Electronic Band Scores and Stage Services Framework

Georg J. Schneider and Dimitri Perepelkin
Computer Science Department, Trier University of Applied Sciences, Main Campus, Trier, Germany

Keywords: Services and Mobility, Application of Mobile Information Systems, Protocols, Standards, Interoperability and Mobility.

Abstract: The paper describes a framework for the mobile display of electronic sheet music using the UPnP infrastructure for bands. Based on this architecture, further services can be integrated, like MIDI sounds or light shows.

1 INTRODUCTION

A cover band, which plays an evening show, usually has a large set list to satisfy the various musical tastes of the audience. Additionally they are usually prepared for song requests from the audience. Each of these songs consists of several sheets of music. The same is true for large orchestras playing symphonies or operas. One of the challenges, which has nothing to do with mastering the instrument, is the turning of the sheets during the play. Another defiance is to keep all these sheets in a straight ordering on the music stand. A disorder in the sheets can cause a major problem during a performance. Confusing scores for different instruments may lead to a similar catastrophe.

For this reason an electronic version of a score would be just the right solution for this issue. The musicians only need a tablet computer. The sheet music could be stored on the devices or on a central unit.

The Brussles Philharmonic used this approach as the first orchestra in 2013. They use a modified version of neoScores (Gustaf 2018) to display the sheet music.

Furthermore the score could also move forward as needed, like a tele prompter, so that the musician will never have to switch to the next sheet.

However as it concerns an evening performance of a band, the situation is more various. Different musicians may join in for certain tunes. They would need the scores for their instrument or the lyrics, when the musician is singer.

Additionally the integration of further services in the system, which belong to the performance, shall be possible as well. For example the playback of MIDI sequences, the light show or the playing of videos in the background shall be flexibly integrated into the system also.

For this reason we suggest a flexible software architecture based on the Universal Plug and Play (short: UPnP) (OCF 2018) protocol in this paper. Together with services running either on mobile clients like tablets or on other eventually specialised devices, an infrastructure for bands can be realized that nicely adapts to the current situation.

The paper is structured as follows. First we will discuss several systems targeting a similar domain as ours. We will sketch related developments using UPnP. We will also discuss approached towards electronic sheet music. Afterwards we will give a short introduction to UPnP. We will then present our framework. Based on these considerations we will describe the realization of the system with the help of the Cling UPnP library. The paper concludes with a short summary and an outlook to further developments of our solution.

2 RELATED WORK

Most systems that support similar situations like the one we are targeting are UPnP based systems that stream media from a server to different clients using the UPnP protocol. One example is BubbleUPnP (BubbleUPnP 2018). The software consists of a media server and an Android client for rendering audio and video data to mobile devices.

Another application with similar functionality is UPnP monkey from Ronald Bruckner. The Android

Schneider, G. and Perepelkin, D.
Electronic Band Scores and Stage Services Framework.
DOI: 10.5220/0006958102690276
In *Proceedings of the 14th International Conference on Web Information Systems and Technologies (WEBIST 2018)*, pages 269-276
ISBN: 978-989-758-324-7

app detects media server and media renderer in a network and can control them individually.

(Sung et. al. 2006) describe an UPnP based framework for multimedia content delivery with content adaptation. In our case however the different versions of the multimedia content belong to the profile an automatic adaptation or blending of different versions is not planned in our system.

(Lai et al. 2011) suggest an architecture for an UPnP-based high performance multimedia content sharing system. Right now the amount of data of our application is not that high yet so that a conventional approach satisfies the needs of our scenario. However their ideas would be helpful if extensive video shows will be a topic in the future.

On the other hand systems like Cantabile (Cantabile 2018) offer possibilities to support life performances with sound effects and MIDI playback. The system also offers the managing of set lists.

(Kurth et. al. 2007), (Kurth et. al. 2008) suggest a digital library framework for managing multimodal music collections. Their system synchronizes e.g. playback of scanned sheet music and audio playback and also offers search capabilities, which are important for large music collections. Right now the requirements of our scenario are still clearly arranged so that an automatic processing of the files is not yet necessary. Furthermore other media such as light or video is not part of their consideration yet but might be an interesting extension of this approach.

A software solution, which supports bands for their life performances, including possibilities to integrate services like sheet music, set list management, midi player, light shows and a flexible extension of the system with further services is not known to the authors so far. The above mentioned systems have extensive functionalities in their specific domain. However an integrated and easily expandable system with a low footprint is needed for the band scenario like the one we are looking at.

For this reason we suggest an architecture based on the UPnP protocol in order to offer the different services in the network. With this approach we can flexibly adapt to different situations and even change the setting in between.

3 UNIVERSAL PLUG AND PLAY

Universal Plug and Play (OCF 2018) is a protocol for ad-hoc networking. The protocol uses IP networks and the HTTP protocol to communicate between entities. Using the protocol a developer can specify different entities like devices that offer services and control points that detect these devices and consume their services. Devices and services are described in form of XML files. Therefore arbitrary devices and services can be defined and a developer is not bound to prior defined specifications. The implementation of the services however is proprietary and hence not part of the protocol. Another important part of the specification is a discovery mechanism to detect new devices in a network. Once a device is detected, it can be queried for the services it offers. As a consequence the XML specification will be transmitted explaining details about the services and the way they can be used. Finally the communication between a control point and a device is done using SOAP (W3C 2018).

Nevertheless, some often needed devices are defined by the UPnP task force, like media server and media renderer.

A physical device can offer different UPnP devices with a variety of services as well as a control point. Like that the interaction with the services on the same physical device or services offered by other devices on the network is transparent to the control point.

4 CONCEPT OF THE FRAMEWORK

The idea of our framework is that it should be possible to add different services to a network, like electronic scores, set list management, MIDI controller, light show but also to make the services adaptable to the situation. For example a band consisting of a singer, a guitar player, a bassist and a drummer need an electronic sheet music service. Each of these musicians will obviously need a different version of the sheet music, which demands for adding extra information to the sheet music, e.g. using metadata for describing the purpose for example the instrument. Especially looking at a situation where one of the musicians is also the singer in one show where in another show there will be a special guest only for singing, it is possible that different sheet music for the same song are needed. Therefore a user management must be part of the framework as well.

The electronic sheet music shall be rendered for example on a tablet computer. The tablets again shall not be dedicated to a specific band member. Whoever uses the tablet shall specify her role and access the appropriate information. The sheet music shall automatically start in parallel on each tablet and scroll forward as the song continues, so that there is no need for manual interaction during the song.

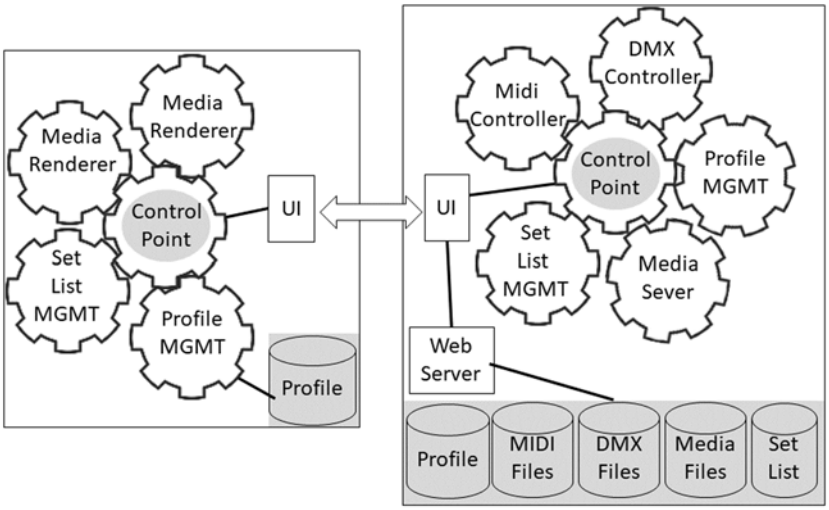

Figure 1: System architecture.

Set list management is another important service for a band. Usually the band will put together a set list for a show in beforehand. However situations where the band wants to adapt to its audience may happen, especially when we are looking at cover bands. Song requests from the audience shall also be respected. As a consequence an easy to use set list handling is needed. Since every band member may be approached, each tablet must provide this service and changes must be communicated immediately to the devices of the other band members. Each band member shall also have the possibility to start the playing of a song. In our case this means that each tablet must be able to give the signal to the other devices to begin displaying the appropriate information and start the other services, which belong to the playing of the song, i.e. the light show.

Today the use of MIDI samples is a feature, which is often used by bands. “MIDI is an industry standard music technology protocol that connects products from many different companies including digital musical instruments, computers, tablets, and smartphones.” (MIDI 2018). Sound effects for example that cannot be easily produced with an instrument or instruments that are not part of the band can thus be integrated into the songs. The playback of these samples must be started at the right time within the song. A MIDI controller can therefore be defined as an UPnP device, which interfaces with the MIDI instruments.

Finally we want to integrate light and video services to improve the visual appearance of the show. Media services on the one hand are easy to integrate, since media server and renderer are already defined from the UPnP task force, as described in the precedent section. Since video services are media services and many video player software already implement an UPnP interface, the integration is straightforward. However as it concerns light services, the situation is different. Light control usually is based on the DMX protocol (Beuth 2000). DMX stands for Digital Multiplex. The protocol defines codes that are interpreted by the light equipment, e.g. like strobe lights or moving head lights. The DMX protocol can be wrapped in an UPnP device as well.

Putting the different pieces together we realize a basic architecture as in figure 1. We see two physical devices offering different basic services for the band. In general the assignment from UPnP devices, shown as gear in the illustration, to the two physical devices is flexible.

On the left hand side there is an example of a setup, which could be used for each band member. The architecture on the right hand side offers some fundamental services for all band members. Looking first at the left hand side, two media renderers are shown. One media renderer will display the lyrics and the sheet music. The other media renderer is in charge with the presentation of videos in the background. Especially this service might also be exploited on another device since further physical devices can be added as well and the arrangement of UPnP devices may be different to any of the illustrated (physical) devices. The control point uses the services offered by the devices. In general it is transparent to the control point if the device resides on the same physical device or if it is controlled over the internet. The set list management must be installed on each device where it shall be possible to modify the set list during a show. The profile management makes it possible to use an arbitrary client and log in e.g. as a guitar player or a

singer. Thus the sheet music for this band member can be rendered on the appropriate device.

On the right hand side a media server is shown. The media server will stream the appropriate sheet music to the clients during the show. This allows to keep all files on one device. Therefore the copying of files to the other devices is not necessary. The use of the different devices is more flexible with this approach. The profile data identify a certain role and this role can change over time. Also spare devices can replace defect devices during a show almost instantaneously. The web server is the central component to copy files to the device. It provides also the interface to author the show, to build the set list and to add light or sound effects to the pieces of music. On the bottom, the different files that might be used during the show are shown.

5 IMPLEMENTATION

Tablet computers are an ideal output device for digital sheet music since they have a decent size and are easy to transport. Therefore we have implemented our system for these devices. However, as the architecture in figure 1 shows on the right hand side, a dedicated device offers more powerful services, like media streaming to several devices and it needs more connection possibilities like DMX or MIDI. For this reason we have decided to add a laptop computer to the technical infrastructure.

As it concerns the tablet computers, we have used Android as the target platform. Therefore we need to build an app, which renders the information on the device.

Figure 2 shows the view of an UPnP inspection tool, which has detected the two devices in the network. The SheetMusic-PC is the central unit providing the streaming services. “Smarty” is one of the tablet computers, which are used to display the sheet music for the band members. As one can see, several services are visible like changing the playlist.

The UPnP control points, devices and services are realized using the Cling UPnP implementation (Line 2018), which is available for Android and Java.

In order to implement an UPnP device with Cling, the Java class that represents a service is annotated.

We show the procedure giving a simple example, which only has one state variable “Address” where the IP address of the data exchange server is stored and one action “GetAdddress” that returns this address. They are part of the service SheetMusicServer, which belongs to the UPnP device SheetMusic-PC (see figure 2).

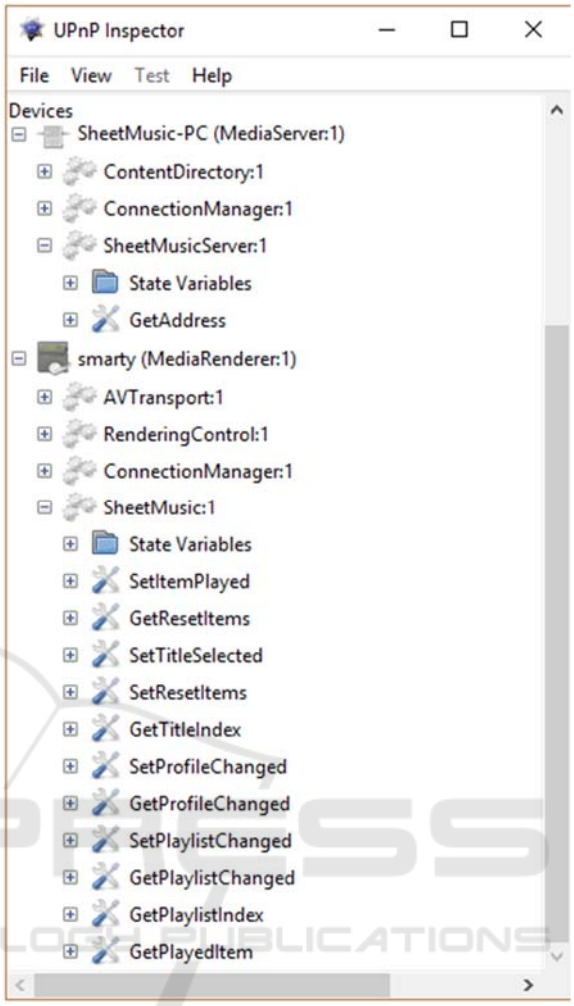

Figure 2: UPnP devices.

First the service is declared giving a service type and a service identifier. This is accomplished with the appropriate annotations starting with “@Upnp”.

```
@UpnpService(
 serviceId =
@UpnpServiceId("SheetMusicServer"),
 serviceType = @UpnpServiceType
  (value="SheetMusicServer",
         version = 1))
```

Afterwards we declare the state variable in a similar way, providing a default value, which is returned when the value of the state variable cannot be determined.

```
@UpnpStateVariables({
 @UpnpStateVariable(
 name="Address",defaultValue="0")})
```

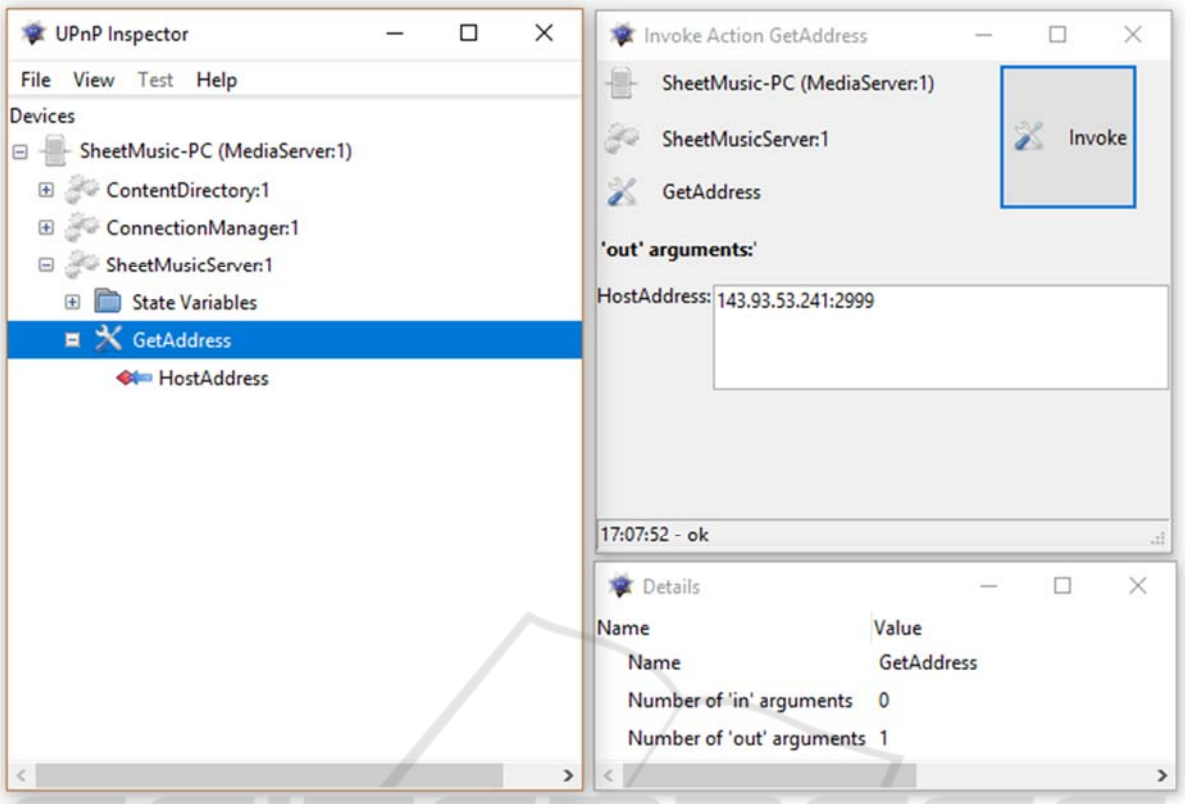

Figure 3: UPnP action invocation.

Finally we declare the action to return the value of the state variable, followed immediately by the code that represents the action. In this case the “out” argument is the host address.

```
@UpnpAction(out =
   @UpnpOutputArgument(
      name="HostAddress"))
```

The Java code returning the address is given below. The “getter” method accesses the attribute, where the data is stored.

```
public String getAddress() {
      return address;}
```

Afterwards the device is created that will contain the service. First we need to create a unique identity of the device, which is used for searching the device in the network.

```
DeviceIdentity identity =
  new DeviceIdentity(
    UDN.uniqueSystemIdentifier(
     "SheetMusic MediaServer"));
```

Subsequently we create the service for the device. The annotated class is used for the creation of this service.

```
LocalService<SheetMusicService>
sheetMusicService = new
AnnotationLocalServiceBinder().read
(SheetMusicService.class);
```

Finally the bits and pieces are put together. The device is created, which offers the service.

```
return new LocalDevice(identity,
type, details, createIcon(),
sheetMusicService);
```

Using this service remotely, a control point would have to add the following Java code. First the control point has to find the appropriate service from the remote UPnP device.

```
Service service =
 device.findService(new
 UDAServiceId("SheetMusicServer"));
```

The desired action, which will be executed is accessed using the service.

```
Action action =
service.getAction("GetAddress");
```

A callback method is defined. It will be executed when the desired action, which is executed on the remote side, successfully returns the data.

```
ActionInvocation invocation = new
  ActionInvocation(action);
ActionCallback callback = new
 ActionCallback(invocation) {
```

The success method handles the successful execution of the action and stores the data received from the remote UPnP service. The failure method will be called if there were any problems during the execution.

```
public void
     success(ActionInvocation
           invocation) {
ActionArgumentValue address =
invocation.getOutput("HostAddress")
;
main.setServerAddress(
     address.toString());
                        }
public void failure(…) {…}
};
```

Finally the service can be executed using the following command.

```
upnpService.getControlPoint()
  .execute(callback);
```

Figure 3 shows the view of an UPnP inspection tool to the UPnP device on the physical device. The SheetMusic-PC device is displayed as well as the services. Concerning the SheetMusicServer, the action and the state variables are visible as well. Especially the description of the service in the bottom right corner shows the “in” and “out” arguments of the “GetAddress” action. Invoking this action from the tool shows the result of the action. In our case the host address and the port are displayed in the text field on the right hand side, to inform the other devices about where to find the data exchange server.

Finding our device using a general UPnP inspection tool, which serves as control point, shows the conformance of the implementation with the UPnP standard and guarantees a flexible usage of the components.

Each control point can gather the information of device and services because the UPnP device automatically generates the following XML description and informs the peers about the URL where to access this description.

The XML file shows the actions offered by a service, as well as “in” and “out” arguments and related state variables. Other entities can subscribe for changes of these state variables and can likewise react, when changes of these variables occur.

```
…
<actionList>
  <action>
    <name>GetAddress</name>
    <argumentList>
      <argument>
       <name>HostAddress</name>
       <direction>out</direction>
       <relatedStateVariable>
        Address
       </relatedStateVariable>
      </argument>
    </argumentList>
  </action>
</actionList>

<serviceStateTable>
  <stateVariable sendEvents="yes">
    <name>Address</name>
    <dataType>string</dataType>
    <defaultValue>0</defaultValue>
  </stateVariable>
</serviceStateTable>
```

Figure 4: Snippet of the generated XML file of the service.

As we can see in figure 4, the action “GetAdress” is listed together with the argument “HostAddress” as “out” argument. Furthermore the part under “serviceStateTable” tells that the change of the address sends out an event, that interested parties can subscribe.

6 EXPERIMERNTAL RESULTS

The system has been tested on various occasions from a band called “El Camino” consisting of the two artists Albert Niesen and Detlef Roth, both guitar players and singers. They used the system productive on stages around Europe. Especially the system proved to be stable, reactive and user friendly in demanding environments such as beach clubs.

Figure 5: Picture of “El Camino” performing using the system.

Figure 5 shows a screenshot from one of the rehearsals. The screen of one of the tablet computers is displayed on the projection behind the artists. The display shows the lyrics, where only a part is visible on the top. Below it shows the chords. On the upper right corner the elapsed time of the tune is displayed as well.

7 CONCLUSIONS

We have presented a system architecture together with an implementation for supporting bands with electronic scores and further services, which are relevant for their performances. Based on the UPnP protocol, the presented system is extensible and may integrate further services as needed. The paper also shows the flexible integration and usage of the implemented services on the example of an UPnP inspection tool.

The systems has been used in real world scenarios and it has proved its usefulness in several shows, especially in rough environments like pubs or beach clubs. In the future we want to add services for light shows through and integration of a service for the DMX protocol.

Right now the MIDI support is not yet realized as an UPnP device. This will also be a next step in the implementation of our system.

Additionally a service for selecting suitable photographs, which relate to the tune and displaying them as slideshows in the background is planned.

In the future, more sophisticated possibilities to display and interact with the sheet music may be desired. The notation of music in an XML format, that would nicely integrate in our platform could be one extension (see Good 2001) or even as a means to generate music (Torrente 2011).

Whereas approaches for real-time music notation (Lee et. al 2010) target a different scenario, i.e. integration of laptop performers with instrumental musicians the approach includes ideas that go beyond our level of interactivity and may be a promising path for further extensions of our system.

ACKNOWLEDGEMENTS

The authors thank Albert Niesen (deceased in December 2017) for his support, ideas and most of all for his great personality and for being much more than a colleague.

REFERENCES

Gustaf, 2018. https://www.gogustaf.com.

Beuth, 2000. DIN 56930-2, Beuth Verlag GmbH, Berlin

BubbleUPnP, 2018. http://bubblesoftapps.com.

Cantabile, 2018. https://www.cantabilesoftware.com

Good, M., 2001. MusicXML: An Internet-Friendly Format for Sheet Music. XML Conference and Expo 2001

Kurth F., Müller M., Fremerey C., Chang Yh, Clausen M., 2007. Automated Synchronisation of scanned sheet music with audio recordings. In: *Proceedings of the 8th International Conference on Music Information Retrieval, 2007, Vienna, Austria.*

Kurth F., Damm D., Fremerey C., Müller M., Clausen M., 2008. A Framework for Managing Multimodal Digitized Music Collections. In: Christensen-Dalsgaard B., Castelli D., Ammitzbøll Jurik B., Lippincott J. (eds) Research and Advanced Technology for Digital Libraries. ECDL 2008. *Lecture Notes in Computer Science, vol 5173. Springer, Berlin, Heidelberg.*

Lai, C. F., Chang, S. Y., Huang, Y. M. Park, J. H., Chao, H. C., 2011. A portable UPnP-based high performance content sharing system for supporting multimedia devices. *The Journal of Supercomputing, Vol. 55, Issue 2, Springer, Berlin, Heidelberg.*

Lee, S.W., Freemen, J., Collela, A., 2012. Real-Time Music Notation, Collaborative Improvisation, and Laptop Ensembles. In: Proc. *12th International Conference on New Interfaces for Musical Expression, 2012, Ann Arbor, Michigan, USA.*

Line GmbH, 2018. http://4thline.org

MIDI, 2018. https://www.midi.org

OCF, 2018. https://openconnectivity.org/developer/specifications/upnp-resources/upnp

Sung, J., Kim, D., Song, H., Kim, J., Yong L., Choi, J., 2006. UPnP based intelligent multimedia service architecture for digital home network. *Proc. of the Fourth IEEE Workshop on Software Technologies for Future Embedded and Ubiquitous Systems and Second International Workshop on Collaborative Computing, Integration, and Assurance (SEUS-WCCIA'06)* .

Torrente, B., Barro Torres, S.J., Dapena, A., Escudero, C.J., 2011. Defining an XML format for sound synthesis. In: *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication, 2011, Seoul, Republic of Korea.*

W3C, 2018. https://www.w3.org/TR/soap/