AviaVox interface protocol

File name:	AviaVox – interface protocol.doc
Author:	Daniël van der Lei (AviaVox) and Merijn Bosma (AviaVox)
Date created:	January 10th 2013
Last updated:	March 23rd 2023
Version:	2.5

Revision history

Ver sion	Release date	Summary of changes	Released by
1.0	12/24/2013	Initial version	Daniel van der Lei
1.1	28/03/2017		Daniel van der Lei
1.2	14/09/2017		Daniel van der Lei
1.3	26/09/2017		Daniel van der Lei
1.4	29/09/2017		Daniel van der Lei
1.5	20/10/2017	- Fixed some minor typos in data protocols - Added languages node to Announcement Audio Request message.	Daniel van der Lei
1.6	03/03/2020	- Added additional ‘line’ layer in ‘Announcement List Response’ to make protocol better suitable for use with large and complex gate announcement screens - Updated ‘Announcement trigger request’ protocol to support complex gate announcement triggers - Document is in concept state and will need further finetuning based on actual projects using this protocol	Daniel van der Lei
1.7	24/06/2020	- Better indication of those messages that are not yet supported. - Now two scenarios are separately described (1. integrated solution 2. FlexiVox)	Daniel van der Lei
1.8	19/08/2020	- Fixed typo in example hashes	Daniel van der Lei
1.9	16/09/2020	- AnnouncementData node now uses Name / Data format	Daniel van der Lei
2.0	25/09/2020	- Added ReturnAnnouncementText option to AnnouncementAudioRequest message	Daniel van der Lei
2.1	17/05/2021	- Added AnnouncementTriggerRequestEx	Daniel van der Lei
2.2	25/06/2021	- Minor changes / typos	Daniel van der Lei
2.3	15/07/2021	- Changes in AnnouncementListRequest - Added chapter with specific scenario examples - Added AnnouncementAudioRequestEx message	Daniel van der Lei
2.4	01/10/2021	- Changes in AnnouncementListResponse. Now includes Text parts. - Changes in AnnouncementTriggerRequestEx. Now optionally includes contextual information about the announcement triggered. - Changes in AuthenticationResponse. Now optionally can include root node information.	Daniel van der Lei
2.5	23/03/2023	- AnnouncementTriggerRequest is now available within the API	Daniel van der Lei

1. Contents

2. Introduction

This document describes the *AviaVox interface protocol (AIP)* which allows 3rd party developers to setup an interface between their systems and the AviaVox Automatic Announcement system.

It provides a system overview, a description of the supported transport mechanism, an explanation of how to authenticate as well as a full description of all commands and instructions that can be exchanged.

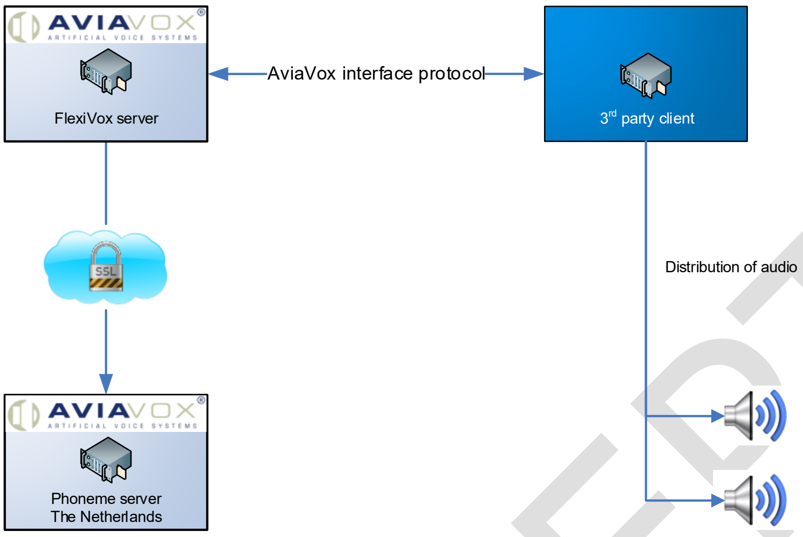
Lastly various scenario's are given containing all messages exchanged for a given setup.

<u>The full AIP specification is in concept state and not as such available for use in any production environment unless explicitly confirmed by AviaVox.</u>

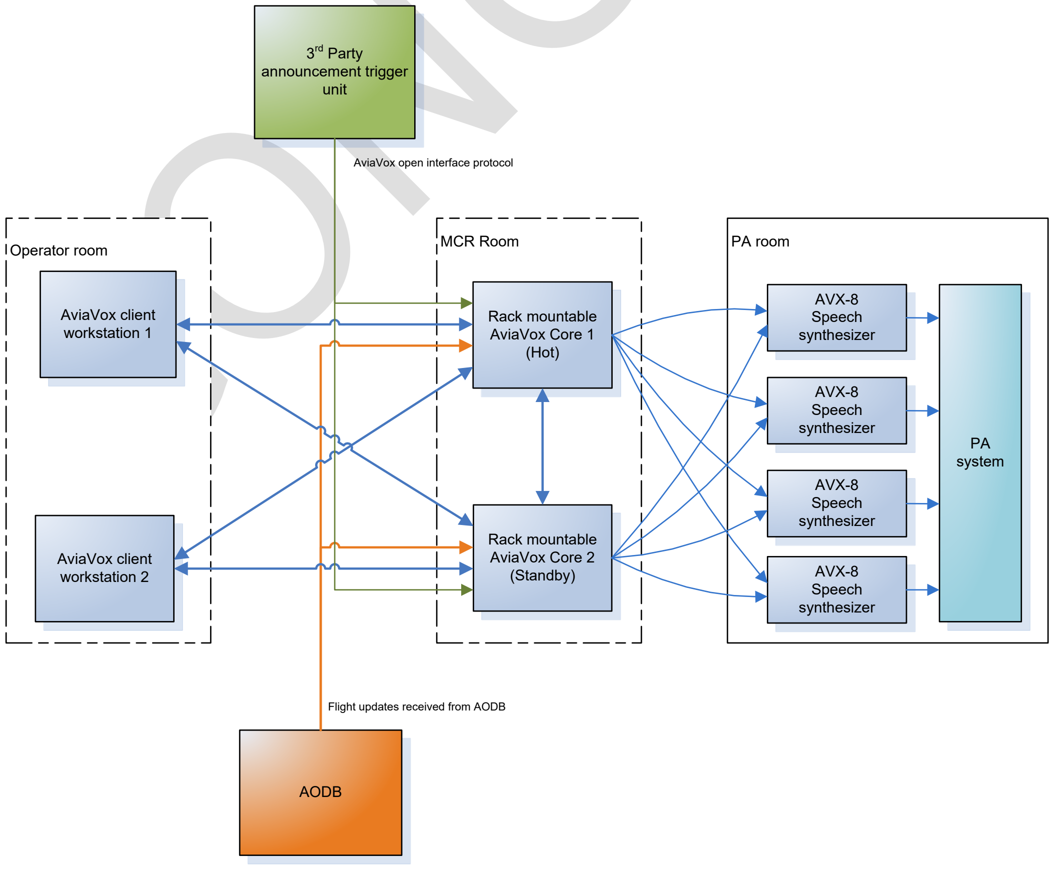
3. System overview

The AviaVox Interface protocol can be used in two different setups:

1. As a standalone system as part of the FlexiVox solution. The 3rd party client receives digital audio data from FlexiVox server and takes care of the distribution of this audio.



2. As an integrated solution combined with the AviaVox ATEC system. The AviaVox Cores are responsible for the generation and distribution of announcements as triggered by either the AODB, the AviaVox clients workstations or by a 3rd party announcement trigger unit (using the AIP protocol)



3rd Party announcement trigger unit
AviaVox open interface protocol
Operator room
AviaVox client workstation 1
AviaVox client workstation 2
MCR Room
Rack mountable AviaVox Core 1 (Hot)
Rack mountable AviaVox Core 2 (Standby)
PA room
AVX-8 Speech synthesizer
AVX-8 Speech synthesizer
AVX-8 Speech synthesizer
AVX-8 Speech synthesizer
PA system
Flight updates received from AODB
AODB

In the integrated solution drawing the various used components are described below:

1. The **AviaVox Core** is the central part of the Automatic Announcement System. The AviaVox Core is responsible for:

   - Retrieving flight information from external systems like an Integration Broker (IB) or Airport Opertional Database (AODB)

   - Interpreting and analysing incoming flight information to automatically generate announcements according to the airports announcement policy.

   - queuing and managing of announcements

   - Sending announcements to the corresponding AVX-8 speech synthesizer.

2. The **AVX-8 speech synthesizer** provides the interface to the PA system and performs the following tasks.

   - Synthesizing of speech

   - Providing audio to PA system

   - Selection of zones within the PA system

   - Reading out busy signals from the PA system.

3. The **AviaVox Client** provides the Graphical User Interface to the Automatic Announcement system and runs on a standard desktop PC. The GUI provides access to:

   - Monitoring of generated announcements

   - Flashback

   - Management of configuration

4. Transport mechanism

Connectivity is based on standard TCP socket connections where AviaVox is server and 3rd party unit is client.

In an integrated solution at any given time, only one AviaVox Core will be active. Only the active core will accept incoming TCP connections from external clients (Figure 4-1)

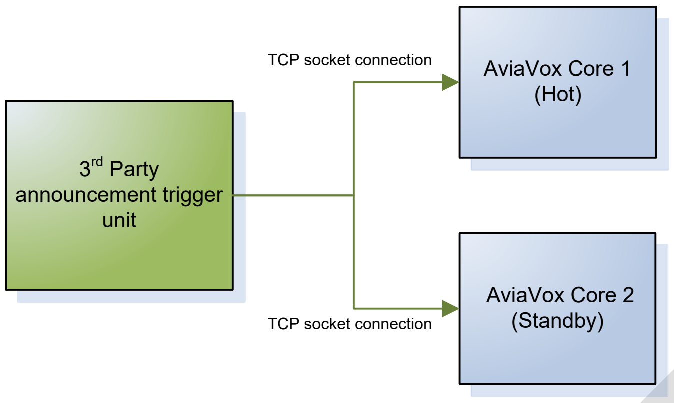


Figure 4-1

In a FlexiVox setup there can be multiple FlexiVox servers available where all servers will accept incoming TCP connections from external clients (Figure 4-2).

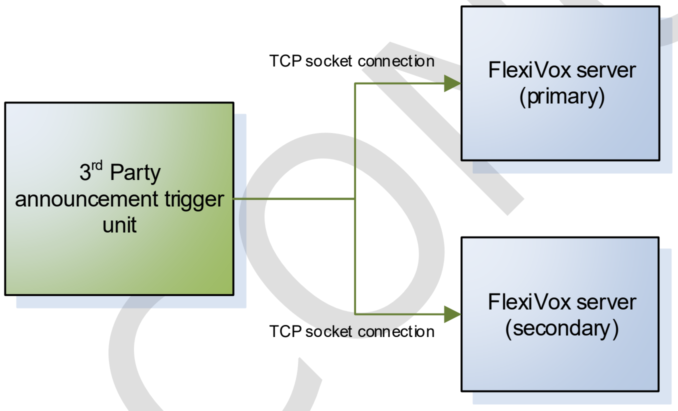

TCP socket connection
FlexiVox server (primary)
3rd Party announcement trigger unit
FlexiVox server (secondary)
TCP socket connection

Figure 4-2

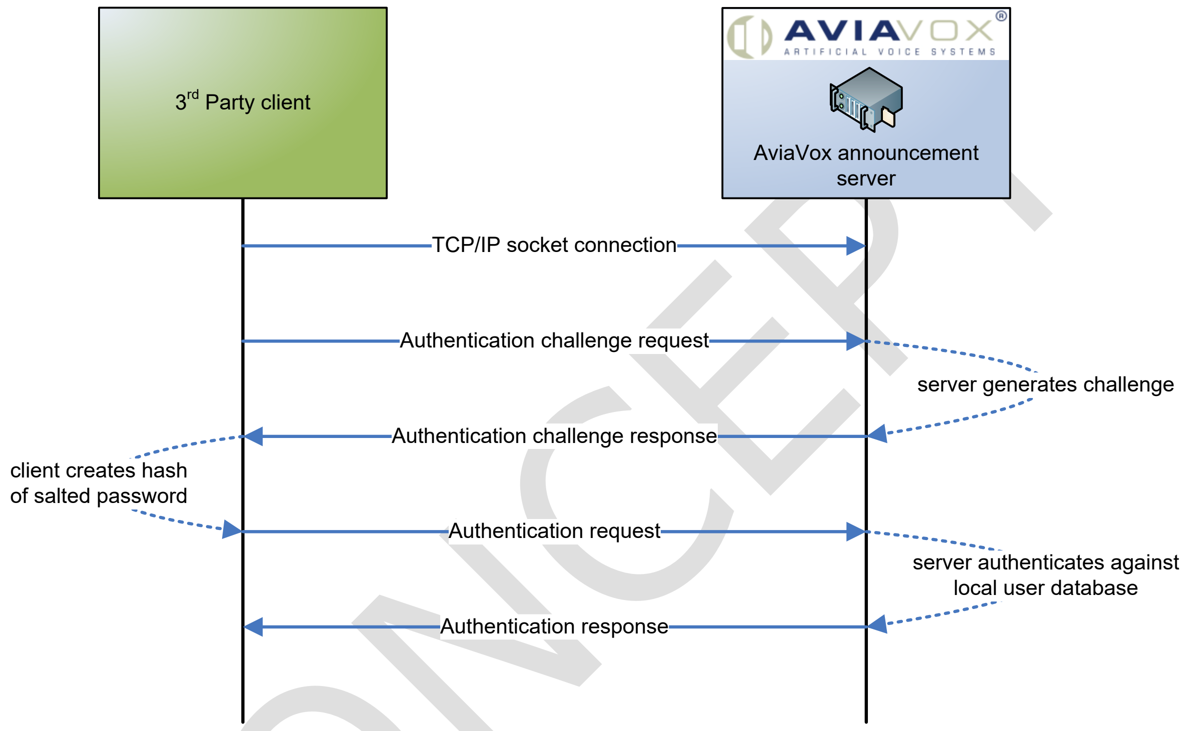
Note. The AIP can also be configured to work with existing / older architectures where Control units are used instead of Cores and Clients. In that case the 3rd party will have to connect directly to any of the Control Units. Only the *master Control Unit* will answer the TCP socket connection.

5. Authentication mechanism

Upon connection, the external client will need to authenticate itself. After successful authentication, the client can send instructions to the server to generate announcements.

Each client will be given a username and password combination. For security reasons, the authentication mechanism avoids sending over passwords in plain text. Instead, it is required for the client to first 'salt' the password and then calculate a hash which is sent to the server for authentication.

The flow of messages during authentication is described in Figure 5-1.



3rd Party client
AVIAVOX ARTIFICIAL VOICE SYSTEMS
AviaVox announcement server
TCP/IP socket connection
Authentication challenge request
server generates challenge
Authentication challenge response
client creates hash of salted password
Authentication request
server authenticates against local user database
Authentication response

Figure 5-1

1. Upon connection, the client will send the authentication challenge request message.
2. The server will respond with a randomly generated challenge code (decimal representation of a 16-bit unsigned integer)
3. The client will use the challenge code to salt the password and calculate a hash
4. The client will send the authentication request message containing the username and hash
5. The server will authenticate against local user database and send back an authentication response message.

5.1. Salt procedure

The password will be salted by making a string concatination containing:

- the password
- the length of the password XOR the challenge received from the server
- the reversed password

<u>Example 1:</u>

Challenge received from server = 5942
Password = test

```
The challenge written in binary:   0001 0111 0011 0110
The password length in binary:     0000 0000 0000 0100
A bitwise XOR will result in:      0001 0111 0011 0010
```

Converted to decimal this is 5938

So, if the password is "test", the data to be salted will be "test5938tset"

<u>Example 2:</u>

Challenge received from server = 61058
Password = strongpassword

```
The challenge written in binary:   1110 1110 1000 0010
The password length in binary:     0000 0000 0000 1110

A bitwise XOR will result in:      1110 1110 1000 1100
```

Converted to decimal this is 61068

If the password is "strongpassword", the data to be salted will be "strongpassword61068drowssapgnorts"

For more information about salting see the following link:

https://en.wikipedia.org/wiki/Salt_(cryptography)

5.2. Hashing procedure

The used hash mechanism is SHA512. The hash value will be sent over as ASCII text in uppercase.

In case of example 1 in chapter 5.1 the hash for “test5938tset” will be:

DED0D07EA35673FA201B26CDDFE4D52EEC5A2626EB12256332D882E20B2FBF93553B8ADCB6
6009D751E81EA9C1A76F5ADE7988C84637F2403364F369F185614F

In case of example 2 in chapter 5.1 the hash for “strongpassword61068drowssapgnorts” will be:

D1D39DAB0496DABE7283332BD0BF1BA253308B0FA59673924B9B0961F76A1BF4860D7543E8E
3B15E49C832B880491F9C69D8BAD4D4F366EB3D7FB7151923BB07

There are several websites available which can calculate these kind of hashes, which can be used as an example. One of them is:
http://passwordsgenerator.net/sha512-hash-generator/

For more information about SHA512 see the following link:

https://en.wikipedia.org/wiki/SHA-2

6. Data

The AviaVox Interface Protocol allows for several different messages to be exchanged between client and server. This chapter lists all messages and describes the exact protocol and field definitions. All data messages are using standard XML format and should be separated by a start of text marker (STX: 0x02) and an end of text marker (ETX: 0x03).

The type of message is always identified by the MessageID element.

Each request sent from a client to server can include an optional ClientID node holding a client generated ID string. When this field is added, any responding message received from the server will include the same ID which allows the client to match the sent request with the answer received.

The following messages are defined within the system:

From client to server:

- Authentication challenge request
- Authentication request
- Heartbeat request
- Announcement list request *(not yet implemented)*
- Announcement list request Extended *(not yet implemented)*
- Announcement trigger request *(not yet implemented)*
- Announcement audio request
- Announcement audio request Extended *(not yet implemented)*
-

From server to client

- Authentication challenge response
- Authentication response
- Heartbeat response
- Announcement list response *(not yet implemented)*
- Announcement audio response
- Error message

6.1. Authentication Challenge Request

6.1.1. Message details

Message Name/ ID	AuthenticationChallengeRequest
Description	This message is generated by client to start the authentication procedure. The server will respond with the message “AuthenticationChallengeResponse”
Trigger	A client should send this message after establishing a TCP connection.
Message handling	Server shall receive the message and respond with a “AuthenticationChallengeResponse” message which contains the challenge that should be used to salt the password.

6.1.2. Message elements

The below table lists all individual data elements within the message:

Message Element	**Description**	**Data Type**	**Mandatory Optional**	**(Example) Value**
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	AuthenticationChallengeRequest
ClientID	Client generated ID	string	O	1234567

6.1.3. Message examples

6.1.3.1. Authentication challenge request sample message:

```
<AIP>
  <MessageID>AuthenticationChallengeRequest</MessageID>
  <ClientID>1234567</ClientID>
</AIP>
```

6.2. *Authentication Challenge Response*

6.2.1. Message details

Message Name/ ID	AuthenticationChallengeResponse
Description	This message is generated by the server in response to an “AuthenticationChallengeRequest” message and will provide a challenge.
Trigger	Sent upon receiving “AuthenticationChallengeRequest” message from client
Message handling	Client shall receive the message and use the challenge to salt the password. A hash is calculated over the salted password and sent back to the server using the “AuthenticationRequest” message.

6.2.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AuthenticationChallengeResponse
ClientID	Client generated ID	string	O	1234567
MessageData	Data node	n/a	M	Contains elements: ○ Challenge
Challenge	Challenge number	integer	M	Number between 0 and 65535

6.2.3. Message example s

6.2.3.1. Authentication challenge response sample message

```
<AIP>
  <MessageID>AuthenticationChallengeResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <Challenge>2345</Challenge>
  </MessageData>
</AIP>
```

6.3. Authentication Request

6.3.1. Message details

Message Name/ ID	AuthenticationRequest
Description	This message is generated by client to authenticate against the server.
Trigger	A client should send this message after receiving an “AuthenticationChallengeResponse” message.
Message handling	Server shall receive the message and authenticate against user database. It will immediately respond with sending an “AuthenticationResponse" message.

6.3.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	AuthenticationRequest
ClientID	Client generated ID	string	O	1234567
MessageData	Data node	n/a	M	Contains elements: ○ Username ○ PasswordHash
Username	Username in plain text	string	M	admin
PasswordHash	Hash	string	M	Hash of salted password

6.3.3. Message examples

6.3.3.1. Authentication request sample message

```
<AIP>
  <MessageID>AuthenticationRequest</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <Username>admin</Username>
<PasswordHash>D1D39DAB0496DABE7283332BD0BF1BA253308B0FA59673924B9B0961F76A1BF4860D7543
E8E3B15E49C832B880491F9C69D8BAD4D4F366EB3D7FB71519 23BB07
</PasswordHash>
  </MessageData>
</AIP>
```

6.4. *Authentication response*

6.4.1. Message details

Message Name/ ID	AuthenticationResponse
Description	This message is generated by the server in response to an “AuthenticationRequest” message and will tell whether the client is authenticated or not
Trigger	Sent upon receiving “AuthenticationRequest” message from client

6.4.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AuthenticationResponse
ClientID	Client generated ID	string	O	1234567
MessageData	Data node	n/a	M	Contains elements: ○ Authenticated ○ RootNode
Authenticated	Element containing the authentication response code	boolean	M	0 = not authenticated 1 = authenticated
RootNode	For some use cases it is necessary to inform the client about a starting point on how to query the available announcements installed within the system. This element can provide such contextual details. Values provided can be used when sending an *Announcement List Request* message (see chapter 6.7), in order to query the announcements available for a given context.	string	O	Format: ID=Value1,Value2,Value3 MessageCategory=TUNNEL,TRAIN

6.4.3. Message examples

6.4.3.1. Authentication response sample message 1

User is not authenticated.

```
<AIP>
  <MessageID>AuthenticationResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <Authenticated>0</Authenticated>
 </MessageData>
</AIP>
```

6.4.3.2. Authentication response sample message 2

User is authenticated.

```
<AIP>
  <MessageID>AuthenticationResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <Authenticated>1</Authenticated>
  </MessageData>
</AIP>
```

6.4.3.3. Authentication response sample message 3

User is authenticated and additional root node information is given. When sending *AnnouncementListRequest* message we can use the conditions element to query all announcements for MessageCategory=TUNNEL, or MessageCategory=TRAIN.

```
<AIP>
  <MessageID>AuthenticationResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <Authenticated>1</Authenticated>
    <RootNode>MessageCategory=TUNNEL,TRAIN</RootNode>
  </MessageData>
</AIP>
```

6.5. *Heartbeat request message*

6.5.1. Message details

Message Name/ ID	HeartbeatRequest
Description	This message is generated by the client in order to monitor the connection. The server will respond with a HeartbeatResponse message.
Trigger	A client should send this message on regular interval basis in order to check if the connection is still healthy.
Message handling	Server shall receive the message and immediately respond with “HeartbeatResponse” message. When the server does not receive data (either a HeartBeatRequest or another MessageID) from a client for more than x seconds the connection will be dropped (where x is a configurable time in seconds).

6.5.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	HeartbeatRequest
ClientID	Client generated ID	Integer / string	O	1234567

6.5.3. Message examples

6.5.3.1. Heartbeat request sample message

```
<AIP>
  <MessageID>HeartbeatRequest</MessageID>
  <ClientID>1234567</ClientID>
</AIP>
```

6.6. Heartbeat response

6.6.1. Message details

Message Name/ ID	HeartbeatResponse
Description	This message is generated by the server and informs the client that the connection is still healthy
Trigger	Response to a “HeartbeatRequest” message.
Message handling	Client will check for the “HeartbeatResponse” message and will re-establish the connection as soon as this message is not received for more than x seconds. (where x should be a configurable value in seconds)

6.6.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	HeartbeatResponse
ClientID	Contains the client generated ID that was provided in the “HeartbeatRequest” message. If the request didn’t hold an ClientID element, this field will remain empty.	string	O	1234567

6.6.3. Message examples

6.6.3.1. Heartbeat response sample message

```
<AIP>
  <MessageID>HeartbeatResponse</MessageID>
  <ClientID>1234567</ClientID>
</AIP>
```

6.7. *Announcement list request (not yet implemented)*

6.7.1. Message details

Message Name/ ID	AnnouncementListRequest
Description	This message is generated by the client and requests the server for a listing of all announcements that may be remotely triggered by the client. This method can be used to create a dynamic GUI on the 3rd party client side containing a button for each announcement available.
Trigger	A client will send this message to the server in order to discover which announcements can be triggered.
Message handling	Server will check for "AnnouncementListRequest" messages and will respond with an "AnnouncementListResponse" message containing all announcements that can be generated.

6.7.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	AnnouncementListRequest
ClientID	Client generated ID	Integer / string	O	1234567
MessageData	Data node	n/a	M	Contains element: ○ Conditions
Conditions	Conditions element provide a way to request a listing for a certain specific context.	n/a	M	Contains one or multiple elements ○ Condition
Condition	Element containing a single condition	n/a	M	Contains attributes ○ ID ○ Value
ID	ID of the condition field	string	M	Examples: Airline Category
Value	Value used to check if the condition is true or false	string	M	Examples: KLM Tunnel Food

6.7.3. Message examples

6.7.3.1. Announcement list request sample message 1

A request to query all possible announcement triggers for

- Airline: KLM

```
<AIP>
  <MessageID>AnnouncementListRequest</MessageID>
  <ClientID>62589</ClientID>
  <MessageData>
    <Conditions>
      <Condition ID='Airline' Value='KLM'/>
    </Conditions>
  </MessageData>
</AIP>
```

The provided conditions node contains contextual information about the type of announcements that are queried. This information will be mirrored against the internal range of loaded announcement configurations. An announcement list response will be generated containing all possible announcements for the given context.

6.7.3.2. Announcement list request sample message 2

A request to query all possible announcement triggers for

- MessageCategory: Tunnel

```
<AIP>
  <MessageID>AnnouncementListRequest</MessageID>
  <ClientID>62589</ClientID>
  <MessageData>
    <Conditions>
      <Condition ID='MessageCategory' Value='Tunnel'/>
    </Conditions>
  </MessageData>
</AIP>
```

The provided *conditions* node contains contextual information about the type of announcements that are being queried. This information is internally mirrored against the set of loaded announcement configurations. An announcement list response will be generated containing all possible announcements for the given context.

The *root node* information optionally included in the *AuthenticationResponse* message can be used as a starting point. E.g. in case the *AuthenticationResponse* message includes the text: **MessageCategory=Tunnel,Train** this means an *AnnouncemenListRequest* can be sent for two different ID / Value pairs.

1. ID = MessageCategory Value = Tunnel
2. ID = MessageCategory Value = Train

Each request will result in a AnnouncementListResponse containing the possible announcements for that context.

Note.

Using a root node as a starting point is only applicable in case of a simple structure of announcements, like the example above where announcement sets are divided into multiple categories.
In other more complex cases root node information cannot be provided automatically due to the wide variety of combinations that can be used to query a set of announcements. In such case there must be a mutual agreement between both client and server on how announcement lists should be queried.

6.8. *Announcement list response (not yet implemented)*

6.8.1. Message details

Message Name/ ID	AnnouncementListResponse
Description	This message is generated by the server and informs the client which announcements may be triggered for a given airline, flight number and gate code
Trigger	Server will send this message as a response to a *AnnouncementListReques*" message If no flight was found in the AviaVox database for the given airline, flight number and gate the response will contain zero announcements

6.8.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node	n/a	M	Contains elements: ○ MessageID ○ ClientID
MessageID	The ID of the message	string	M	AnnouncementListResponse
ClientID	Contains the client generated ID that was provided in the *AnnouncementListRequest* message. If the request didn't hold an ID element, this field will remain empty.	string	O	1234567
MessageData	Data node	n/a	M	Contains elements: ○ Announcement (one or multiple)
Announcement	Node containing all information about one announcement	n/a	O	Contains attributes: ○ ID, Description Contains elements: ○ Line (one or multiple)
@ID	ID of the current announcement	string	M	BOR
@Description	Description of the current announcement.	string	M	pre-boarding business remaining pax
Line	Node containing all information about one line within the announcement	n/a	M	Contains attributes: ○ ID ○ Description ○ Mandatory ○ Default Contains elements: ○ Part (one or multiple) ○ Text (one or multiple)
@ID	ID of the current line	string	M	EB1 WX4 D05
@Description	Description of the current line	string	M	Salutation Welcome to flight Remain seated One item of hand luggage permitted

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
@Mandatory	Attribute defining whether the current line is a mandatory sentence within the announcement	boolean	M	1 = the line is mandatory 0 = the line is optional
@Default	Attribute defining whether the current line should be presented to the user as activated by default	boolean	M	1 = the line should be activated by default 0 = the line should not be activated by default
Part	Element containing a definition of a variable field required to be filled in through the client GUI.	n/a	O	Contains attributes: ○ ID ○ Description ○ Type ○ Mandatory ○ DefaultValue
@ID	Attribute defining the ID of the additionally required field.	string	M	MPS ZDI{2DI2}
@Description	Attribute defining a descriptive text for this field which can be shown within the GUI.	string	M	Number of missing passengers Airline Partner Pax 1 Pax 2

@Type	Attribute defining the current field type	string	M	numeric= numeric field
				string= alphanumeric field
				date = date field (YYYYMMDD)
				time = time field (HHmm)
				datetime = date time field (YYYYMMDDHHmm or ISO8601)

@Mandatory	Attribute defining whether the current field is mandatory	boolean	M	1 = mandatory (without this information the announcement cannot be generated) 0 = optional
@DefaultValue	Attribute defining the default value that should be shown within the GUI	string	M	KL PIC BCP OPP
Text	Element defining a fixed text part	n/a	O	Contains attributes: ○ Text ○ Function
@Text	Attribute defining the fixed text which should be shown within the GUI	string	M	Immediate boarding please at gate
@Function	Attribute defining a function which should be used to determine whether the text should be included (black) or excluded (greyed out).	string	O	If:APT{5BAS} Text part is included as long as the field APT{5BAS} has a value assigned. Text part is excluded when APT{5BAS} is empty. IfN:APT{5BAS} Inverse variant. Text part is included as long as the field APT{5BAS} is empty. Text part is excluded when APT{5BAS} is filled in.
Item	In case there is a fixed list of codes to choose from one or multiple "Item" nodes will hold the choices that can be made.	n/a	O	Contains attributes: ○ ID ○ Description

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
@ID	The ID of the current fixed choice option	string	M	WXS
@Description	The description of the current fixed choice option	string	M	Weather conditions

6.8.3. Message examples

6.8.3.1. Announcement list response sample message 1

```
<AIP>
 <MessageID>AnnouncementListResponse</MessageID>
 <ClientID>62589</ClientID>
 <MessageData>
  <Announcement ID='BOR' Description='Boarding call'>
   <Line ID='D04' Description='[airline] flight [xxx] to [yyy] now boarding please at gate [z]' Mandatory='1' Default='1'>
    <Text Text='Passengers for'/>
    <Part ID='ALN{1BAS}' Description='airline code' Type='string' Mandatory='1' DefaultValue='KL'/>
    <Text Text='flight'/>
    <Part ID='FNB{1BAS}' Description='flight number' Type='string' Mandatory='1' DefaultValue='123'/>
    <Text Text='to'/>
    <Part ID='APT{1BAS}' Description='airport code' Type='string' Mandatory='1' DefaultValue='BCN'/>
    <Text Text='immediate boarding please at gate '/>
    <Part ID='GAT{1BAS}' Description='gate code' Type='alphanumeric' Mandatory='1' DefaultValue='1'/>
   </Line>
  </Announcement>

  <Announcement ID='MPS' Description='Missing passengers'>
   <Line ID='D06' Description='We ask [x] missing passengers for [airline] flight [xxx] to [xxx] to proceed for boarding
through gate [5]' Mandatory='1' Default='1'>
    <Text Text='We ask'/>
    <Part ID='MPS{1BAS}' Description='Nr of missing pax' Type='numeric' Mandatory='1' DefaultValue=''/>
    <Text Text='missing passengers for'/>
    <Part ID='ALN{1BAS}' Description='airline code' Type='string' Mandatory='1' DefaultValue='KL'/>
    <Text Text='flight'/>
    <Part ID='FNB{1BAS}' Description='flight number' Type='string' Mandatory='1' DefaultValue='123'/>
    <Text Text='to'/>
    <Part ID='APT{1BAS}' Description='airport code' Type='string' Mandatory='1' DefaultValue='BCN'/>
    <Text Text='to proceed for boarding through gate'/>
    <Part ID='GAT{1BAS}' Description='gate code' Type='alphanumeric' Mandatory='1' DefaultValue='1'/>
   </Line>
  </Announcement>

  <Announcement ID='DLY' Description='Delay call'>
   <Line ID='C32' Description='We regret to announce that [airline] flight [xxx] to [yyy] will be delayed until [hh:mm] [due to
reason] in [zzz]' Mandatory='1' Default='1'>
    <Text Text='We regret to announce that'/>
    <Part ID='ALN{1BAS}' Description='airline code' Type='string' Mandatory='1' DefaultValue='KL'/>
    <Text Text='flight'/>
    <Part ID='FNB{1BAS}' Description='flight number' Type='string' Mandatory='1' DefaultValue='123'/>
    <Text Text='to'/>
    <Part ID='APT{1BAS}' Description='airport code' Type='string' Mandatory='1' DefaultValue='BCN'/>
    <Text Text='will be delayed until'/>
    <Part ID='TIM{1BAS}' Description='Est time of departure' Type='datetime' Mandatory='1' DefaultValue=''/>
    <Part ID='RSN{5BAS}' Description='Reason of delay' Type='string' Mandatory='0' DefaultValue='' >
     <Item ID='WXS' Description='weather conditions'/>
     <Item ID='TEC' Description='technical reasons'/>
    </Part>
    <Text Text='in' Function='lf:APT{5BAS}' />
    <Part ID='APT{5BAS}'  Description='delay location' Type='string' Mandatory='0' DefaultValue=''/>
   </Line>
  </Announcement>
 </MessageData>
</AIP>
```

The sample message describes 3 possible announcement triggers:

1. A button with label ‘Boarding call’ which triggers the ‘BOR’ announcement.
   Corresponding message trigger to generate this announcement: Announcement Trigger Request Extended sample message 1

2. A button with label ‘Missing passengers’ which triggers the ‘MPS’ announcement. For this announcement information about the number of missing passengers is required. This must be entered via the client GUI through a numeric input field with label ‘Nr of missing pax’
   Corresponding message trigger to generate this announcement: Announcement Trigger Request Extended sample message 2

3. A button with label ‘Delay call’ which triggers the ‘DLY’ announcement. For this announcement, additional information about the time of departure is required. This must be entered via the client GUI through a date/time input field with label ‘Est.time of departure’. An optional reason code can be provided which is provided through a fixed choice list.

6.9. *Announcement trigger request*

6.9.1. Message details

Message Name/ ID	AnnouncementTriggerRequest
Description	This message is generated by the client to instruct the server to generate an announcement and then queue it within the AviaVox system. Only basic meta data is included within the message like 'MessageName', 'Airline', 'FlightNumber', 'Destination' and 'Gate'. When using the *AnnouncementTriggerRequest* message, the required *IDs* and *values* are based on a mutual agreement between 3rd party client and AviaVox. In case a more dynamic interface is required, the *AnnouncementListRequest* in combination with the *AnnouncementTriggerRequestEx* should be used
Trigger	Whenever a client wants to generate an announcement.
Message handling	Server shall receive the message and will attempt to generate the announcement based on the given parameters. If the Feedback field is set to '1' the server will send status updates for this announcement (not yet supported).
Remarks	Every announcement triggered will be announced immediately as soon as the PA system is ready and all required zones are free.

6.9.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AnnouncementTriggerRequest
ClientID	Client generated ID	Integer / string	O	1234567
Feedback	When this field is set, the server will send feedback to the client about activity for given announcement.	boolean	O	0 / 1 *Feedback is not supported. This field is defined here for future purposes.*
MessageData	Data node	n/a	M	Contains elements: ○ AnnouncementData
AnnouncementData	Data node	n/a	M	Contains elements: ○ Announcement The actual number of required elements and their names may vary based on the type of announcement that was selected. As a minimum it must contain all parts from the received announcement listing which are flagged as mandatory
Item	Node containing the details of a single item	n/a	M	Contains attributes: ○ ID ○ Value

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
@ID	ID of the current item	string	M	MessageName Airline
@Value	Value of the current item	string	M	BOR KLM

6.9.3. Message examples

6.9.3.1. Announcement trigger request sample message 1

An example of a ‘boarding’ call request

```
<AIP>
 <MessageID>AnnouncementTriggerRequest</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
  <AnnouncementData>
     <Item ID='MessageName'  Value='BOR'/>
     <Item ID='Airline'  Value='KLM'/>
     <Item ID='FlightNumber'  Value='1234'/>
     <Item ID='Route' Value='BCN'/>
     <Item ID='Gate'  Value='12'/>
  </AnnouncementData>
 </MessageData>
</AIP>
```

6.9.3.2. Announcement trigger request sample message 2

An example of a ‘missing passenger’ call request:

```
<AIP>
 <MessageID>AnnouncementTriggerRequest</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
  <AnnouncementData>
     <Item ID='MessageName'  Value='MPS'/>
     <Item ID='Airline'  Value='KLM'/>
     <Item ID='FlightNumber'  Value='1234'/>
     <Item ID='Route' Value='BCN'/>
     <Item ID='Gate'  Value='12'/>
     <Item ID='MissingPassengers'  Value='5'/>
  </AnnouncementData>
 </MessageData>
</AIP>
```

6.10. Announcement trigger request Extented (not yet implemented)

6.10.1. Message details

Message Name/ ID	AnnouncementTriggerRequestEx
Description	This message is generated by the client to instruct the server to generate an announcement and then queue it within the AviaVox system. The extended announcement trigger request can used by clients who fully dynamically build their own GUI based on the response to a *AnnouncementListRequest* message. This approach results in maximum control of announcement details, but with the added responsibility to include all these details within this trigger request. For clients that do not dynamically build their GUI it is recommended to use the normal *AnnouncementTriggerRequest* message.
Trigger	Whenever a client wants to generate an announcement.
Message handling	Server shall receive the message and will attempt to generate the announcement based on the given parameters. If the Feedback field is set to ‘1’ the server will send status updates for this announcement (not supported).
Remarks	Every announcement triggered will be announced immediately as soon as the PA system is ready and all required zones are free.

6.10.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AnnouncementTriggerRequestEx
ClientID	Client generated ID	Integer / string	O	1234567
Feedback	When this field is set, the server will send feedback to the client about activity for given announcement.	boolean	O	0 / 1 *Feedback is not supported. This field is defined here for future purposes.*
MessageData	Data node	n/a	M	Contains elements: ○ Conditions ○ AnnouncementData
Conditions	Element containing the contextual information for this trigger request. This should be the same information as was used to query the list of announcements using the *AnnouncementListRequest*	n/a	O	Contains elements: ○ Condition

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
	message and is needed to validate that current logged in user is authorized to generate the given type of announcement. See examples in chapter 6.14			
Condition	Element containing a single condition for this announcement	n/a	O	Contains attributes: ○ ID
@ID	Attribute containing the ID of current condition	string	M	MessageCategory Airline
Value	Attribute containing the value of current condition	String	M	Tunnel KLM
AnnouncementData	Data node	n/a	M	Contains elements: ○ Announcement The actual number of required elements and their names may vary based on the type of announcement that was selected. It must contain all parts from the received announcement listing which are flagged as mandatory
Announcement	Node containing the details of the announcement to be generated	n/a	M	Contains attributes: ○ ID Contains elements: ○ Line (optional, one or multiple)
@ID	ID of the announcement	string	M	PRE
Line	Node containing the details of a single sentence	n/a	M	Contains attributes: ○ ID Contains elements: ○ Part (optional, one or multiple)
@ID	ID of the current line	string	M	G03
Part	Node containing the details of a single part	n/a	M	Contains attributes: ○ ID ○ Value

@ID	ID of the current part	string	M	ALN{1BAS}
				ZCO{1BAS}

@Value	Value of the current part	string	M	KLM OPB

6.10.3. Message examples

6.10.3.1. Announcement Trigger Request Extended sample message 1

Example of a ‘Boarding call’ request message (corresponding with the *Announcement list response sample message 1*)

```
<AIP>
 <MessageID>AnnouncementTriggerRequestEx</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
   <AnnouncementData>

     <Conditions>
       <Condition ID='Airline' Value='KLM'/>
     </Conditions>

     <Announcement ID='BOR'>
       <Line ID='D04'>
         <Part  ID='ALN{1BAS}'  Value='KLM'/>
         <Part  ID='FNB{1BAS}'  Value='123'/>
         <Part  ID='APT{1BAS}'  Value='BCN'/>
         <Part  ID='GAT{1BAS}'  Value='12'/>
       </Line>
     </Announcement>
   </AnnouncementData>
 </MessageData>
</AIP>
```

6.10.3.2. Announcement Trigger Request Extended sample message 2

Example of a ‘missing passenger’ request message *(corresponding with the Announcement list response sample message 1)*Announcement list response sample message 1

```
<AIP>
 <MessageID>AnnouncementTriggerRequestEx</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
   <AnnouncementData>

     <Conditions>
       <Condition ID='Airline' Value='KLM'/>
     </Conditions>

     <Announcement ID='MPS'>
       <Line ID='D06'>
         <Part  ID='ALN{1BAS}'  Value='KLM'/>
         <Part  ID='FNB{1BAS}'  Value='123'/>
         <Part  ID='APT{1BAS}'  Value='BCN'/>
         <Part  ID='GAT{1BAS}'  Value='12'/>
         <Part  ID='MPS{1BAS}'  Value='5'/>
       </Line>
     </Announcement>
   </AnnouncementData>
 </MessageData>
</AIP>
```

6.11. *Announcement audio request*

6.11.1. Message details

Message Name/ ID	AnnouncementAudioRequest
Description	This message is generated by the client to request the server to generate an announcement and send back the audio in the given digital format. The announcement data included in this request is based on a pre-agreed set of ID / values. For a 100% dynamic implementation use *AnnouncementAudioRequestEx* instead, in combination with the *AnnouncementListRequest* message.
Trigger	Whenever a client wants to generate an announcement.
Message handling	Server shall receive the message and will attempt to generate the announcement based on the given parameters. Server will respond with an “AnnouncementAudioResponse” message.

6.11.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AnnouncementAudioRequest
ClientID	Client generated ID	Integer / string	O	1234567
MessageData	Data node		M	Contains elements: ○ EncodingFormat ○ BitDepth ○ SampleRate ○ BitRate ○ ReturnAnnouncementText ○ AnnouncementData
EncodingFormat	Type of audio encoding	string	M	PCM, OGG or MP3
BitDepth	Bit depth of the audio returned	integer	O	Values 16 or 24 Only required in case of AudioEncodingFormat = PCM
SampleRate	Sample rate of the audio returned	integer	O	Values 44100 or 48000 Only required in case of AudioEncodingFormat = PCM
BitRate	Bit rate of the audio returned	integer	O	Values 64000, 80000 Only required in case of AudioEncodingFormat MP3 or OGG
ReturnAnnouncementText	Return announcement text	Boolean	O	1 – response will contain announcement text data

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AnnouncementData			M	Contains elements: ○ Item The actual number of item elements may vary based on the type of announcement that was selected. Required items are based on a pre-agreed set of ID / values.
Item	Item node containing Name Data pairs for all announcement related data			Contains attributes: ○ ID ○ Value
@ID	ID attribute	string	M	MessageName Airlines Flightnumber Route GateCode ScheduledTime PublicTime Languages
@Value	Value attribute	string	M	See below examples (marked with $) of data format for various 'IDs'
$MessageName	Announcement ID	string		BOR, FIN, DLY, LAN
$Airlines	Airline code	string		The IATA or ICAO code of the airline. (max 3 characters) Eg: KLM In case of code-shared flight codes should be comma separated: Eg: KLM,BAW,EZY
$Flightnumber	Flight number	string		The number of the flight (max 4 numbers) Eg: 5346 In case of code-shared flight numbers are comma separated: Eg: 5346,4235,6454
$Route	Destination / Origin	string		The destination / origin 3 char IATA code of the flight. EG: BCN In case of multiple destinations codes should be comma separated and in order of the flight. EG: AMS,LHR,JFK
$GateCode	Gate code assigned to the flight	string		The gate code where the flight is due to depart. EG: 24 or A12 or 40A
$ScheduledTime	Scheduled time of the flight	Date/time		Date time using either format yyyymmddhhnn (where n stands for minute) or ISO 8601 standard Eg: 201708171247 2017-08-17T12:47+0200
$PublicTime	Public estimated time	Date/time		Date time using either format yyyymmddhhnn (where n stands for minute) or ISO 8601 standard Eg: 201708171247 2017-08-17T12:47+0200
$Languages	The languages in which the announcement is requested	string		Two-character language codes separated by commas. Eg: It,En

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
				When this field is empy the default language(s) as configured on FlexiVox side will be chosen.

6.11.3. Message examples

6.11.3.1. Announcement Audio Request sample message 1

An example of a ‘No smoking call’ request message in Italian and English where the audio is returned in PCM 16 bit 44.1kHz format:

```
<AIP>
        <MessageID>AnnouncementAudioRequest</MessageID>
        <ClientID>1234567</ClientID>
        <MessageData>
                <EncodingFormat>PCM</EncodingFormat>
                <BitDepth>16</BitDepth>
                <SampleRate>44100</SampleRate>
                <ReturnAnnouncementText>0</ReturnAnnouncemenText>

                <AnnouncementData>
                        <Item ID="MessageName"  Value="NSM"/>
                        <Item ID="Languages"  Value="It,En"/>
                </AnnouncementData>
        </MessageData>
</AIP>
```

6.11.3.2. Announcement Audio Request sample message 2

An example of a ‘final call’ request message where the audio is returned in OGG Vorbis format with bit rate 80 kbps. The response will include announcement texts.

```
<AIP>
        <MessageID>AnnouncementAudioRequest</MessageID>
        <ClientID>2</ClientID>
        <MessageData>
                <EncodingFormat>OGG</EncodingFormat>
                <BitRate>80000</BitRate>
                <ReturnAnnouncementText>1</ReturnAnnouncemenText>
                <AnnouncementData>
                        <Item ID="MessageName"  Value="FIN"/>
                        <Item ID="Airlines"   Value="BA"/>
                        <Item ID="FlightNumber" Value="124"/>
                        <Item ID="Route"  Value="DTM"/>
                        <Item ID="GateCode"  Value="5"/>
                        <Item ID="ScheduledTime"  Value="2020-09-17T12:47+0200"/>
                </AnnouncementData>
        </MessageData>
</AIP>
```

6.12. *Announcement audio request Extented (not yet implemented)*

6.12.1. Message details

Message Name/ ID	AnnouncementAudioRequestEx
Description	This message is generated by the client to request the server to generate an announcement and send back the audio in the given digital format The extended announcement audio request can used by clients who fully dynamically build their own GUI based on the response to a AnnouncementListRequest message. This approach results in maximum control of announcement details, but with the added responsibility to include all details in the resulting request. For clients that do not dynamically build their GUI it is recommended to use the normal *announcement audio request* message.
Trigger	Whenever a client wants to generate an announcement.
Message handling	Server shall receive the message and will attempt to generate the announcement based on the given parameters. Server will respond with an “AnnouncementAudioResponse” message.

6.12.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AnnouncementAudioRequest
ClientID	Client generated ID	Integer / string	O	1234567
MessageData	Data node		M	Contains elements: ○ EncodingFormat ○ BitDepth ○ SampleRate ○ BitRate ○ ReturnAnnouncementText ○ Conditions ○ AnnouncementData
EncodingFormat	Type of audio encoding	string	M	PCM, OGG or MP3
BitDepth	Bit depth of the audio returned	integer	O	Values 16 or 24 Only required in case of AudioEncodingFormat = PCM
SampleRate	Sample rate of the audio returned	integer	O	Values 44100 or 48000 Only required in case of AudioEncodingFormat = PCM

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
BitRate	Bit rate of the audio returned	integer	O	Values 64000, 80000 Only required in case of AudioEncodingFormat MP3 or OGG
ReturnAnnouncemen Text	Return announcement text	Boolean	O	1 – response will contain announcement text data
Conditions	Element containing the contextual information for this trigger request. This should be the same information as was used to query the list of announcements using the *AnnouncementListRequest* message and is needed to validate that current logged in user is authorized to generate the given type of announcement. See examples in chapter 6.14	n/a	O	Contains elements: ○ Condition
Condition	Element containing a single condition for this announcement	n/a	O	Contains attributes: ○ ID
@ID	Attribute containing the ID of current condition	string	M	MessageCategory Airline
Value	Attribute containing the value of current condition	String	M	Tunnel KLM
AnnouncementData	Data node	n/a	M	Contains elements: ○ Announcement The actual number of required elements and their names may vary based on the type of announcement that was selected. It must include all parts from the received announcement listing which were flagged as mandatory. Additionally it may include other parts which were flagged as optional.
Announcement	Node containing the details of the announcement to be generated	n/a	M	Contains attributes: ○ ID Contains elements: ○ Line (optional, one or multiple)
@ID	ID of the announcement	string	M	PRE
Line	Node containing the details of a single sentence	n/a	M	Contains attributes: ○ ID Contains elements: ○ Part (optional, one or multiple)

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
@ID	ID of the current line	string	M	G03
Part	Node containing the details of a single part	n/a	M	Contains attributes: ○ ID ○ Value

@ID	ID of the current part	string	M	ALN{1BAS}
				ZCO{1BAS}

@Value	Value of the current part	string	M	KLM OPB

6.12.3. Message examples

6.12.3.1. Announcement Audio Request Extended Sample message 1

An example of a ‘pre boarding announcement’ request message where the audio will be returned in OGG Vorbis 80kb/s format.

Server will respond with an Announcement audio response message containing the digital encoded audio.

```
<AIP>
  <MessageID>AnnouncementAudioRequestEx</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <EncodingFormat>OGG</EncodingFormat>
    <BitRate>80000</BitRate>

    <AnnouncementData>

      <Conditions>
        <Condition ID='Airline' Value='KLM'/>
      </Conditions>

      <Announcement ID='PRE'>
        <Line ID='WX4'/>
        <Line ID='G03'>
          <Part  ID='ALN{1BAS}'  Value='KLM'/>
          <Part  ID='FNB{1BAS}'  Value='123'/>
          <Part  ID='APT{1BAS}'  Value='BCN'/>
          <Part  ID='ZRO{1BAS}'  Value='OPB'/>
          <Part  ID='ZCO{1BAS}'  Value='STM'/>
        </Line>
        <Line ID='EB1'>
          <Part  ID='ZPX{1EB1}'  Value='BCP'/>
          <Part  ID='GAT{1EB1}'  Value='D15'/>
        </Line>
        <Line ID='WS1'>
          <Part  ID='ALN{1WS1}'  Value='KLM'/>
          <Part  ID='ZBN{1WS1}'  Value='SKT'/>
        </Line>
      </Announcement>
    </AnnouncementData>

  </MessageData>
</AIP>
```

6.13. Announcement audio response

6.13.1. Message details

Message Name/ ID	AnnouncementAudioResponse
Description	This message is generated by the server in response to an “AnnouncementAudioRequest” message and contains the audio in the given digital format
Trigger	Server will send this message as a response to a “AnnouncementAudioResponse” message
Message handling	Server shall receive the message and will attempt to generate the announcement based on the given parameters. Server will respond with an “AnnouncementAudioResponse” message.

6.13.2. Message elements

The below table lists all individual data elements within the message:

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	AnnouncementRequest
ClientID	Contains the client generated ID that was provided in the “AnnouncementAduioRequest” message. If the request didn’t hold an ID element, this field will remain empty.	string	O	1234567
MessageData	Data node		M	Contains elements: ○ EncodingFormat ○ BitDepth ○ SampleRate ○ BitRate ○ AnnouncementText ○ AnnouncementAudio
EncodingFormat	The format in which the audio is encoded		M	PCM, OGG or MP3
BitDepth	The bit depth in which the audio is encoded		O	Only filled in in case of AudioEncodingFormat = PCM. Values 16 or 24
SampleRate	Sample rate in which the audio is encoded		O	Only filled in in case of PCM Values 44100 or 48000
BitRate	Bit rate in which the audio is encoded		O	Only filled in in case of AudioEncodingFormat MP3 or OGG. Values 64000, 80000

Message Element	Description	Data Type	Mandatory Optional	(Example) Value
AnnouncementText	Node containing the texts of each language		O	Contains one or multiple elements ○ Text
Text	Node containing the announcement text of a single language		O	Contains attributes ○ Language ○ Text
@Language	Attribute containing the current language code	string	M	En Au Gr
@Text	Attribute containing the text for current language	string	M	Attention please, now boarding at gate 5
AnnouncementAudio	Base64 encoded digital audio	string	O	

6.13.3. Message examples

Example of an audio response message containing PCM 16 bit 44.1kHz formatted audio:

```
<AIP>
  <MessageID>AnnouncementAudioResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <EncodingFormat>PCM</EncodingFormat>
    <BitDepth>16</BitDepth>
    <SampleRate>44100</SampleRate>
    <AnnouncementAudio>Base64 encoded PCM audio</AnnouncementAudio>
    <AnnouncementText>
      <Text Language='Au' Text='Attention please. Passengers  now boarding at gate 5'/>
    </AnnouncementText>
  </MessageData>
</AIP>
```

Example of an audio response message containing OGG Vorbis 80kb/s formatted audio:

```
<AIP>
  <MessageID>AnnouncementAudioResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <EncodingFormat>OGG</EncodingFormat>
    <BitDepth></BitDepth>
    <SampleRate></SampleRate>
    <BitRate>80000</BitRate>
    <AnnouncementAudio> Base64 encoded OGG audio </AnnouncementAudio>
    <AnnouncementText>
      <Text Language='Au' Text='Attention please. Passengers  now boarding at gate 5'/>
    </AnnouncementText>
  </MessageData>
</AIP>
```

6.14. Dynamic scenarios / examples

Dynamic scenarios are used in setups where the client doesn't know upfront which announcements can be generated. Client sends a requests for an announcement listing for a certain given set of criteria. Server will search for all announcements that match the given criteria and send back an announcement list response. Based on the received response, the client will dynamically setup its GUI and the end user will be able to start generating announcements based on the dynamically generated GUI.

6.14.1. Tunnel announcements requested as digital encoded audio

Step 1:

Client sends *Announcement List Request* to server and asks for all announcements that match the given criteria (MessageCategory = Tunnel)

```
<AIP>
  <MessageID>AnnouncementListRequest</MessageID>
  <ClientID>62589</ClientID>
  <MessageData>
    <Conditions>
      <Condition ID='MessageCategory' Value='Tunnel'/>
    </Conditions>
  </MessageData>
</AIP>
```

The provided conditions will be mirrored against the internal configuration and an announcement list response will be generated containing all possible announcements for given criteria.

Step 2:

Server responds with *Announcement List Response* message and informs the client which announcements are allowed to be generated within the given criteria.

```
<AIP>
 <MessageID>AnnouncementListResponse</MessageID>
 <ClientID>53489</ClientID>

 <MessageData>
  <Announcement ID='TUNNEL_DIVERSION' Description='Tunnel diversion announcement'>
   <Line ID='FVTN1' Description='main sentence' Mandatory='1' Default='1'>
    <Text Text='Passenger cars'/>
    <Text Text='are advised to take'/>
  <Part ID='XXT{1FVTN1} ' Description='Advised direction for passenger cars' Type='string' Mandatory='1'
DefaultValue='EX1'>
      <Item ID='EX1' Description='exit 1'/>
      <Item ID='EX2' Description='exit 2'/>
      <Item ID='EX3' Description='exit 3'/>
      <Item ID='KRE' Description='the Kings road exit'/>
      <Item ID='NRE' Description='the North road exit'/>
     </Part>
    </Line>
  </Announcement>
 </MessageData>
</AIP>
```

This server response describes one announcement with ID: ‘TUNNEL_DIVERSION’ which contains 1 line FVTN1. This line has one additional part XXT{1FVTN1} which requires input from the user. Possible options are EX1, EX2, EX3, KRE and NRE.

Step 3:

Client sends an *Announcement Audio Request Extended* message, requesting the digital encoded audio for a TUNNEL_DIVERSION message. The user chose to use value EX3 (exit 3) for the ‘advised direction’ part. The contextual information used in step 1 is included again so that the server can validate that current logged in user is authorized to generate this announcement.

```
<AIP>
  <MessageID>AnnouncementAudioRequestEx</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <EncodingFormat>OGG</EncodingFormat>
    <BitRate>80000</BitRate>

    <Conditions>
      <Condition ID='MessageCategory' Value='Tunnel'/>
    </Conditions>

    <AnnouncementData>

      <Announcement ID='TUNNEL_DIVERSION'>
        <Line ID='FVTN1'>
          <Part  ID='XXT{1FVTN1}'  Value='EX3'/>
        </Line>
      </Announcement>

    </AnnouncementData>
  </MessageData>
</AIP>
```

Step 4:

Server sends an *Announcement Audio Response* with digital encoded audio containing the TUNNEL_DIVERSION announcement

```
<AIP>
  <MessageID>AnnouncementAudioResponse</MessageID>
  <ClientID>1234567</ClientID>
  <MessageData>
    <EncodingFormat>OGG</EncodingFormat>
    <BitDepth></BitDepth>
    <SampleRate></SampleRate>
    <BitRate>80000</BitRate>
    <AnnouncementAudio> Base64 encoded OGG audio </AnnouncementAudio>
    <AnnouncementText>
      <Text Language='Au' Text=Passenger cars are advised to take exit 3'/>
    </AnnouncementText>
  </MessageData>
</AIP>
```

6.14.2. Airline announcements triggered to be played within the AviaVox system.

Step 1:

Client sends *Announcement List Request* to server and asks for all announcements that match the given criteria (Airline = KLM)

```
<AIP>
 <MessageID>AnnouncementListRequest</MessageID>
 <ClientID>62589</ClientID>
 <MessageData>
   <Conditions>
     <Condition ID='Airline' Value='KLM'/>
   </Conditions>
 </MessageData>
</AIP>
```

The provided conditions will be mirrored against the internal configuration and an announcement list response will be generated containing all possible announcements for given conditions.

Step 2:

Server responds with *Announcement List Response* message and informs the client which announcements are allowed to be generated within the given criteria (Airline = KLM).

```
<AIP>
 <MessageID>AnnouncementListResponse</MessageID>
 <ClientID>53489</ClientID>

 <MessageData>
  <Announcement ID='PRE' Description='pre-boarding'>
   <Line ID='WX4'  Description='salutation' Mandatory='1' Default='1'>
    <Text Text='Good morning - afternoon - evening'/>
   </Line>
   <Line ID='G03' Description='welcome to flight' Mandatory='1' Default='1'>
    <Text Text='Welcome  to'/>
    <Part ID='ALN{1BAS}'  Description='airline code' Type='string' Mandatory='1' DefaultValue='KL'/>
    <Text Text='flight'/>
    <Part ID='FNB{1BAS}'  Description='flight number' Type='string' Mandatory='1' DefaultValue='123'/>
    <Text Text='to'/>
    <Part ID='APT{1BAS}'  Description='airport code' Type='string' Mandatory='1' DefaultValue='BCN'/>
    <Part ID='ZRO{1BAS}'  Description='cooperation' Type='string' Mandatory='0' DefaultValue=''>
     <Item ID='OCW'  Description='operating in cooperation with'/>
     <Item ID='OPB'  Description='operated by'/>
    </Part>
    <Part ID='ZCO{1BAS}'  Description='partnership' Type='string' Mandatory='0' DefaultValue=''>
     <Item ID='OCW'  Description='Oneworld member(s)'/>
     <Item ID='OPB'  Description='Star Alliance Member(s)'/>
     <Item ID='STM'  Description='Skyteam member(s)'/>
    </Part>
    <Part ID='ALN{2BAS}'  Description='partnership airline' Type='string' Mandatory='0' DefaultValue=''/>
   </Line>
   <Line ID='RS1'  Description='remain seated' Mandatory='0' Default='1'>
    <Text Text='Please  remain  seated  for the moment'/>
   </Line>
   <Line ID='EB1'  Description='request for boarding' Mandatory='0' Default='1'>
    <Text Text='We  kindly request  '/>
    <Part ID='ZPX{1EB1}'  Description='pax 1' Type='string' Mandatory='1' DefaultValue='BCP'>
     <Item ID='BCP'  Description='Business class members'/>
     <Item ID='PEC'  Description='Passengers in economy class'/>
     <Item ID='PIC' Description='Passengers traveling with infants or young childeren'/>
    </Part>
    <Text Text='and' Function='If:ZPX{2EB1}'/>
    <Part ID='ZPX{2EB1}' Description='pax 2' Type='string' Mandatory='0' DefaultValue='PIC'>
     <Item ID='BCP' Description='Business class members'/>
     <Item ID='PEC' Description='Passengers in economy class'/>
     <Item ID='PIC' Description='Passengers traveling with infants or young childeren'/>
    </Part>
    <Text Text='to come forward for boarding at gate'/>
    <Part ID='GAT{1EB1}' Description='gate code' Type='string' Mandatory='0' DefaultValue='D15'/>
   </Line>
   <Line ID='RP2' Description='all other passengers remain seated' Mandatory='0' Default='0'>
    <Text Text='All other passengers are requested to remain seated.'/>
   </Line>
   <Line ID='MA1' Description='this is a no smoking flight' Mandatory='0' Default='0'>
    <Text Text='Smoking will not be allowed while on board the aircraft.'/>
   </Line>
   <Line ID='HB1' Description='one item of hand luggage permitted' Mandatory='0' Default='0'>
    <Text Text='On this flight one item of hand luggage per person is permitted.'/>
   </Line>
   <Line ID='HB2' Description='oversize articles will be taken' Mandatory='0' Default='0'>
    <Text Text='On this flight one item of hand luggage per person is permitted.'/>
   </Line>

   <Line ID='DI1' Description='boardingcard ready for inspection' Mandatory='0' Default='0'>
    <Text Text='Please keep your'/>
    <Text Text='boarding card'/>
    <Text Text='and' Function='If:ZID{2DI2}'/>
    <Part ID='ZDI{2DI2}' Description='2nd item' Type='string' Mandatory='0' DefaultValue='OPP'>
     <Item ID='OPP' Description='opened passport'/>
     <Item ID='PPT' Description='passport'/>
    </Part>
    <Text Text='ready for inspection'/>
   </Line>
   <Line ID='WS1' Description='we wish you a pleasant flight' Mandatory='1' Default='1'>
    <Text Text='On behalf of'/>
    <Part ID='ALN{1WS1}' Description='airline' Type='string' Mandatory='1' DefaultValue='KL'/>
    <Text Text='and' Function='If:ZBN{1WS1}'/>
    <Part ID='ZBN{1WS1}' Description='partner' Type='string' Mandatory='0' DefaultValue=''>
     <Item ID='OWO' Description='Oneworld'/>
     <Item ID='SKT' Description='Skyteam'/>
     <Item ID='STA' Description='Star Alliance'/>
    </Part>
    <Text Text='we wish you a pleasant flight'/>
   </Line>
  </Announcement>
 </MessageData>
</AIP>
```

This sample message describes one announcement trigger ‘PRE’ which contains 11 lines.

The lines G03, EB1, DI1 and WS1 have additional parts requiring input from the user. The lines (WX4, RS1, RP2, MA1, HB1, HB2 and HS1 have no additional parts and require no special input from the user.

The lines WX4, G03 and WS1 have the *mandatory* flag enabled, which means these lines cannot be disabled in the actual announcement request. The lines RS1 and EB1 are *not mandatory* and should be presented to the user as lines that are *activated* by default. The lines RP2, MA1, HB1, HB2, HS1 and DI1 are *not mandatory* and should be presented to the user as lines that are *not activated* by default.

The line EB1 contains two parts. Part 1 holds a passenger type code for which 4 possible codes are allowed (BOC, OBC, OPP, PPT). By default, the code OPP should be presented to the user.

Step 3:

Client sends an Announcement Trigger Request Ex message to server.

```
<AIP>
 <MessageID>AnnouncementTriggerRequestEx</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
  <AnnouncementData>

   <Conditions>
    <Condition ID='Airline' Value='KLM'/>
   </Conditions>

   <Announcement ID='PRE'>
    <Line ID='WX4'/>
    <Line ID='G03'>
     <Part  ID='ALN{1BAS}'  Value='KLM'/>
     <Part  ID='FNB{1BAS}'  Value='123'/>
     <Part  ID='APT{1BAS}'  Value='BCN'/>
     <Part  ID='ZRO{1BAS}'  Value='OPB'/>
     <Part  ID='ZCO{1BAS}'  Value='STM'/>
    </Line>
    <Line ID='EB1'>
     <Part  ID='ZPX{1EB1}'  Value='BCP'/>
     <Part  ID='GAT{1EB1}'  Value='D15'/>
    </Line>
    <Line ID='WS1'>
     <Part  ID='ALN{1WS1}'  Value='KLM'/>
     <Part  ID='ZBN{1WS1}'  Value='SKT'/>
    </Line>
   </Announcement>
  </AnnouncementData>
 </MessageData>
</AIP>
```

The announcement request has the lines WX4, G03, EB1 and WS1 activated.
All other sentences as listed in the *AnnouncementListResponse* were not mandatory and have been left out by the user.
In the G03 sentence the optional fields ZRO{1BAS} and ZCO{1BAS} were filled in by the user with codes KLM and SKT.

Step 4:

AviaVox system will process the announcement trigger request and sends the audio to the PA system.

6.15. *Error response*

6.15.1. Message details

Message Name/ ID	ErrorResponse
Description	This message contains details about an error that occurred as a result of a message that was received from the client.
Trigger	Server will send this message as a response any invalid message received from client.
Message handling	Client shall receive the message and use it to log appropriately in order to pinpoint any issues within the interface that has been developed.

6.15.2. Message elements

The below table lists all individual data elements within the message:

Message Element	**Description**	**Data Type**	**M – Mandatory O – Optional**	**(Example) Value**
AIP	Root node		M	Contains elements: ○ MessageID ○ ClientID ○ MessageData
MessageID	The ID of the message	string	M	ErrorResponse
ClientID	Contains the client generated ID that was provided in the last message that was received from the client	string	O	1234567
MessageData	Data node		M	Contains elements: ○ ErrorMessage ○ SourceMessageID
ErrorMessage	Short description of the error that occurred	string	M	InvalidXML
ErrorDetails	Freetext containing details of the error	string	O	Freetext containing details
SourceMessageID	LastMessageID received from client which caused the error	string	O	AnnouncementAuthenticateRequest AnnouncementAudioRequest etc

6.15.3. Message examples

Example of an error response where the client sent an AnnouncementAudioRequest whilst the client was not authenticated yet:

```
<AIP>
 <MessageID>ErrorResponse</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
   <ErrorMessage>NotAuthenticated</ErrorMessage>
   <ErrorDetails>Command not allowed when not authenticated yet </ErrorDetails>
   <SourceMessageID>AnnouncementAudioRequest</SourceMessageID>
 </MessageData>
</AIP>
```

Example of an error response where the client sent an HeartbeatRequest whilst the client was not authenticated yet:

```
<AIP>
 <MessageID>ErrorResponse</MessageID>
 <ClientID>1234567</ClientID>
 <MessageData>
   <ErrorMessage>NotAuthenticated</ErrorMessage>
   <ErrorDetails>Command not allowed when not authenticated yet </ErrorDetails>
   <SourceMessageID>HeartbeatRequest</SourceMessageID>
 </MessageData>
</AIP>
```

Example of an error response where the client sent a message which contained incorrect formatted XML. In this situation, no ClientID or SourceMessageID will be included in the response message

```
<AIP>
 <MessageID>ErrorResponse</MessageID>
 <MessageData>
   <ErrorMessage>InvalidXML</ErrorMessage>
   <ErrorDetails>XSD error message</ErrorDetails>
 </MessageData>
</AIP>
```