequent AgendaUpdate call will contain an empty agenda.

2.6.1.3 SpeakerChange

Input parameters: strSpeakerListXML [STRING]

XML format:

```
<newdataset>
  <seat>
     <title>Minister</title>
     <name>Anne Baastrup</name>
     <usertype>Delegate</usertype>
     <usertable1 />
     <usertable2 />
     <usertable3 />
     <usertable4 />
     <group>Socialist Group</group>
     <groupabbreviation>SG</groupabbreviation>
     <votegroup />
     <seatnumber>5</seatnumber>
     <delegate_id>73</delegate_id>
     <import_id>100589</import_id>
  </seat>
   .
   .
   .
  <seat>
     <title />
     <name>Anita Knakkergaard</name>
     <usertype>Delegate</usertype>
     <usertable1 />
     <usertable2 />
     <usertable3 />
     <usertable4 />
     <group>Danish Peoples Group</group>
     <groupabbreviation>DPG</groupabbreviation>
     <votegroup />
     <seatnumber>4</seatnumber>
     <delegate_id>132</delegate_id>
     <import_id>100576</import_id>
  </seat>
  <timestamp>
     <eventtime>2009-10-27T13:52:54.1023068+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

title	Title of the participant
name	Display name of the participant.
usertype	privileges (meeting role) of the participant (as setup in CAA)
usertable1	as setup in CAA
usertable2	as setup in CAA
usertable3	as setup in CAA
usertable4	as setup in CAA
group	as setup in CAA
groupabbreviation	as setup in CAA
votegroup	as setup in CAA
seatnumber	identifying microphone position

	delegate_id	unique number identifying participant [note two participants may have identical names]
	import_id	unique number identifying participant imported (TingDok Id)
	eventtime	When the event occurred

Return values: empty string [STRING]

Description: Called when speaker changes
XML will contain Seat information for all current speakers. XML will only contain eventtime if there are no speakers.

2.6.1.4 VotingStart

Input parameters: strSubjectNameXML [STRING]

XML format:

```
<newdataset>
  <currentsubject>
     <level>1</level>
     <name>Global Warming</name>
     <description />
     <shortdescription />
     <agenda_id>2653</agenda_id>
     <import_id>32332</import_id>
     <itemnumber>1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>2</level>
     <name>Ice Caps</name>
     <description />
     <shortdescription />
     <agenda_id>2654</agenda_id>
     <import_id />
     <itemnumber>1.1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>3</level>
     <name>Rising Water Levels</name>
     <description />
     <shortdescription />
     <agenda_id>2655</agenda_id>
     <import_id />
     <itemnumber />
  </currentsubject>
  <timestamp>
     <eventtime>2009-10-27T14:07:44.0956233+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

level	Identifies the hierarchy of subject/parent subjects. Level 1 is main, and 2.. are sub items.
name	Name of the current subject as well as name of parent subjects to the current subject
description	agenda item description
shortdescription	agenda item short text description
agenda_id	Unique number for agenda item – SW6000 autogenerated
Import_id	Unique number identifying participant imported (TingDok Id)
itemnumber	Item Number text as input in agenda from CAA.

	Eventtime	When the event occurred

Return values: empty string [STRING]

Description:

Called when voting starts. Part of the system update procedure. In a WS restart sequence, it is called if a voting session is in progress.

If voting session is started without any subject selected, the dataset will be empty.

2.6.1.5 VotingStop

Input parameters: strVotingResultsXML [STRING]

XML format (Complete content of Session dataset):

```
<newdataset>
  <session>
     <session_id>762</session_id>
     <version>1</version>
     <date>2009-10-28T08:46:50.6205579+01:00</date>
     <passedresult>1</passedresult>
     <passedresulttext>Passed</passedresulttext>
     <result1>1</result1>
     <result2>0</result2>
     <result3>0</result3>
     <result4>0</result4>
     <result5>1</result5>
     <result6>0</result6>
     <result7>0</result7>
     <result8>0</result8>
     <result9>0</result9>
     <result1label>Yes</result1label>
     <result2label>No</result2label>
     <result3label>Abstain</result3label>
     <result4label />
     <result5label>Total</result5label>
     <result6label />
     <result7label />
     <result8label />
     <result9label />
     <conclusion>’Conclusion text    ‘</conclusion>
  </session>
  <timestamp>
     <eventtime>2009-10-28T09:45:51.2541477+01:00</eventtime>
  </timestamp>
  <currentsubject>
     <level>1</level>
     <name>Global Warming</name>
     <description />
     <shortdescription />
     <agenda_id>2653</agenda_id>
     <import_id />
     <itemnumber>1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>2</level>
     <name>Ice Caps</name>
     <description />
     <shortdescription />
     <agenda_id>2654</agenda_id>
     <import_id />
     <itemnumber>1.1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>3</level>
     <name>Rising Water Levels</name>
     <description />
     <shortdescription />
     <agenda_id>2655</agenda_id>
     <import_id />
     <itemnumber />
  </currentsubject>
</newdataset>
```

Parameter description:

session_id	Internal identifier for voting session

version	Sequence number identifying session – if revotes are held on a subject the sequence number will be greater than 1 on revotes
date	Voting date
passedresult	1: Vote Passed, 0: Vote Not Passed
passedresulttext	Text to display for vote result
result1 to result9	Calculated results of voting session.
result1label to result9label	Labels associated with calculated voting results.
conclusion	Text overview of voting result
eventime	When the event occurred
level	Identifies the hierarchy of subject/parent subjects. Level 1 is main, and 2.. are sub items.
name	Name of the current subject as well as name of parent subjects to the current subject
description	agenda item description
shortdescription	agenda item short text description
agenda_id	Unique number for agenda item – SW6000 autogenerated
Import_id	Unique number identifying participant imported (TingDok Id)
itemnumber	Item Number text as input in agenda from CAA.

Return values: empty string [STRING]

Description: Called when voting stops or the voting session is cancelled. VotingStop is part of the system update procedure. If WS fails, VotingStop is called when connection is re-established if no voting session is in progress and the current subject has a voting result attached with the corresponding voting result. On startup or WS connection re-established when currently no meeting is active or a meeting is active but no subject is current or a subject is current with no voting result the VotingResult dataset will contain empty tags.

When a voting session completes VotingStop is called with VotingResult dataset containing the result of the voting session. If a voting session is cancelled, VotingStop will be called with VotingResult dataset containing all empty tags.

2.6.1.6 VotingHide

Input parameters: No input

Return values: empty string [STRING]

Description: Called at a specified delay after voting stop calls. To be used by WebService to remove the voting results Webpage from being the active display. The delay interval is retrieved from the database.

2.6.1.7 IndividualVotingResult

Input parameters: strIndividualVotingResultXML[STRING]

XML format (Complete content of Session dataset):

```
<newdataset>
  <currentsubject>
     <level>1</level>
     <name>Global Warming</name>
     <description />
     <shortdescription />
     <agenda_id>2653</agenda_id>
     <import_id />
     <itemnumber>1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>2</level>
     <name>Ice Caps</name>
     <description />
     <shortdescription />
     <agenda_id>2654</agenda_id>
     <import_id />
     <itemnumber>1.1</itemnumber>
  </currentsubject>
  <currentsubject>
     <level>3</level>
     <name>Rising Water Levels</name>
     <description />
     <shortdescription />
     <agenda_id>2655</agenda_id>
     <import_id />
     <itemnumber />
  </currentsubject>
  <result>
     <session_id>762</session_id>
     <name>Anne Grete Holmsgaard</name>
     <group>Socialistisk Folkeparti</group>
     <groupabbreviation>SF</groupabbreviation>
     <votegroup>SF</votegroup>
     <delegate_id>109</delegate_id>
     <import_id>100691</import_id>
     <voteweight>1</voteweight>
     <vote>1</vote>
     <votetext>For</votetext>
  </result>
  <timestamp>
     <eventtime>2009-10-28T09:45:51.2541477+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

level	Identifies the hierarchy of subject/parent subjects. Level 1 is main, and 2.. are sub items.
name	Name of the current subject as well as name of parent subjects to the current subject
name	Agenda item title
description	Agenda item description
shortdescription	Agenda item short text description
import_id	unique number identifying participant imported
agenda_id	unique number for agenda item SW6000 auto-generated
itemnumber	Item Number text as input in agenda from CAA
session_id	Unique identifier for voting session

name	First/Last name of the participant
group	as setup in CAA
groupabbreviation	as setup in CAA
votegroup	as setup in CAA
delegate_id	unique number identifying participant [note two participants may have identical names]
voteweight	as setup in CAA
vote	choice 1-5 according to button pressed
votetext	choice according to voting configuration in use (button text).

Description: Individual voting results is called on the voting session close event. Note individual voting results are only for voting sessions held during a meeting if participants are logged in or on a Prepared participant seat table. Individual voting results are not sent for voting sessions held as secret voting.

2.6.1.8 StatusUpdateStart and StatusUpdateDone

Input parameters: No input

Return values: empty string [STRING]

Description: Used as wrappers for system initialization after reboot/restart. See 2.7.4

When Web Service receives a StatusUpdateStart any current status should be updated accordingly – e.g. after a CUI restart with a running Web Service/Streaming solution.

2.6.1.9 StreamingStart

Input parameters: strStreamingModeXML [STRING]

XML format:

```
<newdataset>
  <streamingmode>
    <mode>Broadcast</mode>
  </streamingmode>
  <timestamp>
     <eventtime>2009-10-27T14:07:44.0956233+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

mode	Streaming mode. Values:
	Broadcast, Archive or BroadcastArchive

Return values: empty string [STRING]

Description: Called by user from within the CUA

2.6.1.10 StreamingStop

Input parameters: strStreamingStopXML

XML format:

```
<newdataset>
  <timestamp>
     <eventtime>2009-10-27T14:07:44.0956233+01:00</eventtime>
  </timestamp>
</newdataset>
```

Return values: empty string [STRING]

Description: Called by user from within the CUA

2.6.1.11 AgendaUpdate

Input parameters: strAgendaXML [STRING]

XML Format (Complete content of Agenda dataset):

```
<newdataset>
  <agenda>
    <level>1</level>
    <sortorder>1</sortorder>
    <iscurrent>1</iscurrent>
    <name>1. Welcome</name>
    <description>Text description 1</description>
    <shortdescription>Short text description 1</shortdescription>
    <agenda_id>65</agenda_id>
    <import_id />
    <itemnumber />
  </agenda>
  <agenda>
    <level>2</level>
    <sortorder>2</sortorder>
    <iscurrent>2</iscurrent>
    <name>1.1. Introduction</name>
    <description />
    <shortdescription />
    <agenda_id>66</agenda_id>
    <import_id />
    <itemnumber />
  </agenda>
  <agendalink>
    <agenda_id>66</agenda_id>
    <sortorder>1</sortorder>
    <linknumber />
    <linkname>SAMPLE.PDF</linkname>
    <linkurl>X:\SAMPLE.PDF</linkurl>
  </agendalink>
  <agenda_event>
    <type>AgendaUpdate</type>
  </agenda_event>
  <timestamp>
    <eventtime>2012-09-17T14:20:07.8621441+02:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

level	Indent level
sortorder	Display order of the items in the xml file. (item number)
iscurrent	If all items are 0, no item is current. If an item is 2, it's the current item, parent items to the current item will be 1 – facilitating identification.
name	Agenda item displayed text – field is formatted according to standard. If no text is entered for an agenda item but a speech type/speaker has been entered these will display in the text.
description	Agenda item description
shortdescription	short text description
import_id	Id originating from import

agenda_id	Unique id for agenda item SW6000 autogenerated
itemnumber	Item Number text as input in agenda from CAA
linknumber	The link number text for an agenda link
linkname	The link name for an agenda link
linkurl	The link target URL for an agenda link. The target can be any URI – typically a file or a web reference (http, ftp etc.).
type	Identifies if the agenda update is due to either a change in subject (AgendaSubject) or a change in agendacontent (AgendaUpdate).
eventtime	When the event occurred

Return values: empty string [STRING]

Description: Called when Agenda content changes, at start and stop of meeting and at startup with event type **AgendaUpdate**. Also called when an agenda item is set active or no agenda item active with event type **AgendaSubject**. Note that if any of the fields are not available in the requested language, these fields will be transmitted in the default language, and if not available in the default language an empty field is transmitted.

2.6.1.12 DelegateDetails

Input parameters: strDelegateInfoXML [STRING]

XML format:

```
<newdataset>
  <delegate>
     <title>Statsministeren</title>
     <first_name>Anders Fogh</first_name>
     <last_name>Rasmussen</last_name>
     <show_name>Statsministeren</show_name>
     <usertype>Minister</usertype>
     <usertable1 />
     <usertable2 />
     <usertable3 />
     <usertable4 />
     <group />
     <votegroup />
     <background>additional information</background>
     <delegate_id>205</delegate_id>
     <import_id>100735</import_id>
     <photos>/9j/4AAQSkZJRgABA …
                …
… ABQAUAFABQAUAFABQAUAFABQAUAFID//ZAA==</photos>
  </delegate>
  <timestamp>
     <eventtime>2009-10-28T10:25:32.4889077+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

title	Participant title
first_name	First name of the Participant
last_name	Last name of the Participant
show_name	Display name according to meeting role
usertype	Meeting role of user. E.g. Delegate or Chairperson
usertable1 to 4	Four user-defined fields, available for entering various text information. E.g. Country or department.

group	As User table, normally used for Party
votegroup	User defined field – used to differentiate participants when using voting formulas
background	User defined field – available for entering text describing details of participant – can be up to 1000 char, and multiline
delegate_id	Id of Participant
import_id	Id originating from import
photos	serialized byte array of participant picture
eventtime	When the event occurred

Note that if any of the fields are not available in the requested language, these fields will be transmitted in the default language, and if not available in the default language an empty field is transmitted – see example.

Description: The DelegateDetails function has a parameter which includes the participant picture and this parameter contains a large amount of data if pictures have been assigned to participants. To reduce communication of large amounts of data to the webservice the DelegateDetails may be omitted depending on the setup in the CAA.

DelegateDetails are sent within the StatusUpdate process (see 2.7.1.6) if a meeting is in progress and participants participating in the meeting are identified – (on a participant list). If a meeting with login mode 'Login using code' is in progress only participants logged in will have details sent. On subsequent login's from other participants the details of these participants will be transmitted as the participants perform login.

Note that the DelegateDetails call has always been made prior to DelegateLogin call – thus details are always available when the DelegateLogin call is made.

2.6.1.13 DelegateLogin

Input parameters: strLoginString [STRING]

XML format:

```
<newdataset>
  <login>
    <seat>30</seat>
    <usertype>Minister</usertype>
    <show_name>Statsministeren</show_name>
    <delegate_id>205</delegate_id>
    <import_id>100735</import_id>
  </login>
  <timestamp>
    <eventtime>2009-11-13T14:00:49.4525549+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

seat	Identification of seat where login is performed
usertype	Meeting role of the user. E.g Delegate or Chairperson
show_name	Display name according to meeting role
delegate_id	Id of Participant
import_id	Id originating from import
eventtime	When the event occurred

Description: The DelegateLogin call is made whenever a participant is logged in to a seat.

For the meeting mode ‘Prepared participant seat table’ a ‘DelegateLogin’ is sent for all participants on the list on meeting start.

For meeting modes where login is required, the ‘DelegateLogin’ call is made when an actual login is performed.

Note that details for the participants are made available in the ‘DelegateDetails’ calls. A participant will only be logged in once for the meeting mode ‘Prepared participant seat table’, while in modes using login e.g. ‘Login using code’ the participant may be logged in several times at different locations, and possibly with different meeting roles.

For mode ‘Login using code on seat’ and ‘Login using code on list’ the ‘DelegateDetails’ call is only made once – but as the participant may be logged in several times with a new meeting role it’s important that the meeting role and ‘show_name’ settings are updated on a new login. Pictures are not updated and a performance gain is achieved by not sending participant details for each login when using these modes. A single call may contain multiple login entries.

2.6.1.14 DelegateLogout

Input parameters: strLogoutString [STRING]

XML format:

```
<newdataset>
  <logout>
    <seat>30</seat>
    <delegate_id>205</delegate_id>
    <import_id>100735</import_id>
  </logout>
  <timestamp>
    <eventtime>2009-11-13T14:00:49.4525549+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

seat	Identification of seat where login is performed
delegate_id	Id of Participant
import_id	Id originating from import
eventtime	When the event occurred

Description: The ‘DelegateLogout’ call is made whenever a participant is logged out from a seat. For meeting modes where login is required, the ‘DelegateLogout’ call is made when an actual logout is performed. When a meeting requiring login is stopped all participants are logged out, this is also true for meeting mode ‘Prepared participant seat table’. A single call may contain multiple logout entries. Stopping a meeting is equivalent of performing logout for all participants.

2.6.1.15 AlertStart

Input parameters: strAlertString [STRING]

XML format:

```
<newdataset>
  <alert>
    <text>A</text>
  </alert>
  <timestamp>
    <eventtime>2009-11-13T14:00:49.4525549+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

alert	Alert text as setup for the meeting
eventtime	When the event occurred

Return values:

empty string [STRING]

Parameter description: text

Description:

This alert originates from a CUA user with appropriate meeting role who activates the alert. Note that alerts are automatically deactivated when a voting session is stopped. The input parameter alert string is the string to display to users identifying this alert.

2.6.1.16 AlertStop

Input parameters: strAlertStop [STRING]

XML format:

```
<newdataset>
  <timestamp>
    <eventtime>2009-11-13T14:06:47.9442398+01:00</eventtime>
  </timestamp>
</newdataset>
```

Parameter description:

eventtime	When the event occurred

Return values: empty string [STRING]

Description: Removes a previously issued alert.

2.6.1.17 WebServiceStatus

Input parameters: No input

Return values: WebServiceStatus [STRING]

See Description

Description: The Web Service can return the current running state. The different states are described below:

Unavailable	Streaming system not available but Web Service is running
Needstatus	Streaming system is just started (could be after a shutdown/reboot). Indicates that the WS needs status/information from the SW6000 WSI
Ready	Status is updated but no streaming has started (not running)
Broadcasting	Broadcasting is running (set after StreamingStart is called)
Archiving	Archiving

BroadcastingArchiving Broadcasting and Archiving are running

If status is Ready, Broadcasting, Archiving or BroadcastingArchiving, the Streaming Start/Stop button on the CUA is enabled. In these states status changes in SW6000 are transmitted to the WS (e.g. AgendaUpdate, SpeakerChange …)

If status is Needstatus, password verification process is executed before any other action will work.

The WebServiceStatus call is issued repeatedly at a 10 second interval to monitor the WebService.

2.6.1.18 RequestDisplayLanguage

Input parameters: No input

Return values: LanguageId [INTEGER]

SW6000 supports multiple languages; the streaming solution should be able to be set up for any of the supported languages in the SW6000 database (as configured by the CAA).

E.g 1033 for English (US), 1030 for Danish and 1053 for Swedish.

If a language id is returned which is not supported in the Database for the current meeting then all strings sent to the WS will be in the default language as defined in the SW6000 database.

2.6.1.19 RequestStreamingSupport

Input parameters: No input

Return values: StreamingSupport [STRING]

Values:

Broadcast

Archiving

BroadcastArchiving

The StreamingSupport value returned tells the SW6000 what configuration the streaming solution supports. Just a direct broadcast (Broadcast), just archiving (Archiving) or simultaneous direct broadcast and archiving (BroadcastArchiving).

2.6.1.20 RequestVoteHideDelay

Input parameters: No input

Return value: VoteHideDelay [STRING] (is convertible to integer)

The VoteHideDelay value returned tells the SW6000 the delay in seconds between a VotingStop call and the subsequent VotingHide call. Default value is 3 seconds.

2.6.1.21 VerifyPassword

Input parameters: strPassword [STRING] WebServicePassword

Return value: strPassword [STRING] StreamingPassword

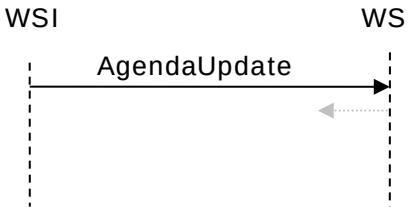
The two passwords are used to establish communication – WebServicePassword is used in the call from Web Service Interface, and Streaming password should be returned in the reply from the WebService. These passwords are setup in the CAA application and should match in order to initiate communication.

2.6.2 Handling Agenda changes

When changes are made in the Agenda, this is sent to the Streaming solution

A change to the Agenda may include insertion or deletion of agenda items as well as changes to the current active subject, including the change to set no agenda item active. Changes also include sending

an empty agenda e.g. on meeting stop. To differentiate between agenda updates involving agenda content and agenda updates changes to the current agenda item the agenda event type is available. An agenda is a hierarchical structure where sub items should be interpreted in relation to their parent items. To facilitate identification of this parent child relationship for the active agenda item, parent agenda items that act as parent to the active agenda item have isCurrent set to 1, and the active agenda item has isCurrent set to 2.



2.6.3 Handling Status request

The WSI must query WS for status every 10 seconds.

If status differs from ‘Ready’, appropriate action is taken. See 2.7.1.15

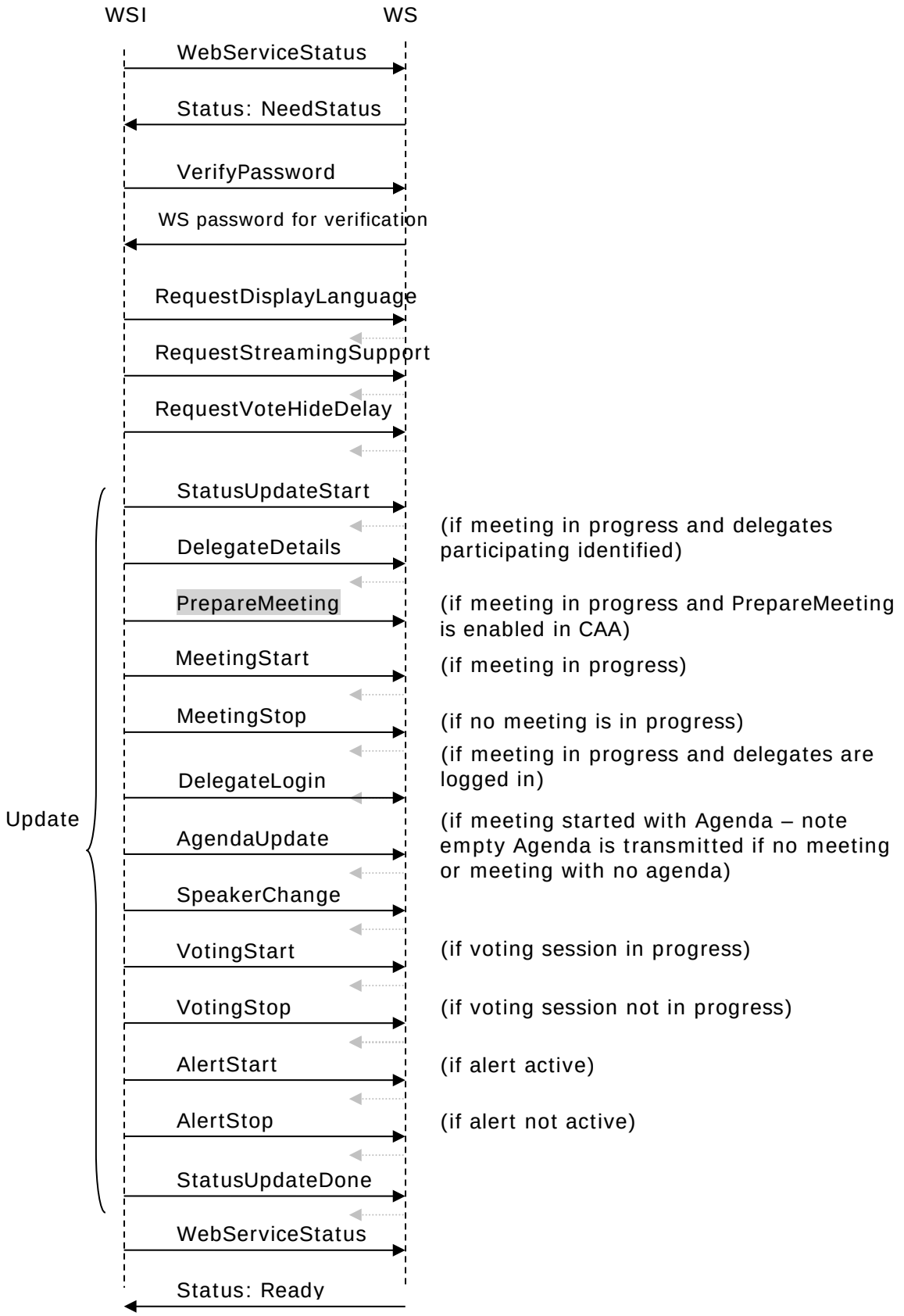
2.6.4 Handling web service and SW6000 reboot/restart/breakdown

The system will handle reboot/restart/breakdown and turn into actual state by itself.

If CUI is restarted on a running Web Service the Web Service will return e.g. Ready or Broadcasting, Archiving or Broadcasting-Archiving. Still status is updated according to flow depicted.

Below is an event diagram of the flow if this happens.

2.6.4.1 Starting up Web Service Interface (CUI) while Web Service is running



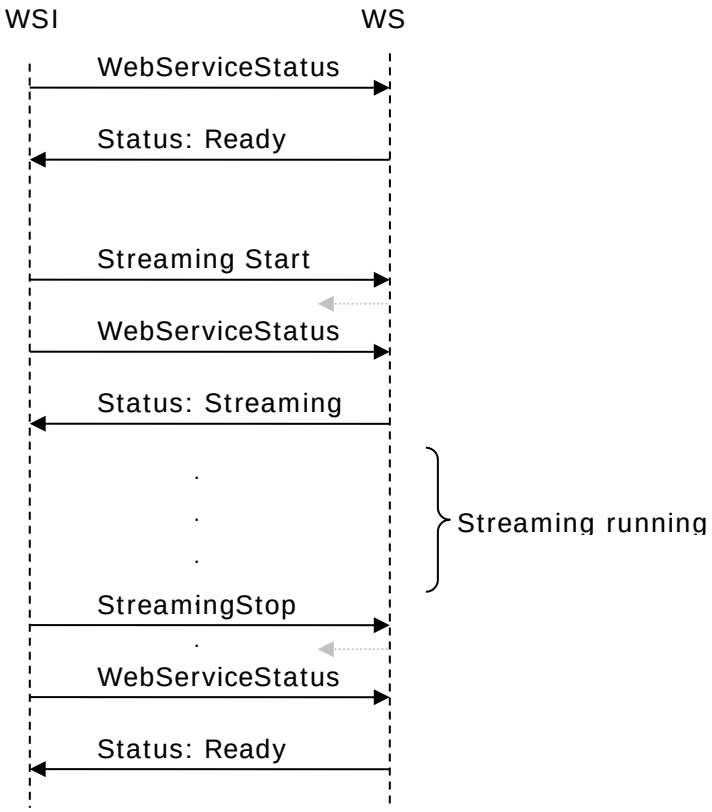
Handling no response/unavailable status from web service

WSI will not send any events to the WS as long as no response or Unavailable status received from WS

On the CUA radio buttons for selecting Streaming mode (Broadcast/Archiving/BroadcastArchiving) is grayed out.

2.6.5 Handling Streaming Start/Stop

The next figure shows the communication flow at Streaming start/stop



3 Meeting Import/Export

3.1 Introduction

The purpose of this chapter is to describe how meeting data can be imported to, and exported from SW6000 using the Conference Administration Application (CAA).

Data can be exported to, or imported from an XML formatted file or by use of a web service call.

The Meeting Import/Export facility can only be used before a meeting is started or after a meeting is stopped.

Upon export of data from the CAA, a schema file is also exported, defining the data format used. To clarify the individual constructs and fields in the file, this document describes the data format used for such import/export operations in greater detail.

3.2 CAA

3.2.1 CAA/Setup/Equipment/Meeting Import/Export

3.2.1.1 Meeting Import/Export Mode

File

The default Import/Export mode for meetings is to/from a file folder.

Please refer to section

3.3 File Import/Export

for specifications.

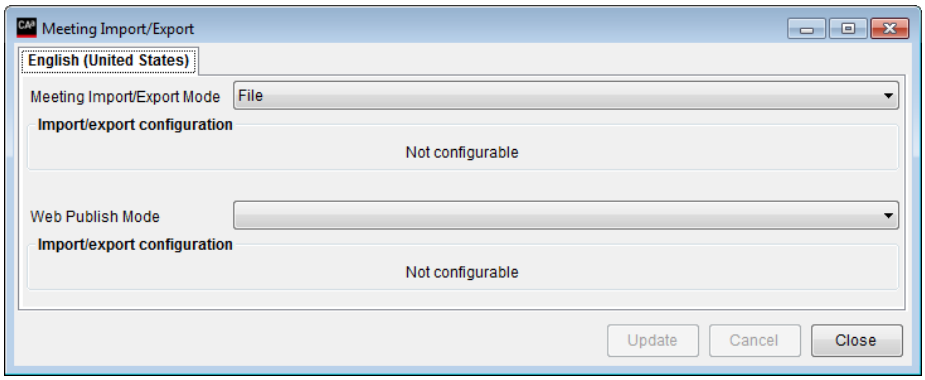


Figure 3.2-A

Web Service

Alternatively Import/Export can be done through a third party web service.

Please refer to section

3.4 Web Service

for specifications.

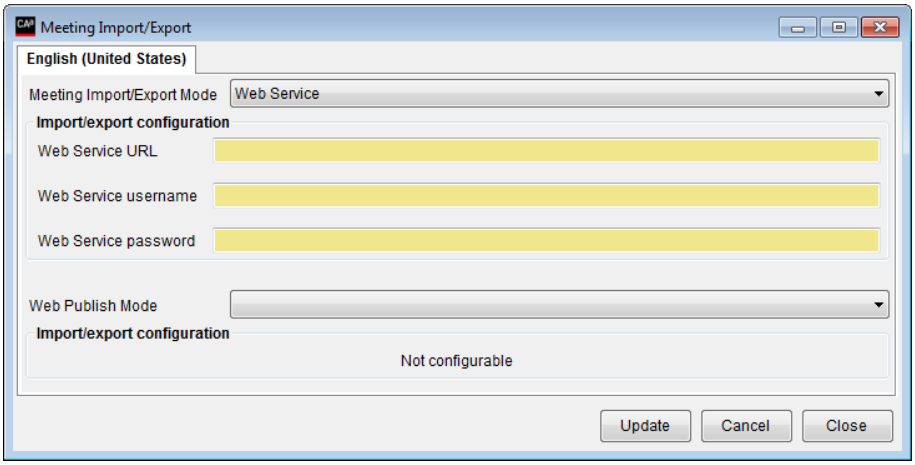


Figure 3.2-B

3.2.1.2 Web Publish Mode

Web Application

The Web Application selection is a ‘beta’ feature with limited functionality.

The selection is only available when the files ***Newtonsoft.Json.dll*** and ***WebMeetingIntegration.dll*** are copied into the folder ‘C:\Program Files (x86)\DIS\Conference Administrator Application\ImportExport’.

The two files are available in the folder ‘WebMeetingIntegration’ in the SW6000 installation package.

When Web Application is selected the available options are shown.

The settings for

URL, User Name, Password Committee and Publish Languages

are all for future use and are not supported.

The entries for ‘Reply Configuration’ and ‘Reply#’ are used for setting the Reply# used for Pre-requests to Speak in the CAA Agenda.

Refer to ‘User Manual SW6000 CAA’ for details.

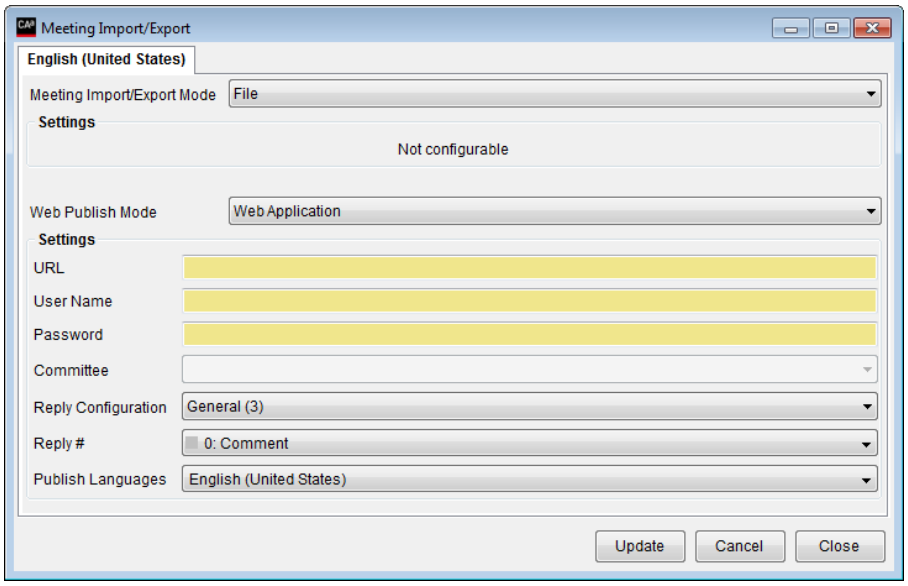
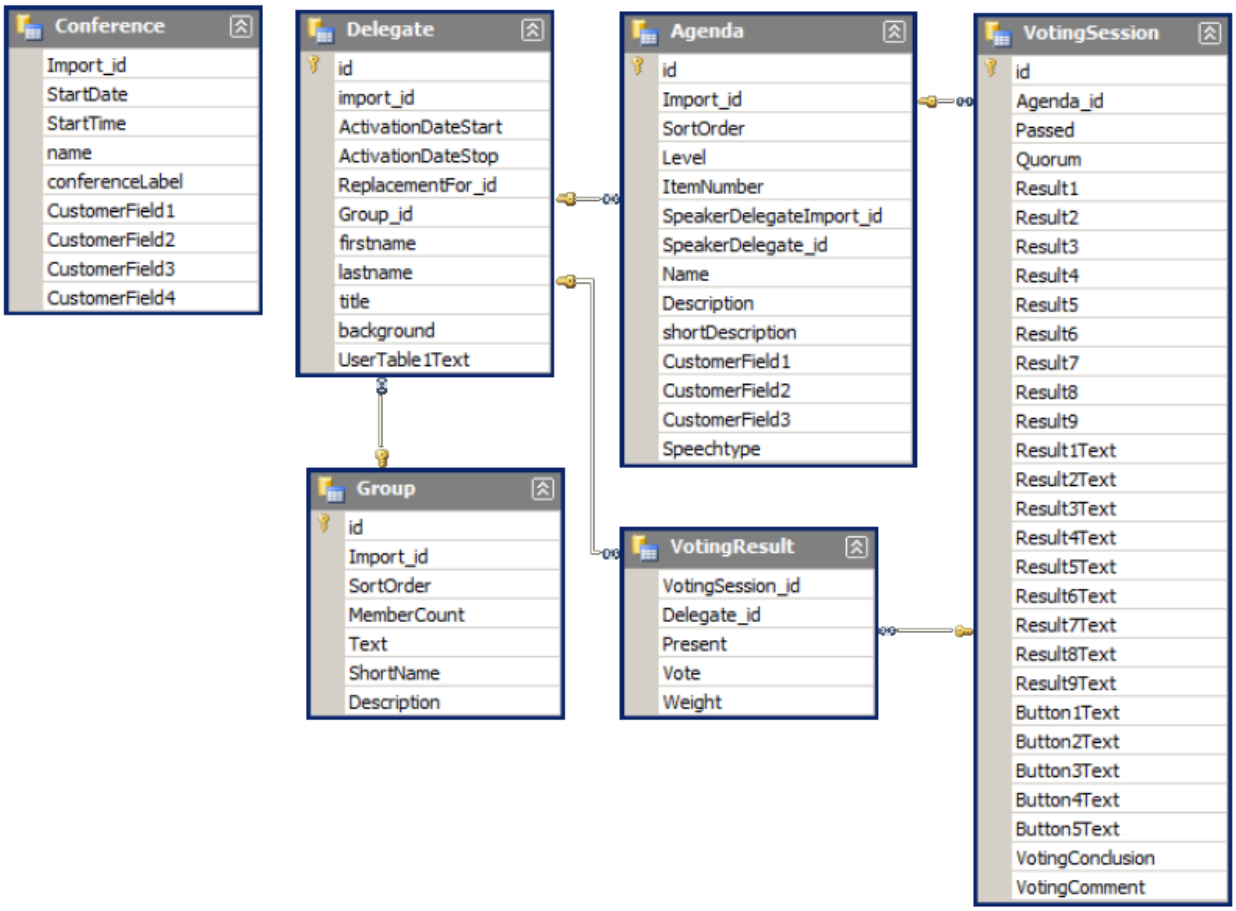


Figure 3.2-C

3.3 File Import/Export

3.3.1 Data format

Generically speaking, the contents of an export file can be schematically represented as depicted below.



That is, the file contains Conference (meeting), Delegate (participants), participants grouping (Group), agenda, voting and participants voting information. The following sections describe the individual data records in detail.

The import file is similar to the export file but does not include voting results.

Each import/export file contains only one meeting record. All other records in the file contain data related to that one meeting.

Each import/export file is mono-lingual. That is, all texts are exported using the language selected in the CAA. If texts are not defined in that language, the default language version of the text will be exported instead.

The body of the import/export data file XML must be contained within an <ExportConferenceDataDataSet></ExportConferenceDataDataSet> tag pair.

3.3.1.1 Conference (Meeting)

The Conference record contains information about the meeting described in the import/export.

Since each import/export file only describes a single meeting, only one Meeting record is allowed in each import/export file.

Field name	Data type	Description
Import_id	Integer	External ID for meeting. The ID the receiver of the export/creator of the import uses for this meeting. This field only exists in export files from SW6000 if the meeting has been previously imported with an Import_id.
StartDate	DateTime	Start date for the meeting. When importing, both the start and stop date for the meeting will be assigned this value.
StartTime	String (max. 10 chars)	A string representation of the start time for the meeting. No parsing of the string will be done.
name	String (max. 50 chars)	The name of the meeting.
conferenceLabel	String (max. 50 chars)	The label of the meeting. This is what is dubbed 'Id' for the meeting in the CAA application.
CustomerField1	String (max. 100 chars)	The content of 'Customer field 1' for the meeting
CustomerField2	String (max. 100 chars)	The content of 'Customer field 2' for the meeting
CustomerField3	String (max. 100 chars)	The content of 'Customer field 3' for the meeting
CustomerField4	String (max. 100 chars)	The content of 'Customer field 4' for the meeting

A meeting record exists within a <Conference></Conference> tag pair in the import/export file.

3.3.1.2 Delegate (Participants)

The delegate record contains information about a participant which is related to the meeting. Since there can be many participants related to a meeting, many delegate records can exist in an import/export file.

Field name	Data type	Description
id	Integer	The ID of the participant as used in this import/export file.
import_id	Integer	External ID for participant. The ID the receiver of the export/creator of the import uses for this participant. A participant import_id must be unique amongst all participants in the SW6000 system – not just amongst the participants in one particular import file. This field only exists in export files from SW6000 if the participant has been previously imported with an Import_id.
ActivationDateStart	DateTime, optional	Start date for the participant replacement specified in ReplacementFor_id field
ActivationDateStop	DateTime, optional	Stop date for the participant replacement specified in ReplacementFor_id field
ReplacementFor_id	Integer, optional	The ID of a participant which this participant replaces.
Group_id	Integer, optional	The ID of the group this participant is a member of.
firstname	String (max. 50 chars)	First name of the participant
lastname	String (max. 50 chars)	Last name of the participant
title	String (max. 50 chars)	Title of the participant
background	String (max. 100 chars)	The background description of the participant
UserTable1Text	String (max. 30 chars)	The content of 'User Table 1' for the participant

When importing it is required that import_id is specified.

import_id will, however, only be specified in an export file if a value has been assigned to the participant.

When importing, it is required that, if specified, ActivationDateStart be an earlier date than ActivationDateStop, or that ActivationDateStop is not specified. If no ActivationDateStart is specified no ActivationDateStop may be specified.

If the participant is already present when importing (based on the import_id field) the participant is updated in the database. Any optional fields not specified will be assigned default values.

A delegate record exists within a <Delegate></Delegate> tag pair in the import/export file.

3.3.1.3 Group

The group record describes a group to which participants can be related. Many groups may be defined in a single import/Export file.

Field name	Data type	Description
id	Integer	The ID of the group used as in this import/export file.
Import_id	Integer	External ID for group. The ID the receiver of the export/creator of the import uses for this group. A group Import_id must be unique amongst all groups in the SW6000 system – not just amongst the groups in one particular import file. This field only exists in export files from SW6000 if the group has been previously imported with an Import_id.
SortOrder	Integer	The sorting priority of the group
Text	String (max. 100 chars)	The name of the group
ShortName	String (max. 30 chars)	The abbreviated name for the group
Description	String (max. 100 chars), optional	Description for the group.

A group record exists within a <Group></Group> tag pair in the import/export file.

3.3.1.4 Agenda

The agenda record describes a single agenda item for the meeting. Many agenda items can exist in a single import/export file.

Field name	Data type	Description
id	Integer	The ID of the agenda item as used in this import/export file.
Import_id	Integer	External ID for agenda item. The ID the receiver of the export/creator of the import uses for this agenda item. This field only exists in export files from SW6000 if the agenda item has been previously imported with an Import_id.
SortOrder	Integer	The position in the agenda of this agenda item. This must be a number in the range 1 to number of agenda items.
Level	Integer	The subordination level of the agenda item. Must be in range 1 to 6, with level 1 being a top level agenda item.
ItemNumber	Integer	The item number of the agenda item.
SpeakerDelegateImport_id	Integer, optional	The external ID for a participant which is going to speak in this agenda item. On import, this field is not used if the SpeakerDelegate_id is specified. It is thus possible to specify SpeakerDelegateImport_id without specifying SpeakerDelegate_id, effectively referring a participant by its Import_id rather than its id field. On export this field is only present if SpeakerDelegate_id is present, and the participant specified has an Import_id value.
SpeakerDelegate_id	Integer, optional	The ID for a participant which is going to speak in this agenda item.
Name	String (max. 1.000 chars)	The name of the agenda item
Description	String (max. 10.000 chars)	A description for the agenda item
shortDescription	String (max. 500 chars)	A short description for the agenda item
CustomerField1	String (max. 100 chars)	The content of 'Customer field 1' for the agenda.
CustomerField2	String (max. 100 chars)	The content of 'Customer field 2' for the agenda.
CustomerField3	String (max. 100 chars)	The content of 'Customer field 3' for the agenda.
Speechtype	String (max. 100 chars)	The speaker type of the agenda item

Import of agenda items is disallowed for a meeting if there has been voted on one or more agenda items.

An agenda record exists within an <Agenda></Agenda> tag pair in the import/export file.

3.3.1.5 VotingSession

The voting session record describes the result of a voting related to an agenda item. Many voting sessions may exist in an export file. Voting sessions, however, cannot be imported, and are thus not allowed in import files.

If no voting has been performed during an agenda item, there will be no voting session record for that agenda item.

Field name	Data type	Description
id	Integer	The ID of the voting session as used in this import/export file.
Agenda_id	Integer	The ID of the agenda item this VotingSession is for
Passed	Integer	Voting result: 1 – Passed 0 – Not passed -1 – Not voted
Quorum	Integer	Quorum check result: 1 – Passed 0 – Not passed -2 – Not checked
Result1	Integer	The result of Result1 of the voting
Result2	Integer	The result of Result2 of the voting
Result3	Integer	The result of Result3 of the voting
Result4	Integer	The result of Result4 of the voting
Result5	Integer	The result of Result5 of the voting
Result6	Integer	The result of Result6 of the voting
Result7	Integer	The result of Result7 of the voting
Result8	Integer	The result of Result8 of the voting
Result9	Integer	The result of Result9 of the voting
Result1Text	String (max. 25 chars)	The name of Result1, i.e. its label for presentation.
Result2Text	String (max. 25 chars)	The name of Result2, i.e. its label for presentation.
Result3Text	String (max. 25 chars)	The name of Result3, i.e. its label for presentation.
Result4Text	String (max. 25 chars)	The name of Result4, i.e. its label for presentation.
Result5Text	String (max. 25 chars)	The name of Result5, i.e. its label for presentation.
Result6Text	String (max. 25 chars)	The name of Result6, i.e. its label for presentation.
Result7Text	String (max. 25 chars)	The name of Result7, i.e. its label for presentation.
Result8Text	String (max. 25 chars)	The name of Result8, i.e. its label for presentation.
Result9Text	String (max. 25 chars)	The name of Result9, i.e. its label for presentation.
Button1Text	String (max. 25 chars)	The name of voting button 1
Button2Text	String (max. 25 chars)	The name of voting button 2
Button3Text	String (max. 25 chars)	The name of voting button 3
Button4Text	String (max. 25 chars)	The name of voting button 4
Button5Text	String (max. 25 chars)	The name of voting button 5
VotingConclusion	String (max. 3500 chars)	Autogenerated conclusion for the voting.
VotingComment	String (max. 3500 chars)	Voting comment, as entered in the CAA after voting.

The Result1-9 fields are calculated using the result formulas defined in the CAA for the relevant voting configuration.

A VotingSession record exists within a <VotingSession></VotingSession> tag pair in the export file.

3.3.1.6 VotingResult

The voting result record describes the voting result of a single participant in a single voting session. Many voting session may exist, so the same participant may have many voting result records, albeit only one voting result per voting session.

Field name	Data type	Description
VotingSession_id	Integer	The ID of the voting session this VotingResult is for
Delegate_id	Integer	The ID of the participant this VotingResult is for
Present	Integer	The presence result of the participant. 0 if the user was not present.
Vote	Integer	The voting button which was pressed by the participant. A value in range 1-5. The vote name of the selected button is found in VotingSession fields ButtonNText.
Weight	Integer	The voting weight of the participant at the time of voting.

A VotingResult record exists within a <VotingResult></VotingResult> tag pair in the export file.

3.3.2 File format XSD

```
<?xml version="1.0" standalone="yes"?>
<xs:schema id="ExportConferenceDataDataSet" xmlns="" xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xs:element name="ExportConferenceDataDataSet" msdata:IsDataSet="true"
msdata:UseCurrentLocale="true">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="Conference">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Import_id" type="xs:int" minOccurs="0" />
              <xs:element name="StartDate" type="xs:dateTime" minOccurs="0" />
              <xs:element name="StartTime" type="xs:string" minOccurs="0" />
              <xs:element name="name" type="xs:string" minOccurs="0" />
              <xs:element name="conferenceLabel" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField1" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField2" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField3" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField4" type="xs:string" minOccurs="0" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Agenda">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:int" minOccurs="0" />
              <xs:element name="Import_id" type="xs:int" minOccurs="0" />
              <xs:element name="SortOrder" type="xs:int" minOccurs="0" />
              <xs:element name="Level" type="xs:int" minOccurs="0" />
              <xs:element name="ItemNumber" type="xs:string" minOccurs="0" />
              <xs:element name="SpeakerDelegateImport_id" type="xs:int" minOccurs="0" />
              <xs:element name="SpeakerDelegate_id" type="xs:int" minOccurs="0" />
              <xs:element name="Name" type="xs:string" minOccurs="0" />
              <xs:element name="Description" type="xs:string" minOccurs="0" />
              <xs:element name="shortDescription" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField1" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField2" type="xs:string" minOccurs="0" />
              <xs:element name="CustomerField3" type="xs:string" minOccurs="0" />
              <xs:element name="Speechtype" type="xs:string" minOccurs="0" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="Group">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:int" minOccurs="0" />
```

```
      <xs:element name="Import_id" type="xs:int" minOccurs="0" />
      <xs:element name="SortOrder" type="xs:int" minOccurs="0" />
      <xs:element name="MemberCount" type="xs:int" minOccurs="0" />
      <xs:element name="Text" type="xs:string" minOccurs="0" />
      <xs:element name="ShortName" type="xs:string" minOccurs="0" />
      <xs:element name="Description" type="xs:string" minOccurs="0" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="Delegate">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="id" type="xs:int" minOccurs="0" />
      <xs:element name="import_id" type="xs:int" minOccurs="0" />
      <xs:element name="ActivationDateStart" type="xs:dateTime" minOccurs="0" />
      <xs:element name="ActivationDateStop" type="xs:dateTime" minOccurs="0" />
      <xs:element name="ReplacementFor_id" type="xs:int" minOccurs="0" />
      <xs:element name="Group_id" type="xs:int" minOccurs="0" />
      <xs:element name="firstname" type="xs:string" minOccurs="0" />
      <xs:element name="lastname" type="xs:string" minOccurs="0" />
      <xs:element name="title" type="xs:string" minOccurs="0" />
      <xs:element name="background" type="xs:string" minOccurs="0" />
      <xs:element name="UserTable1Text" type="xs:string" minOccurs="0" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="VotingSession">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="id" type="xs:int" minOccurs="0" />
      <xs:element name="Agenda_id" type="xs:int" minOccurs="0" />
      <xs:element name="Passed" type="xs:int" minOccurs="0" />
      <xs:element name="Quorum" type="xs:int" minOccurs="0" />
      <xs:element name="Result1" type="xs:int" minOccurs="0" />
      <xs:element name="Result2" type="xs:int" minOccurs="0" />
      <xs:element name="Result3" type="xs:int" minOccurs="0" />
      <xs:element name="Result4" type="xs:int" minOccurs="0" />
      <xs:element name="Result5" type="xs:int" minOccurs="0" />
      <xs:element name="Result6" type="xs:int" minOccurs="0" />
      <xs:element name="Result7" type="xs:int" minOccurs="0" />
      <xs:element name="Result8" type="xs:int" minOccurs="0" />
      <xs:element name="Result9" type="xs:int" minOccurs="0" />
      <xs:element name="Result1Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result2Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result3Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result4Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result5Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result6Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result7Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result8Text" type="xs:string" minOccurs="0" />
      <xs:element name="Result9Text" type="xs:string" minOccurs="0" />
      <xs:element name="Button1Text" type="xs:string" minOccurs="0" />
      <xs:element name="Button2Text" type="xs:string" minOccurs="0" />
      <xs:element name="Button3Text" type="xs:string" minOccurs="0" />
      <xs:element name="Button4Text" type="xs:string" minOccurs="0" />
      <xs:element name="Button5Text" type="xs:string" minOccurs="0" />
      <xs:element name="VotingConclusion" type="xs:string" minOccurs="0" />
      <xs:element name="VotingComment" type="xs:string" minOccurs="0" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="VotingResult">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="VotingSession_id" type="xs:int" minOccurs="0" />
```

```
            <xs:element name="Delegate_id" type="xs:int" minOccurs="0" />
            <xs:element name="Present" type="xs:double" minOccurs="0" />
            <xs:element name="Vote" type="xs:double" minOccurs="0" />
            <xs:element name="Weight" type="xs:double" minOccurs="0" />
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
  <xs:unique name="Constraint1">
    <xs:selector xpath=".//Agenda" />
    <xs:field xpath="id" />
  </xs:unique>
  <xs:unique name="Group_Constraint1" msdata:ConstraintName="Constraint1">
    <xs:selector xpath=".//Group" />
    <xs:field xpath="id" />
  </xs:unique>
  <xs:unique name="Delegate_Constraint1" msdata:ConstraintName="Constraint1">
    <xs:selector xpath=".//Delegate" />
    <xs:field xpath="id" />
  </xs:unique>
  <xs:unique name="VotingSession_Constraint1" msdata:ConstraintName="Constraint1">
    <xs:selector xpath=".//VotingSession" />
    <xs:field xpath="id" />
  </xs:unique>
  <xs:keyref name="VotingResultToDelegate" refer="Delegate_Constraint1">
    <xs:selector xpath=".//VotingResult" />
    <xs:field xpath="Delegate_id" />
  </xs:keyref>
  <xs:keyref name="VotingResultToVotingSession" refer="VotingSession_Constraint1">
    <xs:selector xpath=".//VotingResult" />
    <xs:field xpath="VotingSession_id" />
  </xs:keyref>
  <xs:keyref name="VotingSessionToAgenda" refer="Constraint1">
    <xs:selector xpath=".//VotingSession" />
    <xs:field xpath="Agenda_id" />
  </xs:keyref>
  <xs:keyref name="DelegateToGroup" refer="Group_Constraint1">
    <xs:selector xpath=".//Delegate" />
    <xs:field xpath="Group_id" />
  </xs:keyref>
  <xs:keyref name="AgendaToDelegate" refer="Delegate_Constraint1">
    <xs:selector xpath=".//Agenda" />
    <xs:field xpath="SpeakerDelegate_id" />
  </xs:keyref>
 </xs:element>
</xs:schema>
```

3.3.3 Example XML file

```
<?xml version="1.0" standalone="yes"?>
<ExportConferenceDataDataSet>
  <Conference>
    <StartDate>2011-04-29T00:00:00+02:00</StartDate>
    <StartTime />
    <name>Conference 07 (No login)</name>
    <conferenceLabel>7000</conferenceLabel>
    <CustomerField1 />
    <CustomerField2 />
    <CustomerField3 />
    <CustomerField4 />
  </Conference>
  <Agenda>
    <id>120</id>
    <SortOrder>1</SortOrder>
    <Level>1</Level>
    <Name>1. Agenda</Name>
    <Description>Text description 1</Description>
    <shortDescription>Short text description 1</shortDescription>
  </Agenda>
  <Agenda>
    <id>121</id>
    <SortOrder>2</SortOrder>
    <Level>2</Level>
    <Name>1.1. Agenda</Name>
    <Description />
    <shortDescription />
  </Agenda>
  <Group>
    <id>44</id>
    <SortOrder>1</SortOrder>
    <MemberCount>4</MemberCount>
    <Text>Denmark</Text>
    <ShortName>DK</ShortName>
  </Group>
  <Delegate>
    <id>43</id>
    <Group_id>44</Group_id>
    <firstname>Bent</firstname>
    <lastname>Nielsen</lastname>
    <title>Software Tester</title>
  </Delegate>
  <Delegate>
    <id>48</id>
    <firstname>Benny</firstname>
    <lastname>Pedersen</lastname>
    <title>Software Programmer</title>
  </Delegate>
</ExportConferenceDataDataSet>
```

3.4 Web Service

It is possible to have the SW6000 system call a web service to facilitate import and export of meetings. The following section describes the Web Service calls the SW6000 will make.

Note that this document is not intended to be in-depth in its description of the data passed back and forth between SW6000 and the WebService implementation.

3.4.1 Web Service Functions

The following are the web service functions that SW6000 will try to call. There are two functions relating to import into SW6000 and one relating to export from SW6000.

3.4.1.1 GetConferenceList

The GetConferenceList endpoint must deliver a list of conferences available for import for a given date range. This endpoint does not deliver full meeting data, rather summary data of those conferences.

Parameters:

Parameter Name	Data Type & Format	Description
startDate	Date/time	Specifies the opening date of the date range for which to get available conferences.
endDate	Date/Time	Specifies the opening date of the date range for which to get available conferences.

Returns: ConferenceListData, i.e. an array of ConferenceSummaryData describing the meetings available for import.

Example request:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://dis.cc/IConferenceIoService/GetC
onferenceList</Action>
  </s:Header>
  <s:Body>
    <GetConferenceList xmlns="http://dis.cc">
      <startDate>2017-10-27T00:00:00+02:00</startDate>
      <endDate>2017-11-03T00:00:00+01:00</endDate>
    </GetConferenceList>
  </s:Body>
</s:Envelope>
```

Example response:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <ActivityId CorrelationId="3ac3e6e0-253e-497f-b9de-a0613f3ce768"
xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">00000000-0000-0000-0000-
000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <GetConferenceListResponse xmlns="http://dis.cc">
      <GetConferenceListResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <Conferences>
          <ConferenceSummary>
            <ImportId>10000</ImportId>
            <Name>Test 001</Name>
            <ConferenceLabel>001</ConferenceLabel>
            <StartDate>2016-10-06T00:00:00+02:00</StartDate>
            <StartTime />
            <CustomerField1 />
            <CustomerField2 />
            <CustomerField3 />
            <CustomerField4 />
          </ConferenceSummary>
          <ConferenceSummary>
            <ImportId>10001</ImportId>
            <Name>Login using code</Name>
            <ConferenceLabel />
            <StartDate>2016-10-12T00:00:00+02:00</StartDate>
            <StartTime />
            <CustomerField1 />
            <CustomerField2 />
            <CustomerField3 />
            <CustomerField4 />
          </ConferenceSummary>
        </Conferences>
      </GetConferenceListResult>
    </GetConferenceListResponse>
  </s:Body>
</s:Envelope>
```

3.4.1.2 GetConference

The GetConference endpoint must deliver full meeting data for a single meeting.

Parameters:

Parameter Name	Data Type & Format	Description
importId	integer	The ImportId of the meeting to get. The import id given will be one from a meeting previously returned from GetConferenceList.

Returns:

GetConferenceData, i.e. a collection of meeting, group, delegate and agenda data for the meeting.

Example request

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://dis.cc/IConferenceIoService/GetC
onference</Action>
  </s:Header>
  <s:Body>
    <GetConference xmlns="http://dis.cc">
      <importId>20000</importId>
    </GetConference>
  </s:Body>
</s:Envelope>
```

Example response

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <ActivityId CorrelationId="3f34c573-dc5e-4564-89da-98ed9a1b94d7"
xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">00000000-0000-0000-0000-
000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <GetConferenceResponse xmlns="http://dis.cc">
      <GetConferenceResult xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <Conference>
          <ImportId>20000</ImportId>
          <Name>Test meeting</Name>
          <ConferenceLabel>001</ConferenceLabel>
          <StartDate>2016-10-06T00:00:00+02:00</StartDate>
          <StartTime />
          <CustomerField1 />
          <CustomerField2 />
          <CustomerField3 />
          <CustomerField4 />
        </Conference>
        <Groups>
          <Group>
            <Id>53</Id>
            <ImportId>20001</ImportId>
            <Text>Group A</Text>
            <ShortName />
            <Description />
            <SortOrder>0</SortOrder>
            <MemberCount>0</MemberCount>
          </Group>
          <Group>
            <Id>55</Id>
            <ImportId>20002</ImportId>
```

```
      <Text>Group B</Text>
      <ShortName />
      <Description />
      <SortOrder>0</SortOrder>
      <MemberCount>0</MemberCount>
    </Group>
  </Groups>
  <Delegates>
    <Delegate>
      <Id>56</Id>
      <ImportId>106943</ImportId>
      <Firstname>Delegate</Firstname>
      <Lastname>A</Lastname>
      <Title />
      <Background />
      <GroupId>57</GroupId>
      <ActivationDateStart>2008-10-01T00:00:00+02:00</ActivationDateStart>
      <UserTable1Text />
    </Delegate>
    <Delegate>
      <Id>58</Id>
      <ImportId>106904</ImportId>
      <Firstname>Delegate</Firstname>
      <Lastname>B</Lastname>
      <Title />
      <Background />
      <GroupId>53</GroupId>
      <ActivationDateStart>2008-10-01T00:00:00+02:00</ActivationDateStart>
      <UserTable1Text />
    </Delegate>
  </Delegates>
  <Agenda>
    <Agenda>
      <Id>251</Id>
      <SortOrder>1</SortOrder>
      <Level>1</Level>
      <Name>a</Name>
      <Description />
      <ShortDescription />
      <CustomerField1 />
      <CustomerField2 />
      <CustomerField3 />
      <SpeechType />
    </Agenda>
    <Agenda>
      <Id>252</Id>
      <SortOrder>2</SortOrder>
      <Level>1</Level>
      <Name>b</Name>
      <Description />
      <ShortDescription />
      <CustomerField1 />
      <CustomerField2 />
      <CustomerField3 />
      <SpeechType />
    </Agenda>
    <Agenda>
      <Id>253</Id>
      <SortOrder>3</SortOrder>
      <Level>1</Level>
      <Name>c</Name>
      <Description />
      <ShortDescription />
      <CustomerField1 />
      <CustomerField2 />
```

```
            <CustomerField3 />
            <SpeechType />
          </Agenda>
        </Agenda>
      </GetConferenceResult>
    </GetConferenceResponse>
  </s:Body>
</s:Envelope>
```

3.4.1.3 SetConference

The SetConference endpoint receives full meeting data for a single meeting.

Parameters:

Parameter Name	Data Type & Format	Description
export	SetConferenceData	A collection of meeting, group, delegate, agenda voting sessions and voting result data for a single meeting.

Returns:

The import id of the meeting exported. That is, the external ID SW6000 will associate with the exported meeting. Any future meeting import using this import id will be mapped to the just exported meeting, allowing for export/import roundtrips for updating meeting data in an external system.

Example request:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <Action s:mustUnderstand="1"
xmlns="http://schemas.microsoft.com/ws/2005/05/addressing/none">http://dis.cc/IConferenceIoService/SetC
onference</Action>
  </s:Header>
  <s:Body>
    <SetConference xmlns="http://dis.cc">
      <export xmlns:i="http://www.w3.org/2001/XMLSchema-instance">
        <Conference>
          <Name>Test meeting 2</Name>
          <ConferenceLabel />
          <StartDate>2017-10-27T00:00:00+02:00</StartDate>
          <StartTime />
          <CustomerField1 />
          <CustomerField2 />
          <CustomerField3 />
          <CustomerField4 />
        </Conference>
        <Groups>
          <Group>
            <Id>55</Id>
            <ImportId>20002</ImportId>
            <Text>Group A</Text>
            <ShortName />
            <Description />
            <SortOrder>2</SortOrder>
            <MemberCount>0</MemberCount>
          </Group>
        </Groups>
        <Delegates>
          <Delegate>
            <Id>54</Id>
            <ImportId>106940</ImportId>
            <Firstname>Delegate</Firstname>
            <Lastname>A</Lastname>
```

```
      <Title />
      <Background />
      <GroupId>55</GroupId>
      <ActivationDateStart>2008-10-01T00:00:00+02:00</ActivationDateStart>
      <UserTable1Text />
    </Delegate>
  </Delegates>
  <Agenda>
    <Agenda>
      <Id>804</Id>
      <SortOrder>1</SortOrder>
      <Level>1</Level>
      <ItemNumber />
      <Name>Subject A</Name>
      <Description />
      <ShortDescription />
      <CustomerField1 />
      <CustomerField2 />
      <CustomerField3 />
      <SpeechType />
    </Agenda>
    <Agenda>
      <Id>805</Id>
      <SortOrder>2</SortOrder>
      <Level>1</Level>
      <ItemNumber />
      <Name>Subject B</Name>
      <Description />
      <ShortDescription />
      <CustomerField1 />
      <CustomerField2 />
      <CustomerField3 />
      <SpeechType />
    </Agenda>
  </Agenda>
  <VotingSessions>
    <VotingSession>
      <Id>824</Id>
      <AgendaId>804</AgendaId>
      <Passed>1</Passed>
      <Quorum>-2</Quorum>
      <Result1>1</Result1>
      <Result2>0</Result2>
      <Result3>0</Result3>
      <Result4>0</Result4>
      <Result5>1</Result5>
      <Result6>231</Result6>
      <Result7>0</Result7>
      <Result8>0</Result8>
      <Result9>0</Result9>
      <Result1Text>Yes</Result1Text>
      <Result2Text>Abstain</Result2Text>
      <Result3Text>No</Result3Text>
      <Result4Text />
      <Result5Text>Total present</Result5Text>
      <Result6Text>Seats</Result6Text>
      <Result7Text />
      <Result8Text />
      <Result9Text />
      <Button1Text>Yes</Button1Text>
      <Button2Text>Abstain</Button2Text>
      <Button3Text>No</Button3Text>
      <Button4Text />
      <Button5Text />
      <VotingConclusion>Motion approved
```

```
1 votes for the motion (Delegate A ())

0 votes neither for nor against the motion

0 votes against the motion

</VotingConclusion>
            <VotingComment />
          </VotingSession>
          <VotingSession>
            <Id>825</Id>
            <AgendaId>805</AgendaId>
            <Passed>0</Passed>
            <Quorum>-2</Quorum>
            <Result1>0</Result1>
            <Result2>0</Result2>
            <Result3>1</Result3>
            <Result4>0</Result4>
            <Result5>1</Result5>
            <Result6>231</Result6>
            <Result7>0</Result7>
            <Result8>0</Result8>
            <Result9>0</Result9>
            <Result1Text>Yes</Result1Text>
            <Result2Text>Abstain</Result2Text>
            <Result3Text>No</Result3Text>
            <Result4Text />
            <Result5Text>Total present</Result5Text>
            <Result6Text>Seats</Result6Text>
            <Result7Text />
            <Result8Text />
            <Result9Text />
            <Button1Text>Yes</Button1Text>
            <Button2Text>Abstain</Button2Text>
            <Button3Text>No</Button3Text>
            <Button4Text />
            <Button5Text />
            <VotingConclusion>Motion not approved

0 votes for the motion

0 votes neither for nor against the motion

1 votes against the motion (Delegate A ())

</VotingConclusion>
            <VotingComment />
          </VotingSession>
        </VotingSessions>
        <VotingResults>
          <VotingResult>
            <VotingSessionId>824</VotingSessionId>
            <DelegateId>54</DelegateId>
            <Present>1</Present>
            <Vote>1</Vote>
            <Weight>1</Weight>
          </VotingResult>
          <VotingResult>
            <VotingSessionId>825</VotingSessionId>
            <DelegateId>54</DelegateId>
            <Present>1</Present>
            <Vote>3</Vote>
            <Weight>1</Weight>
          </VotingResult>
```

```
        </VotingResults>
      </export>
    </SetConference>
  </s:Body>
</s:Envelope>
```

Example response:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Header>
    <ActivityId CorrelationId="b00f8f80-7618-43e7-90df-27eadc592432"
xmlns="http://schemas.microsoft.com/2004/09/ServiceModel/Diagnostics">00000000-0000-0000-0000-
000000000000</ActivityId>
  </s:Header>
  <s:Body>
    <SetConferenceResponse xmlns="http://dis.cc">
      <SetConferenceResult>23969</SetConferenceResult>
    </SetConferenceResponse>
  </s:Body>
</s:Envelope>
```

www.shure.com

United States, Canada, Latin America, Caribbean:
Shure Incorporated
5800 West Touhy Avenue
Niles, IL 60714-4608
USA

Phone: +1 847 600 2000
Fax: +1 847 600 1212 (USA)
Fax: +1 847 600 6446
Email: info@shure.com

Europe, Middle East, Africa:

Shure Europe Gmbh
Jakob-Dieffenbacher-Str. 12
75031 Eppingen
Germany

Phone: +49 (0) 7262-9249-100
Fax: +49 (0) 7262-9249-114
Email: info@shure.de

Asia, Pacific:

Shure Asia Limited
22/F, 625 King's Road
North Point, Island East,
Hong Kong

Phone: (+852) 2893-4290
Fax: (+852) 2893-4055
Email: info@shure.com.hk