Exploring Crowd-sourced Music Tagging for Explorative Playback: a Prototype and a User Study

HCI - Software P10

Group IS101F17

Aalborg University
SICT
Selma Lagerlöfsvej 300
DK-9220 Aalborg

AALBORG UNIVERSITY
STUDENT REPORT

SICT
Selma Lagerlöfsvej 300
DK-9220 Aalborg
http://moodle.aau.dk

Title:
Exploring Crowd-sourced Music Tagging for Explorative Playback: a Prototype and a User Study

Theme:
Music exploration

Project Period:
10^{th} semester, Spring 2017

Project Group:
IS101F17

Participant(s):
Michel Laden
Mike Nellemann Gregersen
Sean Skov Them

Supervisor(s):
Jesper Kjeldskov

Page Numbers: 36

Date of Completion:
June 15, 2017

Abstract:

Exploring new music on mainstream music services can be a tedious task. Their recommendation system might be inaccurate in their results, and many services require specific searches on songs. We use an already existing concept and create a digital solution of it, constructing it with focus on exploring new music with the help of music tags. In our research of this project we conduct two user studies, one in the lab and one in the field. We achieve useful information regarding Muse as a tool for music exploration. One disadvantage of our studies has proven to be the limitations of our dataset, which only contains 1 million songs, all from 2010 and prior. However, this dataset is able to match with Spotify, allowing for both online and offline usage of the application. In the light of the obsolete dataset, our test participants find high value in discovering and rediscovering music.

Table of Abbreviations

Acronym	**Full word**
AI	Artificial Intelligence
AAU	Aalborg University
HCI	Human-Computer Interaction
IxD	Interaction Design
MIR	Music Information Retrieval
RFID	Radio-frequency identification
UI	User Interface
GUI	Graphical User Interface

Table 1: Table of abbreviations

Contents

Part I

Prelude

1. Introduction

Music has always been a part of human society and have increasingly filled in people's daily life [6]. The availability of music has increased as more music has been uploaded to the internet. Streaming services have access to tens of millions of songs [7, 8], and uses recommender systems to help the users choose among the music, as the selection can be overwhelmed [8, 9, 10]. These recommender systems are based on metadata and information regarding the user's music preferences. In the field of Music Information Retrieval (MIR) there has been conducted research on collecting music, generate its data, and how to recommend music. This resulted in research through artificial intelligence that extract more data from the music, for instance genre and mood [1].

Other approaches have been to crowd-source data collection, where websites like last.fm use social tags to help tagging the music. The idea is the users provide tags on each song as they think the song should be tagged. These tags are what could be interesting to be able to scout music from.

Muse is a proof-of-concept created by a 5^th semester Interaction Design (IxD) students group in 2016 at Aalborg University (AAU) [2], thought up a physical design where the user takes up to three tags and from these are introduced to music matching those tags. Last.fm have introduced an interactive Venn diagram working with music based on tags, much like Muse. However, last.fm limits their search to two tags, decade and genre only [13]. Providing the user with a broad result compared to the Muse concept, where the user can narrow it further by being more specific using these "loose" tags, like moods (fun, sad), language (Spanish, Danish), meaningful (male/female vocalist) and feelings (sexy, awesome), which last.fm does not take into consideration at all.

This project uses the concept of Muse, convert if from its physical form into a digital one, and explore what is possible to do in practice. To be able to see how tags are works on a larger scale than the limited dataset used for Muse. We develop an application and used it in two studies for exploring and get the experience of music in different manner. To find these results, we conduct user studies both in the lab and in the field. That way it is possible to get controlled environment with valuable feedback and how the application will impact the user over time.

This project holds an article, which is written as if it was to be presented at a CHI conference, upholding all the guidelines required from CHI. Other than the article, this projects holds our problem statement, implementation, process, usability issues found, final conclusion and appendix.

1.1 Problem Statement

In this project we answer the following problem statement:

"How can Muse be build as a digital version of the original concept and used in practice?"

We want to build a digital version of the Muse concept which was made by a group of Interaction students on their 5^{th} semester. This is an exploratory and ambiguous way of interacting and playing music. Furthermore, with the goal of investigating the concept it lead to these more specific questions:

1. *What implications there are when using ambiguous tags?*
2. *How does the amount of tags affect the playlist?*
3. *What is the overall user experience?*

Part II

Article

Exploring Crowd-sourced Music Tagging for Explorative Playback: a Prototype and a User Study

Michel Laden
Aalborg University
Dept. of Computer Science
Selma Lagerlofs Vej 300,
DK-9220
Aalborg, Denmark
mladen10@student.aau.dk

Mike N. Gregersen
Aalborg University
Dept. of Computer Science
Selma Lagerlofs Vej 300,
DK-9220
Aalborg, Denmark
mgrege11@student.aau.dk

Sean S. Them
Aalborg University
Dept. of Computer Science
Selma Lagerlofs Vej 300,
DK-9220
Aalborg, Denmark
sthem12@student.aau.dk

ABSTRACT
Exploring new music on mainstream music services can be a tedious task. Their recommendation system might be inaccurate in their results, and many services require specific searches on songs. We use an already existing concept and create a digital solution of it, constructing it with focus on exploring new music with the help of music tags. In our research of this project we conduct two user studies, one in the lab and one in the field. We achieve useful information regarding dMuse as a tool for music exploration. One disadvantage of our studies has proven to be the limitations of our dataset, which only contains one million songs, all from 2010 and prior. However, this dataset is able to match with Spotify, allowing for both online and offline usage of the application. In the light of the obsolete dataset, our test participants find great value in discovering and rediscovering music.

Author Keywords
Genre; music; music playlist; decade; tag; music categories; exploration; physical-to-digital

INTRODUCTION

Historically, music has always been a part of human society and has increasingly become a larger part of people's daily life as the accessibility has increased. Initially the music that has been available to people where music played by themselves, or by other tribe members. It was required to always be a live performance, and for that reason the options were limited for what audience a musician could have. [6]

These options got expanded as travelling became easier and even more when music recording was invented. Now, the musicians did not have to perform live and people could listen to the music without having to leave their house. Today this has expanded to the digital world where almost all music is available online and through music streaming services, with tens of millions of tracks available for their users [15, 7]. The selection of music have left many users overwhelmed with options, leaving some unable to make any decision at all, except for leaving the player on autoplay or playing a select few songs each time.

Some music services attempts to solve this problem, by making recommender systems or other tools to reduce the amount of music presented [14, 7, 13]. The results of these tools are usually based on metadata for the music and information about the user's music preferences. These user data are sometimes compared to that of other users, to categorize them in groups, finding potential music to recommend.

Research has been conducted in the Music Information Retrieval (MIR) research area about how to collect music, generate its data, and recommend music. This has manifested itself through research in artificial intelligence (AI) to extract extra data from the music like genre and mood. [5]

Other approaches have been to crowd-source data collection, where websites like last.fm uses social tags to help categorizing the music. Part of this data from last.fm has been collected to create The Million Song Dataset that was made for research purposes [4]. This idea of using social tags instead of categorizing regular metadata, like genre, to find and search for music, a new proof-of-concept called Muse [1] has been proposed. Muse is a physical interaction design that facilitates explorative, playful and ambiguous playback of music.

In our research, we found potential in the Muse concept and developed a software prototype to explore how it works in practice by applying it to The Million Song Dataset. We aim to see if The Million Song Dataset can be applied to the Muse concept, and if the data itself could be used to add new features. We seek to find out how well a system like this operates, and to find out how much the results are affected by the number of tags used. This lead to our research question:

How can Muse be build as a digital version of the original concept and used in practice?

Furthermore, with the goal of investigating what implications there are when having ambiguous tags, how the amount of tags affects the playlist, and the overall user experience, we have conducted two user studies, one in the lab and one in the field. The feedback is collected through recorded semi-structured interviews.

Initially we introduce related work, then the Muse concept, our two prototypes, followed by two user studies, a discussion, future work, and lastly a conclusion.

RELATED WORK

The literature in this project has focused the field of human-computer interaction (HCI), on music retrieval and ways of tagging music for that purpose [11]. As the availability of music has vastly increased the last few years the commercial interest has increased with it, and from that many online music information and streaming services has emerged [9].

These online services uses varies methods to help the users find new and known music, often based on what they believe the users likes. The online services have also helped providing data for more research in the area of music recommendation and categorization.

Academic Work

The PhD thesis by Daniel Boland "Engaging with music retrieval" [5] describe many areas and ideas in the research area of music retrieval. He talks about how the vast amount of music available has raised a problem of giving users too many choices in what music to play, making them unable to make a decision. He explains how there are two different characteristics in music listening behaviour, casual and engaged. The casual listener will take the minimal effort to find what music to play, while an engaged user will take advantage of all the tools available to find specific music they want to play, like a single album or individual song. Most people inhibit both behaviours, as the level of engagement is situational. He describes why it is important that finding music is easy, as if it is too complicated, the users would simply not bother and instead use simpler methods like playing the same music as usual or putting a playlist on shuffle.

It is explained how recommenders are used to try and fix this problem, by using personal profiles to suggest music for the users. In this approach there has been attempts to solve the long tail problem, which is the problem that recommenders are often biased towards music that is popular, and by that lesser known music never gets recommended. He talks about how making queries for finding and recommending music can be enhanced, by getting more metadata for the music, where he talks about mood and genre which both can be a lot of work to obtain. Here he explains how MIR has made attempts of using AI to classify music, and how it has obtained a greater accuracy than the average human has in making the classifications.

Part of Daniel Boland's work used data from the Million Song Dataset, which is described in the paper by Thierry Bertin-Mahieux, Daniel P.W. Ellis, Brian Whitman, and Paul Lamere in the paper by the same name. It is the largest dataset in MIR and has been made freely available for research purposes. It contains metadata for a million songs, as well as data about similarities between them and crowd-sourced tags from the online music information service last.fm. [4]

The paper "Social tagging and music information retrieval" by Paul Lamere, talked about social tagging of music and how that information can be useful for studies in MIR, as well as some of the problems that occurred from it. Having the crowd make the tags for music can give extra information to analyze music by, like genre, mood, and other information that cannot be obtained through regular analysis of the music itself. It does however also has tags that are irrelevant, has the same meaning, are spelled a little different, or are misspelled. The paper explained that this problem would have to be sorted out, in order to remove some of the noise in the data. [11]

Commercial Application

Spotify is a online music service with over 30 million songs [15]. It uses standard search for finding and exploring specific music, artists, and albums. It takes advantage in recommending tracks to the user based on what they have been listening to while using the service. The recommendations are made in the form of a special playlist with the recommended songs, and is made in comparison with other users. Spotify also have an additional playlist with recommendations for new releases. [14]

Deezer is like Spotify, an online music service and has more than 43 million songs in its catalog. It also uses standard search, but the search has an additional feature, where it is possible to search for music given one or more tags, which is either a genre or decade [8]. Deezer also uses recommendations, but in the form of a constant flow of songs, instead of a playlist like Spotify. The flow of music is based on what music the user has told Deezer they like in the past, by pressing a like button, as well as avoiding music the user has told Deezer they dislike, and by that forms a special profile for each user. [7]

Last.fm is a music information service website that recommends music based on the user's listening habits. The recommender system is called Audioscrobbler and works by keeping track on what the users listen to. Last.fm differs as it does not have any music itself, but instead links itself to other music services like Spotify, Deezer, YouTube, and more. This means it is able to make recommendations, based on what the user listen to on all services, giving it more data to recommend from. In addition to this, last.fm also has a dataset of crowd-sourced tags for the music, that is used for the recommendations, as well as for a way to let the users find and explore music. [13]

MUSE

Muse is a physical music player [1] that introduces a new way to play music in a more explorative and playful way. It does this by making the selection of music more ambiguous by having the user specify up to three tags to play from instead of choosing specific music tracks. These tags can be either a genre, decade, mood, or theme. The design of Muse can be seen on Figure 1.

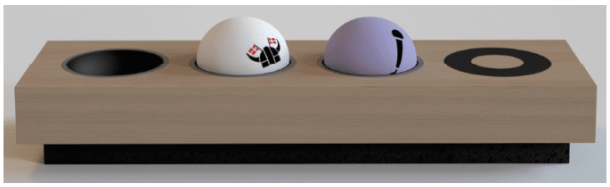

Figure 1: The Muse Music Player

It has a wooden design with three spherical slots, two of which already has a sphere in it, and a wheel with play/pause/skip and volume control. The user selects what music to play by putting spheres into one or more of the slots. A sphere symbolize one of the tags and is identified by the users through the unique design on them that represent the tag. For the example on Figure 1 the tags Danish and Pop has been specified. This means Muse would be playing music that fall into both the category of Danish and Pop. Muse identify what tag the spheres are, by using a Radio-Frequency Identification (RFID) reader in the device and a chip in each of the tag spheres.

Muse was designed in a student project at Aalborg University and was created with the focus on making a physical interactive concept to explore music in a new and playful way. For that reason the project did not focus on testing the system on a large quantity of data. Muse was tested on a small set of songs, each of which had been tagged by the team itself. This allowed the designers to see if the system would work and how the system would be experienced. However, the limited amount of data meant that the designers was not able to tell if the concept would work with a larger dataset as seen in modern music streaming services. It is possible the concept does not scale well. This makes it a good concept to make a follow up study on where the concept is tested in practice with a much larger dataset.

RESEARCH PROTOTYPE

Our project focus on further development and practical use of the Muse concept and studies a prototype that uses a large set of data for a million songs.

First Prototype

As a starting point for the design and functionality of our system, we used the concept of Muse and implemented a digital version of it. The design looked like shown on Figure 2, which is our system with the same setup used for Muse on Figure 1. It has three tag circles that symbolizes the spherical slots from Muse, each with a label for the selected tag, and a circular play/pause/skip button with a circumferencing volume control. When clicking on a tag circle, a list of predefined tags are displayed for the user to select from. We added a wooden background to keep the same aesthetic.

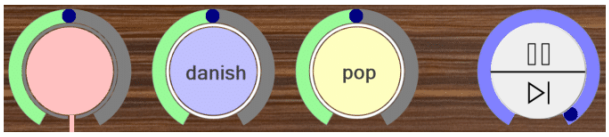

Figure 2: Tags and play/pause part of the dMuse

We connected this simple interaction design to The Million Song Dataset. From that dataset we used information about the title, artist, album, and more for the songs, and the tags associated with those songs. This allowed us to search for songs given the selected tags and allowed us to experiment with the Muse concept for music selection using the large dataset.

Findings

The Million Song Dataset we used, proved to fit well for the Muse concept. It has everything needed to search on tags, as well as enough information to find and play the music. The dataset contained other tags than the traditional ones like genres and decades. Those we named "loose" tags, and consist of tags like "sexy", "awesome", "danish", etc. We also found additional information, such as a value ranging from 0 to 100 of how strong the connection between a tag and a song is. However, we did find some limitations and problems with the Muse concept when using it with a dataset of this size. In our initial step, we used the nine predefined tags that was presented for the Muse concept. This was proven to be a limited selection compared to the 522.366 tags available in The Million Song Dataset. Manually building a list of all these tags would be infeasible, and would create problems when using a single list.

Another problem was, that we could not guarantee that a specific combination of tags would give any results. We analyzed the data by running queries counting all the possible combinations that returns results. We found that the amount of combinations that gave no results far outnumbered the combination that did. For a two tag search only 0,056% gave any results, and for three tags that number was even lower at 0,000013%. Additionally, most combinations that did return results, only had a few songs in them. More than half of the combinations only returns a single song, and there are only few that returns more than ten. We found that the impact the amount of tags combined had on the number results were noticeable, where if searching with a confidence of a 100%, one tag had a maximum of 29.268 result, two tags 1.801, and three tags 409. Lastly the user did not get information about the results when specifying a tag. They did not know how many songs were found for any of the specified tags, or what songs those are. The only information given was the sound of the music currently playing.

Second Prototype

In the light of what we learned from the first prototype, we made a second one with an alternative way of interacting with the data, still closely related to the original concept. This prototype is called dMuse. The new design is shown with a short description on Figure 3. We added a few new features to improve the usage of The Million Song Dataset. The most prominent ones are described in more detail below.

Tag Cloud

By inspiration from Muse and its tag sphere, we decided to implement a tag cloud. We imagined how it would look if we were to store the spheres inside a bag. We simulated this representation by adding a tag cloud that have the labels of the tags in it. The color of the tags are randomized to give variations on the tags, and to entice people to test out new ones. To avoid having the possibility of choosing a tag that returns zero results, it was made such that the tags shown are based on the already selected tags. The tags have different sizes, depending on how many results the tag has when it is combined with the other selected tags. The tag cloud cannot hold all the tags in it, but by using the scroll function on the

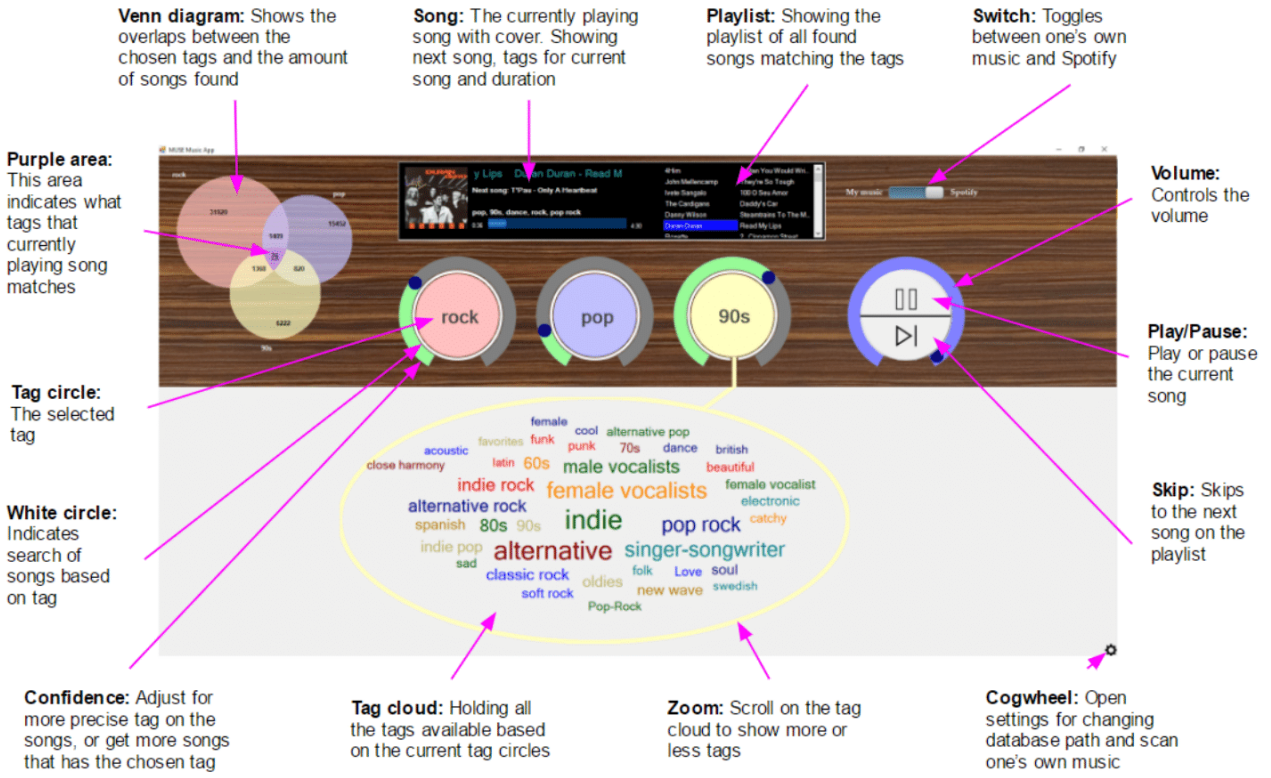

Figure 3: dMuse with explanation of its components

mouse, it is possible, to zoom in and out, to change the number of displayed tags. When a tag is clicked, it gets displayed in the selected tag circle, the search starts, and the possible tags for the next tag circle is displayed in the tag cloud.

Tag Circles

When a tag is selected in the tag cloud, it is represented in the tag circle. A white ring around the tag circle starts to spin, until the search has been completed, to provide feedback of the search process. After that, information about the result is shown in the playlist panel, and the Venn diagram. The search on a tag can be adjusted using the confidence around the tag circle.

Confidence

To make use of the information about the strength of the connection between a tag and a song, an interactive circular progress bar with a handle was added around the tag circle. This allows the user to select how strong the connection between the tag and the songs should be, which is called confidence. A greater confidence would return less songs as it has less mismatches. If confidence is turned down the confidence level goes up, and the user get less songs, and vice verse. When the confidence is changed the search on that tag resets to make the results match the new criteria.

Venn Diagram

To give a quick overview of the search results and provide some feedback about what tags the currently playing song belongs to, we designed a Venn diagram. The diagram consists of up to three circles, one for each tag selected. The circles have matching colors to that of the tag circles to indicate the link between them. In the Venn diagram, numbers are held to indicate the results of the different areas from the tag search. The outer part are the total amount of songs found for a single tag. The intersection between two circles is the amount of songs that have both of those tags, and lastly in the areas covered by all the circles is the amount of songs that matches all three tags at the same time. In addition there is a purple area in the Venn diagram, this area explain what tags the currently playing song belongs to.

Playback Panel

We introduced a panel at the top center of the interface with playback information. This panel holds information about the currently playing song, what song to play next, and a playlist for all songs found through the tag searches. For the currently playing song there is information about the title, the artist, an album cover, and a playback timer. For the next song there is title and artist. The playlist is sorted such that all the songs that covers all tags are at the top and it then builds out, covering fewer and fewer tags, the further down it is progressed. The playlists is also shuffled every time a new search takes place.

Switch
A toggle switch was introduced to allow the user to switch between their own music and Spotify. As the application work on both, the user decides themselves what they like listening to. Being able to listen to their own music can help personalize the use of the application, as well as letting them re-explore their own music library.

USER STUDY 1: LAB EVALUATION

To evaluate how music is explored and how the application is used through the prototype, two user studies were conducted. The first in the lab and the other in the field.

Method

The lab evaluation was a regular lab study method where the participants are tested in a controlled environment. When testing in the lab, distractions and noise are kept at a minimum, and the test participant can focus at the given task at hand. Testing in lab provides good opportunity to get insight into the different features and use. However, it removes the "wild" aspect of natural use from the application [2, 3].

Setup
Before the test was conducted a manuscript were created to ensure the correct preparations were in order:

- Set up computers with headphones and refreshments
- Read practical information for the test participants.
- Provide joint information regarding idea and interaction with the application, as the evaluation does not focus on usability.
- Providing each test participant with a set of tasks for them to solve, created to facilitate interaction.
- When the time is up, the three test participants are interviewed as a group.
- During the interview, the focus will be on the experiences of the program and not its functionality nor usability.
- After the interview is completed, preparation for the next group begins.
- Finally, each interview will be transcribed, and analyzed using grounded theory.

Participants
In the user study, 15 test participants were used, hereof 7 males and 8 females, all ranging between the age of 23 and 35. 8 of the test participants had none to minimal technical/computer skills, hereof 3 males and 5 females, and 7 had semi to high technical/computer skills, hereof 5 males and 3 females.

Procedure
During the user study, three participants were tested simultaneously, they were tested with their back towards each other in a triangle. Having them facing away from each other, and all using headphones help isolating the participants from each other, helping them get their own experience of the application. During the test period, each test participant followed a set of tasks. They were given a pencil and 30-45 minutes to solve the tasks. The test participants did not receive any help except if the application stopped working. While each test participant was using the application, their every move inside the application were logged.

After the test participants finished their tasks, they could play around in the application until everyone was done. When everyone finished, the joint interview began. The test participants were set up in a triangle facing towards the two interviewers. The test administrators focus on the semi-structured interview guide, ensuring the test participants understands and stays on topic. The interviews lasted between 30 and 45 minutes, and was recorded on camera. After the interview ended, the settings of the test area were reset, to prepare for the next group.

Data Collection
We collected data through the joint interviews and the given tasks. The joint interviews were recorded on film and transcribed.

Data Analysis
The method used was a mix of grounded theory from the book Grounded Theory by Glaser, Barney G and Strauss, Anselm L. [10] and Affinity diagram method [16]. The analysis of the findings were made on paper, where all findings were split up and sorted into different codes. Duplicated findings were merged. We used sticky notes to create sub-themes and overall themes for the findings.

Findings

The focus of our findings have been the user experience of the application. How they experience it, how the interaction had impact on them. All of these themes covered an important area from our study, and their sub-themes explained the findings. The findings has been grouped into three major themes, these were as follows:

- **Music**
  - Music Experience
  - Music Categories
- **Context**
  - Collection
  - Interaction Understanding
- **Application**
  - Application Improvements
  - Application Interface

Theme 1: Music
The music theme was about how the test participants experienced the music, and how they discovered/re-discovered music, but also how they understand it. Understanding music, and understanding tags/genres could have a major impact on the general use of the application.

Music Experience
The test participants were delighted about the new way of interacting with music. They found it interesting and exciting to find music using tags. However, from the found interest in discovering new music, a few issues arose as well. The main issue was being unable to rediscover songs or previously found playlists.

After the test participants finished their task they had a tendency of wanting to find music they knew. Some of the participants explained their interest in exploring new music depended on their current mood, thus it was needed to find both music specifically and exploratively. Not all the participants understood how the tags were working in the application. In general most had an idea of what they were getting from searching on specific tags. Most of the test participants had a hard time pinpointing what classified a song into a genre. Some could answer when asked to categorize a song into a genre or decade, while others could not and had no idea what they should anticipate from chosen tags.

The dataset suffered from only having one million songs, as the newest song in the dataset is from 2010. This resulted in the test participants being annoyed by not being able to find music from a newer date. Another issue was that some songs from the dataset did not exist on Spotify, and thus could not be played.

Music Categories
Tags can be understood in various ways, which the test participants mentioned. For instance the tag "sexy" could have multiple interpretations. It could be a sexy song, a sexy voice, a sexy artist or a song containing the word sexy. This was only one tag amongst many, which carries more than one interpretation. We had great input from the participants regarding how to work with the tags. A finding was a function that could save a tag combination into a single tag. This combination could be the favorite tags from a user and allows multiple users to merge their preferred tags and create a combined playlist. To be able to tailor ones search, could open up for a whole different way to use this music exploring application. However, if this were to be included, there should be the option to choose whether the tailored tags be used together as an intersection or a union, to increase the amount of songs found but also to increase precision. However, if this were to be implemented union and intersection should be optional. Implementing this would increase the complexity of the application.

Theme 2: Context
The context theme contains the setting of the application and how it was used. Including the general understanding of the application.

Collection
Through the interviews we found that the test participants preferred to interact with their playlists in multiple ways. Being able to save their playlist, favorite it for future use, and look at rated playlists from other users were suggestions for further development. The only feedback the user gets from the playlist was changing to a different song, seeing what song is currently playing, and which song is coming next. Another issue with the playlist was shuffling. Some test participants wanted the playlist to be sorted, allowing some songs to come concurrently, either by artist or alphabetically. Other test participants wanted to re-shuffle if too many songs with the same artist gathered too close to each other on the playlist.

Interaction Understanding
The Venn diagram was helpful for the test participant regarding understanding the user-interface (UI). It helped them understand what tags the songs that were playing belonged to, and how they were mixed together. The test participants did have a few wishes for the Venn diagram. The test participants wanted to be able to click on the Venn diagram and be allowed to control the output of the music directly. This could for instance be done by clicking on the overlap between two chosen tags, and the application would play the intersection of those two tags.

The tag cloud was met negatively, as many of the test participants found the tags messy with their different colors and sizes, as this restricted their ability to choose tags. It was suggested to keep the tags in clusters, alphabetical, or on a list instead of a cloud, for easier usage.

Theme 3: Application
The application theme contains what could be improved in the application, for instance new features, which could improve the experience of the application, and help to increase the rate of returning users. It is also about how the interface was understood by the participants, both the application itself and the UI.

Application improvements
New features that could improve the user experience were found. For one, application stability. The application was not robust enough as it is, random crashes occurred. Another one was profiling, which would allow the users to tailor the program after preferences, such as favorite tags, songs, playlists, and tailored tags. Another feature found, was to add a fourth tag circle, which could contain; languages, bands, or albums.

The most requested feature found during the interview and lab evaluation was a search field. The test participants want to be able to search for specific music, artists, albums, or tags. It was suggested that the tags to find this song could be presented in the tag circles, allowing the user to find similar music based on a song they like. Another finding was excluding artists or albums from the playlist, allowing the users to avoid songs they do not want to listen to. It was an issue if the user had to skip multiple songs in a row or listen to music they did not like. Hence, a function to exclude artist, albums, or songs could be a solution. The last important finding, was the issue with many duplicates. A duplicated tag could be: "hip hop", "hiphop", and "hip-hop". Each holding some of the same songs, but also different ones, even though they are essentially the same tag. These should be joint to create a better understanding of how searching for music works. We predicted this prior to the study, and it was confirmed doing the evaluation.

Application interface
In the application interface there were interactions that could be difficult to understand how to use. The one with the most issues was the confidence. All but one test participants never figured out how confidence worked. Therefore, it should be more intuitive. It was found that allowing the application to work with smaller screen sizes, such as tablets and mobile phones, will induce use. Responsive design would open for a vast amount of additional users, as users can bring the ap-

plication with them on the go, thus allow more settings for usage. Allowing users to bring the application with them on their phone would give them more interactions and settings than if locked to the PC platform.

USER STUDY 2: FIELD EVALUATION

Method

The field study evaluation was a regular field study method where the user experience was tested in a natural environment. There can be noise and distraction, but for our application it gives a more realistic picture on how it would be used. We did not observe in the wild as the participants were spread across the country, instead we held semi-structured interviews after the test period ended [3, 12].

Setup

The field study was conducted in the test participants home. Before the test, a step-by-step guide was send to the test participants. It included a setup-guide and information about the application. Those who needed help installing the application were guided by phone. The test participants were from different parts of the country to give a larger spectrum of users. They were given no task beforehand unlike user study 1. Every test participant had a log file to provide us extra feedback.

The log showed information about:

- Timestamps
- Tags that were combined
- Play/pause button pressed
- Switch between my music and Spotify
- When a song was skipped
- When volume was changed
- Errors and exceptions

Participants

For the field study, 8 test participants were used, hereof 5 males and 3 females, all ranging between 22 and 54 years of age. 6 of the test participants had none to minimal technical/computer skills, hereof 3 males and 3 females, and 2 had semi to high technical/computer skills, hereof 2 males.

Data Collection

We collected data through logs and interviews. The logs could be used to show how and how much the participants had used the application. The interviews were recorded with a dictaphone and later prepared for the analysis.

Data Analysis

The analysis methods that were used were similar to those in user study 1.

Findings

The focus will be on new findings and recurring findings from user study 1. As in user study 1 there were different main themes an sub-themes, where they each have their own explanation of their given codes found in the analysis. The study provided information regarding the way of exploring tags confirms the findings from user study 1. The three themes are as follows:

- **Context**
  - Context usage
  - Functionality
  - Target group
- **Knowledge**
  - Loose tags
  - Issues
  - Tag understanding
- **Application**
  - Features
  - Library issue
  - UI

Theme 1: Context

The context theme is about the usage of the application and when, how, who, and why it was used.

Context usage

From the log, the usage of the application could be seen. The application was used in a period ranging between two to five days. Over that period participants used the program between half an hour to eight hours per day. It was mainly used in the background during the afternoon while test participants were doing other tasks like homework, eating, or relaxing. One participant also used it in the evening, but most of the others did not. In general the test participants found it useful as a radio that played in the background.

Functionality

The functionality findings were that the music was easily tailored for ones mood, and possibly be a standalone application with its own music library. Another finding was that a test participant mentioned: *"..When I used the application i found old songs that i have not heard for many years, it was surprising that some of these old an good songs showed up.."*. This was interesting because it is possible to explorer forgotten music and not only new and unknown music. Also the test participants expressed the effortlessness in which they could create a playlist, which was usually a hassle with other services.

Target group

A main finding in the target group was the issue when the application was used by more than one person. Test participants found it difficult to decide who should pick the first tag and who the next. When the first tag was selected it narrows down the options for the second tag. This could put the second user in a position where they are only shown tags they do not like. This makes the application difficult to use in some cases of the tag selection. This could state the application as a single-user-application, but needs further explorations and studies to confirm this. We have not focused on multi-user in our evaluation.

Theme 2: Knowledge
The knowledge theme is about the tags of the application and what understanding, experience, and impact they had.

"Loose" tags
In the tag findings, there were also some that were mentioned in user study 1, like some of the test participants also wanted to be able to filter on languages. For the "loose" tags there should be the possibility to like or dislike the tags, which were partly similar to the finding of user study 1: exclude loose tags. A more interesting finding was that special "loose" tags like "sexy" tempts the test participants to try those tags out. In general understanding of the word "tag" was interesting as all the participants always referred to it as "categories". Therefore it make sense to change the name to category instead of tag.

Issues
For tag issues, the problem with the tag precision and the tag cloud was the same as in user study 1. It was mentioned that the tag cloud had a good contrast in the colours, which made it easier to differentiate the tags. It was also found that some tags in sub-tags were identical, which made the test participant confused.

Tag understanding
There were two new categories of how test participants understand the outcome of the music when selecting tags. As in user study 1 there were found the union, the intersection, and the overlap. We introduce two additions, the first is where all the instruments represents the tags. For instance if "classic", "winter" and "pop" were selected, a result could be an orchestra that uses classical instruments with a pop-singer that have made a winter song. An example of such band is Electric Light Orchestra (ELO), that have fused classical and pop instruments. The band was mentioned by a test participant.

The other understanding was the time the artist was active. For instance if a rock-artist was active from "70s" to the "90s" and the given tags were "70s", "90s", and "rock", that artist would be represented in the playlist.

Theme 3: Application
The application theme contains the findings of general application features, library issues, and user interface.

Features
There were a few new findings but also some repeats. For the repeats we had the search field feature and display like fast forward, shuffle and sorting. Some of the new findings were more information about the song/artist. Another was a queue for songs where it is possible to add songs on a list to listen to them later. For the display it was found to add a repeat button.

Library Issue
releases and mainstream music as a choice. There were some general library issues found in the study. The library was huge, but the new and popular music was missing. One of the test participants changed to another music service because of the limitation. To counter this limitation we could include more services to switch between, which would increase the library size. Another finding was that it could be possible to choose among new releases and mainstream music, making it more attractive for an artist or music enthusiast. It would make the searching and exploring new music easier.

UI
For the UI understanding there were some similarities that confirms some of the findings in user study 1, these included improved design, access from mobile, a list instead of the tag cloud, and a larger playlist size. Additionally the test participants found the design easy to understand and use, but there were issues with confidence. A test participant mentioned that it was not intuitive to understand what it should be used for. Another finding was that the application should be accessible from a mobile as a remote control, which could ease the application use, if it was playing in the background like a radio.

DISCUSSION
Overall, our study contributes to understanding how the Muse concept works as a digital version. Here, we discuss **Implications of Tag**, **Implications of Ambiguity** and **User Experience**.

Implications of Tag
Test participants already had an idea of what music that would be presented and played for them, but their expectations were not always met. Some of the participants thought there should be more "rock" in "rock" where others said there was enough.

The precision was inaccurate. For instance a test participant selected some "rap music" and retrieved some "polka". Another one selected "80s" and "rock", where the participant got "90s" music on the playlist. When selecting more than two tags, it also narrowed the third tag circle, and therefore the consequences was a specific third tag circle. This affect the amount of tags to be lowered, making it possible to be specific in the music you want. In addition some of the participants wished to have a fourth tag circle when selecting a tag like "rock" with many sub-tags. However, this would not make sense when choosing specific and finite tags with less sub-tags like "psychadellic" and "chamelleon". This only return one possible tag in the third tag circle, namely "surf rock".

Implications of Ambiguity
During the assignments given under the lab study, the test participants had a hard time understanding what genre and decades were. In the assignment they were asked to insert a genre in a tag circle, and some of them decided to choose a "loose" tag, as they thought it was a genre. Same issue occurred with decades, where the test participants tended to choose a specific year instead of a decade.

This issue was addressed during the semi-structured interview, where it came to light that a lot of the test participants had trouble when trying to classify a genre. However, even though they had a hard time classifying the genre, they still had a presumption of what they would be introduced to when choosing a genre tag. Throughout the entirety of the studies, we were met with a common understanding of the word tag as categories. Most of the test participants did not know each other and were never introduced to the tags as categories. All

referred to the tags as categories. Continuing called it categories even when the test administrators strictly called it tags throughout the entirety of the study.

The test participants had various understanding of how a tag was used in the application. All tags in the application worked as an intersection, except for decades which were as a union. In the end we were able to make five categories of how the participants interpreted the use of tags. The two first ones we found was a union of the three chosen tags and the intersection of them. The third interpretation was as an overlap in decades; if two decades were chosen, then the user would get the end of one decade and start of the next one. The fourth interpretation was that a song was classified in a genre if the song was composed of instruments found in that genre, for instance a rock song having a violin would be able to be found in classic music as well. This is an interesting take, that the instruments solely define the genre and would be able to provide an entire different playlist, however it might not be able to uphold the more standard classification of the genres. The fifth and final interpretation was if a genre was chosen with two decades, then the music the user would be presented with was songs from artists within that genre, which was active within that genre in those decades. This was also an interesting interpretation of the ambiguity of selecting different tags.

These different interpretation of how the tags work with each other, shows that even though we thought that it was a simple setup of tag interaction, it was met differently by our test participants, and would have to be made even clearer.

User Experience

dMuse was a new an interesting way to explorer not only new, but also forgotten music. The participants found it entertaining and easy to use. During the field study some of the participants were new and did not participate in the lab study, however, most of them could easily use the application and figure out how the different functions worked on their own. They found it interesting to work with tags and discover music because it was a new method they have not experienced before. They also found that there were no hassle when creating a playlist, they only needed to choose some tags and leave it in the background. However, some still wished power to be more specific, especially to search for specific music and not get a broad spectrum of unknown music. When it was used in the background it worked more or less like a genre-themed radio station without annoying ads and a radio speaker.

Still some problems were also discovered. It seemed to be a good idea to get music without any hassle, but the consequences was that it can be broad. Therefore many test participants disliked that they could not get new music. They were also annoyed when the selected tags did not provide them with music that they expected, because the tag precision was not perfect. We also found that the more "loose" tags like "sexy", "awesome" etc. were more attractive than other tags like genres and decades. This was because they were new, unknown, and they did not know what to expect.

FUTURE WORK

The dMuse application is still a prototype and requires more attention, in order to be more stable. We present three categories of future work: *the application features*, *context* and *the general music understanding*. One of the main features the application should have to be a final system, is the ability to be responsive to work on other platforms. Other features that must be implemented are all the missing standard music-player features such as fast-forward, repeat, shuffle or sort, and a search field for songs and tags. The user-interface could also be redesign in other ways by interaction designers to give better understanding of the application controls, like the tag cloud, Venn diagram, and tag circles.

In the context part, it could be interesting to create a study that would confirm the hassle of creating a playlist in dMuse is absent, in contrast to using other music services. Another study could be how to use a multi-tagging application with more than one user, as finding of our user study 2 confirmed that there were issues when it was used by more than user. For the tag cloud it would be useful to cluster all similar tags to get a better overview. It would be interesting to expand the tag selection experience, by having multi-combination of tags, for instance a 3x3 where you have "rock", "90s", and "80s" as one, "disco", "pop", and "80s" as another, and lastly "70s", "80s", "metal". This allows multiple users by giving them one tag combination each.

For the music, it was difficult to present new music as The Million Song Dataset lacks this. The dataset also needs to be cleaned and sorted, there were a lot of tags that are similar, which was a known problem [11]. It has also been discussed if the more "loose" tags should be kept or leaved out if they does not make sense for a future application, or instead have a like or dislike function. It would be interesting to offer the user the possibility of saving their tag history and creating a personal playlist. Lastly it should also have the possibility to reverse the music searching process, enabling the user to search for a song and receive the tags that define the song. These tags can then be used to find similar music.

CONCLUSION

This project raised new questions for further studies in the area of music exploration, music listening, and general engaging with music. We developed an application of an already existing concept called Muse, which interaction students from AAU developed on their 5^{th} semester. The application was a functional prototype that is able to play music and presents new ways and thoughts of how one can explore music. It is able to play music from Spotify and ones local music. We used much effort in developing queries that reads The Million Song Dataset, to give the best connection and speed in the app-layer and database-layer.

We did not have an actual focus on the Graphical User Interface (GUI), as the functional part was more important. It was more important to develop and explore the concept to a digital version than replicating the design. The Muse concept was also expanded with a tag cloud instead of spheres, and with confidence to handle the importance of every tag. Another expansion was the Venn-diagram that shows which tag the

currently playing song belong to. The last expansion was the player field and the playlist which shows all the songs that matched the selected tags.

We created two user studies which used a mix of two theoretical frameworks: grounded theory and affinity diagram. they mainly focused on the scientific way of how people interact and understand music in this concept, which underlines the ground of the project. The first user study was an controlled lab evaluation and the second user study was a more free field evaluation. The findings from our two studies can be of worth to the area of music experience in regards to further development of music exploration and understanding.

What we learned was that this 'no hassle' way of listening to music would ease the music experience when listening to music. We also learned that users are different regarding to the way of understanding music and music tags, which was meant to be useful when developing dMuse as an ambiguous concept. dMuse offers not only a music service but a music experience that lets the user explore. The word tag was introduced at the start, but was called a category by all test participants except for one. The "loose" tags were more attractive than the rest, but were also more difficult to understand.

Tags were split into five types of understanding: intersection, union, an overlap between decades, instrumental classification, and artist activity. In general they had an idea of what kind of music that would be presented and played, but their expectations were not always met. They also had a hard time understanding, identifying, and classifying genres and decades. The precision of tags were increased when having three tags, but with popular tags that can potentially return a playlist of 409 song, it would be great to have the option to select a fourth tag to further increase the precision. The test participants did in general find that three tag circles were sufficient.

We learned that users are interested in finding music they already know, therefore it should have the features to search for songs like standard music services already have. A good playlist depend on how the tags were combined. The possibility to switch between intersection and union would give more opportunities for customizing tag searches. Having a history or saving searched tracks would also help.

If there was a search function the user could make the opposite concept. Find a song and get a playlist from the tags of the song, instead of finding songs from tags. This opens up for new ways of listening and interacting with music.

ACKNOWLEDGMENTS

We would like to thank Dimitrios Raptis and Lefteris Papachristos for their help working out the tags in the dataset, allowing dMuse become what it is today. We would also like to thank all our test participants in both user studies for their time and their valuable input. Finally we would like to thank Jesper Kjeldskov for his priceless guiding, without him we would not be able to get this far.

REFERENCES

1. Dong Hyun Kim Tobias Jacobsen Jimmi Bagger Anders Pajbjerg, Kathrine Hansen and Tobias Jørgensen. 2016. Muse: An Interactive Physical Explorative Music Device. *5. semester IxD* 1 (2016).
2. James J. Heckman Armin Falk. 2009. Lab Experiments Are a Major Source of Knowledge in the Social Sciences. Webpage. (2009). Retrieved June 10, 2017 from http://science.sciencemag.org.zorac.aub.aau.dk/content/326/5952/535.full.
3. David Benyon, Phil Turner, and Susan Turner. 2005. *Designing interactive systems: People, activities, contexts, technologies*. Pearson Education.
4. Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. 2011. The Million Song Dataset.. In *ISMIR*, Vol. 2. Unknown, Unknown, 10.
5. Daniel Boland. 2015. *Engaging with music retrieval*. Ph.D. Dissertation. University of Glasgow.
6. Charles Burney. 1789. *A general history of music: from the earliest ages to the present period*. Vol. 1. author.
7. Deezer. 2016a. Find your flow. Press play. Webpage. (2016). Retrieved May 23, 2017 from https://www.deezer.com/features.
8. Deezer. 2016b. Tags in Search. Webpage. (2016). Retrieved May 23, 2017 from http://www.deezer-blog.com/tags-in-search/.
9. Lauren Gil. 2016. history of music streaming. Webpage. (2016). Retrieved June 7, 2017 from https://www.sutori.com/story/history-of-music-streaming.
10. Barney G Glaser and Anselm L Strauss. 2009. *The discovery of grounded theory: Strategies for qualitative research*. Transaction publishers, Unknown.
11. Paul Lamere. 2008. Social tagging and music information retrieval. *Journal of new music research* 37, 2 (2008), 101–114.
12. Tomer Sharon Nigel Bevan. 2002. Field Study. Webpage. (2002). Retrieved June 10, 2017 from http://www.usabilitybok.org/field-study.
13. Varies people. 2016. Last.fm. Webpage. (2016). Retrieved May 23, 2017 from https://en.wikipedia.org/wiki/Last.fm.
14. Spotify. 2016a. Discover. Webpage. (2016). Retrieved May 23, 2017 from https://support.spotify.com/dk/using_spotify/discover_music/discover/.
15. Spotify. 2016b. Spotify About. Webpage. (2016). Retrieved May 23, 2017 from https://press.spotify.com/us/about/.
16. Chauncey Wilson. 2016. Affinity Diagram Process. Webpage. (2016). Retrieved May 23, 2017 from http://dux.typepad.com/files/Method%2022%20of%20100.pdf.

Part III

Project Development

3. Implementation

This section was created to give a better insight on how we implemented different parts of the application. We have divided it into four different sections consisting of **Design**, **Application Features**, **Database**, and **Queries**. **Design** contains the overall design idea for the visual part of the application, **Application Features** about some of the main features the application has, **Database** about the structure of the database and how it was used, and **Query** explain some of the important SQLite queries that we made.

3.1 Design

As explained the in the article the general design was based on the original Muse concept and through iterations additional elements were added to make full use of the concept and dataset. We did not have focus on creating a unique or great design as we only focused on the concept and functionality. The design could be recreated and could be improved in many ways. It was build in Windows Forms, which is outdated and gives limitations for the design. It would be better to make an implementation in Windows Presentation Foundation (WPF) if it should be a Windows application again. Otherwise implementing it as a web application would optimize as it makes it available for other devices than Windows PCs. A web interface would also make it easier to use the web APIs provided by music streaming services, as they do not have an SDK for all languages and platforms.

3.2 Application features

During the development we added features to our prototype, some of which got changed or replaced. Some of these features needed external libraries. What libraries and what additional work we did for each feature will be explained below.

3.2.1 Music player

One of the most essential parts of an application for playing music was the music player. As we both wanted the ability to play music on Spotify and locally stored music, we had to make a player for each. They both worked in a similar ways, but uses different libraries. The music player to use for a song was identified by looking at the source path to see if it was a Spotify link or not.

Spotify

At the beginning we tried to have Deezer as our source for music streaming, but we encountered a problem with only being allowed to play 30 second previews. To fix this we would have needed to use their official music player implemented in JavaScript which was complicated to implement in a C# application. Instead we found an unofficial Spotify API and combined it with data from the official Spotify API, which allowed us to play the full track.

The unofficial Spotify API, is called SpotifyAPI-NET [11], and works by sending request directly to the Spotify application. This was also why it was required that Spotify was running in the background while using our application. When starting our application, the user needs to allow our application

to connect through their Spotify account, which was why it opens the web browser and redirects to Spotify's web page. After the user have agreed, information like the access token will be passed to the API. This was an issue as having a web browser open each time the user start the application can confuse users. A potential fix could be having a web browser inside the application which handles these request through a special window. After the API had the information needed it was possible to play music by sending requests to the Spotify application. For this the API has a few standard calls like play, pause, volume control.

The official Spotify API was used to look up songs to get the playback path needed to start playing them through the player. The request were handled through rests calls. For the playback path request the songs name, artist and album was required.

Local Music

To play music stored locally on the computer, we used a library called NAudio [3]. It can use both local files and streaming. Most music streaming services does however limit the streaming to a 30 second preview for non-official players, which made NAudio unsuitable for that purpose. For playing a local file it needs the its path. We have tested the library on MP3, WAV, and MP4 files, but other music formats were also supported. The player allows for standard playback control like the Spotify API like play, pause, and volume control, but also support skipping to specific points in a song.

3.2.2 Scanning local music

In order to be able to search in the local music, we had to scan them for metadata and use that data to find tags for that song from The Million Song Dataset and last.fm. The scanning was done by having the user select what folder they want scanned. Each file was then first checked up against The Million Song Dataset to find all possible entries that might fit. For this we use the title of the song with different modifications like changing a "&" into "and". The varies results was then compared to the original metadata for the file using the Levenshtein Distance. If any of them beneath a threshold of 50% difference on the title and artist, the one closest was chosen. This was not the perfect solution, but it allow us to find most songs and also reduce the amount of false positives. If a song matched the data, it was gathered and stored in the database inside the application. The database will be described in more detail later.

If the song was not found in this first step, we look it up using the last.fm API. Last.fm have more songs available, making it possible to find most songs. The reason this was the second option, was it only returned up to five tags attached to it, while songs in The Million Song Dataset could have up to a hundred. The procedure for this lookup was almost the same, where we search using different variations on the title and if we found the song we add it to the database for local music. All songs not found in this step will be skipped as we did not have any other options for finding tags.

We made tests with a collection of 4000+ songs, where we were able to find more than 95% of the songs. This collection contained a lot of known songs and the metadata was precise, making them easier to find. It was possible to get a lower hit rate for a collection of lesser known songs and songs with poor metadata.

3.2.3 Tag Cloud

In the beginning we used drop down lists for navigating the available tags. To improve on this we made a tag cloud that gives a better overview and allows for more tags to be shown at the same time. We found a library called Word Cloud [5]. It was a Word Cloud (Tag Cloud) Generator Control for .NET Windows.Forms in C# which takes a strings as input, count all the re-occurrences of each word, and display them in the cloud with varies sizes depending on how many times the word occurred. The library had an object called *IWord*, which contains a words and a count. The cloud uses a list of these

to determine what to display in the cloud and what size. We used this object and filled it up with our tags, and then replaced the count with the number of results the tag was expected to return.

While working with the library we found that it had one error in it. The error was that if there ever were a case where all IWords had the same count for them, the application would crash because the size of the text would be set to zero. Luckily the source code for the library were available and we could fix the code ourselves. The code we modified can be see in Listing 3.1. The changes we made was that we added line 4 and 5. It checks if the error was present and if that was the case the text size was set to the maximum size allowed.

```
1  private Font GetFont(int weight)
2  {
3      float fontSize = (float)(weight - m_MinWordWeight) / (m_MaxWordWeight - m_MinWordWeight)
       * (MaxFontSize - MinFontSize) + MinFontSize;
4      if (m_MaxWordWeight == m_MinWordWeight)
5                  fontSize = MaxFontSize;
6
7      if (m_LastUsedFont.Size!=fontSize)
8      {
9          m_LastUsedFont = new Font(this.FontFamily, fontSize, this.FontStyle);
10     }
11     return m_LastUsedFont;
12 }
```

Listing 3.1: Code modified in the Word Cloud library

3.3 Database

For the application we had two SQLite databases. A small one for local music, which were inside the application, and a large one with all the data from The Million Song Dataset with some modifications. For a future version of the application the large database would have to be accessed through a server rather than on the computer itself.

3.3.1 The Million Song Dataset

As mentioned in the article we used The Million Song Dataset [4] for our data, where we used information about songs and tags. At first we implemented a subset of the dataset consisting of 10.000 songs and it worked we implemented the full dataset. We added a few extra tables to the dataset for some features in the application and the modified version can be seen on Figure 3.1. The dataset originally only had the tables: *songs*, *tids*, *tid_tag*, and *tags*, where songs had information on songs, *tids* had entries pointing at the *track_id* in *songs*, tid_tag a pointer to the rows in tids and tags, and tags with all the tags in the dataset.

We used all data in all tables except from the songs table where we only use *track_id*, *title*, *artist_name*, *release*, *year*, and *duration*. The rest were of no use at for our application.

For the modifications we first added the three tables *TagCloud1*, *TagCloud2*, and *TagCloud3*. These were used to find what tags to show in the tag cloud given what other tags were already selected. They were build using the *tid_tag* table and store the amount of result a tag combination will give, if all tags have a confidence of a 100%. The reason we choose a 100% was that it guarantees that there will always be results, even if the confidence is set to max, and also because it reduced the size of these tables, making the search for what tags to show faster.
Table *TagRelGenre* and *TagRelDec* were added after, as a way to add extra control to what tags to show in the tag cloud. *TagRelGenre* have a relation between genres and their sub-genres, like "rock", with a

sub-genres like "pop rock", "alternative rock" or other. The same apply for the *TagRelDec* table, but with decades instead. The *RelId* value is the main genre/decade, while *TagId* is the sub-genre/decade.

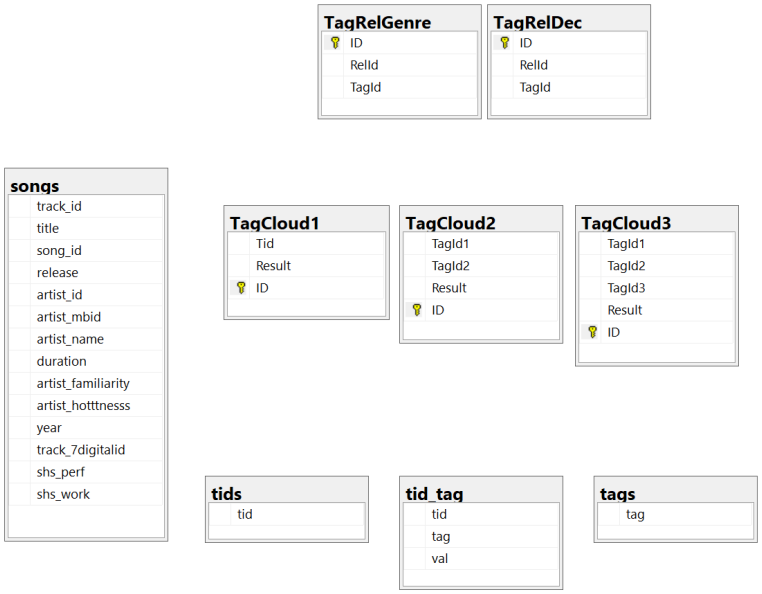

Figure 3.1: Entity model for the modified version of The Million Song Dataset

3.3.2 Local database for local music

The local dataset was similar to the modified version of The Million Song Dataset, with a few changes to make it easier to work with. The entity model for this dataset can be seen on Figure 3.2. The *song* table only have the necessary entries and have a path for where the song was located on the computer. The man in the middle between *song* and *tid_tag* has also been removed by having *tid_tag* link directly to *song*. All tables also use an ID as the primary key, while the tables from The Million Song Dataset all used the row id to find songs. Other than these few differences the table works the exact same way.

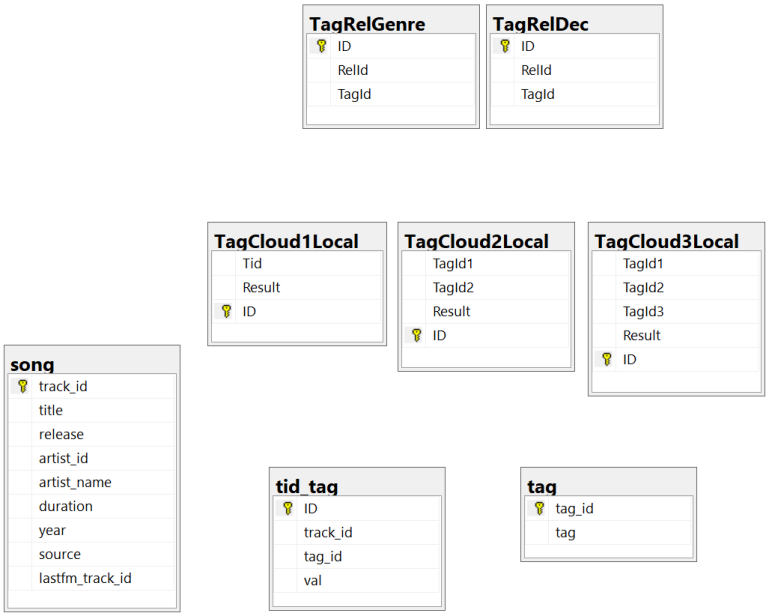

Figure 3.2: Entity model for the local dataset

3.4 Queries

Our queries were one of the major parts of the application because it couples the The Million Song Dataset with the application, and ensures that we can get songs through searching with tags.

3.4.1 Tag Search Queries

First we had two ideas for how searching for songs using a tag could be done. One was to search using all the selected tags at the same time and then only returning the intersection between them all. The other was to search on each tag individual and then take care of the intersection locally. The advantage of the first one was that it was faster as all was done in one single search compared to up to three searches like the second one. The second one has the advantage that a tag can be removed without having to redo the search, that it allows for more local control, and more detailed information about intersections between the results for each tag. We decided to use the second one because it provide the extra information which can be used for better user feedback, and because the search does not have to be redone each time a tag was removed.

When a tag was selected it did not immediately start the search. First there was a check on whether the tag selected was a decade. In this case a sub-query will be made for the decade to be used in the tag search and the application will restart the search for all selected tags. The sub-query makes a check for whether a song was released inside the given decade and works as an extra filter. This was also the reason all tags have to be searched on again, as they have not used this new filter for their original search.

When the check on the decade was done the actually search on the tag starts. The query for this checked for all songs that has this tag associated with them and then filters out the ones that have a lower confidence than the minimum specified by the user. The sub-query for decade will also be included in this search if any of the selected tags are a decade. The result was then returned as a list of songs that can later be compared with the other searches.

3.4.2 Fill TagCloud Tables Queries

There are three tables that contains information about how many results different tag combinations give, namely *TagCloud1/TagCloud1Local*, *TagCloud2/TagCloud2Local*, and *TagCloud3/TagCloud3Local*. The first was for a single tag search, the second for two tags, and the third for three tags. This also means there has to be three different methods to fill out these tables. They all follow the same principle, but the complexity varies.

We start by explaining how the table with a single tag works, as this was the most simple one and the method that the others build upon. The table get its data from a query on the *tid_tag* table. This query looks at all the rows and filter out those that does not have a confidence value above a specified threshold. The results are then grouped by the tag ids with a count on how many occurrences there were of each tag id. These two values are inserted into the table for result count on a single tag. The query for the first tag can be seen in Listing 3.2

```
1 INSERT INTO TagCloud1 (Tid, Result) SELECT tag, COUNT(*) FROM tid_tag WHERE val >=
      [CONFIDENCE_LIMIT] GROUP BY tag ORDER BY COUNT(*) DESC
```

Listing 3.2: Code for interting tag result counts into *TagCloud1*

The second one was a little more complex as it has to take two tags into consideration at the same time. Like the first one it finds all entries and filters out those that does not have a confidence value above a specified threshold. It differs by considering a unique combination of two tag ids. For each possible tag combination in the entries it has to filter out all that does not both match the same song. Afterwards it grouped the entries on the tag combinations and counted how many occurrences there were of each tag combination. The two tag ids and the count for the occurrences were then inserted

to the table for result count on two tags.

The third one was a expansion on the second one where the only difference was that it consider the unique combination of three tags instead of two. This means it ends up with three tag ids and a count for the occurrences of the combination.

3.4.3 Tags For Tag Cloud Queries

When the tag cloud had to be filled with tags, a look up was made in the database, in order to find the tags, that makes sense when compared with the ones already selected. As there were three different tables depending on how many tags to consider, there also had to be three different methods for this. All of these methods follow the same procedure that can be summarized like this:

- Find the IDs for the already selected tags.
- Check if any of them is a genre
- Check if any of them is a decade
- If any of them is a genre find the top 20 sub-genres for it
- Find the top 20 genres
- If any of them is a decade find the top 20 sub-decade for it
- Find the top 20 decades
- Search for additional tags, excluding: genre, decade, sub-genre, and sub-decade.
- Combine all results into a list ordered by number of results and return them

3.5 Usability Issues

During user study in the lab, a few usability issues were encountered. The usability issues found are gathered in Figure 3.3. These issues were collected during the five evaluations in the lab that had three test participants in each, amounting to a total of 15. We will address each of these usability issues below.

Error	Encountered	Level of error
Cloud disappears	2	Cosmetic
Race condition on tags	3	Serious
Socket error	1	Serious
Spotify would not play music	18	Critical
Player stopped (local)	2	Critical
Total crash	1	Critical

Figure 3.3: Usability errors found during lab evaluation

The issue we encountered the most, was Spotify stopped playing music. It was an critical issue when a music service does not play music. We do not really know why the player in the API stops and it happens irregularly, which makes it difficult to track the source of the problem. One of our test participants from the user study in the field said that he encountered this issue, by closing and then opening Spotify and then our application was able to continue working again. We have potentially fixed the source for the issue, but it has not been thoroughly tested and there were still other factors that triggers the problem. This issue can have an impact on the user experience of the application making it a critical problem, which needs to be resolved as soon as possible.

A similar issue we encountered was, the player stopped (local), which was from a test participants who used his own music during the lab evaluation. When this issue was encountered it was first thought of a player issue again, but it seems more accurate to believe that it was because the external hard drive used during the evaluation turned off, as this was observed once during the session. This still have to be tested though as there still could be other factors that could course this problem, like when Spotify stops playing. A warning if the local files were missing, could sort this issue. If the connection to the hard drive was the issue, then it was not as critical as the previous mentioned one.

We were able to fix the issues of the cloud tag disappearing and the race conditions on tags, but it has not been tested thoroughly. The issue with the tags in the tag cloud disappearing happened at random when selecting a tag. At times the shown tags would not be cleared and the text would still be shown, but they would update as soon as the mouse hovered over them, making them disappear. The race condition happened when a user clicked on multiple tag circles in a short time span. They would each start a thread and the tags shown in the tag cloud after would be the tread that finished last, making it possible that the tags would not match the selected tag circle. This was fixed by making a mutex that only let one thread run the code at a time, and a ID for the thread that was known to be the last one started. The last one started would run the code for updating the tags shown, but the others would skip, both to avoid the race condition, but also to save CPU usage.

Total crash of the application happened once during the test period. We do not know if it was a random Windows "Program has stopped working"-error, or if it was a critical issue in the application. We have not been able to reproduce the error and not been able to solve it.

The socket error was when the application tries to login to Spotify multiple times simultaneously on the same port. This was not allowed and crashes the program. This was encountered when clicking on a tag circle before the program had connected to Spotify, triggering the call again. A mutex around the connection code, which only allows one connection at a time, solved the problem.

4. Conclusion

This has been an interesting project to experience and also raised new questions for further studies in the area of music exploration, music listening and general engaging with music. We developed a digital application of an already existing concept called Muse, which interaction students from AAU developed on their 5^{th} semester. The application was a functional prototype that is able to play music and gives new ways and thoughts of how to explore music.

When working on this project we analyzed a lot of data from the user studies, which gave us the opportunity to provide findings in the field of HCI and music-experience. These findings could also open up for new projects and further studies. An example is to create a study that would confirm the hassle of creating a playlist in Muse is absent, in contrast to using other music services. Another could be how to use a multi-tagging music application with more than one user.

We have managed to answer the problem statement, with data from the two user studies.
The word tag was introduced at the start, but was called a category by all test participants except for one. The "loose" tags were more attractive than the rest, but were also more difficult to understand. Tags were split into five types of understanding: intersection, union, an overlap between decades, instrumental classification, and artist activity. The understanding of what a tag was were different from user to user. In general they had an idea of what kind of music that would be presented and played, but their expectations were not always met. They also had a hard time understanding, identifying, and classifying genres and decades.

The precision of tags were increased when having three tags, but with popular tags that can potentially return a playlist of 409 song, it would be great to have the option to select a fourth tag to further increase the precision. The test participants did in general find that three tag circles were sufficient.

A good playlist depend on how the tags were combined. The possibility to switch between intersection and union would give more opportunities for customizing tag searches. Having a history or saving searched tracks would also help.

The general user experience of Muse was that it was an interesting way of exploring unknown music. They found it entertaining, but they wanted the opportunity to search for specific and new music.

For further developing on the application it would need to expanded to a larger audience, and also have added the standard music player features like Spotify has (fast forward, repeat etc.). The further features like reversing the search process to search for music, and retrieve a playlist based on the songs tag. As it is still a prototype there were many things as described in the article and usability issues sections that could make it more stable and valuable for future use.

Acknowledgements

We would like to thank Dimitrios Raptis and Lefteris Papachristos for their help working out the tags in the database, allowing Muse become what it is today. We would also like to thank all our test participants in both user studies for their time and their valuable input. Finally we would like to thank Jesper Kjeldskov for his priceless guiding, without him we would not be able to get this far.

5. Process

In the beginning of this project we planned how to develop this original Muse concept as an digital application. It was an very straight-forward process in the beginning, but it ended up being a more exploratory process, where we had a few bumps on the road to the goal. We read the article from the 5^{th} semester IxD students project where they described the concept in details and how it should work in practice. We included the PhD thesis by Danial Boland 'Engaging with music retrieval' [1] that looks into the problem with 'too much choice' that is the term of too much music, which makes it harder to choose between the music when having all music available.

Figure 5.1: Codes from lab evaluation

We build the application as a Windows Forms application, and used visual studio online with team foundation server as version control. One of the problems when developing this application was the music. Spotify and Deezer which we were interested in using, were closed. In order to have access to their music, you need to implement their official player, or you can only listen to songs as previews (30s). A solution we found was an unofficial Spotify API which connects the application to the Spotify windows application [11]. Another ting was the The Million Song Dataset which was limited from songs to 2010 and earlier.

When we analyzed our findings we used grounded theory and affinity diagram method. When we used grounded theory we identified codes in the statements given by our test participants during the interviews. These codes were then classified into concepts, giving a more broad meaning of the codes. The codes that were coupled together had to show some similarity, see Figure 5.1. When the concepts where created, these were summed up into a category from which we found our general areas. How we searched for similarities can be seen in Figure 5.2. The yellow post-it notes showed the collected codes, the orange the concepts, the green the categories (themes), and the purple the general areas. After multiple it-

Figure 5.2: Data analysis from lab evaluation

erations, where the different codes and concepts were moved around until they were in their right places, we constructed an affinity diagram, which can be viewed in Appendix D.1 and Appendix E.1. This diagram is showing Figure 5.2 in a more readable and understandable manner [14, 15].

When we ended the user studies we also corrected some errors and added features that was found in the first user study.

If the application should be rebuild it would be kind to build it as a WPF (Windows Presentation Foundation) application, which is a newer technology than Windows Forms, or even build as a phone/table/webapplication with a responsive design. This will also make the application be available to a lager audience. From the studies we found that participants wanted to use the application it on their phone, which could be an newsworthy project in itself.

It has been a very interesting project to explore and get results from, especially on the findings part from the user studies, but also the road from having a concept, to bringing it into the world as a prototype. It has been time consuming for weeks when we were developing the application, but the results and our interest drove our motivation. We hope that future studies would dig into our project and get even more findings or other perspecdtives for a music application.

Bibliography

[1] Engaging with music retrieval, Author Boland, Daniel, Year 2015, University of Glasgow Visit: 27-05-2017
http://theses.gla.ac.uk/6727/

[2] Dong Hyun Kim Tobias Jacobsen Jimmi Bagger Anders Pajbjerg, Kathrine Hansen and Tobias Jørgensen. 2016. Muse: An Interactive Physical Explorative Music Device. Device. 5. *semester IxD* (2016)

[3] Naudio: Documentation, Visit: 01-06-2017
https://naudio.codeplex.com/documentation

[4] The Million Song Dataset, Visit: 01-06-2017
https://labrosa.ee.columbia.edu/millionsong/

[5] Word Cloud (Tag Cloud) Generator Control for .NET Windows.Forms in C#, Visit: 01-06-2017
https://www.codeproject.com/Articles/224231/Word-Cloud-Tag-Cloud-Generator-Control-for-NET-Win

[6] A general history of music: from the earliest ages to the present period, Author Burney, Charles, Year 1789, Visit: 10-06-2017
https://play.google.com/store/books/details?id=G-9CAAAAcAAJ&rdid=book-G-9CAAAAcAAJ&rdot=1

[7] Spotify About, Author Spotify, Visit: 10-06-2017
https://press.spotify.com/us/about/

[8] Find your flow. Press play, Author Deezer, Visit: 10-06-2017
https://www.deezer.com/features

[9] Discover, Author Spotify, Visit: 10-06-2017
https://support.spotify.com/dk/using_spotify/discover_music/discover/

[10] Last.fm, Author Varies people, Visit: 10-06-2017
https://en.wikipedia.org/wiki/Last.fm

[11] SpotifyAPI-NET, Author Jonas Dellinger, Visit: 10-06-2017
https://github.com/JohnnyCrazy/SpotifyAPI-NET

[12] Circular ProgressBar for WinForm [.Net3.5+], Author Soroush, Visit: 10-06-2017
https://github.com/falahati/CircularProgressBar

[13] last.fm, Visit: 15-06-2017
https://www.last.fm/

[14] Affinity Diagram Process, Author Chauncey Wilson, 2016. Visit: 15-06-2017
http://www.usabilitynet.org/tools/affinity.htm

[15] The discovery of grounded theory: Strategies for qualitative research, Author Glaser, Barney G and Strauss, Anselm L, Publisher Transaction publishers, 2009. Visit: 15-06-2017

A. Assignment for User Study in Lab

Assignment 1:
- Select a genre in one of the circles
- Listen to at least 4 tracks. You can skip to the next number whenever you want, but we would like you to listen to each number for at least 1 minute.
- Did these numbers match the chosen genre?

Yes ☐ No ☐ Partly ☐

Assignment 2:
- Select a new genre and choose a decade
- Listen to at least 4 tracks, as in assignment 1
- Are these songs matched with the chosen genre and decade?

Yes ☐ No ☐ Partly ☐

Assignment 3:
- Select a new genre, a new decade, and anything else of your choice in the last circle
- Listen to at least 4 numbers, as in previous assignments
- Are these numbers matched with the chosen genre, decade and tag?

Yes ☐ No ☐ Partly ☐

Assignment 4:
- You have genre, decade and tag
- Remove now decade and add a new tag
- Listen to at least 4 numbers, as in previous assignments
- Are these numbers matched with the selected tags?

Yes ☐ No ☐ Partly ☐

Assignment 5:
- Remove all three tags so that nothing is selected
- Choose a combination of three tags yourself. Start from the left
- Set the importance of the three tags
- Listen to at least 2 numbers, as in previous assignment
- Reduce the importance of the three tags
- Listen to at least 2 numbers, as in the previous assignments
- Are the numbers matched with the selected tags?

Yes ☐ No ☐ Partly ☐

Assignment 6:
- Please remove all tags so that nothing is selected
- Select 1 to 3 tags, and their individual importance
- Experiment free with tags and their importance, as you like
- Listen to at least 10 numbers, as in previous tasks
- Are the numbers matched with the selected tags?

Yes ☐ No ☐ Partly ☐

Figure A.1: Assignments for given during lab evaluation

B. Interview for User Study in Lab

1. General music experience
   - E. Why use MUSE?
     - a. What is nice?
   - F. What did you like the most of MUSE?
     - a. Features, functions, music, autoplay, design, .. ?
   - G. Did MUSE play the music you expected?
     - a. If no, what were not expected?
   - H. Did MUSE play music that was surprising?
     - a. Did MUSE play music you did not know, but found interesting or did like?
     - b. Did MUSE play music that did not fit?

2. Playlists
   - F. Were there too much music on the list?
     - a. no? too little? why?
     - b. Should the playlist be smaller?
     - c. Should the playlist be bigger?
   - G. What does it take to get a "good" playlist?
   - H.
   - I. Would it be important for you that MUSE could distinguish between known and unknown music?
     - a. Exclude an artist because you do not know him?
     - b. Preference list?
   - J. How did you feel the music match of what you have chosen?
     - a. Is there enough "Rock" or should there have been less? "Rock"?
   - K. Were there anything that missed about the suggested music?
     - a. Funktioner, features, different music, something you know, something new?

3. Tags
   - A. How did you experience the possible tags regarding to your music library?
     - a. Does it make sense?
     - b. Were there something that surprised your?
   - B. How are your understanding/experience of the chosen "tags" you can pick?
   - C. How do you experience the more solve tags? Would you prefer that there only were like genre and decades?
   - D. Are there too few/many number of tags to choose among compared to your expectations?
   - E. Would it be enough with two tag possibilities or would u prefer more?
   - F. What do you expect to listen to with the "rock", "pop", "punk" tags as the chosen ones?
     - a. What characterize a song a "rock" song?
     - b. What characterize a song a "love" song?
     - c. What makes a song an "oldie"?

Figure B.1: Questions for the joint interview in lab

4. What would you change
   A. What is the best thing about MUSE compared to other known music players?
   B. What is missing in MUSE compared to known music players?
   C. What could improve the usage of MUSE?
      a. What is the worst part about MUSE?
   D. What do you feel is missing from MUSE?
      a. Is there any important quality of life items missing?
      b. Were the any information you felt was missing?
         i. More tags, more information of the playing song?
      c. Where there any functions you felt was missing?
         i. An extra tag circle?
   E. What would make you buy MUSE?
      a. Functions?
   F. Would it be important for you that MUSE could be coupled with other music services?
      a. Which?
         i. Youtube?
         ii. Deezer?
         iii. Others?

5. Tag understandment
   A. How do you understand the word tag?
   B. What do you expect to listen to if you are presented with the following tags: "80s rock" og "90s"?
   C. "70er + 80er + 90er"?
   D. "Classic, Winter, Pop"?
   E. "spring+summer+winter"?
      a. MORE!!
   F. How do you understand the following tags: "80","90" og "rock"?

Figure B.2: Questions for the joint interview in lab

C. Interview for User Study in Field

0. Field brug
   A. Hvordan har du brugt MUSE?
   B. Hvornår brugte du mest MUSE?
   C. Hvornår var det fedest at bruge MUSE?
      a. Hvornår var det ikke?
   D. Hvor meget har du brugt MUSE?
      a. Check log, se forbrug, og clicks

Hvis MUSE ikke har været brugt
   A. Hvorfor har du ikke brugt MUSE i testperioden?
   B. Hvad afholder dig?

Figure C.1: Additional questions for field, also using same questions from lab

D. Affinity Diagram from Lab

Application	App improvements	New features	App Stability
			Profiling
			Extra Tag-circle
			List instead of Cloud
			Discover while listening to music
			Search field for music and tags
			Display features (show current song, duration, fast forward, release year)
			Include more services (Mix local with service)
			Remove duplicate songs and tags
			Exclude artist and (loose) tags
	App interface	App Understanding	App usage
			Confidence understanding and misunderstanding
			App target group
			Intersection, Union & Overlap
		UI Understanding	Bad user interface
			Introduction
			Editable interface
			Responsive design and mobile integration
			Larger playlist size, playlist paging, larger music display
			Intuitive user interface
			Tag cloud understanding
			Intuitive Venn Diagram
Context	Collection	Playlist	Favorite playlist
			Rated playlist
			Save playlist
			Shuffle, anti-shuffle and mix albums
			View artist albums
	Interaction	Venn Diagram	Interaction with Venn Diagram
			Information and percentage on Venn Diagram
		Cloud	Amount of tags in Cloud
			Clusters in Cloud
			Year with sub-year
			Cloud always visible
			Empty Cloud issue
			Cloud tags size and color
Music	Music Experience	Discovery	Rediscovery issue
			New way to interact with tags
		Music Understanding	Playlist makes sense of tags
			Music flow
			Unknown and known music
			Button for Union and Intersection
		Song Improvement	Lyrics
			Similar songs
			Music videos
			1 mio. songs limitations, newer library, bigger library
	Music Categories	Tag	Tag history
			Tag understanding and misunderstanding
			Combination of tags, subtags and tailoring tags
			More tags, more information and improved precision

Figure D.1: Findings from user study 1 interviews using affinity diagram

E. Affinity Diagram from Field Study

Application	App improvements	Features	Search field for music and tags
			Display features (fast forward, repeat, shuffle, sorting)
			Queue for songs
			More song/artist information
		Library Issue	Add service to increase library
			Change service for specific music
			Include new releases and mainstream music as choices
	UI understanding	UI	Improved design
			Access MUSE from mobile, like a remote
			List instead of cloud
			Confidence bar issues
			Easy understandable design
			Playlist sizes
Context	Current context	Field usage	Used between 2-5 days
			Used like a radio
			Time of use (Afternoon)(evening)
			Used in the background
		Target group	Issue when used by more than one person
		Functionality	Music is easily tailored for one's mood
			Unique way to combine tags than other services - Standalone
			Exploring new and old music
			Differ between known and unknown
			No hassle when creating playlist and easy to use application
Knowledge	Tag	'Loose tags'	Likes / dislikes 'Loose' tags
			'Loose' tags tempts the user
			Language tags
			Good to discover new tags/genres
			Collect tags in a more broad sense
		Issues	Tag precision issue
			Identical tags
			Tag cloud
		Tag Understanding	Union
			Intersection
			Overlap end 80s, start 90s
			Songs that holds instruments fitting for the chosen genres (intersection)
			Artist was active in the period (Intersection)
			Tag understanding/misunderstanding

Figure E.1: Findings from user study 2 interviews using affinity diagram