MEET: A Technical Framework Enabling Ubiquitous Music in a Social Context

Simon Lind Damgaard
Department of Computer Science
Aalborg University
sdamga09@student.aau.dk

Liv Stahl Madsen
Department of Computer Science
Aalborg University
lsma09@student.aau.dk

Henrik Sørensen
Department of Computer Science
Aalborg University
hsoere09@student.aau.dk

ABSTRACT

As music gets digitalized and as mobile devices gets more powerful, possibilities for listening to your own music everywhere emerges. Synchronization is a technique often used to keep data consistent across multiple devices. It is however not suitable for extensive music collections because of the large amount of data that needs to be transferred and stored redundantly. Increasing availability of broadband Internet connections on mobile devices creates new opportunities for ubiquitous music and circumvent the shortcomings of synchronization. Through an exploratory technology analysis of existing music sharing systems we found an unexplored area which concerns the aspects of listening to music in a social context. Our solution tries to fill this gap by allowing users to metaphorically bring along their music collection using their mobile as a key token to their home music library. This enables the users to pick out what music to contribute to a common playlist and stream it directly from home. The system is a technical platform for further user interaction development aimed at a future usability field test.

Author Keywords

Music sharing, Mobile music, Streaming, Ubiquitous computing, Social context

INTRODUCTION

The growing extent of digital music usage and the wide distribution of powerful Internet and media enabled mobile devices, creates new opportunities for the rapidly growing research field of ubiquitous music [23]. In a Human Computer Interaction (HCI) perspective, there are several angles of approach that are interesting in research related to relations between people and music.

This paper focuses on creating a ubiquitous home music library that seeks not only to obtain accessibility for the single user personally, but also to let the user share music in a social context.

Through a technology analysis, existing multiple device music systems are examined and the results are used to develop a new approach that lets the user metaphorically carry the home music library around at all times. Ubiquity is obtained by introducing the mobile device as a key token, granting access to the personal home music library, as opposed to having the data stored on the device itself. The goal is to represent the music library as a tangible item that features a user-controlled sharing mechanism, allowing multiple users to share parts of or their entire personal library to a common library in a social context. An example is creating playlists at a party from a combination of each user's personal music, accessed through their mobile device.

Many people have a large personal digital music collection at home [17] which they would like to listen to regardless of physical location. Synchronization is often used to keep information consistent across several devices and is appropriate for small amounts of data like emails, calendars and contact information. Despite efficient sound compression techniques, it is however inconvenient to keep large amounts of data, like a music library, synchronized with mobile devices because of limited storage space. If music is instantly accessible without the receiver being tied to a physical location, there simply is no need to replicate potentially large amounts of data.

As an increasing number of mobile devices are continuously online, feasible possibilities for media streaming opens up to accommodate the shortcomings of synchronization. Streaming is a suitable solution that has become increasingly popular for both audio and video through services such as YouTube [14], WiMP [11] or Sputnik [12]. The transfer bandwidth in this case is limited, but transmissions only occur when the media are needed.

The paper is structured with an initial insight into related work followed by the technology analysis which leads to the introduction of the solution called MEET. The architectural design of MEET is described, a user scenario for the system is given and the essentials of the implementation and system setup are explained. The paper is concluded and interesting aspects of future research is looked into as well.

RELATED WORK

Several studies are made concerning music in a social context. It is, i.a., proposed that choice of music is tied to an individual's identity and that a music collection can be very personal [22]. Because of the anonymous nature of online music sharing, the social aspect of conventional music sharing disappears. Research has focused more on how people share and exchange music rather than how people listen to music together [18]. The distinction between the personal music collection and a large scale music sharing service are examined in various studies [25][20][17].

A study of an early music sharing initiative shows how an attempt is made to make digital music sharing more intimate and less anonymous. This is, e.g., achieved by limiting the sharing space to a local subnetwork and reintroducing the human factor in form of personal playlists [26].

More specific technical suggestions are made as part of the music sharing research. One is Push!Music that creates recommendations for nearby users that share listening patterns. It also allows users to manually recommend songs wirelessly to friends. The study showed that the features were well received, but users tended to favor the direct music exchange opposed to the more anonymous autonomous recommendations [21]. Another approach, called SuperMusic, offers music streaming and context aware recommendations to avoid locally stored music collections [24].

A group of technologies are built around the concept of discovering new music through examining music collections of people with similar taste in music. Implementations of such technologies are seen in tunA [15] , BluetunA [16] and Music Buddy [19]. These systems do not focus on the actual sharing of music, but rather the social aspect of discovering new music. Music Buddy lets users explore other user's playlists online. The same principle is used in tunA with mobile devices and Wi-Fi and an attempt to make this even less location dependent is seen with BluetunA over Bluetooth.

METHOD

This study performs exploratory research by examining already existing music sharing systems, to unveil unexplored concerns of this area. Part of the research is to choose systems which are representative for interesting fundamental differences in music sharing.Only systems that are established in their respective domains are chosen. The format is a technology analysis where the systems are compared and categorized to discover strengths and weaknesses.

The technology analysis forms the basis to develop a new concept which seeks to accommodate an absence in existing approaches. The concept is implemented as a functional prototype with focus on the underlying technical framework based on a specific user scenario. Establishing a solid framework facilitates future user interaction development with the aim at conducting a field test in a situation like the user scenario.

Testing of the prototype's scalability is conducted to ensure that it can endure usability testing in a real-life situation, where the scale and user interaction is not fully controlled. Results can be used to optimize the system prior to field testing.

TECHNOLOGY ANALYSIS

The purpose of the technology analysis conducted in this study is not to create a comprehensive categorization of all existing music sharing services. Instead, it is used as part of a process to discover and address a shortage in making music sharing ubiquitous. The analysis is based on four systems, chosen because of their different approaches to provide music sharing capabilities.

- **iTunes [4]:** Apple's iTunes represents systems which rely on synchronization for music sharing across multiple devices, including mobile devices such as iPhones and iPods. iTunes does provide two functionalities of sharing and listening to media files within a Local Area Network (LAN); one is the so-called Home Sharing that lets the user stream and transfer media with up to five computers, in the household, using the same iTunes login [6]. The other, lets the user stream and play music from the home library to speakers containing the AirPlay wireless technology within a LAN [5].
- **DLNA [1]:** Digital Living Network Alliance (DLNA) is not a system itself but a collaboration of consumer electronics, PC and mobile companies with the goal of making devices compatible across brands and making media sharing easier. It is aimed at the digital home and relies on a LAN and streaming between devices rather than file exchange. Between DNLA certified devices, media are then made ubiquitous.
  A plausible usage scenario is a media center setup with a large flatscreen TV as a renderer. Several devices such as smartphones, laptops and gaming consoles can be used as servers making the entire media collection spread across several devices but be accessible from a single place such as the TV.
- **Orb Live [8]:** This solution consists of a desktop application, that hosts different types of media, and a mobile application where the media are streamed onto. For usage, the users must create an Orb Live user account, that both the server and the client application must be logged into for connection establishment. This solution give users the opportunity to access their home media collection on the run.
  Basically, Orb Live supports media accessibility for one or more mobile accounts from one media server. This means that Orb Live gives the option to log users' friends into their media library server. This enables the users to choose exactly what to share with whom. However, this restricts the users, logged in to another media collection, by not being able to log in to their own media collection. Orb Live furthermore has a community for users on their website, where they can send messages to friends with media links or publish the link on the public profile on Orb Live's website [9].
- **Grooveshark [2]:** The primary difference between Grooveshark and the other systems presented in this analysis, is that it is an online music service where the shared music is not that of the users. When a user account is created, such features as playlist management and music upload is available. The music can then be streamed either from the website, a desktop application or a mobile application. Using the website as a jukebox is free while using the Grooveshark mobile application requires a so called Grooveshark Anywhere subscription which requires a monthly fee. Even though the user can contribute with personal music by uploading to Grooveshark's collection, this is the only solution examined which is not based entirely on a personal media collection. TDC Play [10] and WiMP supplied by the Danish phone companies TDC and

Telenor, respectively, are music services similar to Grooveshark. They offer their broadband customers access to millions of music tracks which are available as long as the user is a customer at this company.

Table 1 summarizes the sharing methods of the presented systems. One thing that is missing for each of the four systems is the focus of listening to the music in a social context. However, DLNA and Grooveshark have partial social features but mainly focuse on single user functionalities. DLNA lets users connect to a centralized player and listen to their private music collection in a social context, as long as they are connected to the same LAN and as long as the media files are stored on a DLNA certified device. Grooveshark lets users create user defined playlists and share them with other users. With global streaming, the music is accessible everywhere but the service lacks the personal aspect of having a music collection of your own, saying something about your personality. iTunes gives the opportunity to access and stream from up to five music computer libraries within the same LAN, however, the five computers must use the same iTunes login which makes the service mainly a single user service. Orb Live lets the users access their media files everywhere but the service is created for single users and only music from a single music collection can be accessed at a time.

Solution	Synchronization	Streaming		Social Context
		Global	Local	
iTunes	X		X	
DLNA			X	/
Orb Live		X		
Grooveshark		X		/

Table 1. Sharing method differences between investigated music sharing solutions.

MEET

What the technology analysis has revealed is a rather unexplored area of music sharing that has motivated a different approach. The notion of making a home music library ubiquitous is already there, but the primary goal of existing solutions is to stream music from a library at home to the device currently used by the user. This aims at enabling a single user to listen to the music anywhere.

Music is however also something that people listen to together and something that people use to express themselves. By sharing music with friends they might discover new music and inspire others. The motivation for MEET is to create a ubiquitous music library in a social context by combining the ubiquity of global streaming from systems such as Orb Live and the distributed library used locally in a DLNA setup. This makes it possible for every user to access a home music library from anywhere and share it in a common library at social events such as parties. Not only does this allow each user to personally contribute to the music played, it also opens up for opportunities to create different ways to interact with the music at such events.

MEET is a system which enables users to access their home music library from their mobile and grant access to a player. The mobile can hence be seen as a key token to the users music library, which can be shared with others when the user allows it. The system further allows the user to decide, not only what should be accessible on the phone, but also what to grant access to, from the phone, to a friend's common library. This functionality is called double filtering, since the user first defines a private library accessible on the phone and later, on the phone, defines a public library which is shared with friends.

Analogy

Two analogies can be used to describe the different aspects of the solution.

For the ubiquitous part, it can be seen as if everywhere the user goes, he carries along a speaker with a very long cable leading back to a stereo at home and a long-distance remote. At any time he can play his own music for himself or together with others. He can however not pass the music along to others, as the it is physically located at home.

The other part is the social aspect. This is like a potluck where every attendee brings along a dish, each contributing to what, put together, makes up a complete dinner. In this case, everyone brings the music they want to share with the people present and together they can create a common music library, where everyone is represented.

Design Goals

In order to make this approach successful there are three design goals that are considered important.

1. **Ubiquity:** To be able to access the music from any given location is key. The user should not be concerned about where the music is stored and should therefore not have to transfer anything when listening to the music from different locations.
2. **Ease-of-use:** For this approach to work in practice it has to be simple, easy to configure and easy to use. The feeling of ubiquity can easily be spoiled if, e.g., the connection to the music library is a 10 step procedure. The configuration and connection mechanisms should be as transparent as possible to the user. The goal is to make the system feel just as intuitive as bringing along a collection of CDs, finding the desired track and playing it on a stereo.
3. **Community:** An important part of this approach is to emphasize the social aspect of music. The system should support features that encourage multiple users to listen and explore music together at the same physical location. This enables a concept of music experienced everywhere together.

ARCHITECTURAL DESIGN

In order to reach the design goal of ubiquity, users need to be able to share their music from anywhere. This is achieved by creating a client/server system where the server hosts the home music library. The client is an application able to play the desired songs from server.

Since most users can be assumed to own a mobile and carry it with them at all times, this device is a central part of the system and considered a key token granting access to shared music. From the mobile, users are able to decide which songs from the home music library a certain player has access to as well as the ability to connect to and disconnect from players. Sease et al. [25] describe how mechanisms that allow the user to scaffold the extent of their sharing should exist. This is the exact purpose of the introduced double filtering. Users can choose which folders to share from home, to form a private library, and additionally choose which songs from this selections to share in a specific context as a public library. The concept is not about sharing all music at all times to everyone, but instead to contribute with a specific selection of music for different occasions.

Combining the client/server architecture with the mobile as a key token, results in a system architecture as shown in Figure 1.

The actual streaming of music from the library server to the player does not need to be routed through the mobile. Having full control of the stream through the mobile application would conceptually be a nice feature but could just as easily be achieved by issuing commands from the mobile application to the library server or player. By streaming directly from the library server to the player, the amount of traffic through a user's mobile data connection is greatly reduced. Since affordable flat rate mobile data contracts with high up- and download bandwidth, at the time of writing, are not available in Denmark, this is considered a better solution.

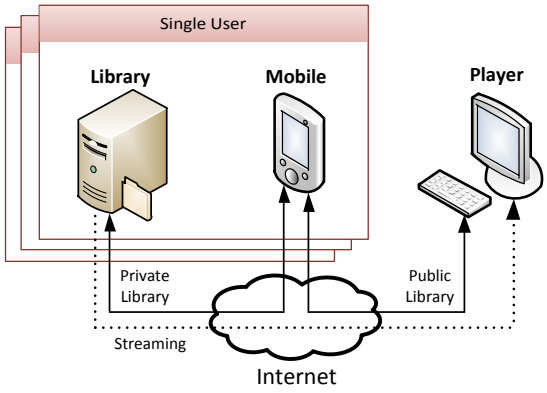

Figure 1. Conceptual system architecture.

Increasing the ease-of-use of the system and reaching the second design goal involves simplifying the initial setup of the system as well as the actual sharing of music. Having both the music library and player contained in a single desktop application caters to the former. The only additional setup action besides the installation of the application is registering the library in the mobile application. The player requires no additional setup.

The community design goal aims at enabling music listening and discovery in a social context. The platform for this interaction will be letting users share music in a single common playlist from where the songs can be played regardless of which library server they originate from. This will be the target for this first prototype and form the basis of further development exploring the HCI possibilities of the system.

Local vs Global Device Discovery

In the technology analysis both LAN and global Internet are encountered as device discovery solutions to a multiple device music system.

LAN device discovery can relatively easily be achieved through protocols such as DLNA. A player broadcasts it's presence and when users arrive at a social event and log on to the LAN, their mobiles can automatically check for players and choose to connect to one of them as shown in Figure 2. When the event is over, the mobile disconnects from the network which triggers the end of streaming from the associated home library server. The disadvantage of this approach is the need to hand out the Wi-Fi password to all users or making the network open and insecure.

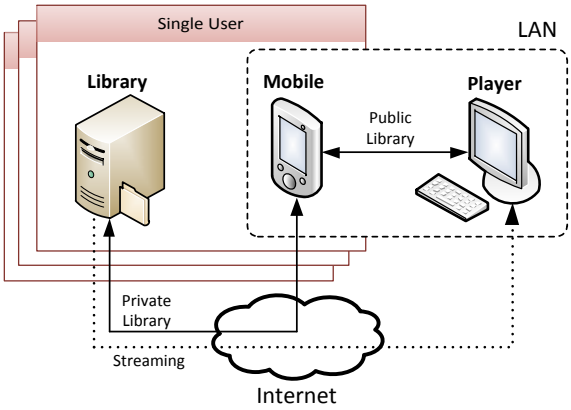

Figure 2. System architecture with LAN device discovery.

Global Internet device discovery eliminates the need for a LAN entirely but the presence of, e.g., a player, cannot easily be broadcast to a nearby mobile and the establishment of the connections between the library, mobile and player requires extra work.

A solution to this problem is an online player directory with searchable parameters, e.g., geographical, making it possible for users to search their way to the desired player. A disadvantage of this approach is that each user can potentially have several players online. This will make the direc-

tory very large and, in turn, make it harder to find the correct player. Another disadvantage is that it requires a careful setup procedure of all players as they will never be found, if they are set up with inaccurate information. Lastly, it will simply seem cumbersome having to spend time finding a player online when standing right next to it.

A more elegant solution is asking the user to enter a key, generated by a facilitating web server, into the player as done by, e.g., iTunes in regards to connecting an iPhone for use as a remote control of the media player. This will identify a specific mobile and thereby a specific user. When the key is entered, the user uses the mobile to identify which songs to share and send this information to the player. The music will then be available in the common library of the player and ready for playback. This way, the only setup needed for players is to connect them to the Internet. Users can then connect to any player as long as they have access to the key entry functionality.

With the addition of a web server, named facilitator, handling the key generation and the connections between library, mobile and player, the final system architecture ends up as shown in Figure 3.

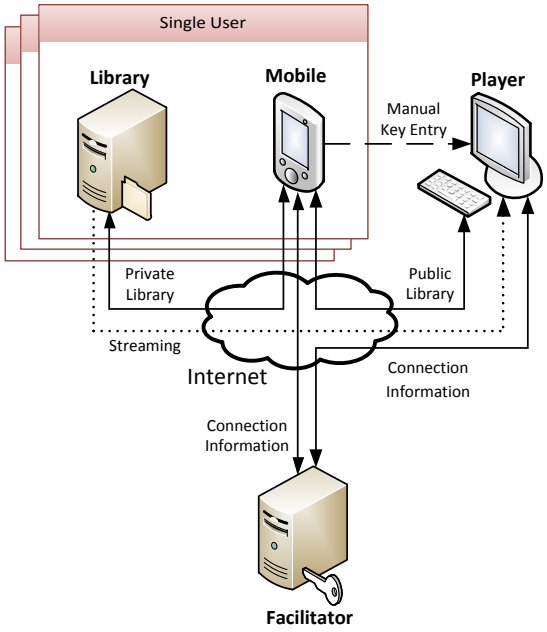

Figure 3. System architecture with online facilitator.

USER SCENARIO

To create a clear illustration of how the system is set up and used, we introduce three fictional persons, Alice, Bob and Charlie. Of the four system architecture components, users only interact directly with three; the library server, the mobile application and the player application. The mobile application works as a key token for a music sharing connection, which indicates that the person with the mobile decides when and what to share with others.

Bob and Alice have both installed the library server on each of their computers at home. They are attending Charlie's party on Friday and would like to be able to add some of their own music to the party playlist. When the server is, installed Alice chooses to add her entire music collection to the music library while Bob only adds one album.

This is illustrated in Figure 4 which is a state diagram showing the different states of the Library Server.

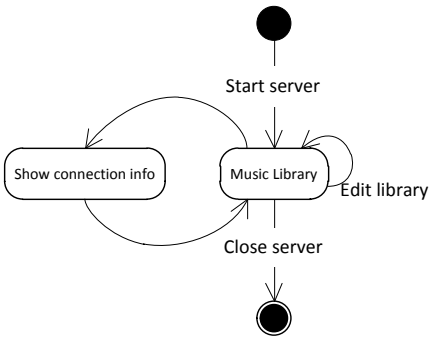

Figure 4. State diagram of the library server.

Both Alice and Bob also installs the mobile application on their smartphone, and configure it with the library server IP. Figure 4 illustrates how to display this library server information.

Charlie has a limited music collection and therefore installs the player desktop application to be able to play some of his friends' music at his party on Friday.

Connection Setup

When Bob arrives at Charlie's he wants to play his album. He starts the mobile application and requests a connection key from the web server, which Charlie enters in the player. Figure 5 shows screenshots from the mobile application and the player, respectively.

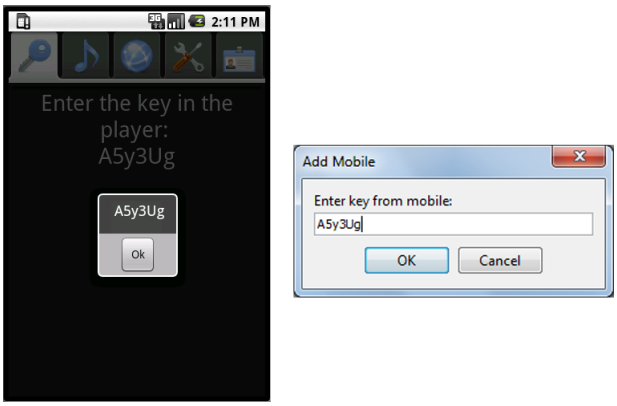

Figure 5. Key from the mobile application entered in the player.

When the key is entered in the player Bob moves on to the next step in the mobile application and his one album, selected from home, is displayed, showed in Figure 6.

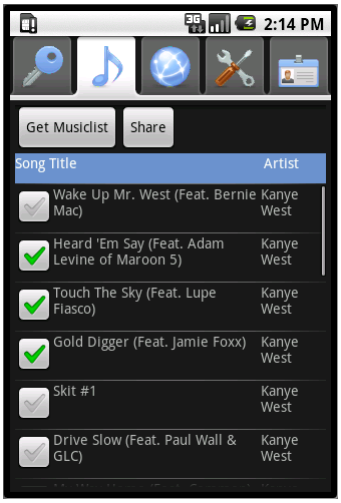

Figure 6. Private music displayed in the mobile application.

Now Bob can choose which songs to add to Charlie's playlist from his music library at home. Bob chooses to add three songs to the common library, clicks Share and is redirected to a list of shared songs, as shown in Figure 7.

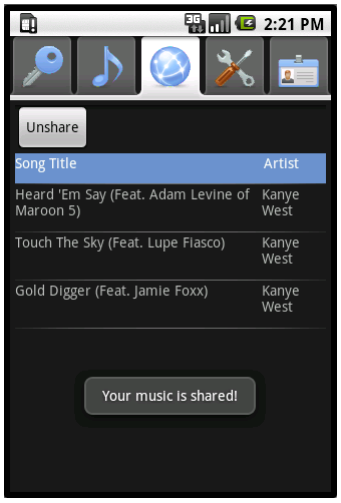

Figure 7. Public music displayed in the mobile application.

Now Charlie has access to Bob's three shared songs and is able to play them directly from Bob's library at home.

The process of retrieving music from the library server to the mobile application and to select what to share, is shown in Figure 8.

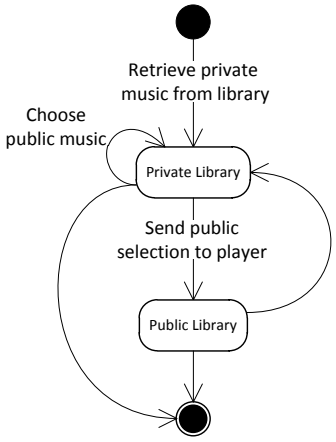

Figure 8. State diagram of the mobile application.

Alice arrives at Charlie's and wants to play some of her own music. Like Bob she uses her mobile to request a connection key and enters it in the player. Her entire music collection is included in her home music library, but Bob and Charlie mostly like pop. Alice's mobile displays her entire music library and she picks out three pop albums which she knows Bob and Charlie like. These are then added to the common library and displayed on Charlie's player.

The functionality of the player, e.g., entering a key to connect a mobile and selecting music for the common playlist, is shown in Figure 9.

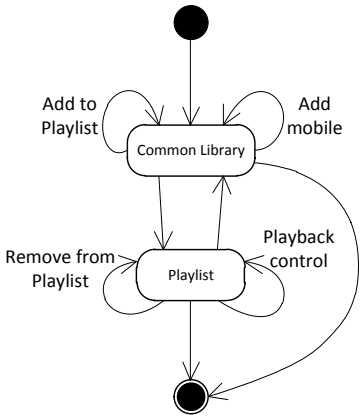

Figure 9. State diagram of the player.

Music Playback

Now the common library contains Charlie's own music as well as three songs from Bob and three albums from Alice. Alice chooses to add some of her own songs as well as some of Bob's, from the common library to the playlist. When the songs are played from the guests' music libraries, the songs are streamed directly from their library servers to Charlie's player.

Disconnect Sharing Connection

It is late and Charlie says goodbye to his guests. Alice and Bob want to make sure that when they leave the party, Charlie no longer has access to their home music library. In the mobile application's menu they make use of the function Unshare *and the player can no longer connect to the library servers at Alice's and Bob's homes, respectively.*

IMPLEMENTATION

MEET fulfills the design goals using the architecture previously presented and the user scenario can be executed successfully using the implemented prototype.

The system is developed using different languages and tools. The player desktop application and the library server is developed in Java, the facilitator is developed in PHP with a MySQL database and the mobile application is developed for the Android platform. An XML-RPC implementation is used by all components of the system to handle remote procedure calls. It is a lightweight protocol that uses XML over HTTP. XML is furthermore used as the data transfer format in other parts of the system, e.g., exchange of song information. The Real Time Streaming Protocol (RTSP) over TCP is used for server and client streaming control, and the Real-time Transport Protocol (RTP) over UDP is used for streaming the actual data.

The system consists of four main components; the facilitator, the library server, the player and the mobile application. The library server and the player are contained in one application but works as two different components of the system. The facilitator manages each music sharing connection and the mobile application administrates what music to share.
Combined, the four components make up a system enabling users to listen to and discover music in a social context.

Connection Flow

Figure 10 shows a sequence diagram visualizing the establishment of a music sharing connection.

To establish the connection several calls are made between the mobile application, the facilitator and the player. The facilitator initially generates a connection key on a mobile application request. This key is entered in the player which updates the facilitator entry, identified by the key, with the player address. The mobile application receives the player address from the facilitator and lastly, the mobile application sends the Library server address to the player. A connection is now set up, the public music list is transferred and available for music streaming from library to player.

Facilitator

The role of the facilitator is to manage the connections created between the mobile application and the player desktop applications. The facilitator holds data related to connection between mobile application and player. Each time a mobile application requests a key, an entry is created in the database which is updated when the key is entered in the player and the player address is sent. This means that for each connection, a database entry, containing a generated connection key and a player address, is created. Figure 11 shows entries in the facilitator database.
The mobile application makes use of XML-RPC calls to request a connection key and later to request the player address. Also, the player makes use of XML-RPC calls when the key is entered and a check for key existence is made to the database.

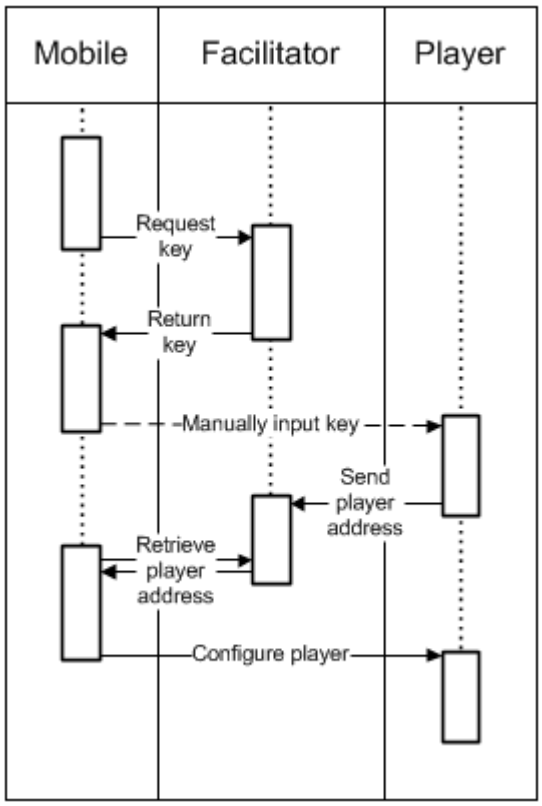

Figure 10. Connection establishment.

Key	Player IP
mY7fri	90.185.90.145
hTR78v	127.147.99.22

Figure 11. Facilitator database table.

Library Server

The library server manages the user's shared music and has functionality to stream music to players over the Internet.
It contains two XML-RPC handlers; one holding Library server methods and one holding player methods. The Library Server lets the user create a music Library, by adding music folders, and the handler contains a method to return this library. This method is used when the mobile application has created a connection with a player, and needs the list of the home music library to further pick out music to share for that connection. For this purpose, the mobile application

needs the IP address from the Library Server. To ease, the configuration the library has a functionality to show this address to the user. With the mobile application's knowledge of the library server's address, and the player's address from the facilitator, a connection for music streaming is created.
To stream music from Library servers to players, an implementation of a RTSP server is made, which uses RTP for actual audio data transfer. The player can start playing the music before the entire track has been transferred. When packets arrive they are added sequentially in a buffer and by the last packet, the music stream is ended. The XML-RPC and streaming functionalities let the mobile application request music from the home music library and enables streaming of music shared from the mobile application from library to player.

Player

The player is a desktop application which visualizes the music from a number of library servers in one playlist. From the users perspective the music is gathered one place, where as it is really distributed over possibly multiple computers worldwide.
The player consists of a GUI and an XML-RPC handler for the library server with two methods. A method to configure the player with the library servers address and another to update the GUI with the shared music list defined on the mobile application. The GUI has two main functionalities:

- *Add Mobile* displays a dialog box where the user must enter a key from a mobile application. When the key is entered the facilitator is called with a check to see if the key already exists. If it does, the mobile application is updated with the player address such that music can be shared.
- The GUI has normal playback functionality, such as play, stop, pause, next and previous song, to be able to play the music in the common library.

Each time the *Add Mobile* functionality is used to create a new music sharing connection, an instance of a custom data model, called *User Connection*, is created. A *User Connection* contains all relevant data to a music sharing connection, i.e., the connection key, the library Server address and the currently shared music list. The connection key works as an ID for the*User Connection*, such that each time the mobile application updates the shared music list, the connection key is used to update the related*User Connection's* music list.

Mobile Application

The mobile application is implemented for smartphones using Android OS. To start a connection, the user requests a key from the facilitator. This key must be entered in the player GUI and from here, the connection setup starts. When the key is entered in the player, the mobile application continues by requesting a list of music from the home music library. From this list, the user can further pick out the tracks which should be shared with the given player. The selected music is added to the common library on the player.
When a user wishes to stop sharing music, the mobile application features a function to call the facilitator to delete the database entry created between the player and the library server. Concurrently, the mobile application also contacts the player to delete the existing instance of the *User Connection* holding the music sharing connection.

SYSTEM SETUP

Before using the system for the first time there is a short setup procedure for each system component. This is a one-time procedure, and the system is afterwards ready for use.

Library Setup

When the library server is installed, the home music library should be defined, selecting the music which should be available via the mobile application. This is done using the *Manage Library* function created to add music folders to the home music library. At any time the contents of this library can be altered.

Mobile Application Setup

To grant access to the home music library the mobile application must be installed on a smartphone. Initially, the mobile application must be configured with the library IP address. This information is easily accessible from the library server desktop application via the *Display Library Address* function. Figure 12 shows screenshots of these functionalities, from the library and the mobile application, respectively. The configuration enables the mobile application to retrieve a list of music from the home music library and provide the player with the needed connection information. The library server's address is saved and is available the next time the application is used.

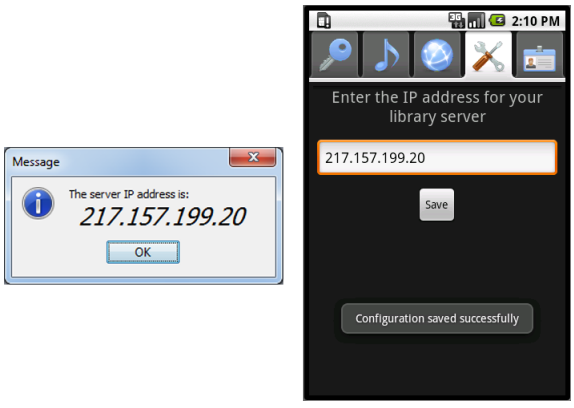

Figure 12. Library IP address entered in the mobile application.

Player Setup

To be able to create a playlist from friends' home music collections, the player desktop application must be installed. Installation of the player application is the only thing needed to host a common music library

SCALABILITY TEST

The main focus of this study has been to develop a technical framework to support further HCI studies within the field of ubiquitous music. Scalability can have significant influence on the general user experience and the ability to conduct a usability field test.

The generated XML file containing music metadata grows linearly to the number of songs. An XML file containing 5,000 songs has a size of approximately 1.42 MB, given that each song has complete ID3 v. 1.1 tags [3]. The test setup consisted of the mobile application running on an Android powered mobile, a varying size of the music library from 100 to 10,000 tracks and a 1 Mbit 3G connection on the mobile. Through several tests, a linear tendency could be observed as expected. A library of 5,000 music items could be retrieved within 30 seconds, however, the 3G connection turned out to be the source of the greatest deviations in transfer time, thus creating time differences of up to 10 seconds for this test condition. The test with 10,000 tracks could not be handled by the mobile application. The shown linear tendency is satisfactory in relations to scalability, however the test has shown that effort must be put into optimization according to the restrictions of mobile resources such as 3G connection and limited memory. Reducing transfer time can significantly improve user experience, taking into account the potential impatience of users. It is however unlikely that the music library is changed while using the mobile application, thus the task of retrieving the music library on the phone is not a frequent task. The task of sending selected music to a player is similar and will probably be used more frequently, but the size of the transfered XML file is inherently smaller due to the double filtering.

DISCUSSION AND CONCLUSION

The MEET prototype is a proof of concept, creating a platform for further development. The current status of the prototype is that it enables users to create a private music library, connect to a player using a mobile device and choose a subset of the library to share with others in a common library. Only the technical backbone of the system was developed meaning little detail was given to, e.g., the user interface.

One of the ideas that has been implemented is allowing the user to perform a double filtering of the music collection. This gives the user a more refined control of what to share, when to share and with whom to share. The system furthermore fulfills the design goal of ubiquitous music by using wireless technologies and streaming to overcome physical boundaries of accessing and sharing music. The need for redundant data is no longer present. As long as an Internet connection is available, users carry along their personal music collection, metaphorically, using the mobile device as an access granting key token.

The test has shown, that effort must be put into scalability issues to obtain a better user experience for large music collections. Additional testing, such as a reliability test, must be conducted to ensure a smooth field test.

FUTURE WORK

The current solution requires the ports used for communication with both the library server and the player to be forwarded. This is acceptable for expert users with access to their own router but mainstream users would probably find this an insurmountable challenge. Exploring the technologies Universal Plug and Play (uPnP) [13] and the Network Address Translation Port Mapping Protocol (NAT-PMP) [7] could yield a solution to the problem.

To explore the community aspect of the system, not only is an improved user interface needed for the existing functionality, but new functionality shall be developed. This can, e.g., be implementing a song voting system for the mobile and making the player prioritize the songs with most votes. Many different voting schemes and user interfaces for the functionality can be imagined on both the mobile and the player.

The technical framework at this point, substantiates future user interface development with the aim at conducting usability tests outside a formal usability laboratory. A field test in a real life situation will be used to evaluate the interaction of the concept, but in order to succeed and obtain useful data, not only does the prototype have to be stable enough to represent a fully functional system, it also needs an intuitive and user friendly interface. To ensure a satisfactory level of stability, a reliability test with several library servers connected to a single player needs to be conducted. The test will be carried out by repeating a playlist, consisting of tracks from the libraries connected, for a number of hours.

These initiatives will be the main focus of the spring semester of 2011 where a workshop will be held in the beginning of the semester with the aim of brainstorming user interfaces for the system. A selection of these will be implemented and a usability field test will form the basis of the final thesis.

ACKNOWLEDGEMENT

A special thanks to our supervisor Jesper Kjeldskov, for support and guidance through this semester.

REFERENCES

1. DLNA. http://www.dlna.org.
2. Grooveshark. http://grooveshark.com.
3. ID3 Tags. http://www.id3.org.
4. iTunes. http://www.apple.com/itunes.
5. iTunes AirPlay. http://www.apple.com/itunes/airplay/.
6. iTunes Home Sharing. http://support.apple.com/kb/ht3819.
7. NAT-PMP. http://tools.ietf.org/html/draft-cheshire-nat-pmp-03.
8. Orb Live. http://orb.com.
9. Orb Live: sharing your media. http://www.orb.com/en/sharing_your_media.

10. TDC Play. http://www.tdcplay.dk.

11. Telenor WiMP. http://www.telenor.dk/wimp.

12. TV2 Sputnik. http://www.sputnik.tv2.dk//.

13. UPnP. http://www.upnp.org/.

14. YouTube. http://www.youtube.com/.

15. A. Bassoli, J. Moore, and S. Agamanolis. tuna: Socialising music sharing on the move. In *Consuming Music Together: social and collaborative aspects of music consumption technologies*, pages 151–172. Springer, 2006.

16. S. Baumann, A. Bassoli, B. Jung, and M. Wisniowski. Bluetuna: Let your neighbour know what music you like. *CHI '07: CHI '07 extended abstracts on Human factors in computing systems*, 2007.

17. J. Brinegar and R. Capra. Understanding personal digital music collections. *Proceedings of the 73rd ASIS&T Annual Meeting*, 47, 2010.

18. B. Brown and A. Sellen. Sharing and listening to music. In *Consuming Music Together: social and collaborative aspects of music consumption technologies*, pages 37–56. Springer, 2006.

19. B. Brown, A. J. Sellen, and E. Geelhoed. Music sharing as a computer supported collaborative application. *Proceedings of the Seventh European Conference on Computer-Supported Cooperative Work*, 2001.

20. S. J. Cunningham, M. Jones, and S. Jones. Organizing digital music for use: An examination of personal music collection. *Proceedings of the 5th International Symposium on Music Information Retrieval*, 2004.

21. M. Hå konsson, M. Rost, M. Jacobsson, and L. E. Holmquist. Facilitating mobile music sharing and social interaction with push!music. *HICSS '07 Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, 2007.

22. D. J. Hargreaves, D. Miell, and R. A. R. Macdonald. What are musical identities, and why are they important? In *Musical Identities*, chapter 1, pages 1–20. Oxford University Press, 2002.

23. L. E. Holmquist. Ubiquitous music. *Magazine: interactions - Ambient intelligence: exploring our living environment*, 12 Issue 4, 2005.

24. A. Lehtiniemi. Evaluating supermusic: streaming context-aware mobile music service. *ACE '08 Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, 2008.

25. R. Sease and D. W. McDonald. Musical fingerprints: Collaboration around home media collections. *GROUP'09: International Conference on Supporting Group Work*, 2009.

26. A. Voida, R. E. Grinter, and N. Ducheneau. Social practices around itunes. In *Consuming Music Together: social and collaborative aspects of music consumption technologies*, pages 57–83. Springer, 2006.