KU LEUVEN

FACULTY OF SCIENCE

”This DJ is very bad”: Exploring the design of control features for group music recommender systems during social gatherings

Tom MARTENS

Supervisor: Prof. dr. K. Verbert
Mentor: *Ir. O. Luis Alvarado Rodriguez*
Assessor: *Dr. D. Sileo*
Assessor: *Prof. dr. ir. B. De Decker*

Thesis presented in fulfillment of the requirements for the degree of Master of Science in applied informatics: Artificial intelligence

Academic year 2020-2021

Preface

This thesis is conducted at the faculty of Science by a student of applied informatics, option: artificial intelligence. It was a very interesting and educational process.

First of all, I would like to thank Oscar Luis Alvarado Rodriguez for mentoring me during this thesis. His provided insights were a significant contribution to this work. The weekly meetings were very useful in keeping me motivated and keeping me on track. Furthermore, I am very grateful for prof. dr. Katrien Verbert's and the augment group's feedback.

I would also like to thank the KULeuven for this opportunity for writing this thesis. This final work will also be the final step for graduating. That is why I would like to thank my parents who gave me the opportunity to achieve this diploma. Finally, I am thankful for everyone who participated, supported me or showed interest during the thesis.

Tom Martens

Contents

Abstract

Interest in group recommender systems is growing in the past years and as it seems this not going to stop anytime soon. A lot of existing group recommender systems appear as "black boxes" for the users. They don't know how the recommender system comes up with recommendations or how to interact with it. Existing research already investigated possible ways to improve the transparency, controllability,... of individual recommender systems. However, a great deal of this research is still lacking for group recommender systems. This research is a contribution on designing possible controls for a mobile music group recommender application and their possible influence on the social dynamics of the group. Two prototypes were created and improved throughout three iterations. The first iteration was executed to find the expectations of potential users regarding the controls of a group music recommender system. The researcher also presented two low-fidelity prototypes with different types of controls, to collect feedback. One democratic version, where votes are needed for songs to be played and one veto-oriented version, where there is one member with almost all responsibility regarding the music being played. A thematic analysis revealed that such an application should require little attention, as the focus should be on socializing. The results of the first iteration were used to create two high-fidelity prototypes, one democratic and one veto-oriented. Both prototypes used the Spotify API to play songs and to request the logged-in user's top tracks. A second user study revealed the low cognitive load and high usability of both versions. A low cognitive load and high usability implies that the user needs to spend less effort and time on the application, causing more time for socializing. The obtained scores of both versions were too similar to make a final decision on which prototype is best. The effect of a music group recommender system on the social dynamics is discussed in the third iteration. It is a subject where there is not a lot of research about. As this iteration is an exploratory study it revealed some interesting results, which require further investigation. The existence of a music group recommender system can positively influence the integration of members in the group. Requesting songs is an easy topic to start a conversation. Also, participants mentioned feeling more connected to a group with similar taste in music. Both applications have a functionality where songs can be requested. The rejection of a requested song might negatively influence the group-feeling and behaviour of a user. It is therefore important to take this into account when designing such a functionality. The democratic version has the most positive impact on the social dynamics, this is why it was preferred over the veto-oriented version in general.

Samenvatting

De belangstelling voor groepsaanbevelingssystemen is de laatste jaren toegenomen en het ziet er naar uit dat deze belangstelling niet snel zal verdwijnen. Veel van de bestaande groepsaanbevelingssystemen lijken een magische doos voor de gebruikers. Ze weten niet hoe het systeem tot aanbevelingen komt of hoe ze met het systeem moeten interageren. Bestaand onderzoek heeft reeds mogelijke manieren onderzocht om de transparantie, controleerbaarheid,... van individuele aanbevelingssystemen te verbeteren. Uit veel van dit onderzoek ontbreekt echter nog voor groepsaanbevelingssystemen. Dit onderzoek is een bijdrage tot het ontwerpen van mogelijke controls voor een mobiel groepsaanbevelingssysteem en de mogelijke invloed hiervan op de sociale dynamiek van de groep. Twee prototypes werden geconstrueerd en verbeterd gedurende drie iteraties. De eerste iteratie werd uitgevoerd om de verwachtingen van mogelijke gebruikers ten aanzien van de besturing van een groepsaanbevelingssysteem voor muziek te achterhalen. Om feedback te verzamelen presenteerde de onderzoeker ook twee mogelijke prototypes met verschillende soorten controls. Een democratische versie, waar stemmen nodig zijn om nummers af te spelen en een veto-gerichte versie, waarbij er een lid is met bijna alle verantwoordelijkheid over de muziek. Uit een thematische analyse bleek dat een dergelijke applicatie weinig aandacht zou moeten vereisen, omdat de nadruk moet liggen op het socialiseren. De resultaten van de eerste iteratie werden gebruikt om twee high-fidelity prototypes te maken, een democratische en een veto-gerichte. Beide prototypes gebruikten de Spotify API om liedjes af te spelen en om de topnummers van ingelogde gebruikers op te vragen. Het tweede onderzoek toonde de lage cognitieve belasting en hoge gebruiksvriendelijkheid van beide versies. Een lage cognitieve belasting en hoge gebruiksvriendelijkheid impliceren dat de gebruikers minder moeite en tijd moeten besteden aan de applicatie, waardoor er meer tijd overblijft voor het socialiseren. De behaalde scores van beide versies waren te gelijkaardig om een eindbeslissing te nemen over welk prototype het beste is. Het effect van een aanbevelingssysteem voor muziekgroepen op de sociale dynamiek wordt besproken in de derde iteratie. Het is een onderwerp waar nog niet veel onderzoek naar is gedaan. Ondanks dat deze iteratie een verkennende studie is, heeft ze enkele interessante resultaten opgeleverd, die mogelijks verder onderzoek vereisen. Het bestaan van een aanbevelingssysteem voor muziekgroepen kan een positieve invloed hebben op de integratie van leden in de groep. Het aanvragen van liedjes is een gemakkelijk onderwerp om een gesprek over te beginnen. Deelnemers gaven ook aan zich meer verbonden te voelen met een groep met een gelijkaardige muzieksmaak. Beide toepassingen hebben een functie waarmee liedjes kunnen worden aangevraagd. De afwijzing van een aangevraagd liedje kan een negatieve invloed hebben op het groepsgevoel en op het gedrag van een gebruiker. Het is daarom belangrijk hiermee rekening te houden bij het ontwerpen van een dergelijke functionaliteit. De democratische versie heeft de meest positieve impact op de sociale dynamiek, daarom kreeg deze uiteindelijk de voorkeur boven de veto-gerichte versie.

List of Figures and Tables

List of Figures

List of Tables

List of Abbreviations and Symbols

Abbreviations

RFID	Radio-Frequency Identification
SUS	System Usability Scale
TLX	Task Load Index
API	Application Programming Interface
JSON	JavaScript Object Notation

Chapter 1

Introduction

The introduction describes why this work is relevant, which specific research questions were answered and the methodology to find these answers. To conclude this introduction, an overview is given of all chapters and a brief intro into these.

1.1 Motivation

Overall recommender systems are used to help the user make the right decisions. It does so by lowering the information overload of all possible items by recommending the items which it thinks fit the most for the current user. Depending on the feedback of the user it will change the recommended items or it will recommend similar items. There are a number of different activities where such recommender systems can be used, for example listening to music, selecting a series or a movie to watch, choosing what to eat,... These examples are things that an individual can do, but are also activities which can be performed in group. This is a reason why an interest is developed in group recommender systems. This interest in group recommender systems is growing the past years and as it seems it is not going to stop anytime soon [17]. Making decisions in a group is more difficult than making individual decisions. The group recommender system provides help. It will suggest an item for the group by taking into account the preferences of all members of the current group. This way it lowers the information overload for the group and helps the group in making the right decision. The way the individual preferences are used in the group recommender system depends on the used aggregation function. There are also other factors which need to be considered such as the context and group dynamics.

Group recommender systems are deployed in a lot of different domains and a lot of applications are jumping on the train in developing their own group recommender system. Music applications like Spotify[1] are adding more and more functionality for groups to listen to music together. They implemented a group session[2] where it is possible for a group to listen to the same music. They also offer possibilities to make music playlists together, which can be listened to individually or in group. On top

[1]https://www.spotify.com
[2]https://support.spotify.com/us/article/group-session/

of this they provide a ’Family Mix’[3], which is a playlist for a group based on the individual musical preferences.

Because of the growing popularity of these group recommender systems, it is important to develop the best possible interface. This work contributes to this goal, by focusing on the controllability of a mobile music group recommender system. The effect of different types of controls on groups of people was measured in terms of cognitive load and usability. These are important elements to measure, because the attendants of a certain social event should focus on the event, instead of focusing on their smartphone. The implemented interface requires little focus (low cognitive load) and is easy to use (high usability). To the best of my knowledge, the influence on group dynamics are not yet considered in any other papers about music group recommender systems. Cohesion is one of these group dynamics and as this is an important feature of a group, this research explored the influence of the interface on group cohesion.

1.2 Problem statement

The influence of controls for a mobile music group recommender system on the cognitive load, cohesion and usability of groups hasn’t been studied before to the best of my knowledge. There are a lot of aspects which need more investigation regarding music group recommender systems and group recommender systems in general. Controllability is one of those aspects. This study covers how to develop controls for a mobile music group recommender systems and measure what the effect of these controls are on the group. Specifically, we tried to answer these research questions:

1. Which are people’s expectations when attending a social event (party) regarding different levels of control for a mobile music group recommender system?
2. In what ways do different levels of control affect cognitive load and usability for a group of users on the mobile group recommender system?
3. In what ways do different levels of control affect group cohesion?

1.3 Methodology

These research questions were answered through three different user studies. Each of them focused on answering one question. The first iteration answered the first research question by gathering the expectations and the opinions on the first prototypes of the potential users. With these expectations in mind, the researcher updated his prototypes. The second user study investigated the usability and cognitive load of the finished prototype. The final user study was performed to find the influence of the interface on cohesion. User study 1 and user study 3 were conducted in groups of

[3]https://support.spotify.com/nl/article/family-mix/

people, because this is the way to design the best possible interface for a group. All three user studies will be explained more thoroughly in the corresponding sections.

1.4 Overview

An overview of previous research is given in the literature review (chapter 2). Individual recommender systems and there recommendations strategies are discussed. Next, it explains how group recommender systems are build and what aggregation methods are used to recommend items. Some state-of-the-art group recommender systems are shown as well. Furthermore, the chapter shows how important the interface of such a system is, with the focus on the controllability. Existing research about controllability is used to highlight this importance. Chapter 2 is finalised with an introduction into the theory about group cohesion.

Chapter 3 discusses the design and results of the first user study. This iteration aimed to find the expectations of the participants regarding a mobile music group recommender system's interface. Two prototypes were created as examples. An explanation of the design decisions and the functionality of both prototypes is included. The findings were examined using a thematic analysis.

The results of the first user study were used to develop two high-fidelity prototypes, which were used in the second iteration. The design of the prototypes and the user study are explained in chapter 4. The second iteration's aim was to find out whether both prototypes were easy to use and required a low cognitive load. The results of this iteration are discussed at the end of this chapter.

The final user study was performed to find the influence of such a mobile music group recommender system on the social dynamics of the group. Chapter 5 explains the design of this iteration and a thematic analysis was used for the processing of the results.

Chapter 6 connects the results of all iterations. It gives general insights obtained in this research together with an answer on the research questions.

Finally, this work is concluded with chapter 7. This part summarises the results of the three iterations. Suggestions for further research are also included in this chapter.

Chapter 2

Literature review

This part is subdivided into five different sections. It starts with a small introduction about individual recommender systems. Next, group recommender systems will be discussed, with the focus on how they can be generated and what the implications are of such systems. This part is followed by some existing work about recommender systems with the focus on controllability. Finally some theory about cohesion is presented. Each of these sections are supported with existing research. This literature study is summarised in the conclusion.

2.1 Individual recommender systems

Recommender systems are algorithms which are designed to help the user to choose the right item from a big list of items. It does so by providing the user with the most relevant items from this list [42]. This way it avoids information overload for the user and it makes the user able to go through the relevant information in an efficient way. Such an algorithm is applicable on a lot of different domains. Companies like Spotify[1], YouTube[2], Netflix[3] or Amazon[4] already use it to suggest relevant items to their customers.

There are different methodologies to recommend items to the users. They differ in the way of selecting items for recommendation. Based on this difference, there are three different methods: collaborative, content-based and knowledge-based filtering. There are also hybrid algorithms which combine two or more techniques to profit from the benefits of each technique or to avoid disadvantages [4]. In the next part, each of these methods is briefly introduced.

2.1.1 Collaborative filtering

Recommender systems that use collaborative filtering look at past interactions between users and items to detect similar users/items and make predictions based

[1] www.spotify.com
[2] www.youtube.com
[3] www.netflix.com
[4] www.amazon.com

on this similarity. It calculates recommended items using ratings given by the user [4]. Amazon uses a collaborative filtering approach by finding items that are similar to the items that the user rated in the past or has bought before [25].

Depending on whether the system creates a model from the user-item interactions or not, the system is memory-based or model-based. If the algorithm needs to record all interactions and uses a nearest-neighbour algorithm to find similar items or users, it is memory based. If it generates a model from the user-item interactions, it is model-based [33]. There are two approaches in the memory-based method: user-centred and item-centred.

The user-centred approach follows the assumption that people with similar tastes will rate things similarly [41]. In this case, certain items are recommended, because similar users liked these items. Users are said to be similar when they gave the same rating to certain items. When the recommender system wants to predict the rating of a certain item, it will look for the ratings of similar users. Based on these ratings the recommender system predicts the grade for the item. In the music domain this means that the system recommends music based on songs the user listened to or liked in the past. The system will then look for other users who like the same kind of music. These other users' music is then used recommended to the current user.

In the other case of the item-based approach, items are recommended because of similarities between the items. The prediction for the rating of an item is calculated from the ratings of similar items by the current user. Items are said to be similar when most of the users interacted with them in the same way. On the music domain this can be applied as recommending a song when it is similar to a liked song of the current user. [40]

The model-based approach generates a model from the past user-item interactions and predicts or recommends items based on this model[33].

2.1.2 Content-based filtering

Content-based filtering recommends items to users based upon a description of the item and a profile of the user's interests [4]. The algorithm collects the characteristics of items that the user likes and creates a user profile based on these. When a new item is discovered, it is compared with this profile. If it matches the profile, then it is recommended to the user. The profile of the user can be updated and changed through feedback, provided by the user [37] [32]. Items are recommended because they have similar characteristics of items that the user liked in the past. It assumes that items which are alike will all get similar ratings by the user. In Spotify there are different characteristics which describe songs, some of them are energy, danceability, instrumentalness,... When a user only likes songs with high danceability, then the suggested songs will have high danceability.

2.1.3 Knowledge-based filtering

This algorithm uses knowledge about users and products for generating recommendations. It expects that the user already has some domain knowledge [38]. The user must either apply constraints or requirements. In both cases the user gets recommendations which fulfil these requirements or constraints [17]. An example of a movie, knowledge-based recommender system (using requirements) asks the user first which movie he/she likes and based on this movie it recommends similar movies. When the user isn't satisfied with the movie because it is isn't funny enough, he/she has the possibility to request another movie by adding the "comedy" feature [10]. The same reasoning can be applied for the music domain. So the knowledge based algorithm makes it possible to request similar items as the provided item and to add filters.

2.2 Group recommender systems

There are a lot of situations where a group of users has to decide on a certain topic, like selecting a movie to watch or selecting a song to listen to. Group recommender systems are developed and being updated to help in this decision process. Group recommender systems are very similar to individual recommender system in the aspect of recommending items. Individual recommender systems therefore form the basis for the group recommender system.

However there are a lot more aspects which need to be considered in a group. The satisfaction of the recommended item can differ for an individual or for a group, because members influence each other. When one member strongly dislikes a certain item, it is possible that other members will dislike it more than they would if the item was recommended to them individually. The opinion of other members can also have an impact, this is called conformity [26].

Together with these aspects, the recommender system must take into account all the member's preferences and suggest items based on these. There are some issues on how to combine these individual user models. The individual preferences are either collected using collaborative or content-based filtering, which are explained in the previous section. There are two strategies to combine the individual preferences, aggregating recommendations or aggregating profiles. When the system aggregates recommendations, it produces recommendations for each individual and aggregates these into a group recommendation. On the other hand, an aggregated model for the group is constructed from the individual profiles [37].

2.2.1 Aggregation strategies

The difficult part for a group recommender system is on how to aggregate such a model or recommendations using the individual's likes and dislikes. There are different aggregation strategies for this problem, the main used functions are shown here [17]:

- **Least misery:** Take the minimum of the individual ratings as the predicted group rating. This way you avoid to recommend an item which is very disliked by one person in the group.

- **Average:** Take the average of every individual's rating as the predicted group rating.

- **Average without misery:** Takes the average of every individual's rating, except items which have a rating below a certain threshold are removed from the possible recommended items.

- **Voting:** Choose the item with the most votes.

- **Approval Voting:** Counts the individuals who rated the item higher than a certain threshold and the item with the highest count is the most suited.

- **Most respected person:** The rating of the most respected person predicts the group rating.

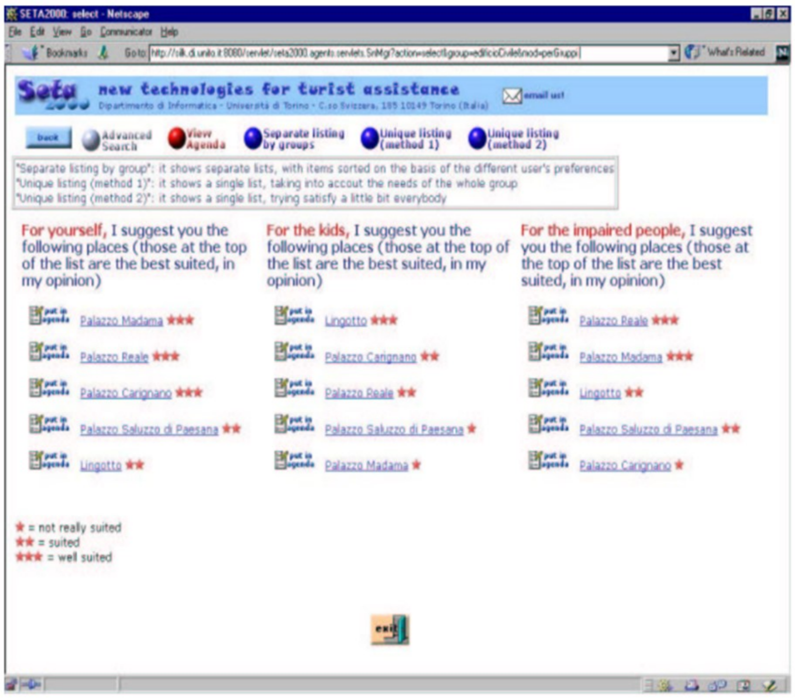
2.2.2 Existing work of group recommender systems

The theoretical part of a group recommender system has now been introduced. In what follows some of the best known, already existing group recommender systems are shown, together with the used aggregation functions:

- **PolyLens:** Is a group recommender system for movies and builds further on the individual recommender variant MovieLens. In PolyLens they use aggregated recommendations to recommend movies to the group. The function used in this application is the least misery strategy [30].

- **Intrigue:** Is an application for a group of users who go on a vacation together and want to visit places. The group recommender system suggests spots based on the characteristics of subgroups. Figure 2.1 shows a list of recommended items for each subgroup. The item which suits the group the most is on top of the list. Here they also use aggregated recommendations to recommend places of interest for the group. The used aggregation function is the average strategy [5].

- **Yu's tv recommender:** Recommends television programs for a group to watch (Figure 2.2). It recommends programs based on the individual preferences for program features (genre, actors,...). The middle part of the interface shows the recommended programs to watch, which are ordered according their group scores. Yu's tv recommender uses aggregated models to recommend a program to the group. This application also uses the average strategy to aggregate the models [49].



SETA2000: select - Netscape

new technologies for turist assistance

Dipartimento di Informatica - Università di Torino - C.so Svizzera, 185 10149 Torino (Italia)

"Separate listing by group": it shows separate lists, with items sorted on the basis of the different user's preferences
"Unique listing (method 1)": it shows a single list, taking into accout the needs of the whole group
"Unique listing (method 2)": it shows a single list, trying satisfy a little bit everybody

For yourself, I suggest you the following places (those at the top of the list are the best suited, in my opinion)	For the kids, I suggest you the following places (those at the top of the list are the best suited, in my opinion)	For the impaired people, I suggest you the following places (those at the top of the list are the best suited, in my opinion)
Palazzo Madama ★★★	Lingotto ★★★	Palazzo Reale ★★★
Palazzo Reale ★★★	Palazzo Carignano ★★	Palazzo Madama ★★★
Palazzo Carignano ★★★	Palazzo Reale ★★	Lingotto ★★
Palazzo Saluzzo di Paesana ★★	Palazzo Saluzzo di Paesana ★	Palazzo Saluzzo di Paesana ★★
Lingotto ★★	Palazzo Madama ★	Palazzo Carignano ★

★ = not really suited
★★ = suited
★★★ = well suited

back
Advanced Search
View Agenda
Separate listing by groups
Unique listing (method 1)
Unique listing (method 2)
email us!
put in agenda
exit

FIGURE 2.1: Intrigue, recommended places for subgroups [5]

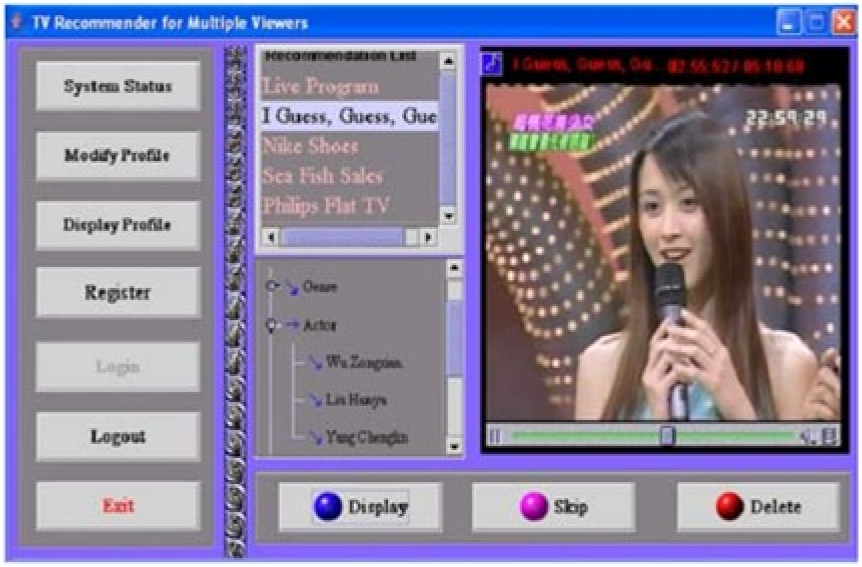


FIGURE 2.2: Main screen of Yu's recommender system [49]

2.2.3 Existing work of music group recommender systems

These examples emphasize the broad applicability of a group recommender system. Other possible domains can be news sites, libraries, games and many more. However, in this research the focus remains on developing an interface for a music group recommender system. This is why a section about existing and relevant work in this domain is included:

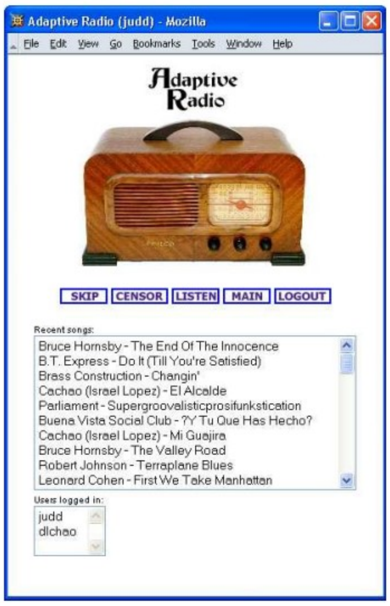
- **MusicFX:** Chooses a radio station in a gym by taking the present members into account. When someone leaves or enters the fitness centre, the radio station updates [27]. The recommender system aggregates the individual profiles to generate a group profile. It applies a variant of the average without misery strategy. The research concludes that most of the members preferred the music chosen by MusicFX.

- **Flytrap:** Is an application which plays music depending on who is currently present. The presence is recorded using RFID-chips. The recommended music is based on the individual profiles of the members. Based on these profiles, the recommender system calculates which songs satisfy the most, these songs have the highest probability to be played [16].

- **Adaptive Radio:** Uses negative preferences to generate a playlist for a group of people. The first time the recommender system is used, it considers each song to be relevant unless told otherwise. Users only have the possibility to dislike a song, using the "censor"-button on the interface (see Figure 2.3). When a song is disliked, it is removed from the playlist together with similar songs [15].

- **GroupFun:** This is an application that helps a group of friends in creating a group playlist (Figure 2.4). Users can manage their own music and share their music with their friends. They can invite friends to share their music or to rate other's music. Using these ratings the recommender system is able to aggregate each member's preferences into recommended songs for the group [35] [34].

- **PartyVote:** When a user joins the "party" he/she votes for a song, artist, album or genre. These votes are used to select the music for the social event. Based on the votes, the recommender system defines an area of songs which are playable and which will be liked by most of the users (Figure 2.5). The system promotes voting by ensuring that one song of his selected album, genre or artist will be played within the next ten songs [44].


Adaptive Radio (judd) - Mozilla
File Edit View Go Bookmarks Tools Window Help
Adaptive Radio
SKIP
CENSOR
LISTEN
MAIN
LOGOUT

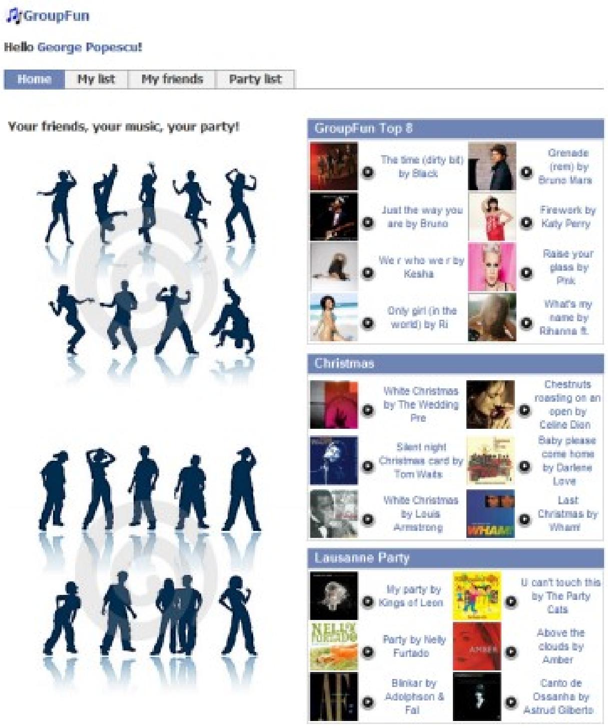
Recent songs:

Bruce Hornsby - The End Of The Innocence
B.T. Express - Do It (Till You're Satisfied)
Brass Construction - Changin'
Cachao (Israel Lopez) - El Alcalde
Parliament - Supergroovalisticprosifunkstication
Buena Vista Social Club - ?Y Tu Que Has Hecho?
Cachao (Israel Lopez) - Mi Guajira
Bruce Hornsby - The Valley Road
Robert Johnson - Terraplane Blues
Leonard Cohen - First We Take Manhattan

Users logged in:

judd
dlchao

FIGURE 2.3: Interface of Adaptive radio [15]



GroupFun Top 8

The time (dirty bit) by Black
Grenade (rem) by Bruno Mars
Just the way you are by Bruno
Firework by Katy Perry
We r who we r by Kesha
Raise your glass by Pink
Only girl (in the world) by Ri
What's my name by Rihanna ft.

Christmas

White Christmas by The Wedding Pre
Chestnuts roasting on an open by Celine Dion
Silent night Christmas card by Tom Waits
Baby please come home by Darlene Love
White Christmas by Louis Armstrong
Last Christmas by Wham!

Lausanne Party

My party by Kings of Leon
U can't touch this by The Party Cats
Party by Nelly Furtado
Above the clouds by Amber
Blinkar by Adolphson & Fal
Canto de Ossanha by Astrud Gilberto

FIGURE 2.4: Main screen of GroupFun [34]

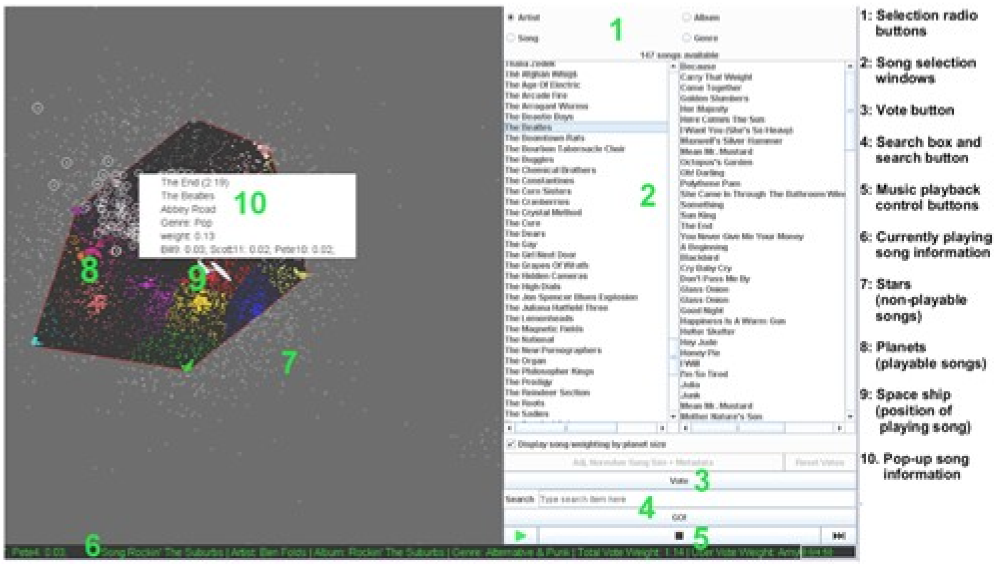

1
2
3
4
5
6
7
8
9
10
1: Selection radio buttons
2: Song selection windows
3: Vote button
4: Search box and search button
5: Music playback control buttons
6: Currently playing song information
7: Stars (non-playable songs)
8: Planets (playable songs)
9: Space ship (position of playing song)
10. Pop-up song information

Figure 2.5: Partyvote interface: left side is the area of playable songs, right area is used to vote for a song/artist/album/genre [44]

- **Perception of fairness in group music recommender systems:** This research highlights the influence of fairness on the users perception of the group recommender system. It concludes that people with an openness personality don't seem to find fairness that important. Openness indicates that a person will easier try new experiences and accept the predicted items by the system. Next, participants with a conscientiousness personality will faster report when they perceive unfairness. Finally, the choice of the ranking-algorithm can have an influence on the perceived fairness [22].

2.3 Controllability

The interface of an application has a lot of influence on the perception and experience of the application. It can improve the efficiency, trust, ease-of-use, satisfaction and a lot of other things for users. When designing an interface for a music recommender system there are a lot of different elements which need to be considered. Examples of such factors are transparency, controllability, diversity, novelty,... These are all features which need to be studied while developing the best possible interface [36]. A lot of research has already been executed for these features in individual recommender systems. In contrast with group recommender systems, where a lot of these subjects have not been touched yet.

Recommender systems often act as a "black box". The users don't know how the recommender system works and if they want to change the behaviour of it there are no means or they don't know how to do so. The system just requires input, it does some processing and it gives output. Without justifying why this output is relevant. Research has discovered that providing explanations and justifications can improve the satisfaction, trust, efficiency,... [46]. In recommender systems it also helps the user in correctly revising the input when given an undesirable output [43]. Processing output from a "black box" system can often be confusing for the users. Because of this it might produce undesirable user behaviour.

Recommender systems with no transparency are known to negatively affect user satisfaction [47]. These explanations can either be textual or interactive visualisations. Millecamp et al. conducted a research where he measures the effect of personal characteristics on explanations. Results show that people with a low need for cognition prefer textual explanations [28]. Other research performed by Tsai C.H. et al. found that combining explanations with controls have an additive and positive effect on the user experience [47].

Because there are so many items to consider when designing the most appropriate interface, this research will only focus on the controllability of the recommender system. Controllability indicates how much the system supports the user to configure the recommender process to improve the recommendations [21]. A system without controls often negatively influences the user's perception about the recommendations [19]. To address this problem, controls are often added. Randomly inserting controls into a recommender system is not useful when the user doesn't interact with them or when they are confusing. Therefore it is important to measure the effect of these controls. The following part consists of previous work where controls were developed for individual recommender systems and the effects of these were measured.

2.3.1 Existing work controllability

Tasteweights is an application developed by Svetlin Bostandjiev et al. It has an interactive interface which provides explanations for the generated recommendations together with controls for the recommender system (Figure 2.6). In this application music is recommended either social- (Facebook[5]), expert- (Wikipedia[6]) or content-based (Twitter[7]). The weight of each method is changeable using the sliders on the interface. From this research one of the conclusions is, that the addition of the controls improve the generated recommendations [7].

[5]www.facebook.com
[6]www.wikipedia.com
[7]www.twitter.com

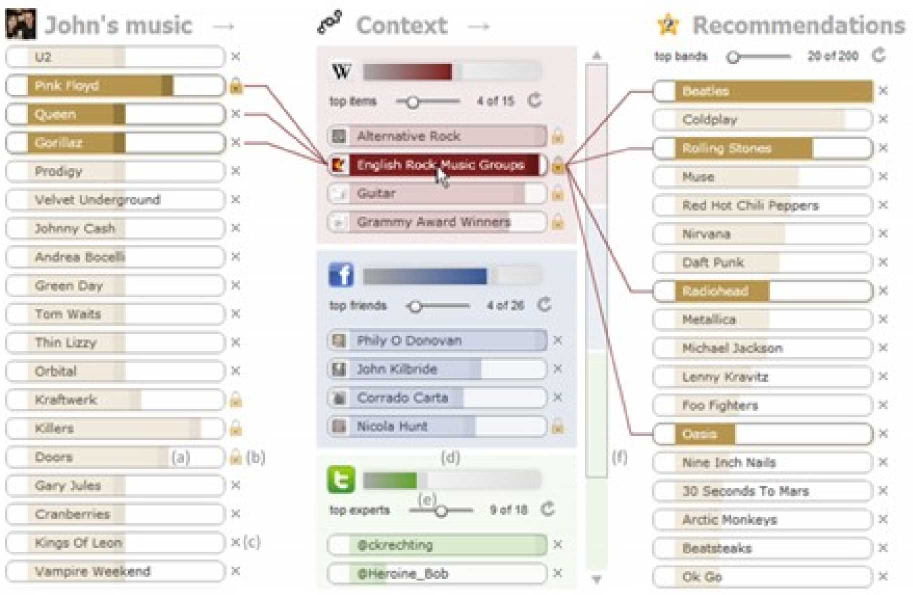


Figure 2.6: TasteWeights interface [7]

In the research conducted by F. Maxwell Harper et al. they investigated the effects of adding controls. They explored whether users like to have control over their recommendations. Next they examined whether the recommended items using controls differ a lot from the original recommendations. Finally the research considered if the users converge to a common "best tuning" setting. The implemented controls alter the item popularity and the item age. The research is conducted on a movie recommender system, MovieLens. The results show that people liked the ability to control recommendations, there was a significant difference in the original recommendations list and the tuned list and that there is no globally optimal setting [19].

Jin Y et al. measured the effect of different levels of controllability on the cognitive load. The cognitive load theory focuses on the cognitive resources which are used while solving a problem or interacting with an interface in this case [14]. Cognitive load is often measured using the NASA task index questionnaire [20]. In both studies [23] [24] they developed an interface with three different levels of control. In Figure 2.7, you see three different areas in the interface. Area "a" consists of all top tracks, artists and genres of the current user. These can be dragged to area "b". The grey sliders change the weight of each category. Based on these weights and the dragged songs, artists and genres of area "a", the recommender system calculates a suiting playlist, which is shown in area "c". As a result from both studies they conclude that recommender systems with high level controls produce the best recommendations, but require the highest cognitive load. Therefore most of the people prefer low or middle level control. Acceptance of recommendations is influenced by the presence and types of controls. Finally, it concludes that personal characteristics, like music sophistication, can have a significant influence on the interaction with the interface.

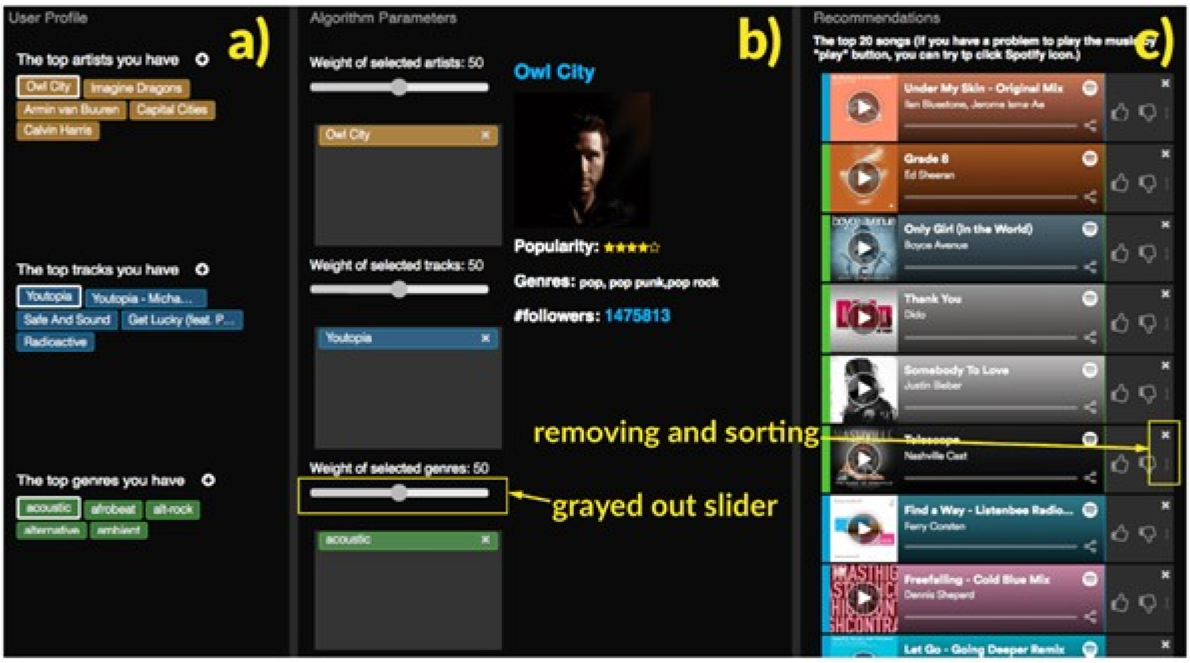


FIGURE 2.7: Used interface to measure the influence of controls on cognitive load [24]

Millecamp et al. investigated [29] the effect of personal characteristics on music recommender user interfaces. Two different types of visual controls are implemented and the effect of personal characteristics on each type of controls is measured. One interface uses sliders as controls, while the other interface has a radar chart as depicted in Figure 2.8. From this research they determined that people with higher musical sophistication interact more with the radar chart. When using radar charts, users discover a significant higher number of songs compared to the sliders.

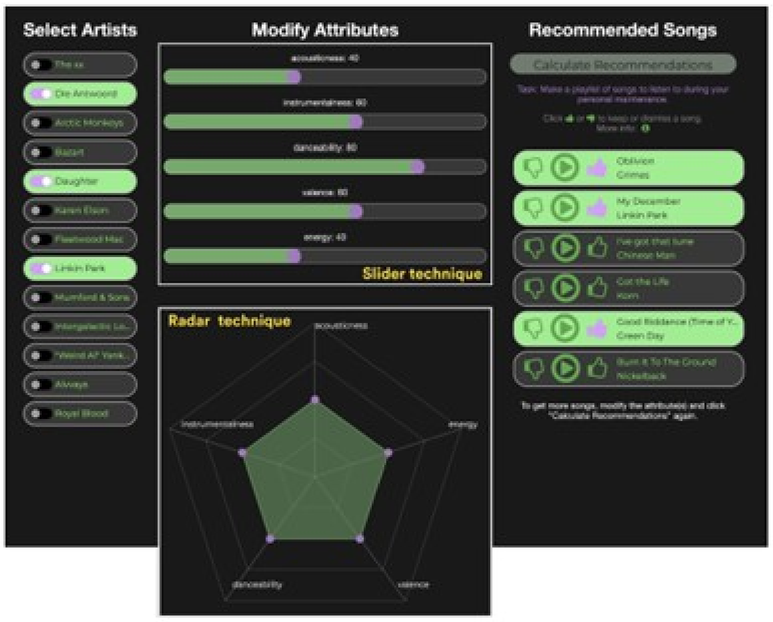


Figure 2.8: interface with sliders (upper middle part) and radar chart (lower middle part) [29]

2.3.2 Controllability in group recommender systems

Group recommender systems is still a relatively new concept in the world of recommender systems. The section about controllability already shows existing work in different domains. The focus of this existing work is developing a recommender system for groups, without taking in consideration the effect of the interface on the users experience. However this is an important aspect when designing the best possible system. As it can improve efficiency, trust, satisfaction,... for the users. This study is a contribution to the research on controllability in the music domain for mobile group recommender systems. In this case, controllability is considered as the members' influence on the played music. Other investigations in recommending music to groups implemented different ways of such controls, but never explored the influence of different types of controls on the users interaction with the system.

Adaptive radio uses negative elicitations to prevent certain songs of being played by pressing a dislike button [15]. Party vote uses a voting mechanism to find a region of commonly liked songs within the group [44]. MusicFX is an application which takes individual profiles into account and calculates a suitable playlist based on these profiles. This list is then submitted to the host. He/she is responsible for selecting the appropriate music [27]. GroupFun uses individual ratings to calculate a playlist for a group of users [35]. These examples show different ways of implementing controls.

2.4 Group dynamics

When items are recommended to groups, it may have an influence on the group dynamics. Group dynamics are interpersonal processes that occur in groups over time. An example of such a group dynamic is cohesion, as groups tend to become more cohesive over time [18].

2.4.1 Cohesion

Carron et al. defines cohesion as "a dynamic process that is reflected in the tendency for a group to stick together and remain united in the pursuit of its instrumental objectives and/or for the satisfaction of member affective needs" [11]. Through the history of research on cohesiveness there exist a lot of different definitions, models or perceptions about the cohesion theory [11] [13] [39]. This makes it difficult for researchers to know how to measure or to conceptualise cohesion. All sorts of conceptual models are created to describe cohesion.

Some researchers describe cohesion as a uni-dimensional construct, while others consider it to be multi-dimensional. Most of the research however defines cohesion as a multidimensional construct, where a distinction is made between task and social cohesion [39]. This distinction is based on the fact that the group is either created to complete a task or not. Social cohesion is the desire for a member to remain in the group because of the positive relationship with the group members. While task cohesion refers to the attraction to the group because of a shared task [48].

Furthermore, research isn't clear about the fact whether cohesion should be multilevel or not. From Salas E. et al. they conclude that cohesion should be considered on both the individual's level as well as the group's level [39].

An example of such a model for cohesion in sports teams is described by Carron A.V. In this research they make a distinction between the group and the individual in the analyses of group dynamics. The individual's perception about the group is called group integration (group). The way in which the group satisfies the personal needs and objectives is the individual attraction to the group (individual). These levels are then both divided in social and task cohesion. This division is also supported by the group dynamics theory [12]. Figure 2.9 shows a schematic overview of the conceptual model.

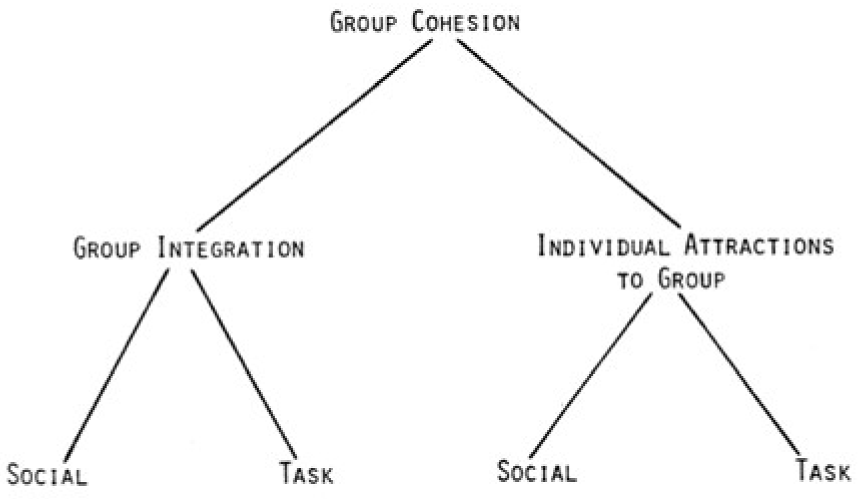


Figure 2.9: Schematic overview of conceptual model of cohesion [12]

There is no existing work where the influence of a music group recommender system on group cohesion is measured. This is why this will be an exploratory study in finding some of these influences, but it leaves the door open for further research in this domain.

2.5 Conclusion literature review

This chapter explains the basic theory for individual recommender systems. Recommender systems either use collaborative, content-based, knowledge based filtering or a hybrid strategy to recommend items to a user. Based on these individual models, this chapter clarifies how group recommender systems are developed and which aggregation strategies can be used for recommending items. Some existing work is cited to show examples. Next, the importance of controllability is explained, together with some existing research for individual recommender systems. Finally, the chapter finishes with a small introduction in cohesion. This final part concludes that there are many ways to conceptualise or measure cohesion. We adapt the conceptual model where cohesion is multilevel (group and individual level) and multidimensional (social and task cohesion) as this model is applied by most of the research [39].

Chapter 3

User study 1

The goal of the first user study was to find out what the expectations of groups of users are in terms of controllability in a music group recommender system, when they attend a social event (e.g. party).

3.1 First user study design

The method used to discover these expectations consisted of four main parts. The first part was done individually, the following parts were performed in groups of three people. Groups were used because the researcher was designing an interface for a group of people and this is the best way to evaluate and design such an interface. This user study was performed online due to COVID.

In the first part a sensitizing activity was applied where the users explored the music recommender system they were using. The goal here was to make the users aware about the existence of such a recommender system, that they developed a certain degree of confidence over the subject and maybe even formed an opinion about it. The used questionnaire for this sensitizing activity is added as an appendix (see Appendix A). The questionnaires were built and distributed using Qualtrics[1].

This part was followed with a small introductory discussion about their expectations in terms of controls for a music group recommender system.

After this introductory discussion, a co-design session was initiated. The goal for the groups was to build their own interface for a music group recommender system using tools provided by the researcher. The group of users had to collaborate to create an interface which had all their desired functionalities. This would help the researcher understand the wishes and expectations of the participants.

The last section of this user study consisted of a part where the researcher proposed his own created prototypes. The groups were asked to discuss these prototypes and compare them with the interface they developed in the co-design session. The researcher wanted to collect as much information about the opinions of the groups (what they like and don't like).

[1]www.qualtrics.com

The ultimate goal of this user study was to use this feedback regarding the prototypes, together with the created interfaces by the groups to update the prototypes. So that the updated prototypes consider the users' preferences.

For the first user study two different prototypes were created. The difference between the two was the type of music group recommender controls. In the first prototype a democratic strategy was used, while in the second prototype a veto power approach was used. Both prototypes were created using Figma[2].

3.2 Prototype 1: Democratic strategy

3.2.1 Design rationale

In this situation every member of the party is considered equal, everybody has the same influence on the played music. It is an approach were the group decides which songs are played.

The recommender system calculates songs based on the group profile and adds these songs to the playlist. When the system makes a bad recommendation, users have the ability to dislike the song. When a song added by the recommender system reaches a certain threshold of dislikes, it is removed from the playlist. This method of using negative preferences to avoid certain music of being played is derived from previous research. In Adaptive Radio negative preferences are used to avoid certain music [15] (see chapter 2). This approach also has the advantage that users only need to use their smartphone when they dislike a song. If the recommender system works properly, disliking a song should happen rarely. Because of this, the user is capable of fully enjoying the party.

The users also have the ability to propose songs. Songs proposed by users only become playable when they receive enough likes from other attendees of the event. This voting strategy is something that also has been used in previous research. PartyVote asks users to vote for songs, genres, albums or artists. Based on these votes, it calculates an area of songs, which fit the group [44]. Another example of a group music recommender system with a voting mechanism is Jukola. The users have the ability to nominate songs. Four of those nominated songs, together with songs picked from the host's playlist are presented as the votable songs. Users have one vote to choose the next song to be played. The most popular song is selected [31].

[2] www.figma.com

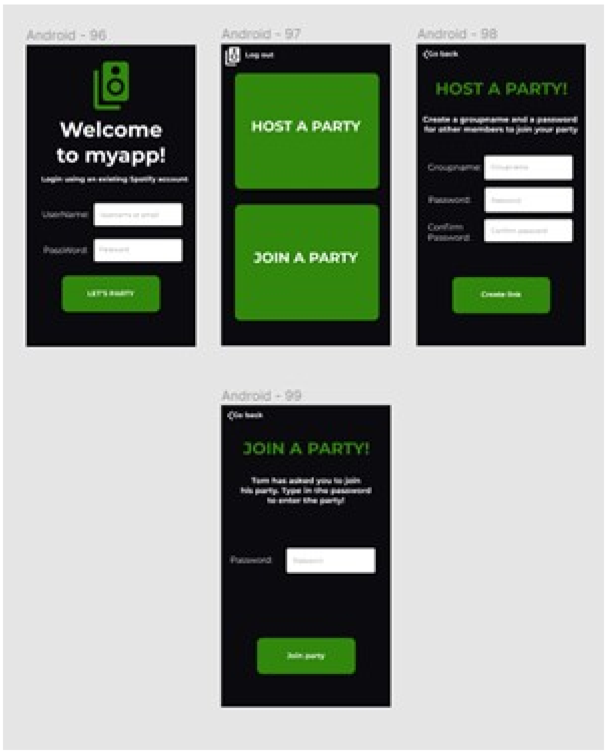
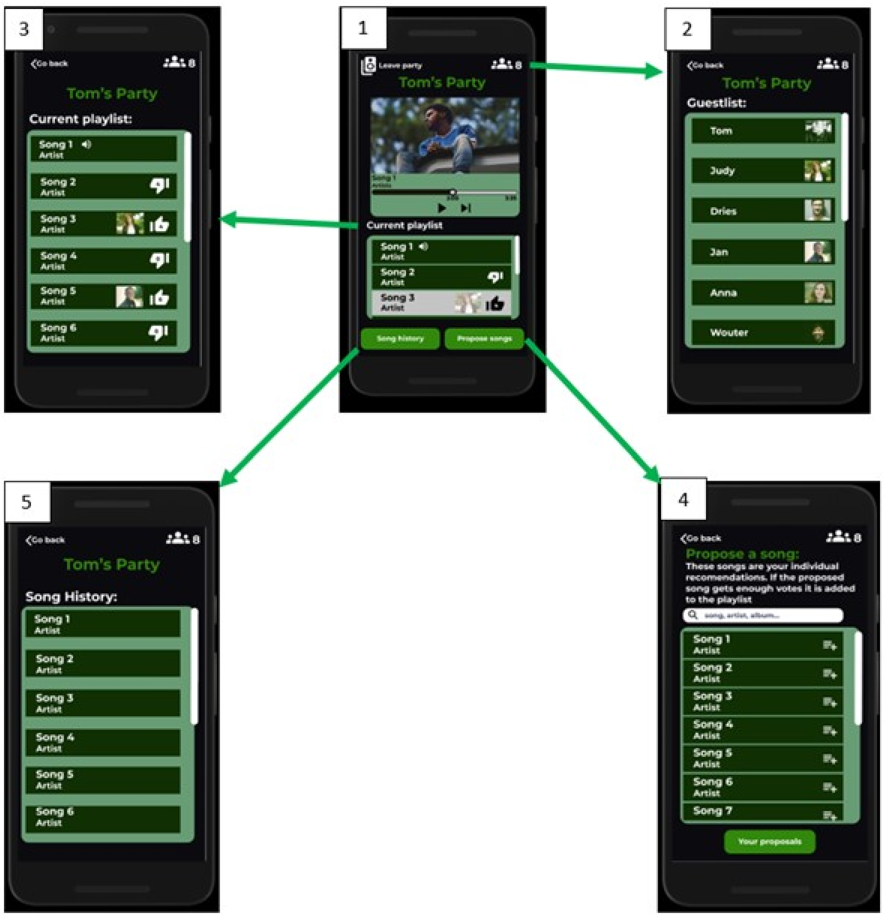


FIGURE 3.1: Login screen low fidelity prototype

3.2.2 Walk through of the interface

First the user needs to log in using an existing Spotify account. The next screen asks the user if he/she is going to host a party or join an existing party. Depending on what the user clicks, a different screen pops up. In the case of hosting a party, the user is asked to fill in a group name and a group password. A link is created for the host, this link can be used as an invitation for other people to join the party. In the case of joining a party, the user clicked on the provided link and he/she has to fill in the password to join the party. This set of screens is shown in Figure 3.1 above.

The rest of the interface is the same for every member of the party. When the user completed the setup, he/she will be shown screen 1 (Figure 3.2). This screen contains the currently playing song, controls to start, pause or skip a song, the amount of people in the party, a playlist which contains the songs which are going to be played next and two buttons ("song history" and "propose a song").



These songs are your individual recomendations. If the proposed song gets enough votes it is added to the playlist

Figure 3.2: Overview prototype, democratic strategy. Screen 1: main screen. Screen 2: Guestlist. Screen 3: Current playlist. Screen 4: Screen for proposing a song. Screen 5: History of played songs

The symbol in the top right corner opens a list with all current members of the party together with their profile pictures (screen 2). If the user clicks on "current playlist", the list with the songs which are going to be played next is shown (screen 3).

Each song is accompanied by either a like- or dislike-button. Each song that is added by the group recommender system is accompanied by a dislike button. The button is used for users to send feedback to the recommender system.

Songs that are proposed by a user have a like-button next to it. A user is capable of proposing a song to the rest of the group. To do this he/she has to press the "propose a song"-button. This action opens screen 4, where a song can be searched to propose. When a song is proposed by a user, it is virtually added to the current playlist. But it is not playable until it receives enough votes from the rest of the members. Song 3 in Figure 3.2 is an example of a proposed song with too little votes. User proposed songs are accompanied with the profile picture of the user who proposed it.

Each user can find their previous proposals by pressing the "Your proposals"-button in screen 4. The "song history"-button is used to get the list of all song which have been played in the past (screen 5).

3.3 Prototype 2: Veto power

3.3.1 Design rationale

In the veto power approach, one person has more power than the rest of the group. In most cases this person will be the host of the event. The host is responsible for the songs which are being played and for the creation of the playlist. The other members of the group are capable of stating their preferences by submitting requests to the host. The host is also capable of removing and adding songs from and to the current playlist.

This veto-oriented approach is also used in previous works. WE-DECIDE is an example of such an application. It is used to choose the right restaurant. Users are asked to rate all different possibilities individually. Those ratings are then used to make group recommendations with corresponding explanations. These group recommendations are then sent to the users. They can choose to change their preferences based on those recommendations. After a certain deadline has passed, the final group recommendations are sent to the host. He/she has to take the final decision, while taking the group recommendations into account [45].

3.3.2 Walk through of the interface

The process of joining or setting up a party is the same as for the previous prototype. The only difference between both prototypes is that when you host a party, you become the DJ and you have more power/responsibility. The host has a different interface than the other users. Once a user (not the host) joins a party, he/she gets screen 1 (Figure 3.3). This screen shows the currently playing song, the current playlist with songs which are going to be played next, two buttons ("song history"-button and "request a song!"-button) and the amount of people in the party. In the figure below you can see the functionality of each button.

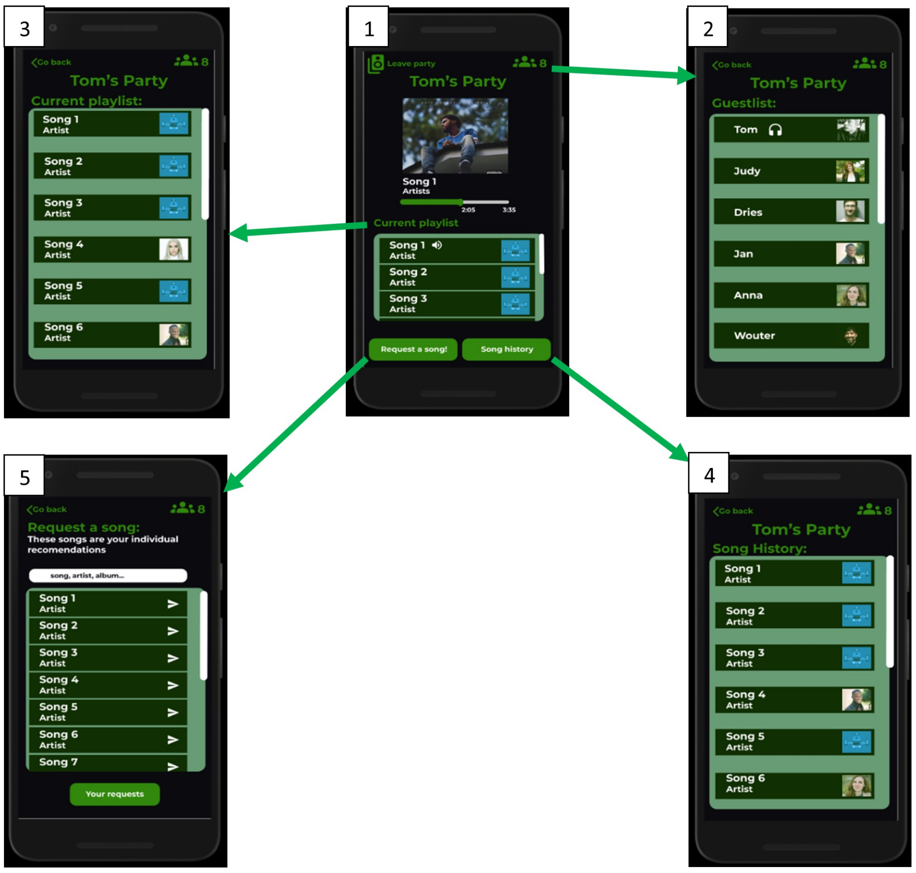


Figure 3.3: Overview prototype: Veto power approach (not host). Screen 1: Main screen. Screen 2: Guestlist. Screen 3: Current playlist. Screen 4: History of played songs. Screen 4: Screen to request a song.

The functionality of the icon in the top right button is the same as in the previous interface. The same can be said about the functionality of the "song history"-button.

If the user taps on "current playlist", the list with the songs which are going to be played next is shown (screen 3). A picture is added next to each song in the list. When this picture is a robot, it means that this song was calculated by the group recommender system. If the picture is a profile picture, it means that a user has requested this song and this request was approved by the host.

The "request a song"-button opens screen 5, where the user is able to send requests. The user can request songs from his personal recommendations or using the search bar.

This screen has an extra button, to see which previous requests were already sent by the current user (Figure 3.4). Each symbol next to a song represents the status of the request. The red symbol means that the request is rejected, the green symbol means that the requested song is added to the playlist and the orange symbol means that the request is being processed by the host.

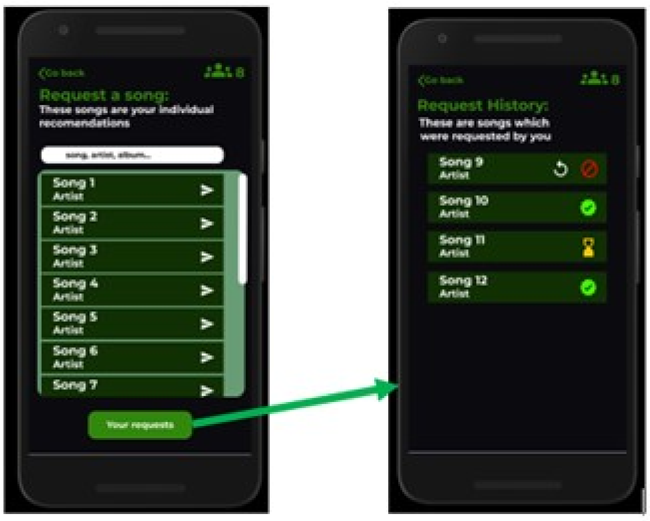


FIGURE 3.4: Overview of screens to request a song

The host has a similar interface (Figure 3.5), but he/she has more power than the other members. The difference is that the host can pause or skip the current song (screen 1), he/she can remove songs from the current playlist (screen 1), the host can process incoming requests (screen 2), access the requests he/she rejected (screen 3) and is able to add songs to the current playlist (screen 4).

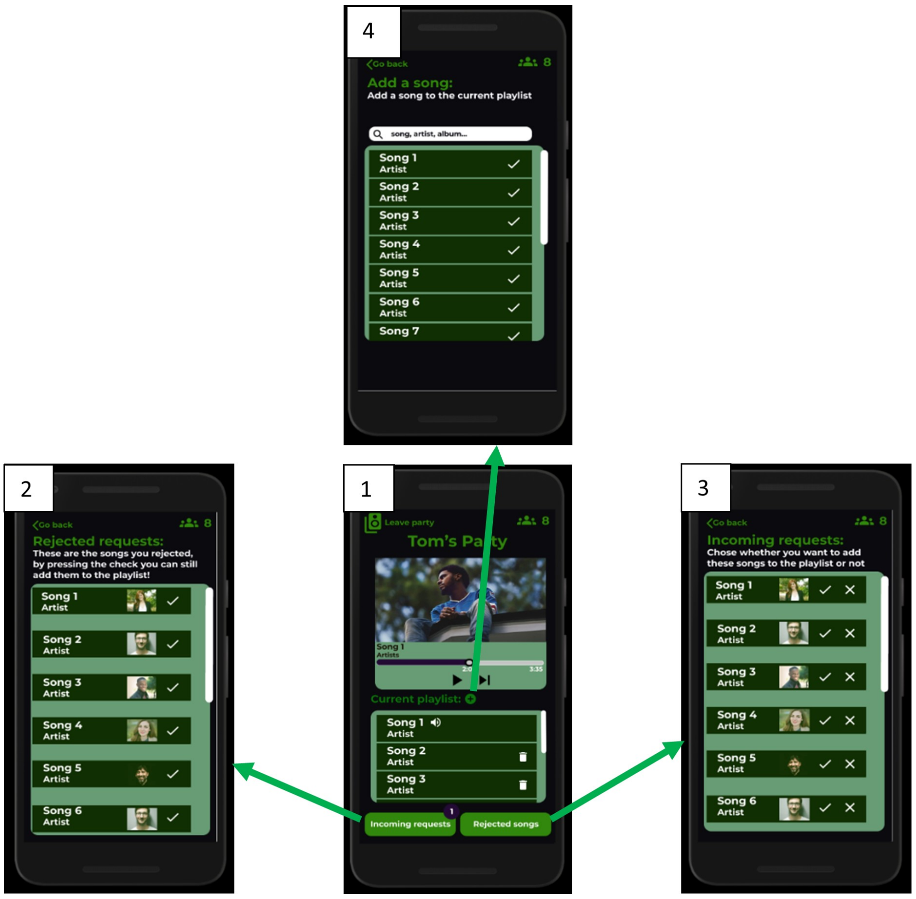


Figure 3.5: Overview of screens: veto power approach (host). Screen 1: Main screen. Screen 2: List of rejected requests. Screen 3: List of incoming requests. Screen 4: Screen to add a song to the playlist.

3.4 Results user study 1

3.4.1 First iteration recruitment

The amount of users for the first user was fifteen participants, divided in five groups of three people. Recruited participants were fellow students and acquaintances, contacted through Facebook[3]. All participating test users are Belgian. The user

[3] www.facebook.com

study was applied on people who have an active subscription on any music platform which uses a recommender system. Other constraints for the users were that they were subscribed for at least a year and used the application frequently (weekly). Most of the participants were aged between 22 and 24 years old and the average age was 26 years. The major part of the participants were still studying (60%) and 20% of the users were female. All recruited participants were either average or advanced computer users and they all knew that Spotify offered personalized recommendations. Only 27% of participants used these recommendations to listen to, most of the time. The remaining participants sometimes listened to them. Figure 3.6 shows the time the users spent on Spotify per week. An overview with the demographics for each participant is given in Table 3.1 .

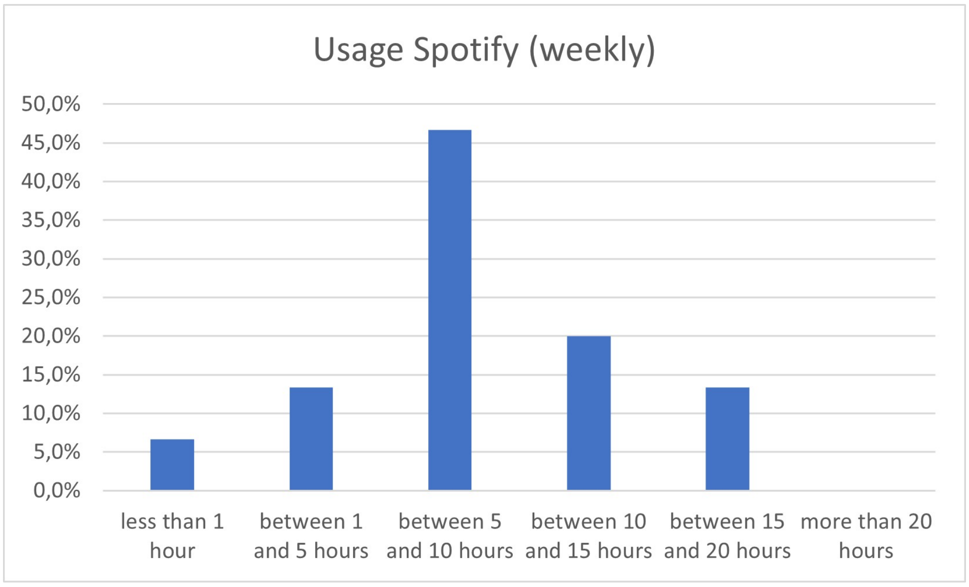


FIGURE 3.6: Time recruited participants spent on the Spotify application per week (user study 1)

ID	Age	Profession	Education level	Computer expertise	Weekly usage	Group
P4	22	Student	Bachelor's degree	Advanced	10-15 hours	1
P5	23	Student	Bachelor's degree	Average	10-15 hours	1
P27	23	Student	Master's degree	Average	5-10 hours	1
P12	24	Student	Master's degree	Advanced	15-20 hours	2
P13	22	Student	Master's degree	Advanced	15-20 hours	2
P14	24	Student	Secondary education	Average	5-10 hours	2
P2	23	Consultant	Master's degree	Average	5-10 hours	3
P15	23	Student	Master's degree	Average	10-15 hours	3
P16	23	Consultant	Master's degree	Advanced	< 1 hour	3
P1	23	Sales Engineer	Master's degree	Average	5-10 hours	4
P3	23	Student	Bachelor's degree	Advanced	5-10 hours	4
P29	23	Student	Master's degree	Advanced	1-5 hours	4
P8	26	Career counsellor	Bachelor's degree	Average	5 - 10 hours	5
P9	27	Consultant	Master's degree	Average	5-10 hours	5
P11	60	Manager	Master's degree	Average	1-5 hours	5

Table 3.1: Demographics of recruited participants for user study 1

3.4.2 Thematic analysis

The goal of this user study was to find the expectations for a mobile music group recommender system's interface. These expectations are used to improve the design of the interfaces of both presented prototypes. A thematic analyses was used to process the results, because this is an accessible and flexible tool to identify, analyze and report patterns within qualitative data. The guide provided by Braun et al [8] was used. First, we familiarized our self with the data by listening to/reading the data repetitively and making annotations if necessary. Once familiar with the data, we analyzed the data by finding relevant features and labelling them. This part of the process is called coding. It is important to associate each code with the corresponding quote of the participant. Next, identifying each code that is relevant to the research question and grouping codes together so they formed themes. Some iterations were needed to find the right themes for this research question. The found themes are discussed here:

- **The relevance of designing to focus on the social aspect of the event:** The participants prioritized the social aspect of the event over the music being played or the application controlling the music. They preferred focusing on interacting with the people present at the party, rather than with the application. The amount of time spend on the application should be as minimal as possible. An example citation of P4 was: "I would rather have the focus on the social part then on the application".

Those concerns were also expressed when presenting the veto-oriented prototype. The participants feared that the host couldn't fully enjoy the social aspect, because he/she had to process incoming requests. P16 said: "Is there a possibility for the host to always accept incoming requests?". While P14 shared this opinion with: "Maybe assign co-hosts to divide the workload".

A comparable trend was visible with the prototype which uses a democratic approach. A song only becomes playable when it receives enough votes. People need to be on their smartphone for a song to be played. Almost all groups highlighted this problem with some of the following citations. P4 stated: "When there is low engagement, suggested songs should be automatically added". P1 said: "the disadvantage is that everyone is continuously busy with liking or disliking songs". P12 confirmed this statement: "There needs to be a way of expressing your opinion without being on your phone most of the time". Next, P11 agreed by saying: "I want to work on the playlist before the party, but during the party I want to focus on the people present". To conclude P2 said: "The app should make life easier when you meet with your friends. The focus shouldn't be on the music, but on interacting with friends".

These examples show that is important for the user to have a simple application, which doesn't need much time/effort when interacting with it. Because almost all participants mentioned that the focus should be on socializing, it seems that this should be the centre of attention when designing a music group recommender application.

- **Contradictory preferences between the host- and voting-oriented designs:**
  On the one hand the members of the group felt that it was important to always have one person who is in charge of the organizational part of the application. The participants often highlighted the fact that only one person should always be able to skip or pause the current song, preventing other users from abusing these functionalities. This person should have the final responsibility. However it is up to the host whether he uses this functionality or not. P4 highlighted this concept with the following quote: "If the party is getting out of hand, the host should have the ability to skip certain songs". P8 confirmed this opinion with: "The ability to skip a song might be abused when everyone has this power." On top of this functionality, P1 and P8 stated that they would like that the host has the power to kick someone out of the party. P1 referred to this functionality as: "If someone is suggesting terrible music, it can be nice to remove him from the event".

  While on the other hand most of the groups preferred the democratic-oriented design. Group 2 is an example of these groups, they stated: "It is nicer to select the music as a group". The next theme will give more examples on why the democratic approach was preferred 80% of the time.

  The majority of the groups often asked if it was possible to integrate both prototypes. Because they want the advantages of having a host and selecting the music as a group. They want an application which is group oriented, but where there exists the possibility for the host to take control when necessary. P4 expressed this concern as follows: "I prefer the voting-approach, but I feel that there should always be a host who can take over when needed". Putting these statements together indicates that the users prefer a music recommender application were you can decide which music is played as a group, but the option of a host is always there to guide the party when necessary.

- **Importance of the group-oriented design:**
  As mentioned earlier, 80% of the groups preferred the democratic prototype. P13, P12 and P1 highlighted the group oriented design of the democratic approach which they preferred. P12 said: "it is nice to select the music as a group and not one person who has the final decision". P1 commented on the democratic prototype: "With this approach you create a better group feeling". P13 also suggested that adding lyrics might even improve the group dynamics.

  Other statements relating to this topic discussed the fact that they didn't want one user to dominate the music. It could ruin the group dynamic of the social event. P12 brought this issue up by saying: "You want to avoid that one person is playing his/her individual music by requesting songs all the time". P29 also indicated this problem with the veto power approach: "With this approach, the music of the host will dominate the event". P11 and P9 agreed with this concern by proposing a limited number of requests per user.

- **Design features for group music recommender systems:**
  This theme highlights some of the most requested functionalities from the participants.

  - *Ability for the attendants to request or to add a song in the playlist:* A repeating subject in the user studies was the ability to add or request songs. All groups expressed the fact that they liked the ability to request or add songs in the current playlist. For example, P5 communicated: "the option to request or add a song to the playlist is a nice feature". P15 confirmed this by stating: "I would like it to be possible to add songs from every smartphone".
  - *Function to set the theme or a genre of the music on the social event:* Most participants preferred a functionality for setting the theme for the event. P3, P16, P11 and P14 contributed that it might be nice to have a functionality which can set the theme or genre of the event. P11 said: "There should be a distinction in played music between party or talking with friends". P8 compared such a functionality with the one implemented in Spotify where you can choose between upbeat or chill music in the family mix playlist[4]. P9 suggested: "only accept songs that fit in the current genre".

[4]https://support.spotify.com/nl/article/family-mix/

- *Connect the group application with the personal music streaming accounts:* Users frequently stated that they would want to link the group application with their personal music streaming accounts. It was important for them to have the option to save played songs or to save the song history as a playlist in their personal accounts. Both P16 and P3 wanted such a feature. P16 asked: "Is there an option to save the used playlist of the event?". P3 backed this up with: "Is it easy to link a song from this app to my Spotify account?".
- *Add a song preview with each song:* One of the participants suggested adding a preview with each song. A preview can be useful for the veto-power approach, the host can use this function when a song is requested he/she doesn't know. In the democratic approach however, it might induce long screen times which need to be avoided. P1 suggested this functionality: "Maybe you can add a preview next to each song, so that attendants who don't know the song know what to vote".

3.5 Conclusion user study 1

The results of this user study shows the significant importance of the social aspect during an event where music is played. A lot of participants found socializing the most important part. They prefer an application which needs little time and effort, so that they can spend more of it on interacting with the members of the group. This result indicates the relevance of the perceived ease-of-use and cognitive load of the application, which were measured in the next iteration. Furthermore, most of the groups preferred the democratic approach, because it creates a group feeling. The fact that the participant highlighted this topic, without being asked about, indicates the importance of this aspect. The host's music might dominate when the veto-oriented approach is applied. However, most of the participants agreed on the fact that there should always be one person with more responsibility. He/she has the ability to override the music and remove annoying people from the application. This result is in contradiction with the previous one and because of this no hard conclusions can be made on which prototype is better. Finally, some suggestions were made for the application. For example, the application could have a way of setting a certain genre/theme according to the participants. Or the users would like a functionality where they can add songs or playlists to their personal music streaming accounts.

Chapter 4

User study 2

The results from the previous user study were used to develop two high-fidelity prototypes with different types of music group recommender controls: 1) A democratic version and 2) a veto-power version. The aim of this user study was measuring the usability and the cognitive load.

The previous iteration highlighted the importance of socializing during an event. To avoid that users are on their smartphone during the social event, both prototypes need to be easy to use and require a low work load. An easy and intuitive application will cause more time for socializing. This theme of designing an application with the focus on socializing emphasizes the relevance of this user study, where the cognitive load and usability of the prototypes are measured. If there is a clear preference towards one of the prototypes, the preferred prototype will be selected for the final user study of this thesis. The participants also had the opportunity to discuss the usability and to highlight some flaws in either one of the systems.

A within subject user study is chosen to compare both prototypes. The participants used for this study are different from the previous user study.

4.1 Second user study design

The study itself consisted of three main parts and it was conducted on individuals. Before the session, the user was sent a questionnaire for his/her demographic information (see Appendix A). In the second phase, the participant was invited to play around with the prototypes, using tasks provided by the researcher. The tasks were designed such that all aspects and functionalities of the application are explored. An overview of all different tasks is included in Table B.2 and Table B.3 in Appendix B. The order of the tasks were randomized using a Latin square distribution for each participant, to avoid biasing. Sometimes a certain task was left out for one prototype, because it was the same for both prototypes: finding the members of the party, indicating which songs were added by the system or a user and looking for the song history. Each test started with task 1, since this is logging into the application. For the same reason and to counteract the learning effect, the order in which the prototypes were presented was changed for each participant.

After exploring each prototype, the test user was invited to fill in a questionnaire about the usability and the task load of the application. The usability and cognitive load were measured using the NASA task load index questionnaire [20] and the SUS-questionnaire [9]. Both questionnaires were combined and sent to participants using Qualtrics. The used questionnaire to measure the cognitive load and ease-of-use is added at the end in Appendix C. After this part, the user study was finalized with some open questions to support the provided results, some examples are included in Table B.1 in Appendix B.

The personal account of the researcher was provided to the participants if they did not have a personal Spotify account. This implied that the personal recommendations didn't fit the profile of the participant, which was mentioned to the participants. This solution is justified, because the focus of this user study was on the ease-of-use and the cognitive load and not on the quality of the recommendations.

4.2 Prototypes

The used prototypes for this user study were developed from the prototypes which were presented in the first user study. The functionality of both versions remained the same as they implemented all the main required and desired functionalities, which were derived from the thematic analysis of user study 1.

4.2.1 Technical implementation

Both prototypes are developed using Flutter[1] in Android Studio [2] and the Spotify API [3]. The Spotify API was chosen since it is free and has a clear documentation of its functionalities. For an application to use the Spotify API it needs to be registered in Spotify for Developers. After the registration a client ID is generated for each prototype. This unique ID is used in every request made by the application to the API. Flutter also offers a Spotify package[3] which implements part of the functionality that is in the official API. This package was used to retrieve access tokens, since these together with the client ID were required to make requests to the API. For the application to request personal recommendations, permission from the user was needed. When the user logged in for the first time on the prototype, a pop-up asked authorization to access information from their profile. A flow from the log-in to making a request to the API is presented in Figure 4.1. A request returns JSON files from which the requested information can be subtracted. For example, the Spotify API is used to retrieve personal recommendations of the logged-in user. These recommendations are the songs to which the user listened most in the recent past. This functionality requires that the current logged-in user has a Spotify Premium account. The API is also used to search certain songs and to retrieve song ID's, song titles, artists, covers and the duration of songs. When given a song ID, the Spotify package from Flutter was used to play songs.

[1]www.flutter.dev

[2]www.developer.android.com/studio

[3]www.pub.dev/packages/spotify_sdk

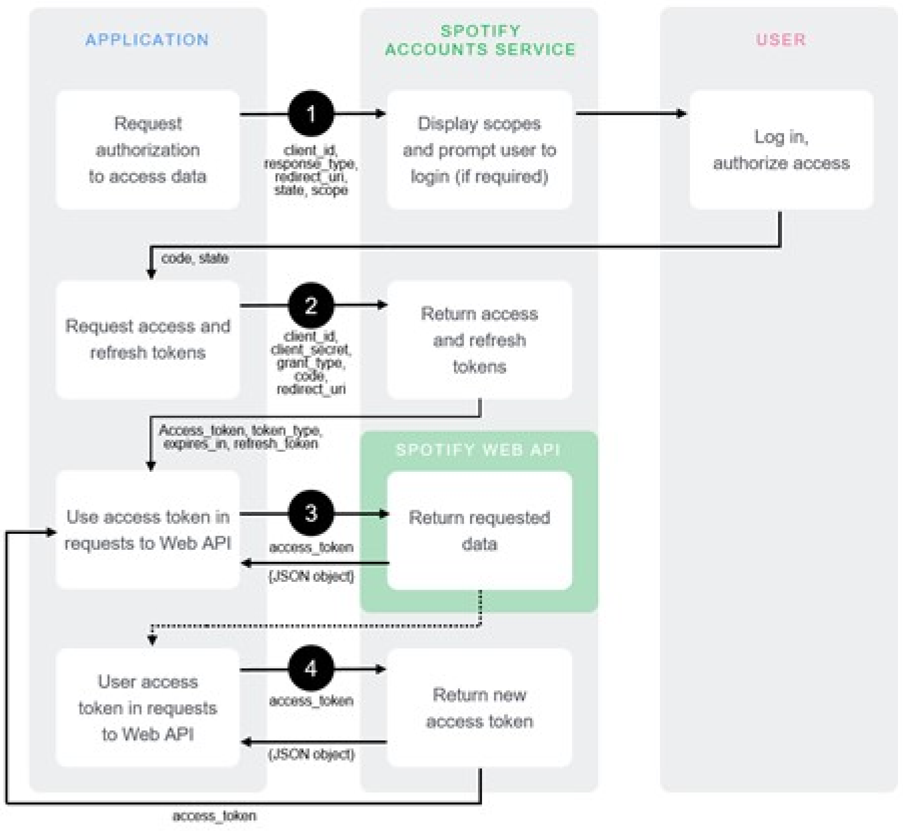

APPLICATION
SPOTIFY ACCOUNTS SERVICE
USER
Request authorization to access data
1
client_id, response_type, redirect_uri, state, scope
Display scopes and prompt user to login (if required)
Log in, authorize access
code, state
Request access and refresh tokens
2
client_id, client_secret, grant_type, code, redirect_uri
Return access and refresh tokens
Access_token, token_type, expires_in, refresh_token
SPOTIFY WEB API
Use access token in requests to Web API
3
access_token
{JSON object}
Return requested data
User access token in requests to Web API
4
access_token
{JSON object}
Return new access token
access_token

Figure 4.1: Authorization flow [1]

Since this user study was taken on individuals and not groups, the behaviour of other (non-real) users was simulated. The group consisted of four people: one real test user and three simulated users. Since these simulated "users" have no real Spotify account, it is impossible to calculate recommendations for the group. To solve this problem an already existing playlist from Spotify was chosen, which represented the calculated songs. This simplification is justified because these songs just need to show that the system automatically calculates and adds songs in the playlist. The algorithm which is used to calculate these songs is less important. On top of this, to show that other users can request songs as well, both prototypes show some requests made by these virtual users.

4.2.2 Interface

The changes made from the low-fidelity prototypes are mentioned together with an overview of both applications.

Design rationale

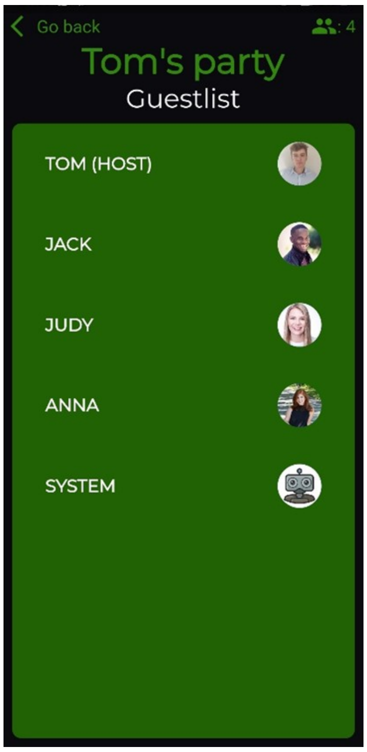
The main difference between both prototypes is the way in which the music is selected. The democratic version emphasizes the group aspect and votes are needed for songs to be played. The veto-oriented version has one person with all responsibility regarding the music being played. Small changes were made to the interfaces of the low-fidelity prototypes to make things more clear:

- First of all, to keep the focus of the application on the playlist, the section of the currently playing song is made smaller. It makes it also possible to make the other sections of the application bigger. This new layout makes the interface cleaner and more comprehensible. The songs inside the current playlist can appear bigger. It also causes the icons next to the songs to be bigger, which is more user friendly.
- To make the functionality of the buttons more intuitive, icons are added inside the "Song History"-, "Incoming requests"- and "Propose songs"- buttons. The "Song History"-button has a clock, the "Incoming requests"- and the "Propose songs"- buttons are accompanied with a musical note.
- The bottom two buttons appear darker to make the contrast with the rest of the interface clearer. Other small color changes were made to make the interface cleaner.
- The currently playing song is also removed from the current playlist section of the interface. There was no real advantage of showing this, it only took up place.
- The songs in the "Song history"-section are accompanied with the time when the song was played. It clarifies the order in which the songs were played in the past. The person/system who added the song is also illustrated next to the song title. A small explanatory text is added below the "Song history"-title as well.
- To make clear that songs are calculated and added by the recommender system, the system appears as a member in the guest-list together with an avatar of a robot.
- In the democratic version, Songs which are not playable yet are accompanied with a white thumbs-up icon, because a black icon can have a bad co-notation.

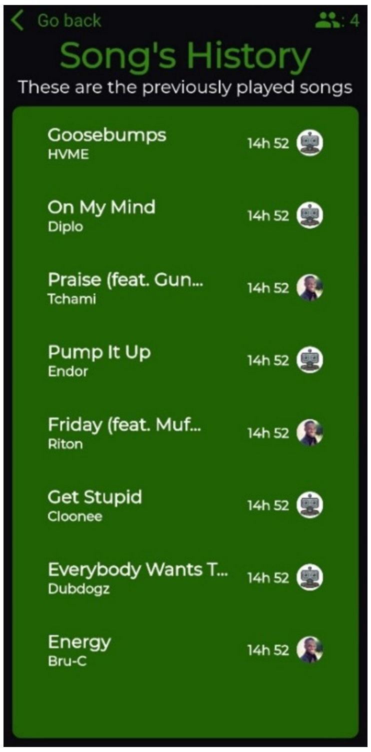
  In the request a song section, the previously requested songs are added on top of the list.
- The order of the bottom buttons in the veto-oriented version were changed to keep the order the same for both prototypes.

Common elements

In both prototypes you can find the members of the social event by pressing the group-icon in the top right corner. It opens the guest-list, as demonstrated on the left screen of Figure 4.2. Also, both prototypes have a song history screen, which is shown in Figure 4.2. It is a list of songs which have been played in the past.


Go back
4
Tom's party
Guestlist

TOM (HOST)
JACK
JUDY
ANNA
SYSTEM


Go back
4
Song's History
These are the previously played songs

Goosebumps HVME	14h 52	
On My Mind Diplo	14h 52	
Praise (feat. Gun... Tchami	14h 52	
Pump It Up Endor	14h 52	
Friday (feat. Muf... Riton	14h 52	
Get Stupid Cloonee	14h 52	
Everybody Wants T... Dubdogz	14h 52	
Energy Bru-C	14h 52	

FIGURE 4.2: Guestlist (left), Song History screen (right)

Democratic version

The most important characteristic of this version is that all members of the group have the same amount of power in deciding which music is played. As you can see in the main screen on Figure 4.3, song 1, 3 , 5 and 7 were already proposed to the group by the simulated users. Songs suggested by other users can only be played when they receive enough votes from the group. Newly proposed songs are added at the bottom of the playlist. The ones that receive three upvotes are added to the playlist and appear green. The proposed songs which require more votes appear grey. If a song without enough votes arrives at the top of the playlist, it is removed from the playlist. In this user study there is one song proposed by a user that requires one more like to be accepted (song 5). All other songs requested by users are already added to the playlist.

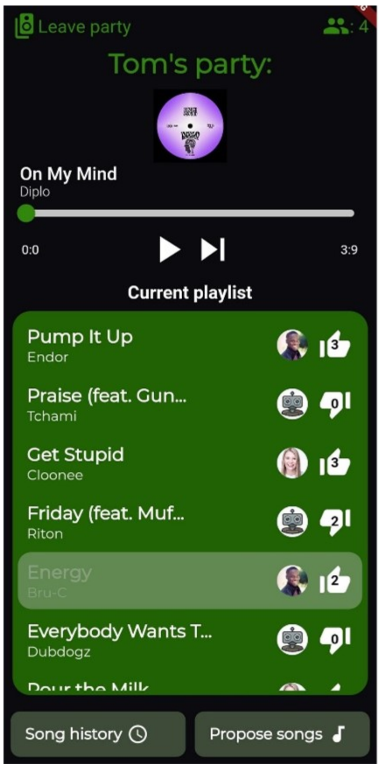
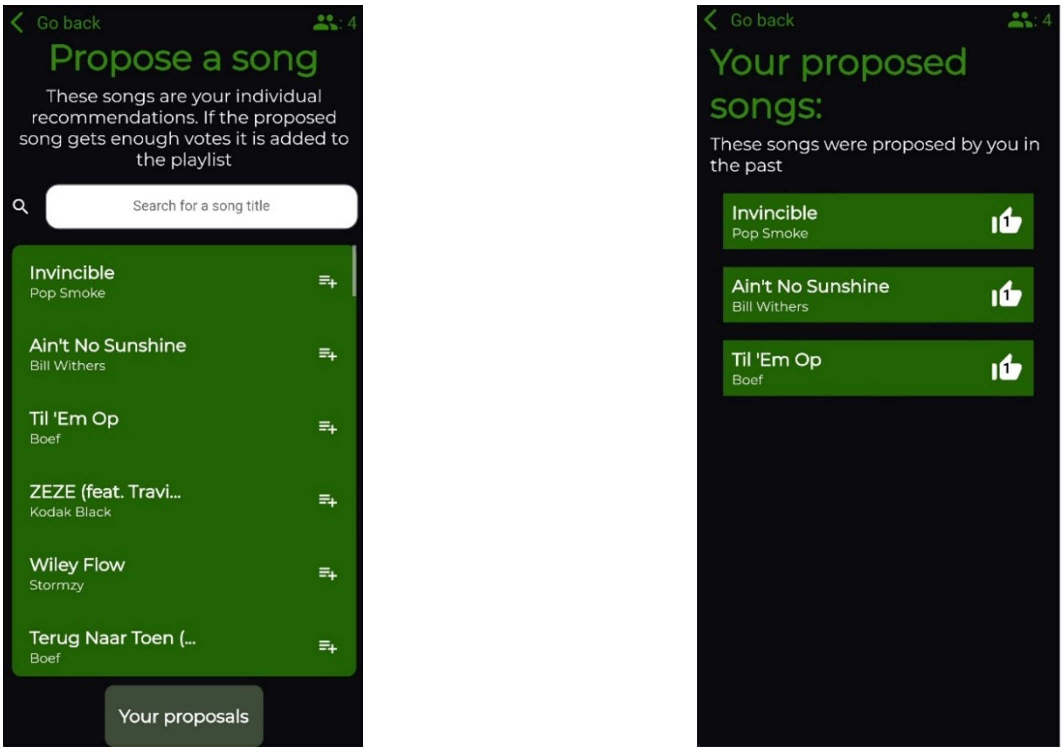


Figure 4.3: Main screen of democratic version

As mentioned before, the system keeps calculating and adding songs to the playlist. These songs are accompanied by a robot-avatar and a dislike-icon (song 2, 4 and 6). Songs added by the recommender system can be removed if enough members (three in this case) press the dislike button. For this user study, all calculated songs have zero dislikes, except one. One song has two downvotes, which means only one downvote is needed to remove this song from the playlist (song 4).

Figure 4.4 shows an overview of all remaining screens in this prototype. The left screen is used to propose a song to the group. You can propose a song from your personal recommendations or search a specific song, using the search bar. The right screen shows an overview of your previous sent proposals, accompanied with the amount of likes each proposal got.


Go back 4

Propose a song

These songs are your individual recommendations. If the proposed song gets enough votes it is added to the playlist

Search for a song title

Invincible
Pop Smoke

Ain't No Sunshine
Bill Withers

Til 'Em Op
Boef

ZEZE (feat. Travi...
Kodak Black

Wiley Flow
Stormzy

Terug Naar Toen (...
Boef

Your proposals

Go back 4

Your proposed songs:

These songs were proposed by you in the past

Invincible
Pop Smoke

Ain't No Sunshine
Bill Withers

Til 'Em Op
Boef

FIGURE 4.4: Screen to propose a song from your personal recommendations or a specific searched song (left), an overview of your previous sent proposals (right)

Veto-power version

The focus of this version is the fact that there is one user, the host/DJ, who has more power than the rest of the members. When a user logs in, he/she needs to select which role he/she will fulfil in the event, on the screen in Figure 4.5. The host has the final decision in which songs are played, his/her main screen is shown on Figure 4.6. The host can add, remove, play or skip songs. The host can press the litter bin next to a song to remove it from the playlist. Songs can be added by pressing the plus icon, next to "current playlist". It will lead to his personal recommendations, represented in Figure 4.7. He/she can choose to add a song from this list or he/she can look up songs. Songs are added by pressing the icon on the right of the song title. Because of the different functionalities between the members and the host, they have a different main screen. Figure 4.6 shows the main screen for the people who didn't host the event.

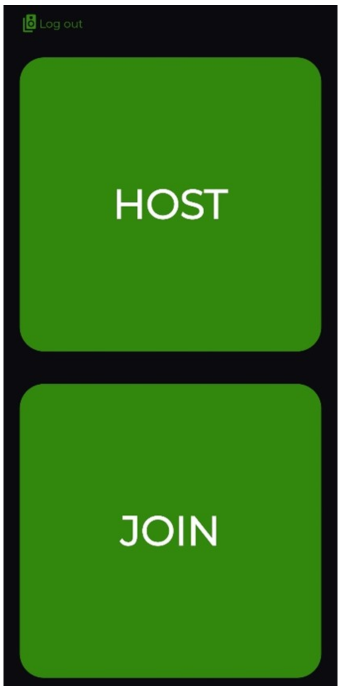
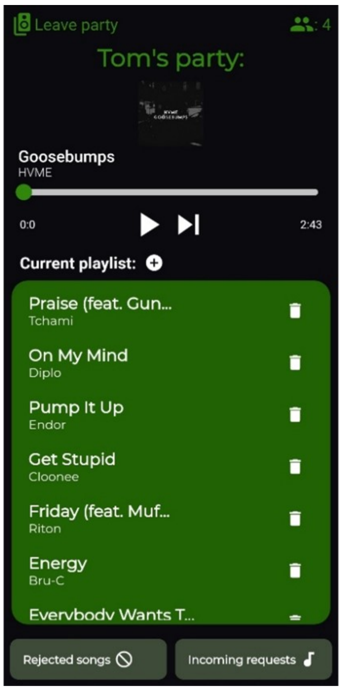


Figure 4.5: Select role screen



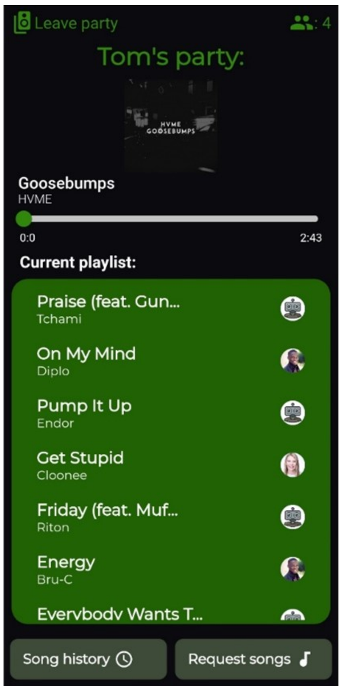
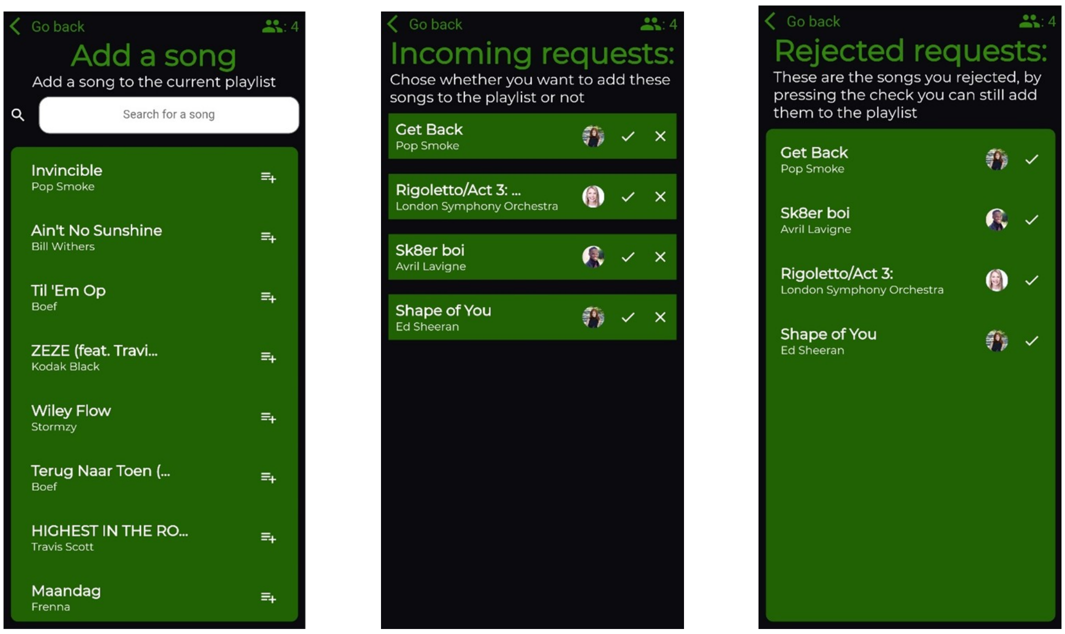


Figure 4.6: Main screen for the DJ/Host (left), Main screen for non-DJ/hosts (right)

Users can sent requests to the host. The host can accept or deny these requests. Approved songs are added at the bottom of the playlist and rejected requests are saved in a separate section (Figure 4.7). Incoming requests can be accessed in the "incoming requests"-screen, illustrated in Figure 4.7.


Go back 4

Add a song

Add a song to the current playlist

Search for a song

Invincible
Pop Smoke

Ain't No Sunshine
Bill Withers

Til 'Em Op
Boef

ZEZE (feat. Travi...
Kodak Black

Wiley Flow
Stormzy

Terug Naar Toen (...
Boef

HIGHEST IN THE RO...
Travis Scott

Maandag
Frenna

Go back 4

Incoming requests:

Chose whether you want to add these songs to the playlist or not

Get Back
Pop Smoke

Rigoletto/Act 3: ...
London Symphony Orchestra

Sk8er boi
Avril Lavigne

Shape of You
Ed Sheeran

Go back 4

Rejected requests:

These are the songs you rejected, by pressing the check you can still add them to the playlist

Get Back
Pop Smoke

Sk8er boi
Avril Lavigne

Rigoletto/Act 3:
London Symphony Orchestra

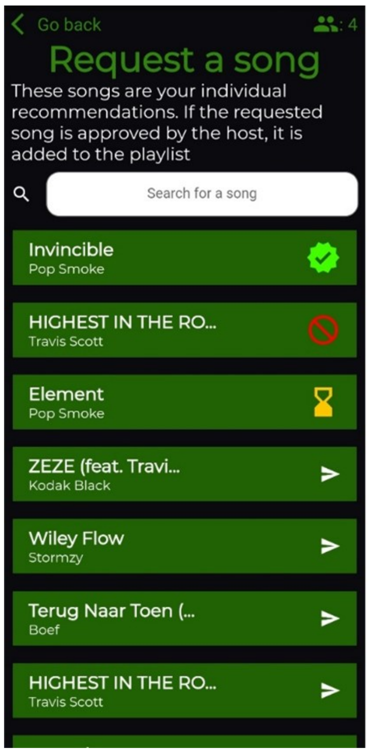
Shape of You
Ed Sheeran

FIGURE 4.7: Screen to add a song (left),Incoming requests screen(middle), Rejected request screen (right)

On the main screen of the other users than the host each song is accompanied either with a robot-avatar or a profile picture. Songs recommended and added by the system are accompanied with the robot avatar. A profile picture next to a song indicates that this user requested the song and it was approved by the host.

These people can only influence the played music by sending requests to the host. Pressing the "request songs"-button on the main screen will take them to the corresponding section. The screen in Figure 4.8 is shown, it gives a list of personal recommendations. The user can also choose to request a specific song, using the search bar. A request is sent by pressing the send-icon next to the song.

The previous sent requests are saved on top of the list and are accompanied with a symbol. This icon represents the status of the request. A green check means the song is approved and will be played soon. Next, a red icon indicates the rejection of the requested song. Finally, the orange hourglass signifies the user that the request is still pending.


Go back

: 4

Request a song

These songs are your individual recommendations. If the requested song is approved by the host, it is added to the playlist

Search for a song

Invincible
Pop Smoke

HIGHEST IN THE RO...
Travis Scott

Element
Pop Smoke

ZEZE (feat. Travi...
Kodak Black

Wiley Flow
Stormzy

Terug Naar Toen (...
Boef

HIGHEST IN THE RO...
Travis Scott

Figure 4.8: Request a song screen

4.3 Results user study 2

4.3.1 Second iteration recruitment

The amount of users recruited for the second iteration was 26 participants. An even number was chosen for an even counterbalancing: the amount of people who received the veto power approach first was equal to the number of participants who received the democratic approach first. There were no specific restrictions for the participants. Participants were contacted through Facebook or acquaintances were recruited for this user study.

Most of the participants were students and aged between 20 and 23 years, as these are the main target audience of the application. The average age of the participants was 26 years old and 30% of them were female. All participants were either average or advanced computer users. Five participants didn't have an active Spotify subscription, one mainly used YouTube[4] to listen to music and the others didn't use any music streaming platform. Figure 4.9 shows the time the recruited users spent on Spotify. 80% of the participants knew that Spotify offered personalized recommendations, but only 27% often listen to these recommendations.Table D.1 in Appendix D shows tables with the individual demographics of each participant together with the received order of tasks.

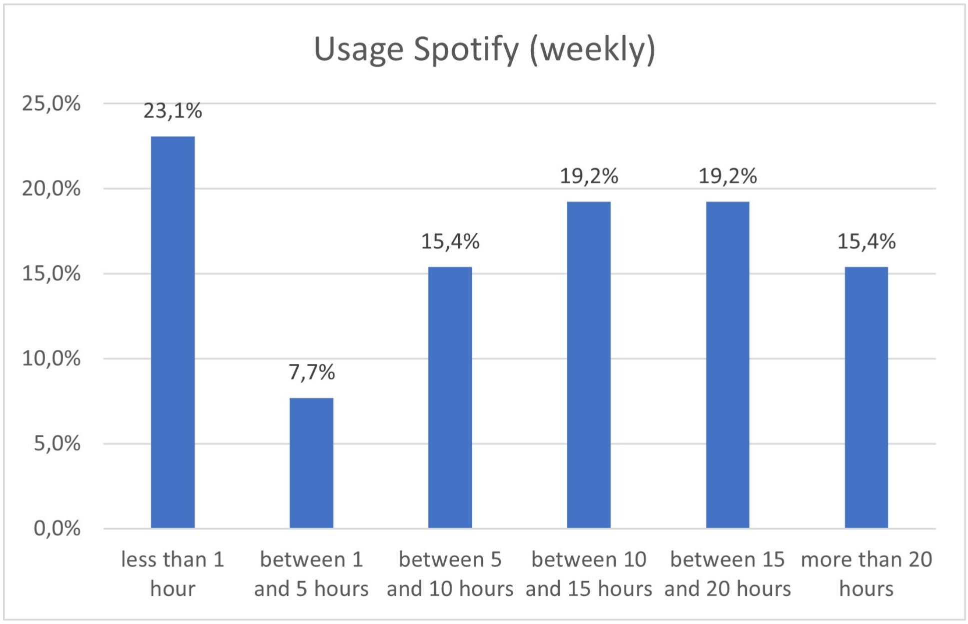


FIGURE 4.9: Time recruited participants spent on the Spotify application per week (iteration 2)

[4]www.youtube.com

4.4 Usability results

The overall results of the usability and cognitive load show that both applications have an easy-to-use interface and require a low cognitive load. Figure 4.10 shows the box-plots of the average SUS-score for both versions. These descriptive results show a slight difference in terms of usability. The democratic version has a total average SUS-score of 85 and the veto-oriented version has a score 87,5. A Wilcoxon Signed Rank test was performed to find out if this small difference was significant. This test confirmed that the usability of the veto-oriented version is significantly better (p<0,05), as shown in Table 4.1.

To see if these scores are sufficient, the scale on Figure 4.11 is used [6]. It illustrates that both applications score excellent.

	P
Average SUS-democratic vs SUS-veto	0,005

Table 4.1: Wilcoxon Signed Rank test on average total SUS-scores (veto vs democratic version)

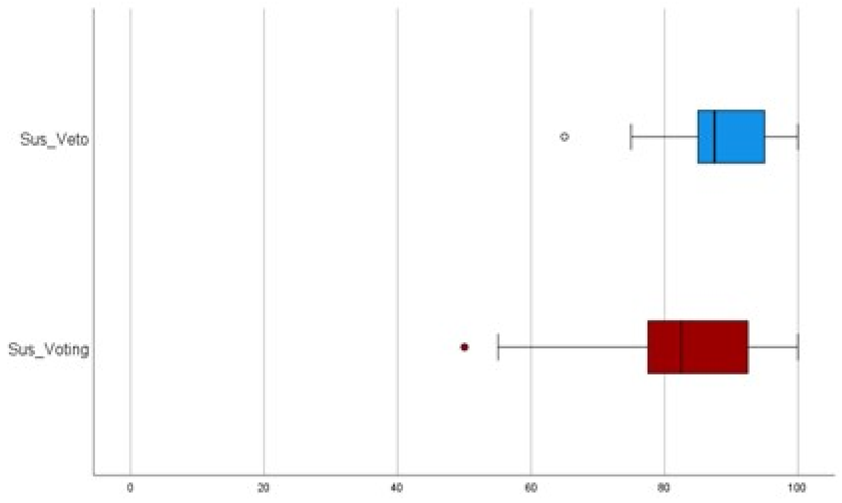


Figure 4.10: Box plot of average total SUS-score (blue:veto and red:democratic)

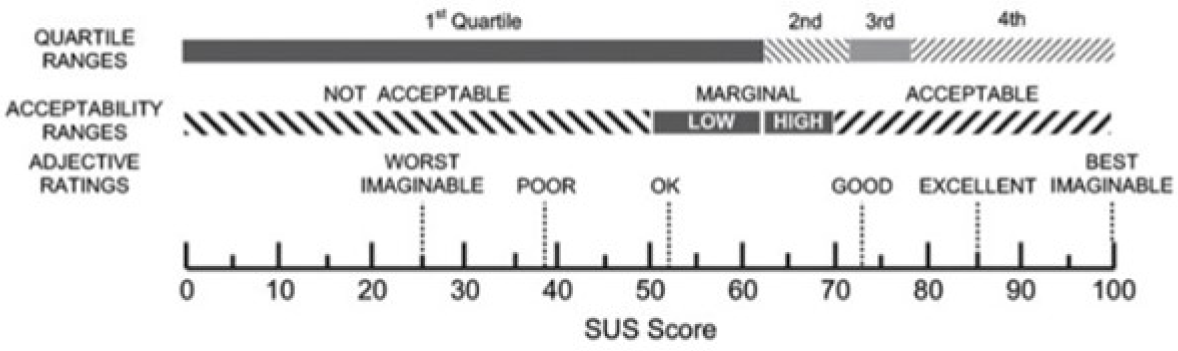


FIGURE 4.11: Scale to see if SUS score is sufficient (red dot: score of democratic version, blue dot: score of veto-oriented version) [6]

However the qualitative results showed that there were some functionalities in the democratic approach which were not clear for the participants. It was not clear that songs needed votes to become playable and how many likes a song needs for this. The same ambiguity was established for songs added by the recommender system. These are removed when received a certain amount of downvotes. One participant stated: "The difference between the upvote and downvote button is not clear". Another stated: "I didn't know that a song became playable when it had three upvotes". A similar statement was: "I didn't know that a song with three downvotes is removed from the playlist". Finally, one participant mentioned: "I didn't know that the grey song wasn't added yet and required upvotes."

Further analysis of the qualitative results showed that the participants voted for the veto-oriented approach, because it was easier to use and less complex. To see whether this result is supported by the quantitative measures, the individual answers of the SUS-questionnaire are analysed. Figure 4.12 and Figure 4.13 show the box plots of the individual answers. The focus here is on the questions which measure ease-of-use and complexity. Both questions seem to be answered in favour of the veto-oriented approach.

To see whether this difference is significant, a Wilcoxon Signed Rank test was performed, as shown in Table 4.2. This test revealed that the difference in ease-of-use is significant (p<0,05), but not for the complexity. It supports the retrieved result from the qualitative analysis, that the veto-oriented version is easier to use.

However since this Wilcoxon Signed Rank test is executed on ten different questions, it is possible that one of them is significant due to noisiness. The more measurements we take, the bigger the chance that the noise will make a measurement meaningful while in reality it's not. To avoid an error of type I (incorrectly rejecting the null-hypothesis), the Bonferroni correction is applied. The Bonferroni correction [2] takes this noisiness into account. The difference is only significant when the p-value is below 0,05 divided by the number of variables. So for this case it means for a difference to be significant: $p < 0,05/10 \Rightarrow p < 0,005$. In this case, it means that the difference in ease-of-use is not significant. To find out if there might be a significant difference, another user study is needed with more participants. Other differences were not significant.

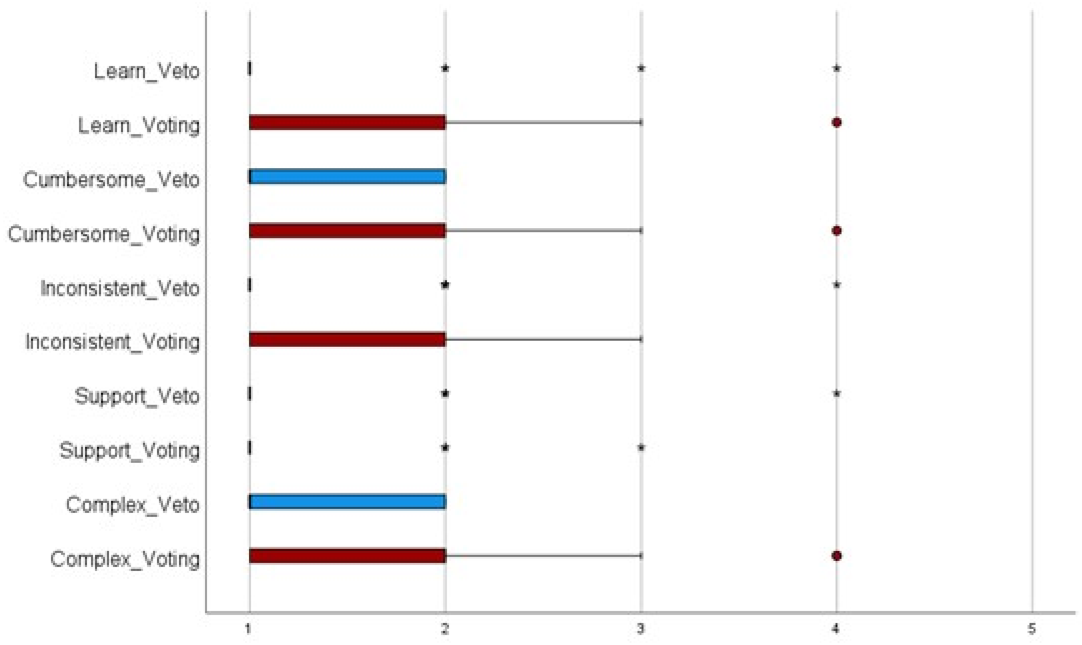


Figure 4.12: Box plots of the individual answers, only the ones which measure the negative aspects of the SUS-questionnaire

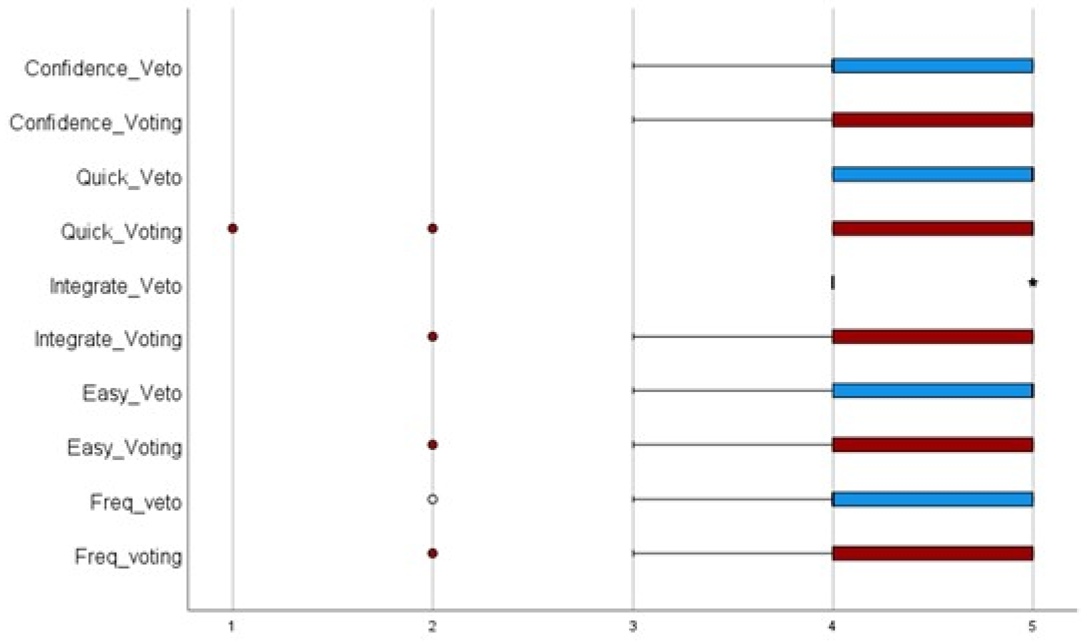


FIGURE 4.13: Box plots of the individual answers, only the ones which measure the positive aspects, of the SUS-questionnaire.

Question	**P-value**
Frequency	0,225
Complexity	0,088
Easy to use	0,046
Needed support of technical person	0,414
Well integrated	0,285
Inconsistency	0,058
Learned quickly	0,070
Cumbersome	0,218
Confidence	0,070
Needed to learn a lot	0,161

TABLE 4.2: Resulst of Wilcoxon Signed Rank test on questions of SUS-questionnaire

4.5 Cognitive load results

The same analysis was used for the results of the raw NASA-TLX questionnaire. The analysis of the quantitative results show a slight preference towards the veto-oriented approach, as it seems that this version scores better on all questions. The difference between both versions is the biggest for mental load and effort. The results of the individual questions are displayed in the box-plots of Figure 4.14.

A Wilcoxon Signed Rank test was used to find out if there was a significant difference between the two versions. The results of the test are summarised in Table 4.3. Only the question which measured the performance was answered significantly different (p<0,05), in favour of the veto-oriented approach. The participants seemed to be more successful in completing the given tasks on this version. This success can further confirm that the democratic version is more difficult to use and more complex. After executing the Bonferroni correction to account for the noisiness of the data, we see that this significance disappears: $p < 0,05/6 \Rightarrow p < 0,008$.

All other differences were not significant. The absence of significant differences can again confirm that the difference between the two prototypes is small.

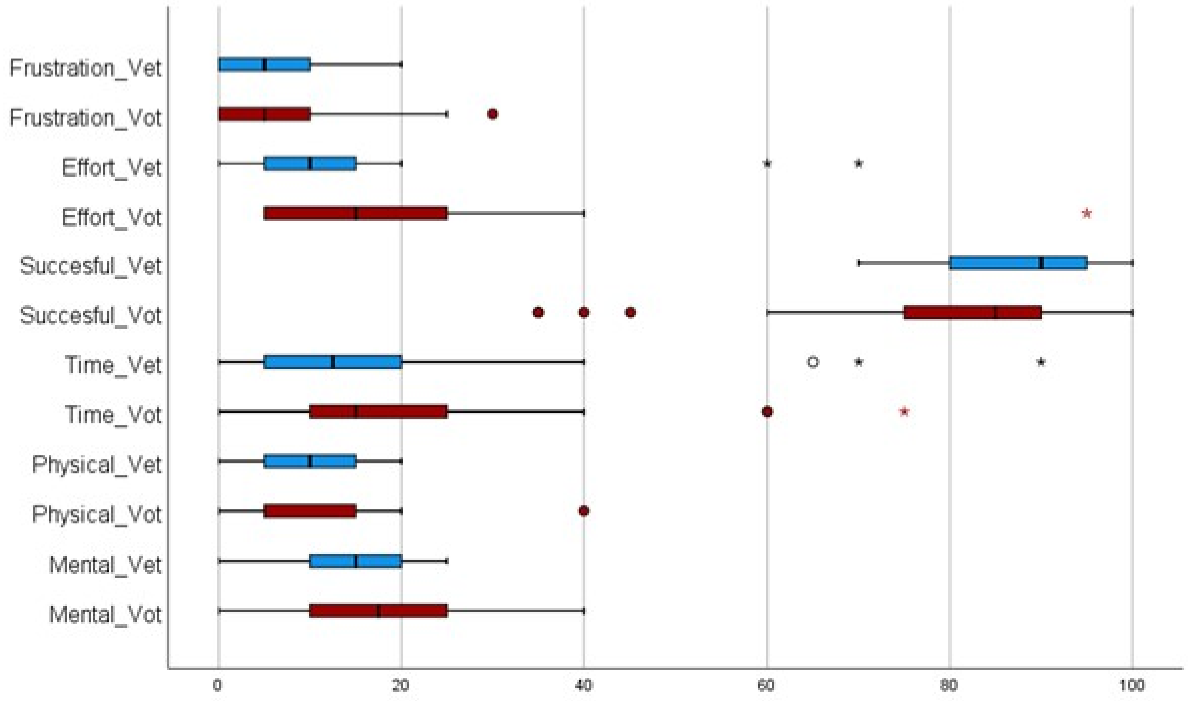


Figure 4.14: Box-plots of the individual questions of the NASA-TLX questionnaire

Question	P-value
Mental load	0,097
Physical load	0,772
Time pressure	0,464
Performance	0,043
Effort	0,189
Frustration	0,297

Table 4.3: Results of Wilcoxon Signed Rank test on questions of NASA-TLX questionnaire

After these quantitative results, the conclusion might be that the veto-oriented version is slightly better. The difference between both versions was rather small. Further analysis of the qualitative results of this user study showed that the participants prefer an approach where the music is selected as a group. But since the veto-oriented version was easier to use and the design was less complex, it was voted for by most of the users. This result together with the small difference in usability and cognitive load makes it difficult to make a hard conclusion on which prototype is best.

4.6 Conclusion user study 2

This user study was conducted to find out if both presented prototypes were easy to use and require a low cognitive load. The conclusion from the quantitative results is that this is indeed the case. Both prototypes have an excellent score on the SUS-scale and have low scores regarding the work load. Further analysis of the descriptive statistics showed that it seems that participants prefer the veto-oriented approach. Inferential statistics are used to find out if this result would also be applicable when the same test is conducted on different populations. This seems to only be the case for the questions which measured the ease-of-use for the SUS-questionnaire and the question which measures the performance for the NASA-TLX questionnaire. However, when noisiness of data is taken into account using the Bonferroni correction, this significance disappears. The analysis of the qualitative results revealed that participants voted for the veto-oriented approach because it was easier to use and less complex and not because this version has a host. When the participant was asked whether he/she likes one person choosing the music or to select the music as a group, they almost always preferred the latter. Taking all this into account, it makes it difficult to make a hard conclusion on which prototype is best. The qualitative results also showed that there was some confusion regarding the functionality of adding and removing songs in the democratic version.

Chapter 5

User study 3

The aim of this final user study was to find out what the influence of both prototypes is on the social dynamics of an event, with the focus on group cohesion. The participants were invited to think about possible effects of a veto-oriented or a democratic group music application on these social dynamics.

As discussed in the literature review, cohesion is considered to be a multidimensional and multilevel construct. It is multidimensional because it can be subdivided in social and task cohesion. The focus of this user study lies on the social cohesion of the event, since this is the most important aspect. The goal of the user study is therefore also to find out if it is possible to design an application which improves the social dynamics of a group. A small part will also focus on the task cohesion. Cohesion is also multilevel as it can be seen as the individual attraction to the group and group integration.

All those aspects were measured using a discussion, guided with questions from the researcher. These questions were generated based on the here described concept of cohesion.

5.1 Third user study design

This user study was conducted on groups of 2 or 3 participants, reused from the second iteration. Participants were reused here, because they already knew how both applications work since they tested them before. This implied that they didn't need much sensitizing. Another advantage that came with this approach is that it made it easier for them to think about the possible effects on group cohesion.

The study was initialised with a small sensitizing activity were the functionality of both applications was briefly explained by the researcher. The participants also had the opportunity to play around with both versions to refresh their memory.

After this part a discussion was started by the researcher. Table 5.1 shows questions which were asked to initiate and guide the discussion. These questions were constructed either from previous obtained results, by the researcher himself or by the conceptual framework used in this work:

Question 1: This question was used to find out if such an application has an influence on the task cohesion. Does this app induce a feeling of responsibility to complete a certain task and is this good or bad? Can this task-feeling make you feel more integrated in the group? (Question constructed from the conceptual framework: task cohesion)

Question 2: The goal was to find out if such an application might induce getting to know people more easily. Would you be more inclined to approach someone if he/she added a certain song to the playlist, for example. This can improve the integration of people into the group. (Question constructed from the conceptual framework: group integration, individual attraction and social cohesion)

Question 3: Applications like these can influence the bonding of the group. This question was posed to find out if this could be the case and whether it improves or worsen the bonding of the group. Can it have an impact on how well you get along with other members? (Question constructed from the conceptual framework: group integration and social cohesion)

Question 4: This question is coupled with question 2 as it aims to find out whether an application like this makes other people more approachable. (Question constructed from the conceptual framework: group integration, individual attraction and social cohesion)

Question 5: People not having the application might feel excluded from the group or might get annoyed or they might want to download the app to feel part of the group. Can the application have an influence on how attracted people might be towards a group? (Question constructed from the conceptual framework: individual attraction to the group and social cohesion)

Question 6: Songs proposed by users can be rejected either by the group or by the host, depending on the version. This can influence the feeling of belonging to the group for this user. The question is posed to find out what the effect of rejection of each version can be on the social behaviour of attendees. (Question constructed from the conceptual framework: group integration and social cohesion)

Question 7: Both versions require that people need to be on their phone. The host needs to handle incoming requests in the veto-oriented approach. And users need to vote for songs, for them to become playable in the democratic version. This was something participants mentioned in previous iterations. This question was posed to find out if there might be a better solution. (Question constructed from results of previous iterations)

Question 8: An open question to find out some extra insights on the influence of such an application on the social dynamics. (Constructed by researcher)

Question 9: To find out if this discussion might have an influence on their opinion. This question is also used to see if there was a clear preference towards one of the two prototypes and why. (Constructed by researcher)

Q1. When you are in a party and you use this kind of application, do you feel some kind of responsibility?
Q2. In what ways do these two applications affect getting to know other people inside the party?
Q3. How do these applications affect the atmosphere of the social event?
Q4. How does such an application affect you, when you meet these people again outside of the event/ in other contexts?
Q5. What happens with people who don't use the app, in comparison to people who do?
Q6. How does a song, suggested by a user, being rejected (either by the group or by the host) affect the group feeling for that person?
Q7. In both versions people need to be on their phone. Do you have an alternative to solve this issue?
Q8. How can these kind of applications improve the social dynamics/group cohesion during a party?
Q9. Having this discussion in mind, which prototype do you prefer and why?

TABLE 5.1: Questions posed by researcher to guide the discussion

5.2 Prototypes

The prototypes used in this iteration were the same as in the previous iteration: one democratic version and one veto-oriented version.

5.2.1 Design rationale

Small changes were made to the democratic version, since the previous iteration revealed some ambiguities in the functionality of this version. It wasn't clear that songs needed votes to become playable and what the amount of votes needed was. The same issue were found with songs added by the recommender system. It was not clear that these could be removed from the playlist when received enough down-votes. To make these functionalities more clear, changes were made to the interface of the democratic approach. The dislike-icon next to each calculated song was changed to a delete-icon, to make clear that the song will be removed once it receives enough dislikes. It also helps improving the contrast between the like and dislike button, they were very similar in the previous version. A counter was also added inside of the dislike icon to show how many more dislikes each song needs to be removed. The same reasoning was used for the like icon, which is accompanied by a song proposed by a user. Figure 5.1 shows the updated interface for the democratic version. The veto-oriented prototype remained the same, since there were no issues with this application.

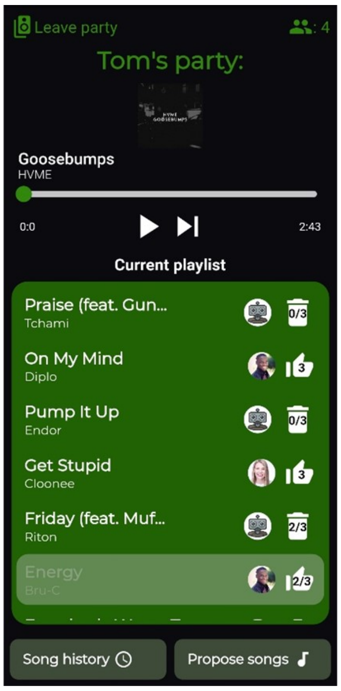


Figure 5.1: Updated main screen from user study 2 for democratic approach

5.3 Results user study 3

5.3.1 Third iteration recruitment

This user study was conducted on three groups of three participants and one group of two participants. As mentioned before participants of the previous user study were reused. 19% of participants were female. The average age of the test-users was 25,7 years old. Figure 5.2 shows the time the used participants spent on their music streaming applications. Most of the participants considered themselves to be advanced computer users (65%). The remaining participants were average computer users. They all knew that Spotify offered personalised recommendations, except one. However only 36% really listened to these recommendations. Table 5.2 shows the individual demographics of each participant.

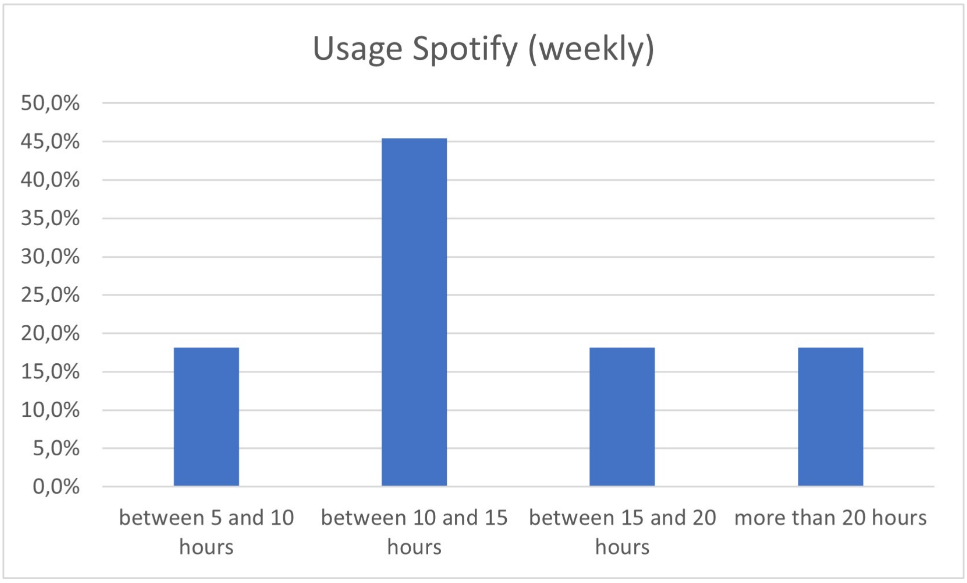


FIGURE 5.2: Time recruited participants spent on the Spotify application per week (user study 3)

ID	Age	Profession	Education level	Computer expertise	Weekly streaming	Group
P1	22	Student	Secondary education	Advanced	5 - 10 hours	1
P2	22	Student	Bachelor's degree	Average	> 20 hours	1
P3	22	Student	Bachelor's degree	Advanced	10 - 15 hours	1
P4	22	Student	Bachelor's degree	Advanced	15 - 20 hours	2
P5	22	Student	Bachelor's degree	Advanced	10 - 15 hours	2
P6	22	Student	Bachelor's degree	Advanced	15 - 20 hours	2
P7	22	Student	Bachelor's degree	Advanced	10 - 15 hours	3
P8	23	Student	Bachelor's degree	Average	10 - 15 hours	3
P9	27	Belgian Defence Officer	Master's degree	Average	10 - 15 hours	4
P10	25	Student	Master's degree	Average	> 20 hours	4
P11	54	Servant	Bachelor's degree	Average	5 - 10 hours	4

TABLE 5.2: Demographics of recruited participants for user study 3

5.3.2 Thematic analysis

To analyse the workshops for this user study a thematic analysis is used. It is the same method used in user study 1 (see chapter 3). The first step is familiarization with the data, where the data is analysed and labelled. Later, themes are formed by putting relevant labels together. Some iterations might be necessary for finding the best themes. Using this method, following themes were found:

- **Don’t hurt my feelings:**
  The application should be designed in such a way that the feelings of the users are never hurt. In particular when a proposed song is rejected, because they might never want to propose a song again or might feel excluded from the group. P3 said: "In a situation where you don’t know everyone and your song is rejected, you might not want to suggest other songs". P10 answered something similar on the effect of a song being rejected: "I would think this is a pity, certainly when you don’t know a lot of people". P5 confirmed: "For some people it might have a negative impact". P6 added: "certainly when you are new in the group".

  Participants stated that the rejection of a song in the democratic version is harsher. It stings, because it feels like the group rejected the song. While a rejected song in the veto-oriented version hits less hard, because only one person denied the request. This difference is mentioned by P7: "When you use the version with the host it will have a smaller impact, in comparison with receiving zero up-votes from the group." P8 answered: "Indeed, in the veto-oriented version I would request a new song more easily if my previous song was rejected."

- **Create a design without obligations:**
  Participants often liked the fact that they can express their musical taste, but don’t have an obligation to do so. They don’t feel any responsibility towards a task, because if they choose to not use the application music is played anyway. P4 said: "The less responsibility the better. It is nice that you have the ability to propose a song, but you don’t have to do this". P5 confirmed: "The less obligations you feel, the better". P2 also stated: "The application doesn’t cause any obligations, but if I want to listen to a song, I would just add it". P10 shared this opinion with: "It is nice to share your taste, but it doesn’t feel like an obligation". Finally, P1 and P11 mentioned that it is nice to have an influence on the music, if you want to.

  The quality of the recommender system can have an influence on this feeling of responsibility. If bad music is being played, users might feel more obligated to request songs. P1 said: "If the bot plays nice music, then I would just leave it. If it doesn’t work well, I might feel more obligated to intervene". P3 confirmed this.

- **Improves group integration:**
  The use of either prototype can have a positive influence on the integration of a member in the group. This theme was expressed by several participants. They stated that when someone requests a song which you really like, the chance of starting a conversation is bigger. All groups highlighted the aspect that the application causes a topic to talk about and it is an icebreaker to start a conversation. P4 mentioned: "When someone plays a nice apres-ski song, I would approach them". P6 backed it up with: "If someone plays an unpopular song that you like as well, it can cause a conversation". P11 also agrees by

saying: "If, for example someone requests a song of a band that I also listen to, I would approach him and ask him where this interest comes from". When there are new members in a group, the application might also help getting to know them.

Also, participants mentioned that they feel more integrated when music similar to their taste is played. P1 said: "If I am new to a group and they play similar music, I might feel a better bond with them". P3 stated: "Songs receiving likes also improves the feeling of belonging to the group". P7 mentioned: "Having people with similar tastes in music, causes a common interest and a nice subject to talk about". P11 confirmed: "I feel more integrated when the requested music, is music I like to listen to".

- **Control over the time table:**
  A functionality to shift the order of or to skip songs is requested by several participants. This flaw is highlighted in the democratic version, since there is no possibility to do so. Group 2 mentioned this: "If a long song is played, you might want the functionality to skip this". P7 stated: "If the group is hyped for a certain song, they have to wait the entire playlist for it to play".

- **Democratic version:**
  Overall the democratic approach is favoured by all groups. All members found this the best way to go when attending a small social event. The democratic aspect causes the group feeling of selecting music together. P5 stated: "In the democratic version everybody together is selecting the music and this creates a group feeling". P1 also brought this up: "You can better avoid music which isn't suited as a group in the democratic version".

  The democratic version is also better in dividing the responsibility over all members. P4 stated: "With the democratic version the responsibility is divided over all members". And as mentioned in the second theme, the feeling of being obligated to do something should be avoided.

  Besides, with the democratic approach the participants felt that their opinion would be appreciated more, because it is easy to like or dislike songs. P8 mentioned that this has benefits for people who normally don't have a lot of input.

  However, the democratic version does have some disadvantages which were mentioned by the participants. Songs rejected by the group hurt more then in the veto-oriented approach. As shown in the first theme, users might then feel less appreciated and could doubt to propose other songs.

- **Veto-oriented version:**
  The veto-oriented approach however has some advantages as well. Participants highlighted the benefits of having this version on bigger events. One person should then be selected responsible for the music. The responsible person will have to focus on the application most of the time, but this is always the case in bigger events.

The rejection of a song stings less, because the users know that only one person rejected it.

- **The relevance of designing to focus on the social aspect of the event:** A theme that was also found in the first iteration is mentioned several times in this iteration as well. Participants found it important that the main subject is still socializing and not interacting with the application. They would find it annoying if the application took over the event. This concern was mentioned by almost all groups (3 groups).

  Possible solutions to the current prototypes were mentioned by some participants. P9 suggested: "Maybe you can create the playlist two hours before the start of the event, that way people have to spend less time on their smartphones during the event." Another suggested solution for the veto-oriented approach was to accept all incoming request, but keep the functionality to remove songs. The host then only needs to use his phone to remove songs if necessary and not to accept songs (P6). Another improvement could be adding notifications for the host when requests are made (P3). The recurrence of this theme emphasises the importance of designing an application which requires little attention.

5.4 Conclusion

This iteration was performed to examine the influence of a mobile group music recommender system on the social dynamics of an event. Two different prototypes are discussed more specifically, a democratic version and a veto-oriented version. The participants were invited to discuss the social effects in groups of two or three people. The results of these discussions showed some themes which need to be taken in consideration when developing a group music recommender application.

First of all, it is important to design the application in a way that it does not hurt people's feelings. When songs, requested by a user, are rejected, it might upset the user. It is sensitive because the user might feel excluded from the group and might not want to request other songs.

Next, the existence of the application should not generate a feeling of a responsibility. Users like the fact that they have the choice to influence the playlist or not. When a user doesn't use the application, music keeps on playing because of the recommender system. This theme might indicate that the task cohesion in this situation should be avoided. The users don't want to feel a responsibility towards a task. It is better to have a strong social cohesion.

Both applications can improve the integration of members in the group. The fact that you can see who requested songs, makes it possible to approach them. It creates a subject to talk about and it is also a conversation starter. Participants also mentioned that when a group listens to similar music they feel connected with them. In conclusion, a group music recommender application can positively influence the integration in the group and the social cohesion.

Individual music streaming applications have the functionality to shift the order of the playlist or to skip songs. It seems that users also want this functionality in group music recommender applications. When a bad song is playing, they want the ability to skip it. Or when the group is hyped for a certain song, it should be possible to put this song on top of the playlist. The integration of these functionalities is not without implications however. How should it be implemented? There are several possible issues to take into account when implementing this functionality. It can cause social conflicts, if someone wants to change the order, but someone else disagrees. Should there just be one person with this functionality? Then the democratic aspect fades away. Should the songs be ordered according to the amount of likes? This might cause proposed songs never being played. This an invitation for further research to explore this subject.

The democratic version is preferred by all groups, because it creates the biggest feeling of being a united group. Participants apparently find it important to select the music as a group, when they meet in small social events. The voting system makes it possible for every member to express their opinion. This power can make users feel more part of the group and it makes it easier for shy people to have an influence on the music. Because the music is selected by the group, the possible feeling of responsibility is divided over all attendees. When a proposed song is rejected, it might have a negative impact on the user who proposed it. The negative feeling is thought to be worse in the democratic version, because it might feel like the group disliked it.

The veto-oriented approach might not be the preferred version, but it still has some advantages. In bigger events, where you know only a small part of the group, this version is thought to be more suitable. The negative effect of a requested song being rejected is also smaller.

A final recurring theme, is the importance of designing an application with the focus on the social aspect. This theme was introduced in the first iteration and repeated here.

Chapter 6

Discussion

This work is a contribution to the research in investigating the effect of different levels of controls for a mobile music group recommender system. Two possible prototypes were introduced and improved throughout three iterations. Those applications were used to track down possible points of attention when designing mobile group music recommender applications. This research tries to answer the three research questions posed in the beginning:

1. Which are people's expectations when attending a social event (party) regarding different levels of control for a mobile music group recommender system?
2. In what ways do different levels of control affect cognitive load and usability for a group of users on the mobile group recommender system?
3. In what ways do different levels of control affect group cohesion?

This thesis was organized following three iterations. The first iteration was performed to find an answer to the first research question. Participants' expectations regarding different levels of control for a mobile group music recommender system were explored using a co-design session and two low-fidelity prototypes. The second research question was answered in the subsequent iteration. Using an online questionnaire, the usability and cognitive load were estimated for two high-fidelity prototypes with different types of controls. The final user study revealed the effect of a music group recommender system on the social dynamics of an event. Participants were invited to workshops to discuss the impact of a music group recommender system on the social dynamics. The results of these iterations revealed interesting takeaways, discussed in the following sections.

6.1 Final decision on prototype

This work examined the differences between a democratic and a veto-oriented version. Advantages and disadvantages were put side by side when deciding which prototype was best. Throughout this research it was hard to decide between one or the other. After two iterations ((chapter 3) and (chapter 4)) there was no clear preference.

Nevertheless, a third iteration (chapter 5) fostered a decision towards the democratic approach. This final iteration also gave advantages and disadvantages of each version.

6.1.1 Democratic approach

In general, it seems that the democratic version does a better job in generating the feeling of a united group. The selection of music in group creates a feeling of unification, it is something you do together. This was also the most used argument, when participants were asked why they chose for the democratic version. Since they preferred the version which induced the feeling of a united group, it appears that this feeling can improve the user's experience with the recommender system.

Furthermore, it seems that participants dislike the feeling of being obligated to do something. The democratic approach avoids this feeling by dividing the responsibility of selecting music over all attendees. When designing a group music recommender application with controls it can be valuable to confirm that participants can influence the music if they feel the need to do so, but it should not feel mandatory.

The voting mechanism also gives the opportunity to express your opinion. This opportunity can be an asset for introvert/shy people. In a normal setting, their opinions might be skipped. In the democratic approach, everyone's point of view is taken into account and equally valuable. Therefore it creates a chance for everyone to express their musical taste. This finding can indicate the possible importance of keeping the gap to express your opinion in a democratic recommender system small.

However, this prototype also revealed an implication. When a proposed song didn't receive votes it is able to hurt the user who requested it. He/she might feel excluded from the group, because it could appear like the group deliberately didn't vote for his/her song. The rejection of songs might therefore negatively influence the user's experience with the group recommender system. This is an invitation to consider this possible negative effect when designing a democratic music group recommender application.

6.1.2 Veto-oriented approach

Some of the disadvantages of the veto-oriented version are the advantages of the democratic version, because of the opposite nature of both prototypes. Firstly, the feeling of selecting the music as a unified group is more absent in this version. There is one person who has the final say. Secondly, people still have the opportunity to express their tastes, but it is not guaranteed that the host considers these.

In contrast with the democratic version, it seemed that the veto-oriented approach generated a big responsibility for the host. He/she has the task to select the music during the whole event. For the host it appears to be mandatory, because he/she is the only one with the ability to add or remove songs. This obligated feeling should be avoided in small events. Participants provided scenarios however where the veto-oriented approach might be more suitable. In bigger events, where you only know a small part of the attendees, it is better to have one person in charge. A DJ is chosen to play music during (a part of) the event. This approach is currently used by almost all big events. This finding can be useful when designing a group music recommender application for bigger events.

Another advantage of this version is the lighter footprint of a rejected song. When a song gets rejected, the user who proposed it knows that only one person disagrees. Hence, it will have a smaller chance of worsening the feeling of being integrated in the group.

6.1.3 Integration of both versions

Considering these advantages and disadvantages, integrating both approaches might result in the best possible version. Participants showed this tendency when evaluating the democratic version in the first iteration (chapter 3), where a contradiction was found: Users preferred selecting music as a group, but still wanted one person to take over when things get out of hand. The third iteration (chapter 5) revealed a similar tendency, where the participants mentioned the lack of functionalities to change the time table of the playlist.

6.2 Relevance of social dynamics in group music recommender systems

The present thesis has provided an introductory knowledge to understand the effects of a music group recommender system on social dynamics. The importance of the social aspect already became clear in the first iteration (chapter 3). Participants mentioned the importance of socializing when attending an event. Socializing is prioritised over interacting with the recommender system. An easy to use application ensures that attendees of an event can focus on socializing with other people. When further exploring the social dynamics in the third iteration (chapter 5), this feeling was confirmed. Some of the most crucial findings about social dynamics found in the third iteration are stated in this section.

6.2.1 Impact on users' feelings in groups during an event

It seems that users feelings have a relevant role in groups of people during an event such as a party. This is a relevant design consideration for this kind of applications. The thematic analysis of the third iteration revealed that it is important to avoid users' feelings getting hurt. In general, you must be careful about users' feelings. A recommender system might scar the relation with the group and alter users' social behaviour in a negative way. This finding might be applicable to group recommender systems in general, however future research needs to confirm this.

For example, a song being rejected, in a music group recommender system, was a possible way of hurting people's feelings in this research. It might cause that the user doesn't want to request other songs ever again. Or he/she might not feel appreciated by the group anymore. Therefore, it is important to design the application in such a way that it wraps this rejection as nice as possible, to minimise any possible negative effect. Further research can dig deeper in what other possible ways people might get hurt and feasible ways to avoid this. It is important to consider these implications when designing, because if the application didn't exist those implications wouldn't exist either.

6.2.2 Group recommender systems influencing social integration

Using a group recommender system in general can be a stimulus for the integration of members in the group. Participants indicated that requested songs might provoke other users to approach the user who requested it. It is an easy subject to talk about and it facilitates starting a conversation. The profile picture of a user next to the song facilitates this integration. The application causes a social effect which is normally more absent in events without it. Adding profile pictures can be something to keep in mind when developing a music group recommender system. Caution should be taken however, because a user might not want the group to know that they proposed a certain song.

When a group listens to similar music, people feel a connection with the group, improving the group cohesion. Participants often mentioned the improvement on the bonding when people listen to the same kind of music.

It is also possible that the application has a negative influence on the social dynamics. What happens if the group plays music dissimilar to your music taste? Or might the use of the application cause the creation of separate groups? It is important to take these possible negative effects in regard when designing a music group recommender system. Further research might explore possible ways to promote these social effects inside a social event. Another invitation is to explore the negative effects of a music group recommender system on the social dynamics.

6.2.3 Conclusion relevance of social dynamics

The results of this section show some of the possible influences of a music group recommender application on the social dynamics. Therefore, this is a strong invitation for other researchers or app developers to consider the social dynamics when evaluating/designing a music group recommender system and maybe a group recommender system in general.

6.3 Reflection on methodology

Due to the mixed nature of the different iterations, where quantitative and qualitative measurements were used, it was interesting to notice that the results of the first two iterations (chapter 3 and chapter 4) were not sufficient to decide between one of the two prototypes. Comparing the veto-oriented version with the democratic version in terms of usability and cognitive load didn't reveal a clear preference. It is when the comparison between both prototypes in terms of their effect on social dynamics was explored (third iteration chapter 5), that participants considered the democratic approach to be better. The third iteration revealed advantages and disadvantages of both prototypes, which made it possible to make a final conclusion. The final iteration revealed some implications on the social dynamics for the veto-oriented approach, for example. These valuable kind of results were lacking in the second iteration. Therefore this is an invitation to encourage mixing quantitative and qualitative research.

6.4 Design recommendations

To conclude, this research provided answers to the posed research questions.

The users expectations were found in the first iteration. Because of the contradictory results in terms of having a host or selecting the music as a group, it was difficult to select one preferred version. Users expect an application with a low need of attention. The emphasis of an event is and remains on socializing. Therefore, when designing an application where the social context is important, the application should be easy-to-use and require a low cognitive load.

The results of the second iteration didn't reveal big differences between the two versions. This made it again hard to decide on one of the two versions. The different methodologies of voting and having one person with a veto didn't seem to alter the usability or cognitive load much. Both applications scored sufficiently on both aspects.

Finally, the final iteration showed a tendency towards the democratic approach. The qualitative approach of this iteration revealed advantages and disadvantages which were absent in the results of the second iteration. This result encourages combining both quantitative and qualitative research. The final iteration also revealed the possible effects of such an application on the social dynamics. It can improve the group integration, but it can also worsen the group feeling (rejecting songs). This user study emphasises the relevance of considering these effects when designing a music group recommender application. Some possible design takeaways were revealed as well.

A group recommender system generates opportunities of uniting groups. The democratic approach revealed positive influences on the unification of the group. This group feeling can have a positive influence on the experience with the recommender system. Therefore it might be interesting to consider this while developing one.

For small social events, design an application, where interaction with it, is not mandatory. A possible way of achieving this, is by distributing responsibility over different group members. One responsible person seems to be more suitable for bigger events.

The design of the interface can create opportunities for introvert/shy people to express their opinions. The voting mechanism lowered the gap to express musical taste.

Consider users' feelings when designing group recommender systems. The existence of a recommender system can alter the behavior of groups, positively or negatively.

The addition of profile pictures can be a possible way of improving group integration. It opens a door for conversation with someone who requested a song.

Chapter 7

Conclusion

This work investigates what implications there are when designing a mobile music group recommender application and what possible effects such an application can have.

The first iteration consisted of workshops where potential users could state their expectations. They had the opportunity to design their own interface in a co-design session. The researcher also presented his own low fidelity prototypes, a democratic version and a veto-oriented version, allowing the participants to discuss possible advantages or disadvantages. The results of this iteration revealed some interesting themes:

- The relevance of designing to focus on the social aspect of the event:
  The application should require a low cognitive load and high usability. This will prevent users from being on their smartphone most of the time.

- Contradictory preferences between the veto-oriented and democratic designs:
  Participants preferred selecting the music as one group, but they would like to have one person who can take over control when necessary.

- Importance of group-oriented design:
  Participants highlighted the group-oriented design of the democratic version, which pleased them.

The two low fidelity prototypes were converted to real applications for the second iteration. The usability and cognitive load of both applications were measured, to find out whether one of them was clearly better. An easy-to-use application which requires low cognitive load makes sure that participants can focus on socializing. The second iteration was performed to make sure that both developed prototypes ensured these specifications. The results of this user study were not so fruitful unfortunately. Both applications had a sufficient and similar score for usability and cognitive load. There was no clear preference towards one prototype, which made it impossible to decide which version was best.

The third and final user study explored the effect of such an application on the social dynamics of an event. This iteration was conducted in workshops with groups of two to three people. The effect of a group recommender system on social dynamics is something relatively new in the research on group recommender systems. It revealed some interesting themes:

- Don't hurt my feelings:
  It is important to be cautious about users feelings when designing a music group recommender application. The application should avoid users getting hurt. Participants pointed out that a user might feel hurt, because his/her requested song got rejected.

- Create a design without obligations:
  Users pointed out the option of requesting songs. They liked that it was not mandatory to use the application. Music will always be played even if they didn't use the application.

- Improves group integration:
  The existence of a group recommender system might improve the integration of members in the group. It creates a topic to talk about and is an easy way to start a conversation. People also feel connected with other people who have similar taste in music.

- Control over the time table:
  The lack of possibilities to change the order of songs or to skip songs in the democratic version was often mentioned by the participants. They would like have a functionality where it is possible to do so.

- The relevance of designing to focus on the social aspect of the event:
  This theme also occurred in the first iteration. The fact that it reoccurs here emphasises its importance.

- Veto-oriented approach:
  In bigger events the veto-oriented approach might perform better according to the participants. Songs being rejected in the veto-oriented version also sting less for the user who requested the song. A rejection in the democratic version might appear as the group rejecting the song.

This final iteration also made it possible to make a final decision on which prototype was best. All groups of the final iterations chose the democratic version. Participants who voted for the veto-oriented application in previous iterations changed their minds, because of the positive effects on the social dynamics of the democratic version. The democratic version distributes the responsibility of playing music over all present members. Next, it causes a group feeling, because you select the music as one group. Finally, everyone's opinion is equally important. It causes users to feel more appreciated by the group.

7.1 Suggestions for further research

The songs in the current democratic prototype needed a majority of votes to become playable. Songs were ranked in the playlist based on their chronological order (FIFO). However, there are other possible voting and ranking methods. Further research can investigate possible alternatives and determine the most suitable one.

This research explores the effect of a music group recommender system on the social dynamics. It is an exploratory study in this topic, but it already showed some promising results. These results indicate the relevance of considering those social dynamics when designing a group music recommender application. Those considerations are possibly also important for other group recommender systems. Currently, there is not much research about this topic yet. Therefore this is an invitation to further research possible effects of a group recommender system on social dynamics. Some specific invitations on this topic are made in chapter 6:

- Can recommender systems hurt users' feelings? If so, how do you avoid this?
- How do you promote social dynamics using a group recommender system?
- What are possible negative influences on the social dynamics?

Participants preferred the democratic approach in small social events. For bigger events they thought that the veto-oriented version is more suitable. Another iteration might be needed to confirm or deny whether a veto-oriented application is indeed better for bigger events. Yet another research might reveal possible advantages or implications of the integration of both versions.

A more specific suggestion for further research is to find out if there are significant differences between the democratic and the veto-oriented version. The second user study revealed that the veto-oriented version significantly scored better in ease-of-use (SUS-questionnaire) and performance(NASA-tlx questionnaire). When the noisiness of data was taken in regard, this significance disappeared. Further research might investigate whether there is a significance in a bigger population.

Appendices

Appendix A

Demographics questionnaire (user study 1 and 2) and sensitizing activity (user study 1)

Q1 Fill in the user-ID provided by the researcher (p1, p5 for example)
Q2 How old are you?
Q3 Nationality
Q4 Last education level achieved?

- Master's degree (1)
- Bachelor's degree (2)
- Doctoral degree (3)
- Secondary education (4)
- Primary education (5)

Q5 Current profession or occupation Q6 How would you rate yourself as a computer user

- No experience (1)
- Beginner (2)
- Average (3)
- Advanced (4)
- Expert (5)

Q7 For how long are you an active user of Spotify?

- < 1 year (1)
- Between 1 year and 3 years (2)
- Between 3 years and 5 years (3)
- 5 years (4)

Q8 Amount of time per week using Spotify

- Less than 1 hour (1)
- Between 1 and 5 hours (2)
- Between 5 and 10 hours (3)
- Between 10 and 15 hours (4)
- Between 15 and 20 hours (5)
- More than 20 hours (6)

Q9 Did you know Spotify offers you music recommendations?

- Definitely yes (1)
- Probably yes (2)
- Might or might not (3)
- Probably not (4)
- Definitely not (5)

Q10 Did you know these music recommendations are personalised?

- Yes (1)
- No (2)

Q11 How often do you consider these recommendations useful for you to decide which song to listen to?

- Every time (1)
- Most of the time (2)
- Sometimes (3)
- Almost never (4)
- Never (5)

Q12 How often do you listen to music that is recommended to you?

- Never (1)
- Rarely (2)
- Sometimes (3)
- A lot (4)
- All the time (5)

Q13 To which degree do you feel that you understand why certain music is recommended?

- A great deal (1)
- A lot (2)
- A moderate amount (3)
- A little (4)
- None at all (5)

Q14 How much control do you think you have over the content that is recommended?

- A great deal (1)
- A lot (2)
- A moderate amount (3)
- A little (4)
- None at all (5)

Please open Spotify on your smartphone and explore your recommendations. Go to the library in Spotify (bottom right of your screen) -> click on create playlist (fill in a random name) -> Look at the recommended songs when pressing the "add songs" button. Q16 From your perspective, how well does your music recommendation system give you a sense of understanding how the recommendations are produced?

- Extremely well (1)
- Very well (2)
- Moderately well (3)
- Slightly well (4)
- Not well at all (5)

Q17 From your perspective, how well does your music recommendations system gives you a sense of control over these recommendations?

- Extremely well (1)
- Very well (2)
- Moderately well (3)
- Slightly well (4)
- Not well at all (5)

Q18 From your perspective, how well does you music recommendation system gives you and other users awareness of the music recommendation system's existence and its functionality?

- Extremely well (1)
- Very well (2)
- Moderately well (3)
- Slightly well (4)
- Not well at all (5)

Q19 Please grade your music recommendation system in terms of its capacity to describe your current user profile which is used to calculate your personal recommendations (-3 to 3)
Q20 Please grade your music recommendation system in terms of its capacity to offer you ways to tune or manage your user profile for the music recommendations (-3 to 3)
Q21 Please grade your music recommendation system in terms of its capacity to avoid music recommendations based on previous music consumption (-3 to 3)
Q22 Please grade your music recommendation system in terms of its capacity to offer recommendations to a group (-3 to 3)

Appendix B

Questions posed in user study 2

Open Questions
1. What did you like the most about each prototype? Why?
2. What did you like the least about each prototype? Why?
3. Are there any elements missing in either of the prototypes?
4. Were some functionalities unclear in either prototypes?
5. Which prototype has your preference? Why?

TABLE B.1: Open questions posed at the end of user study 2

Democratic approach
1. Log-in, play the first song and skip the currently playing song.
2. Add a song to the playlist and check if it is added.
3. There is one song requested by a user,
but it doesn't have enough votes to be playable. Add this song to the playlist.
4. There is one song added to the playlist by the system,
but the group doesn't really like this song. Remove it from the playlist.
5. Look-up your previously proposed songs.
6. Find the members of the social event.
7. Find the songs which were played in the past.

TABLE B.2: tasks for democratic approach

Veto oriented approach Host	Non-Host
1. Imagine you're organizing an event and you log in the application	1. Imagine you are invited to someone's party
2. Add a song to the playlist and check if it is added	2. There are two types of songs: songs added by the system and songs proposed/added by a user. Give an example of both cases and explain why.
3. Remove a song from the playlist	3. Find the participants of the social event
4. Approve and deny at least one of the requests of the group members	4. Find the previously played songs
Check if the approved request is added	5. Request a song
The denied request are saved, find them	6. Find your previous requests
5. Find the participants of the event	Explain the symbols next to your previous proposals

Table B.3: tasks for veto power approach

Appendix C

Questionnaire for cognitive load and system usability

Q1 Which prototype was presented to you first? (If you are not certain ask the researcher)

- Veto power approach (1)
- Voting approach (2)

Q2 I think that I would like to use this system frequently (grade from strongly disagree (1) to strongly agree (5))
Q3 I found the system unnecessarily complex (grade from strongly disagree (1) to strongly agree (5))
Q4 I thought the system was easy to use (grade from strongly disagree (1) to strongly agree (5))
Q5 I think that I would need the support of a technical person to be able to use this system (grade from strongly disagree (1) to strongly agree (5))
Q6 I found the various functions in this system were well integrated (grade from strongly disagree (1) to strongly agree (5))
Q7 I thought there was too much inconsistency in this system (grade from strongly disagree (1) to strongly agree (5))
Q8 I would imagine that most people would learn to use this system very quickly (grade from strongly disagree (1) to strongly agree (5))
Q9 I found the system very cumbersome to use (grade from strongly disagree (1) to strongly agree (5))
Q10 I felt very confident using the system (grade from strongly disagree (1) to strongly agree (5))
Q11 I needed to learn a lot of things before I could get going with this system (grade from strongly disagree (1) to strongly agree (5))
Q12 How much mental and perceptual activity was required? Was the task easy or demanding, simple or complex? (grade from very low (0) to very high (100))
Q13 How much physical activity was required? Was the task easy or demanding,

slack or strenuous? (grade from very low (0) to very high (100))
Q14 How much time pressure did you feel due to the pace at which the tasks or task elements occurred? Was the pace slow or rapid? (grade from very low (0) to very high (100))
Q15 How successful were you in performing the task? How satisfied were you with your performance? (grade from very low (0) to very high (100))
Q16 How hard did you have to work (mentally and physically) to accomplish your level of performance? (grade from very low (0) to very high (100))
Q17 How insecure, discouraged, irritated, stressed and annoyed were you? (grade from very low (0) to very high (100))

Appendix D

Demographics user study 2

ID	Age	Education	Occupation	Active user of Spotify	Spotify usage weekly	Order voting version	order veto (not host)	order veto (host)
P1	22	Secondary education	Student	3 - 5 years	5 - 10 hours	3 4 2 5 7 6	2 5 6 4	2 3 4 5
P2	22	Bachelor's degree	Student	3 - 5 years	> 20 hours	2 5 3 4 6 7	3 5 6 2 4	3 4 2
P3	22	Bachelor's degree	Student	> 5 years	15 - 20 hours	2 5 3 4 7 6	5 6 2 4	2 3 4 5
P4	23	Master's degree	Project Engineer	< 1 year	< 1 hour	2 4 3 6 7	5 6 4	2 3 5 4
P5	22	Secondary education	Student	3 - 5 years	5 - 10 hours	4 3 2 6 7 5	3 4 5 6	2 4 5 3
P6	22	Bachelor's degree	Student	> 5 years	10 - 15 hours	3 4 2 7 6 5	5 6 4 2	4 3 2 5
P7	22	Bachelor's degree	Student	1 - 3 years	10 - 15 hours	2 4 3 7 5	3 2 5 6	2 4 3
P8	22	Bachelor's degree	Student	3 - 5 years	15 - 20 hours	4 3 2 6 7 5	2 5 4 6	4 3 2 5
P9	57	Master's degree	Servant	< 1 year	< 1 hour	6 4 3 2 7 5	3 4 5 6	2 3 4
P10	24	Master's degree	Student	1 - 3 years	10 - 15 hours	3 4 6 2 7	2 4 5 6	3 2 4 5
P11	23	Master's degree	Student	< 1 year	> 20 hours	2 5 4 3 7 6	5 6 2 4	2 3 4
P12	22	Secondary education	Student	3 - 5 years	5 - 10 hours	7 6 4 3 2 5	2 4 5 6	4 2 3 5
P13	20	Secondary education	Student	1 - 3 years	1 - 5 hours	2 5 7 6 3 4	4 2 5 6	3 2 4 5
P14	23	Bachelor's degree	Student	1 - 3 years	15 - 20 hours	4 7 6 2 3 5	4 2 5 6 3	2 3 4
P15	20	Bachelor's degree	Student	3 - 5 years	< 1 hour	6 7 3 4 2 5	2 4 5 6	3 5 4 2
P16	23	Bachelor's degree	Student	3 - 5 years	10 - 15 hours	3 4 7 6 2 5	2 4 3 5 6	4 2 3
P17	27	Master's degree	Belgian Defence Officer	> 5 years	10 - 15 hours	6 7 3 4 2 5	2 4 5 6	3 4 2 5
P18	25	Master's degree	Student	> 5 years	> 20 hours	2 5 6 7 4 3	3 4 5 6 2	4 2 3
P19	54	Bachelor's degree	Servant	1 - 3 years	5 - 10 hours	7 6 4 2 3 5	3 4 5 6 2	2 3 4
P20	25	Master's degree	Student	3 - 5 years	1 - 5 hours	6 7 4 3 2 5	2 4 5 6	5 2 3 4
P21	23	Secondary education	Truck driver	< 1 year	< 1 hour	6 2 5 3 4 7	2 4 5 6	3 4 2 5
P22	22	Bachelor's degree	Student	3 - 5 years	15 - 20 hours	2 5 3 6 7 4	2 5 6 4	4 2 3 5
P23	56	Master's degree	Servant	< 1 year	< 1 hour	6 4 2 3 5	2 3 4 5 6	4 2 3
P24	23	Bachelor's degree	Student	3 - 5 years	15 - 20 hours	7 6 3 4 2 5	2 4 5 6	5 3 2 4
P25	18	Secondary education	Student	< 1 year	< 1 hour	4 3 2 5 7 6	3 5 6 4 2	2 4 3
P26	21	Bachelor's degree	Student	3 - 5 years	> 20 hours	2 3 4 7 5 6	2 4 5 6	5 3 4 2

Table D.1: Demographics of recruited participants for user study 2

Appendix E

Paper

This appendix contains a draft of the paper which I will try to submit in one of the upcoming conferences. A possible deadline for submission might be 15 July or 15 October.

"This DJ is very bad": Exploring the design of control features for group music recommender systems during social gatherings

ANONYMOUS AUTHOR(S)*

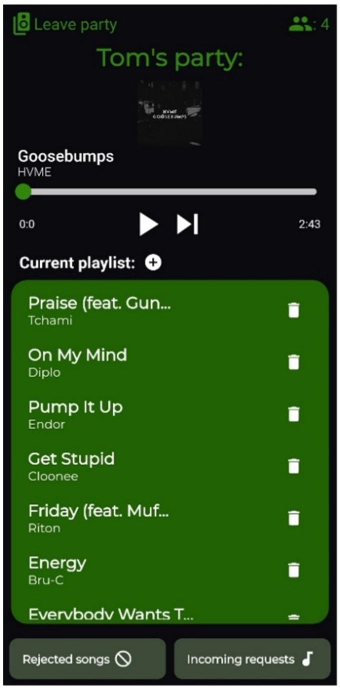
This research is a contribution on designing possible controls for a mobile music group recommender application. Two prototypes were created and improved throughout three iterations. The first iteration was executed to find the expectations of potential users. A thematic analysis revealed that such an application should require little attention, as the focus should be on socializing. The results of the first iteration were used to create two high-fidelity prototypes. One democratic version, where votes are needed for songs to be played. And one veto-oriented version, where there is one member with veto power. A second user study revealed the low cognitive load and high usability of both versions. A low cognitive load and high usability imply more time for socializing. The effect of a music group recommender system on the social dynamics is discussed in the third iteration. The existence of a music group recommender system can positively influence the integration of members in the group. Requesting songs is an easy topic to start a conversation about. Also, participants mentioned feeling more connected to a group with similar taste in music. The rejection of a requested song might negatively influence the group-feeling and behaviour of a user.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

Additional Key Words and Phrases: datasets, neural networks, gaze detection, text tagging

ACM Reference Format:
Anonymous Author(s). 2018. "This DJ is very bad": Exploring the design of control features for group music recommender systems during social gatherings. In *Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION



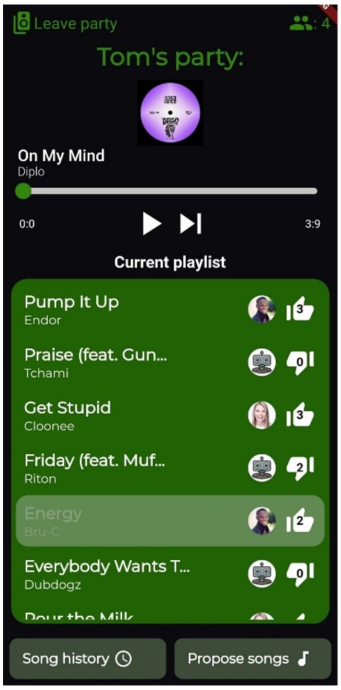


Fig. 1. Main screen veto-oriented version (left), Main screen democratic version (right)

Overall recommender systems are used to help the user make the right decisions. It does so by lowering the information overload of all possible items and recommending the items which it thinks fits the most for the current user. Depending on the feedback of the user it will change the recommended items or it will recommend similar items if the users liked the previous recommended item. There are a number of different activities where such recommender systems can be used, for example listening to music, selecting a series or a movie to watch, choosing what to eat,... These are things that an individual user can do, but these are also activities which can be executed in group. This is a reason why an interest is developed in group recommender systems. This interest in group recommender systems is growing the past years and as it seems it is not going to stop anytime soon [?]. Making decisions in a group is a lot more difficult than making individual decisions. The group recommender system provides help. It will suggest an item for the group by taking in to account the preferences of all members of the current group. This way it lowers the information overload for the group and helps the group in making the right decision. The way the individual preferences are used in the group recommender system depends on the used aggregation function. But there are other factors which need be taken into considerations such as the context and group dynamics.

Group recommender systems are deployed in a lot of different domains and a lot of applications are jumping on the train in developing their own group recommender system. Music applications like Spotify[1] are adding more and more functionality for groups to listen to music together. They implemented a group session[2] where it is possible for a group of people to listen to the same music.

[1]https://www.spotify.com
[2]https://support.spotify.com/us/article/group-session/

They also offer possibilities to make music playlists together, which can be listened to individually or in group. On top of this they provide a 'Family Mix'[3] which is music a group of Spotify users likes to listen to.

Because of the growing popularity of these group recommender systems, it is important to develop the best possible interface. This work contributes to a part of this goal, by focusing on the controllability of a mobile music group recommender system. The effect of the interface on the groups of people will also be measured in terms of cognitive load and perceived ease-of-use. These are important elements to measure, because the attendants of a certain social event should focus on the event, instead of focusing on their smartphone. The implemented interface requires little focus (low cognitive load) and is easy to use (high usability). To the best of our knowledge, the influence on group dynamics are not yet considered in any other papers about group recommender systems. Cohesion is one of these group dynamics and as this is an important feature of a group, this research will explore the influence of the interface on group cohesion. Following research questions will be answered more specifically:

(1) Which are people's expectations when attending a social event (party) regarding different levels of control for a mobile music group recommender system?
(2) In what ways do different levels of control affect cognitive load and perceived ease-of-use for a group of users on the mobile group recommender system?
(3) In what ways do different levels of control affect group cohesion?

2 IMPLEMENTATION

Both prototypes are implemented using Flutter and the Spotify API. Starting from two low-fidelity prototypes, two high-fidelity versions were created and improved throughout three iterations. The functionality of each version is briefly explained in the following subsections. Figure shows both interfaces.

2.1 Democratic version

The democratic version emphasizes the group aspect and votes are needed for songs to be played. Songs suggested by other users can only be played when they receive enough votes from the group. Newly proposed songs are added at the bottom of the playlist. The ones that received three upvotes are added to the playlist and appear green. The proposed songs which require more votes appear grey. If a song without enough votes arrives at the top of the playlist, it is removed from the playlist. In figure ... there is one song proposed by a user that requires one more like to be accepted (song 5). All other songs requested by users are already added to the playlist.

The system keeps calculating and adding songs to the playlist. These songs are accompanied by a robot-avatar and a dislike-icon. Songs added by the recommender system can be removed if enough members (three in this case) press the dislike button. All calculated songs in figure ... have zero dislikes, except one. One song has two downvotes, which means only one downvote is needed to remove this song from the playlist (song 4).

2.2 Veto-oriented version

The focus of this version is the fact that there is one user, the host/DJ, who has more power than the rest of the members. When a user logs in, he/she needs to select which role he/she will fulfil in the event, on the screen in ??. The host has the final decision in which songs are played, his/her main screen is shown on Figure 1. The host can add, remove, play or skip songs. The host can press the litter bin next to a song to remove it from the playlist. Songs can be added by pressing the

[3] https://support.spotify.com/nl/article/family-mix/

plus icon, next to "current playlist". It will lead to his personal recommendations, represented in **??**. He/she can choose to add a song from this list or he/she can look up songs. Songs are added by pressing the icon on the right of the song title. The main screen of the host is shown on figure...

3 FIRST USER STUDY

The goal of the first user study was to find out what the expectations of groups of users are in terms of controllability in a music group recommender system, when they attend a social event (e.g. party).

3.1 First user study design

The first iteration started with a small discussion about their expectations in terms of controls for a music group recommender system.

After this introductory discussion, a co-design session was initiated. The goal for the groups was to build their own interface for a music group recommender system. The group of users had to collaborate to create an interface which had all their desired functionalities. This would help the researcher understand the wishes and expectations of the participants.

The last section of this user study consisted of a part where the researcher proposed his own created prototypes. The presented prototypes were the low-fidelity versions of the ones presented in section 2, they were developed using Figma. The groups were asked to discuss these prototypes and compare them with the interface they developed in the co-design session.

3.2 Results first user study

3.3 First iteration recruitment

15 participants, divided in 5 groups of 3, were recruited for the first iteration. All participating test users are Belgian. The user study was applied on people who have an active subscription on any music platform which uses a recommender system. Other constraints for the users were that they were subscribed for at least a year and used the application frequently (weekly). Most of the participants were aged between 22 and 24 years old and the average age was 26 years. The major part of the participants were still studying (60%) and 20% of the users were female. All recruited participants were either average or advanced computer users and they all knew that Spotify offered personalized recommendations. Only 27% of participants used these recommendations to listen to, most of the time. The remaining participants sometimes listened to them. An overview with the demographics for each participant is given in Table 1 .

ID	Age	Profession	Education level	Computer expertise	Weekly usage	Group
P4	22	Student	Bachelor's degree	Advanced	10-15 hours	1
P5	23	Student	Bachelor's degree	Average	10-15 hours	1
P27	23	Student	Master's degree	Average	5-10 hours	1
P12	24	Student	Master's degree	Advanced	15-20 hours	2
P13	22	Student	Master's degree	Advanced	15-20 hours	2
P14	24	Student	Secondary education	Average	5-10 hours	2
P2	23	Consultant	Master's degree	Average	5-10 hours	3
P15	23	Student	Master's degree	Average	10-15 hours	3
P16	23	Consultant	Master's degree	Advanced	< 1 hour	3
P1	23	Sales Engineer	Master's degree	Average	5-10 hours	4
P3	23	Student	Bachelor's degree	Advanced	5-10 hours	4
P29	23	Student	Master's degree	Advanced	1-5 hours	4
P8	26	Career counsellor	Bachelor's degree	Average	5 - 10 hours	5
P9	27	Consultant	Master's degree	Average	5-10 hours	5
P11	60	Manager	Master's degree	Average	1-5 hours	5

Table 1. Demographics of recruited participants for user study 1

3.4 Results thematic analysis

A thematic analyses was used to process the results, because this is an accessible and flexible tool to identify, analyze and report patterns within qualitative data. The guide provided by Braun et al [?] was used. First, we familiarized our self with the data by listening to/reading the data repetitively and making annotations if necessary. Once familiar with the data, we analyzed the data by finding relevant features and labelling them. This part of the process is called coding. It is important to associate each code with the corresponding quote of the participant. Next, identifying each code that is relevant to the research question and grouping codes together so they formed themes. Some iterations were needed to find the right themes for this research question. The found themes are discussed here:

- **The relevance of designing to focus on the social aspect of the event:**
  Participants preferred focusing on interacting with the people present at the party, rather than with the application. The amount of time spend on the application should be as minimal as possible. An example citation of P4 was: "I would rather have the focus on the social part then on the application".
  Those concerns were expressed for both prototypes. For the veto-oriented approach, P16 said: "Is there a possibility for the host to always accept incoming requests? This functionality can be useful when the host doesn't want to be on his phone". In the democratic approach, P1 stated: "the disadvantage is that everyone is continuously busy with liking or disliking songs".
- **Contradictory preferences between the host- and voting-oriented designs:**
  On the one hand the members of the group felt that it was important to always have one person who is in charge of the organizational part of the application. This person should have the final responsibility. However it is up to the host whether he uses this functionality or not. P4 highlighted this concept with the following quote: "If the party is getting out of hand, the host should have the ability to skip certain songs". But, 80% of the groups preferred the democratic prototype. P13, P12 and P1 highlighted the group oriented design of the

democratic approach which they preferred. P12 said: "it is nice to select the music as a group and not one person who has the final decision".

4 SECOND USER STUDY

The results from the previous user study were used to develop the two high-fidelity prototypes presented in section 2. The aim of this user study is measuring the usability and the cognitive load. The participants also had the opportunity to discuss the usability and to highlight some flaws in either one of the systems.

A within subject user study is chosen to compare both prototypes. The participants used from this study are different from the previous user study.

4.1 Second user study design

Participant were invited to play around with the prototypes, using tasks provided by the researcher. The tasks are designed such that all aspects and functionalities of the application are explored. The order of the tasks were randomized using a Latin square distribution for each participant, to avoid biasing. For the same reason and to counteract the learning effect, the order in which the prototypes were presented was changed for each participant.

After exploring each prototype, the test user is invited to fill in a questionnaire about the usability and the task load of the application. The usability and cognitive load were measured using the NASA task load index questionnaire [**?**] and the SUS-questionnaire [**?**].

The user study was finalized with some open questions to support the provided results.

4.2 Results user study 2

4.2.1 Second iteration recruitment. The amount of users recruited for the second iteration was 26 participants. An even number was chosen for an even counterbalancing: the amount of people who received the veto power approach first was equal to the number of participants who received the democratic approach first.

Most of the participants were students and aged between 20 and 23 years, as these are the main target audience of the application. The average age of the participants was 26 years old and 30% of them were female. All participants were either average or advanced computer users. Five participants didn't have an active subscription on Spotify, one mainly used YouTube[4] to listen to music and the others didn't use a music streaming platform. 80% of the participants knew that Spotify offered personalized recommendations, but only 27% often listen to these recommendations.**??** (TODO add table) in **??** shows tables with the individual demographics of each participant.

4.2.2 Usability results. The overall results of the usability and cognitive load show that both applications have an easy-to-use interface and require a low cognitive load. The democratic version has a total average SUS-score of 85 and the veto-oriented version has a score 87,5.

(TODO na meeting met Nyi Nyi)

4.2.3 Cognitive load results. Comparing both versions in terms of cognitive load reveals that they had very similar scores. There seemed to be a small preference towards the veto-oriented approach in the descriptive statistics. To verify whether this difference was significant, a Wilcoxon Signed Rank test was performed on all individual questions of the NASA-TLX questionnaire. This revealed a significant difference in the performance, in favour of the veto-oriented approach.

[4]www.youtube.com

4.2.4 Qualitative results. Analysis of the qualitative results showed that the participants voted for the veto-oriented approach, because it was easier to use and less complex. But the participants prefer an approach where the music is selected as a group, when asked.

The results also showed that there were some functionalities in the democratic approach which were not clear for the participants. It was not clear that songs needed votes to become playable and how many likes a song needs for this. The same ambiguity was established for songs added by the recommender system. These are removed when received a certain amount of downvotes.

5 THIRD USER STUDY

The aim of this final user study was to find out what the influence of both prototypes is on the social dynamics of an event, with the focus on group cohesion. The participants were invited to think about possible effects of a veto-oriented or a democratic group music application on these social dynamics.

5.1 Third user study design

This user study was conducted in workshops with groups of 2 or 3 participants, reused from the second iteration. Participants were reused here, because they already knew how both applications work since they tested them before.

The study was initialised with a small sensitizing activity were the functionality of both applications was refreshed.

After this part a discussion was started by the researcher. The discussion was initiated and guided with questions of the researcher. These questions were constructed either from previous obtained results, by the researcher himself or by the conceptual framework used in this work:

The feedback of the second iteration was used to update the interface for the democratic version. Some small changes were made to make the functionality of the voting-icons clearer. The dislike-icon next to each calculated song was changed to a delete-icon, to make clear that the song will be removed once it receives enough dislikes. It also helps improving the contrast between the like and dislike button, they were very similar in the previous version. A counter was also added inside of the dislike icon to show how many more dislikes each song needs to be removed. The same reasoning was used for the like icon, which is accompanied by a song proposed by a user. The veto-oriented prototype remained the same, as there were no issues with it.

5.2 Results user study 3

5.2.1 Third iteration recruitment. his user study was conducted on three groups of three participants and one group of two participants. As mentioned before participants of the previous user study were reused. 19% of participants were female. The average age of the test-users was 25,7 years old. Most of the participants considered themselves to be advanced computer users (65%). The remaining participants were average computer users. They all knew that Spotify offered personalised recommendations, except one. However only 36% really listened to these recommendations. Table 2 shows the individual demographics of each participant.

ID	Age	Profession	Education level	Computer expertise	Weekly streaming	Group
P1	22	Student	Secondary education	Advanced	5 - 10 hours	1
P2	22	Student	Bachelor's degree	Average	> 20 hours	1
P3	22	Student	Bachelor's degree	Advanced	10 - 15 hours	1
P4	22	Student	Bachelor's degree	Advanced	15 - 20 hours	2
P5	22	Student	Bachelor's degree	Advanced	10 - 15 hours	2
P6	22	Student	Bachelor's degree	Advanced	15 - 20 hours	2
P7	22	Student	Bachelor's degree	Advanced	10 - 15 hours	3
P8	23	Student	Bachelor's degree	Average	10 - 15 hours	3
P9	27	Belgian Defence Officer	Master's degree	Average	10 - 15 hours	4
P10	25	Student	Master's degree	Average	> 20 hours	4
P11	54	Servant	Bachelor's degree	Average	5 - 10 hours	4

Table 2. Demographics of recruited participants for user study 3

5.2.2 Results thematic analysis. To analyse the workshops for this user study a thematic analysis is used. It is the same method used in user study 1 (see section 3). Using this method, following themes were found:

- **Don't hurt my feelings:**
  The application should be designed in such a way that the feelings of the users are never hurt. In particular when a proposed song is rejected, because they might never want to propose a song again or might feel excluded from the group. P3 said: "In a situation where you don't know everyone and your song is rejected, you might not want to suggest other songs".
- **Create a design without obligations:**
  Participants often liked the fact that they can express their musical taste, but don't have an obligation to do so. They don't feel any responsibility towards a task, because if they choose to not use the application music is played anyways. P4 said: "The less responsibility the better. It is nice that you have the ability to propose a song, but you don't have to do this".
  The quality of the recommender system can have an influence on this feeling of responsibility. If bad music is being played, users might feel more obligated to request songs. P1 said: "If the bot plays nice music, then I would just leave it. If it doesn't work well, I might feel more obligated to intervene".
- **Improves group integration:**
  he use of either prototype can have a positive influence on the integration of a member in the group. This theme was expressed by several participants. They stated that when someone requests a song which you really like, the chance of starting a conversation is bigger. All groups highlighted the aspect that the application causes a topic to talk about and it is an icebreaker to start a conversation. P4 mentioned: "When someone plays a nice apres-ski song, I would approach them".
  Also, participants mentioned that they feel more integrated when music similar to your taste is played. P1 said: "If I am new to a group and they play similar music, I might feel a better bond with them". P3 stated: "Songs receiving likes also improves the feeling of belonging to the group".
- **Control over the time table:**
  A functionality to shift the order of or to skip songs is requested by several participants. This flaw is highlighted in the democratic version, since there is no possibility to do so. Group 2

mentioned this: "If a long song is played, you might want the functionality to skip this". P7 stated: "If the group is hyped for a certain song, they have to wait the entire playlist for it to play".

- **The relevance of designing to focus on the social aspect of the event:** A theme that was also found in the first iteration is mentioned several times in this iteration as well. Participants found it important that the main subject is still socializing and not interacting with the application. They would find it annoying if the application took over the event. This concern was mentioned by almost all groups (3 groups).

Overall the democratic approach is favoured by all groups. All members found this the best way to go when attending a small social event. The democratic aspect causes the group feeling of selecting music together. P5 answered: "In the democratic version everybody together is selecting the music and this creates a group feeling". P1 also brought this up: "You can better avoid music which isn't suited as a group in the democratic version".

The democratic version is also better in dividing the responsibility over all members. P4 stated: "With the democratic version the responsibility is divided over all members". And As mentioned in the second theme, the feeling of being obligated to do something should be avoided.

Next, with the democratic approach the participants felt that their opinion would be appreciated more, because it is easy to like or dislike songs. P8 mentioned that this has benefits for people who normally don't have a lot of input.

However, the democratic version does have some disadvantages which were mentioned by the participants. Songs rejected by the group hurt more then in the veto-oriented approach. As shown in the first theme, users might then feel less appreciated and could doubt to propose other songs.

The veto-oriented approach however has some advantages as well. Participants highlighted the benefits of having this version on bigger events. One person should then be selected responsible for the music. The responsible person will have to focus on the application most of the time, but this is always the case in bigger events.

The rejection of a song stings less, because the users know that only one person rejected it.

6 DISCUSSION

This research is the first to design and investigate the effect of different levels of controls for a mobile music group recommender system. Two possible prototypes were introduced and improved throughout this work. Those applications were used to track down possible points of attention when designing mobile group music recommender applications. Three different user studies were executed to answer the three research questions posed in the beginning.

7 RELEVANCE OF SOCIAL DYNAMICS IN GROUP MUSIC RECOMMENDER SYSTEMS

This work is an exploratory study in finding the effects of a music group recommender system on social dynamics. The importance of the social aspect already became clear in the first iteration. Participants mentioned the importance of designing an application which doesn't need much attention. When further exploring the social dynamics in the third iteration, this feeling was confirmed. Socializing is prioritised over interacting with the recommender system. An easy to use application ensures that attendees of an event can focus on socializing with other people. In general, there is a lot more to explore in this area, but this research already revealed some interesting takeaways.

Designing an application that takes the users feelings into account is one of those takeaways. The thematic analysis revealed that it is important to avoid users' feelings getting hurt. In general, you must be careful about users' feelings. It might scar their relation with the group and alter his/her social behaviour in a negative way. This finding might be applicable to group recommender systems in general, however future research needs to confirm this.

A song being rejected, in a music group recommender system, was a possible way of hurting people's feelings in this research. It might cause that the user doesn't want to request other songs ever again. Or he/she might not feel appreciated by the group anymore. Therefore, it is important to design the application in such a way that it wraps this rejection as nice as possible, to minimise any possible negative effect. Further research can dig deeper in what other possible ways people might get hurt and feasible ways to avoid this. It is important to consider these implications when designing, because if the application didn't exist those implications wouldn't exist either.

Using a group recommender system in general can be a stimulus for the integration of members in the group. Participants indicated that requested songs might provoke other users to approach the user who requested it. It is an easy subject to talk about and it facilitates starting a conversation. When a group listens to similar music, people feel a connection with the group, improving the group cohesion. The application causes a social effect which is normally more absent in events without it.

It is also possible that the application has a negative influence on the social dynamics. What happens if the group plays music dissimilar to your music taste? Or might the use of the application cause the creation of separate groups? It is important to take these possible negative effects in regard when designing a music group recommender system. Further research might explore possible ways to promote this social effects inside a social event. Another invitation is to explore the negative effects of a music group recommender system on the social dynamics.

The results of this work, show some of the possible influences of a music group recommender application on the social dynamics. Therefore, this is a strong invitation for other researchers or app developers to consider the social dynamics when evaluating/designing a music group recommender system and maybe a group recommender system in general.

8 FINAL DECISION ON PROTOTYPE

This work examined the differences between a democratic and a veto-oriented version. Advantages and disadvantages were put side by side when deciding which prototype is best. Throughout this research it was hard to decide between one or the other. After two iterations there was no clear preference towards one. Supported by the results of the third iteration, the democratic version was chosen.

This final iteration also gave advantages and disadvantages of each version. In general, it seems that the democratic version does a better job in generating the feeling of a united group. It divides the responsibility to avoid feeling obligated to add music. The voting mechanism also gives the opportunity to share your opinion. However, when a proposed song doesn't receive votes it might hurt the user who requested it. This is an invitation to consider this possible negative effect when designing a democratic music group recommender application.

Participants provided scenarios however where the veto-oriented approach might be more suitable. It might work better in bigger events with more people, for example.

Finally, integrating both version might result in the best possible version. Participants showed this tendency when evaluating the democratic version. They mentioned the lack of functionalities to change the time table of the playlist. The first iteration revealed a similar tendency, where a contradiction was found: Users preferred selecting music as a group, but still wanted one person to take over when things get out of hand.

9 REFLECTION ON METHODOLOGY

Due to the mixed nature of the different iterations, where quantitative and qualitative measurements were used, it was interesting to notice that the results of the second iteration were not sufficient to decide between one of the two prototypes. Comparing the veto-oriented version with the democratic version in terms of usability and cognitive load didn't reveal a clear preference. It is when the comparison between both prototypes in terms of their effect on social dynamics was explored, that participants considered the democratic approach to be better. The third iteration revealed advantages and disadvantages of both prototypes, which made it possible to make a final conclusion. The third iteration revealed some implications on the social dynamics for the veto-oriented approach, for example. These valuable kind of results were lacking in the second iteration. Therefore this is an invitation to encourage mixing quantitative and qualitative research.

10 CONCLUSION

To conclude, this research provided answers to the posed research questions.

The users expectations were found in the first iteration. Because of the contradictory results in terms of having a host or selecting the music as a group, it was difficult to select one preferred version. Users expect an application with a low need of attention. The emphasis of an event is and remains on socializing.

The results of the second iteration didn't reveal big differences between the two versions. This made it again hard to decide on one of the two versions. The different methodologies of voting and having one person with a veto didn't seem to alter the usability or cognitive load much. Both applications scored sufficiently on both aspects.

Finally, the final iteration revealed the possible effects of such an application on the social dynamics. It can improve the group integration, but it can also worsen the group feeling (rejecting songs). This user study emphasises the relevance of considering these effects when designing a music group recommender application.

10.1 Suggestions for further research

This research explores the effect of a music group recommender system on the social dynamics. It is an exploratory study in this topic, but it already showed some promising results. These results indicate the relevance of considering those social dynamics when designing a group music recommender application. Those considerations are possibly also important for other group recommender systems. Currently, there is not much research about this topic yet. Therefore this is an invitation to further research possible effects of a group recommender system on social dynamics. Some specific invitations are made in the part above.

Participants preferred the democratic approach in small social events. For bigger events they thought that the veto-oriented version is more suitable. Another iteration might be needed to confirm or deny whether a veto-oriented application is indeed better for bigger events.

A more specific suggestion for further research is to find out if there are significant differences between the democratic and the veto-oriented version. The second user study revealed that the veto-oriented version significantly scored better in ease-of-use (SUS-questionnaire) and performance(NASA-tlx questionnaire). When the noisiness of data was taken in regard, this significance disappeared. Further research might investigate whether their still is a significance in a bigger population.

Bibliography

[1] Authorization guide, spotify for developers. https://developer.spotify.com/documentation/general/guides/authorization-guide/. Accessed: 2021-04-20.

[2] The misleading effect of noise: The multiple comparisons problem. https://towardsdatascience.com/the-multiple-comparisons-problem-e5573e8b9578. Accessed: 2021-05-1.

[3] Web api reference, spotify for developers. https://developer.spotify.com/documentation/web-api/reference/. Accessed: 2021-04-20.

[4] C. C. Aggarwal. An introduction to recommender systems. In *Recommender systems*, pages 1–28. Springer, 2016.

[5] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso. Tailoring the recommendation of tourist information to heterogeneous user groups. In *Workshop on adaptive hypermedia*, pages 280–295. Springer, 2001.

[6] A. Bangor, P. T. Kortum, and J. T. Miller. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction*, 24(6):574–594, 2008.

[7] S. Bostandjiev, J. O'Donovan, and T. Höllerer. Tasteweights: a visual interactive hybrid recommender system. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 35–42, 2012.

[8] V. Braun and V. Clarke. Thematic analysis. 2012.

[9] J. Brooke. Sus: a "quick and dirty" usability scale. *Usability evaluation in industry*, 189, 1996.

[10] R. Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175–186, 2000.

[11] A. V. Carron and L. R. Brawley. Cohesion: Conceptual and measurement issues. *Small Group Research*, 43(6):726–743, 2012.

[12] A. V. Carron, W. N. Widmeyer, and L. R. Brawley. The development of an instrument to assess cohesion in sport teams: The group environment questionnaire. *Journal of Sport and Exercise psychology*, 7(3):244–266, 1985.

[13] M. Casey-Campbell and M. L. Martens. Sticking it all together: A critical assessment of the group cohesion–performance literature. *International Journal of Management Reviews*, 11(2):223–246, 2009.

[14] P. Chandler and J. Sweller. Cognitive load theory and the format of instruction. *Cognition and instruction*, 8(4):293–332, 1991.

[15] D. L. Chao, J. Balthrop, and S. Forrest. Adaptive radio: achieving consensus using negative preferences. In *Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, pages 120–123, 2005.

[16] A. Crossen, J. Budzik, and K. J. Hammond. Flytrap: intelligent group music recommendation. In *Proceedings of the 7th international conference on Intelligent user interfaces*, pages 184–185, 2002.

[17] A. Felfernig, L. Boratto, M. Stettinger, and M. Tkalčič. *Group recommender systems: An introduction.* Springer, 2018.

[18] D. R. Forsyth. *Group dynamics.* Cengage Learning, 2018.

[19] F. M. Harper, F. Xu, H. Kaur, K. Condiff, S. Chang, and L. Terveen. Putting users in control of their recommendations. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 3–10, 2015.

[20] S. G. Hart and L. E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In *Advances in psychology*, volume 52, pages 139–183. Elsevier, 1988.

[21] C. He, D. Parra, and K. Verbert. Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities. *Expert Systems with Applications*, 56:9–27, 2016.

[22] N. N. Htun, E. Lecluse, and K. Verbert. Perception of fairness in group music recommender systems. In *26th International Conference on Intelligent User Interfaces*, pages 302–306, 2021.

[23] Y. Jin, B. Cardoso, and K. Verbert. How do different levels of user control affect cognitive load and acceptance of recommendations? In *Jin, Y., Cardoso, B. and Verbert, K., 2017, August. How do different levels of user control affect cognitive load and acceptance of recommendations?. In Proceedings of the 4th Joint Workshop on Interfaces and Human Decision Making for Recommender Systems co-located with ACM Conference on Recommender Systems (RecSys 2017)*, volume 1884, pages 35–42. CEUR Workshop Proceedings, 2017.

[24] Y. Jin, N. Tintarev, and K. Verbert. Effects of personal characteristics on music recommender systems with different levels of controllability. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pages 13–21, 2018.

[25] G. Linden, B. Smith, and J. York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[26] J. Masthoff. Group recommender systems: Combining individual models. In *Recommender systems handbook*, pages 677–702. Springer, 2011.

[27] J. F. McCarthy and T. D. Anagnost. Musicfx: an arbiter of group preferences for computer supported collaborative workouts. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 363–372, 1998.

[28] M. Millecamp, N. N. Htun, C. Conati, and K. Verbert. To explain or not to explain: the effects of personal characteristics when explaining music recommendations. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, pages 397–407, 2019.

[29] M. Millecamp, N. N. Htun, Y. Jin, and K. Verbert. Controlling spotify recommendations: effects of personal characteristics on music recommender user interfaces. In *Proceedings of the 26th Conference on user modeling, adaptation and personalization*, pages 101–109, 2018.

[30] M. O'connor, D. Cosley, J. A. Konstan, and J. Riedl. Polylens: A recommender system for groups of users. In *ECSCW 2001*, pages 199–218. Springer, 2001.

[31] K. O'Hara, M. Lipson, M. Jansen, A. Unger, H. Jeffries, and P. Macer. Jukola: democratic music choice in a public space. In *Proceedings of the 5th conference on Designing interactive systems: processes, practices, methods, and techniques*, pages 145–154, 2004.

[32] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*, pages 325–341. Springer, 2007.

[33] D. M. Pennock, E. J. Horvitz, S. Lawrence, and C. L. Giles. Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach. *arXiv preprint arXiv:1301.3885*, 2013.

[34] G. Popescu. Designing a voting mechanism in the groupfun music recommender system. In *International Conference on Human-Computer Interaction*, pages 383–386. Springer, 2013.

[35] G. Popescu and P. Pu. What's the best music you have? designing music recommendation for group enjoyment in groupfun. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pages 1673–1678. 2012.

[36] P. Pu, L. Chen, and R. Hu. A user-centric evaluation framework for recommender systems. In *Proceedings of the fifth ACM conference on Recommender systems*, pages 157–164, 2011.

[37] F. Ricci, L. Rokach, and B. Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

[38] F. Ricci, L. Rokach, and B. Shapira. Recommender systems: introduction and challenges. In *Recommender systems handbook*, pages 1–34. Springer, 2015.

[39] E. Salas, R. Grossman, A. M. Hughes, and C. W. Coultas. Measuring team cohesion: Observations from the science. *Human factors*, 57(3):365–374, 2015.

[40] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, 2001.

[41] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen. Collaborative filtering recommender systems. In *The adaptive web*, pages 291–324. Springer, 2007.

[42] J. B. Schafer, J. Konstan, and J. Riedl. Recommender systems in e-commerce. In *Proceedings of the 1st ACM conference on Electronic commerce*, pages 158–166, 1999.

[43] R. Sinha and K. Swearingen. The role of transparency in recommender systems. In *CHI'02 extended abstracts on Human factors in computing systems*, pages 830–831, 2002.

[44] D. Sprague, F. Wu, and M. Tory. Music selection using the partyvote democratic jukebox. In *Proceedings of the working conference on Advanced visual interfaces*, pages 433–436, 2008.

[45] M. Stettinger, G. Ninaus, M. Jeran, F. Reinfrank, and S. Reiterer. We-decide: A decision support environment for groups of users. In *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, pages 382–391. Springer, 2013.

[46] N. Tintarev and J. Masthoff. A survey of explanations in recommender systems. In *2007 IEEE 23rd international conference on data engineering workshop*, pages 801–810. IEEE, 2007.

[47] C.-H. Tsai and P. Brusilovsky. The effects of controllability and explainability in a social recommender system. *User Modeling and User-Adapted Interaction*, pages 1–37, 2020.

[48] A. E. Van Vianen and C. K. De Dreu. Personality in teams: Its relationship to social cohesion, task cohesion, and team performance. *European Journal of Work and Organizational Psychology*, 10(2):97–120, 2001.

[49] Z. Yu, X. Zhou, Y. Hao, and J. Gu. Tv program recommendation for multiple viewers based on user profile merging. *User modeling and user-adapted interaction*, 16(1):63–82, 2006.

AFDELING
Straat nr bus 0000
3000 LEUVEN, BELGIË
tel. + 32 16 00 00 00
fax + 32 16 00 00 00
www.kuleuven.be