NTNU
Norwegian University of
Science and Technology

Design and Evaluation of an Personalized Mobile Tourist Application

Magnar Wium

Master of Science in Computer Science
Submission date: June 2010
Supervisor: John Krogstie, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

Mobile applications supporting tourists with travel information can make use of information about the user's location, time and personal preferences to provide personalized recommendations. This could be a solution to the problem of displaying information and navigating on small mobile devices, as it allow tourists to receive information that fit very well with their current situation and needs.However, filtering of information introduce new challenges in terms of facilitating user control and transparency. This project shall through the design and evaluation of a personalized mobile tourism application answer the following

How should personalized mobile tourist application be designed to,

1)Allow tourists to easily find information on- and physically locate points of interests?
2)Facilitate transparency and user control in the personalization process?
3)Gather information on the user's preferences in way that is non-obtrusive, accurate and requires little effort from user?

Assignment given: 15. January 2010
Supervisor: John Krogstie, IDI

Abstract

Mobile applications supporting tourists with travel information can make use of information about the user's location, time and personal preferences to provide personalized recommendations. This could be a solution to the problem of displaying information and navigating on small mobile devices, as it allow tourists to receive information that fit very well with their current situation and needs. However, filtering of information introduce new challenges in terms of facilitating user control and transparency. In this thesis we have developed and evaluated a personalized mobile tourism applications based on collaborative filtering that have tried to meet these challenges. The design of this application is based on experience from similar projects and research on interaction design in recommender systems. The user evaluation of our system suggests that our approach is feasible, but more research must be done to predict the acceptance of this application among tourists.

Preface

This report is a documentation of the project work performed as part of the Master's thesis in Computer Science spring 2010 by Magnar Wium.

I would like to thank my supervisor John Krogstie for his firm guidance and useful comments along the way.

Trondheim, June 11nd, 2010

Contents

Chapter 1

Introduction

The tourism industry is regarded as one of the biggest sectors in the world, generating an estimated eleven percent of the global gross domestic product[21]. Supporting the vast number of tourists that every day explores new territory with information services evidently becomes a vital task[28]. Today most tourists relay on static information such as guide books or other forms of printed material to locate points of interest. These sources of travel information has many problems as for example to much or outdated information[42]. As advanced mobile technology has become standard pocket accessory for most of the world citizens many has the option to substitute the guide book with online mobile information services. There exists however a challenges when it comes to enabling access to information services on hand held devices. The small screen and limited interaction options of mobile phones make it a difficult platform for navigating and searching trough large amounts of data[34]. Researchers of information system has found that context awareness might solve the problem of information overload and contribute to user acceptance of mobile information systems[34].
One very interesting class of context aware systems are so called personalized applications[34]. In the tourist domain a personalized mobile application will use knowledge of the tourist personal preferences and other context properties as the time and location of the user to find points of interest with minimal effort on the user's part.

Most of the personalized applications proposed in literature make us of a key piece of technology called recommender systems[21]. These systems were originally developed to solve the information overload problem in e-commerce, and have contributed greatly to success of companies as Amazon and Ebay[31].However, despite that online ventures have had enormous success with personalization few commercial system delivering tourist information on a mobile platform exists today[21].
The lessons learned from a decade of research on personalized mobile tourist

application points towards the importance of not making personalization process act as an impenetrable black box for the user[7]. When the system makes wrong deductions about the user's situation or needs, this approach will confine the user and make searching from information a frustrating endeavour. Research on the user experience of recommender systems is coherent with this, and shows that transparency is a key property for facilitating efficient usage, trust and general satisfaction in such systems[36].
Transparency in recommender system means understand why a certain recommendation was given, and how the users interaction with the system affects the recommendations[16].Incorporating transparency into the recommendation process has not received much attention in the design of earlier personalized mobile tourist prototype systems[30].
Traditional recommender system relay on feedback from the user such as ratings and analysis of browsing patterns to generate a profile of preferences. The general rule of thumb is that the more direct feedback the user provides the better the recommendations become, and there is there important to quickly collect a base set of preferences[26].
A common approach among the personalized mobile tourist applications discussed in literature is to require the user to specify and manage a profile of interest. This can become tedious, especially without apparent connection between the interest profile and received recommendations[40]. Research on traditional recommender system has on the other hand showed that the process of getting user feedback can be made a fun experience, allowing the system to relay more on accurate direct feedback[39].
The goal of this thesis is to develop and evaluate personalized mobile tourist application that incorporates the lessons learned from research on recommender systems into its design. We will take a pragmatic approach, focusing on developing a system that is easy and pleasant to use. More specifically our research will be guided the following research questions:
How should a personalized mobile tourism application be designed in order to:

1. Allow tourists to easily find information on- and physically locate points of interests?

2. Facilitates transparency and user control in the personalization process?

3. Gather information on the user's preferences in way that is non-obtrusive, accurate and requires little effort from user?

Part I

Background

Chapter 2

Location based services

In this chapter we will introduce location based services. We will also define and briefly explore context awareness. This chapter will serve as the basis of chapter 4 where we go into a specific class of location based services.

2.1 What is location based services?

According to Virrantaus et al. location based services are[41] :

> *..information services accessible with mobile devices thorough the mobile network and utilizing the ability to make use of the location of the mobile device*

Another, more specific definition is given by Küpper[22] :

> *Location-based Services (LBSs) are mobile services for providing information that has been created, compiled, selected or filtered under consideration of the users' current locations or those of other persons or mobile devices. Typical examples are restaurant finders, buddy trackers, navigation services or applications in the areas of mobile marketing and mobile gaming. The attractiveness of LBSs is due to the fact that users are not required to enter location information manually but are automatically pinpointed and tracked.*

Mobile users can be said to be a class of users that are able to make use of information services on the move[8]. The goal of mobile users is often to locate something or someone. Roaming in unfamiliar territory many will also find themselves wanting to get an overview of the new environment by

learning about what objects or people it consists of. To reach these goals, users will have to find answers to question like[38]:

1. Where am I?
2. Where can I find what I need?
3. Where is the most relevant place for what I currently need?

In light of the previous stated definitions, LBS can be said to be applications that exploit their knowledge of the mobile device to help answer these question for the mobile user. Location based services can be classified as push or pull services depending on whether they deliver information directly requested from the user(pull), or deliver information which are either not or indirectly requested from the user(push)[38].

2.2 Context awareness

LBS an advantage compared to conventional application in the sense that they can adapt their content and presentation accordingly to the context of the situation they are currently being used in. What exactly is meant by context and how to classify types of context has been been the subject for much discussion in information system research. A general definitions given by Dey et al.[4]

> *Context is any information that can be used to characterize the situation of an entity. An entity is person, place or object that is considered relevant to the interaction between a user and application, including the user and application themselves*

Going from this broad notion of context it can be useful to see a a specific classification. The following was proposed by Nivala et al. in relation to a map based LSB designed for supporting users with context aware information on hiking trails[38]:

- The user and purpose of use: Demographic information, personal preferences etc .
- Location: The users geographical (or relative) position.
- Time : This can be the time of day, or more longer intervals such as morning, afternoon or evening, day of the week, month, season of the year etc.

- Navigation history: The places the user has visited

Context awareness relates to how the applications adapt to information such as stated above . For a definition we turn back to Dey et al.[4]:

> *A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task*

The way context aware systems work to provide the user with relevant information, the adapting strategy, can have major implications for the for the user experience. We will come back to this in relation to personalized mobile tourism applications in chapter 4.

Chapter 3

Recommender systems

In this chapter we will try to answers the question

- What is recommender systems?
- and how does recommender system work?

In addition we will discuss some research on interaction design with such systems.

3.1 What is recommender systems?

Recommender systems are tools for decision support in the information jungle. The first systems gained popularity in the late 90s as enchantments to web based e-commerce. Users of online platforms for shopping, such as Ebay.com, was in early days of e-commerce faced with a vast number of items for sale.The user was however only interested in a very small subset of for purchase. The job the of recommender systems is to support the user in the search process by providing personalized recommendations or meta information such as ratings and user reviews to make Internet based shopping easier. The benefits for the user is more efficient shopping as well increased confidence in that they are making the right decision. For the companies integrating recommender system into their solutions the primary benefit is increased revenue, as they can present a small set of items to customers that are likely to find these particular items interesting. Further, companies can benefit from that the users get attached to differentiating attributes recommender system represents. Many implementation use long-term interaction to improve recommendations, thus making it less favorable for the user to switch to a competitor[31].
At the time of writing the use of recommender system has long gone beyond

online shopping. The technology is today used in domains spanning from health informatics to tourism[27, 21].

3.2 How does recommender systems work?

Recommender system comes in many flavors, both in the internal mechanisms for generating the recommendations and aspects of the user interface[31]. However, the core of any implementation is some form of information filtering based on a model of the users preferences. The most common filtering techniques used today is[6]:

- Content based filtering
- Collaborative filtering

Systems combining these two filtering techniques, so called hybrid-systems, has in recent years become popular as way to levering the problematic aspects two respectively methods[6]. We will however not go into this approach as it is out side the scope of this thesis.

3.2.1 Content based filtering

In a content-based system, the objects of interest are denoted by their associated features. For example, in a recommender system providing recommendations of restaurants the associated features of restaurant can be type off cuisine served, price level or whether or not the particular place have out door service. To find a restaurant recommendation for the user, the content based filtering algorithm can conceptually be said to work like this: The system has collected a set of features parried with the probability of liking for each user. To find a restaurant the user probably will like, the system do a comparison between the set and the various restaurants in the system. The comparison can be some sort of heuristics or simply a similarity calculation as the cosine measure. The system then returns the restaurants that represents the best fit.
Content-based filtering can be very useful when much descriptive data is available on the items. This can be codified and represented internally as an ontology of features, giving and rich description of the domain applicable for similarity calculation against the users preferences. Nevertheless, to be useful a content based scheme must have this ontology and means to classify items. Because of this, content based filtering can be problematic when the amount of information available on each item is sparse or difficult to interpret with computer algorithms.

Further, content based recommendations can only be based on the explicit information available on a certain item. If we take a movie recommender system as an example, content based approaches would use information as actors's names, plot summaries etc. as features to find recommendations for a user. The movie itself completely opaque to the system, and it can not go beyond the user probability for liking for instance; horror movies from the 50s when it's searching for good recommendations. This scheme makes it difficult to provide the user with refreshing new items[6]. The other main filtering algorithm, collaborative filtering, does not have this problem.

3.2.2 Collaborative filtering

Collobrative filtering is by far the most widespread recommender technique because of it's power and simplicity[17]. The greatest strength of collaborative techniques is that they are completely independent of any machine-readable representation of the objects being recommended, and work well for complex objects such as music and movies where variations in taste are responsible for much of the variation in preferences[17]. Collaborative filtering is often referred to as social filtering as the technique recognize commonalities between users on the basis of their ratings, and generate new recommendations based on inter-user comparisons[6].
This process can be said to mimic the social process of recommending items among friends, as the system gives recommendation to users based on what people that are similar to the user in terms of ratings has liked in the past. An important distinction between movie recommendations around the coffee machine and in commercial collaborative system is that number of "friends" in such system can reach into the millions.
This brings us to one of the disadvantages of the collaborative approach. While content based systems are depended on a rich ontology, collaborative systems are depended on many active user to ensure a proper distribution of ratings across all the items in the system. A obvious challenge is to be able present recommendations on new items, that not yet has been given many ratings from users. This problem is known as the sparsity problem.
Further, both collaborative and content based filtering(as well as approaches combining the two) relies on knowledge of the user in order to do deductions in feature space or calculate correlation with other users. We will in the next section touch how this knowledge may be acquired.

3.3 Implicit versus explicit collection of user preferences

A common feature for all the recommender techniques is, as we have describe, the need for some sort of representation of the users preferences in order to be able to provide recommendations[29]. This representation often referred to as the user model(or profile)[21]. The model can been seen as the systems perception of the user. Conceptually, features parried with probability make up the model in content based system and a set of ratings make up the model in collaborative systems. The complexities of this representation vary from simple vector based models to multi dimensional matrices. The user models can be individual, meaning that each user in the system can be mapped to a distinct profile, or several user can partly share models. This scheme is called stereo typing / group modeling and has in later years been proven a valuable approach to quickly provide good recommendations[6].
However, regardless of the complexities involved or which filtering algorithm chosen, the information to build the model or classify the user in the correct stereotype must come from somewhere. This somewhere is preferable the user. There are two ways of collecting this information. The system can explicitly require the user to input some information about his or her preferences, or it can analysis how the user interact with the system and use this information to implicitly come up with a model[26].
The seemingly favorable technique of using implicit collection of information to the model, since it require the least effort of the user, is not necessarily the best approach as the accuracy of implicit analysis preform very poorly compared to approaches that relies more on on direct feedback[26]. Most commercial system, that not have access to purchases history, use a combination of implicit and explicit collection preference. Further, research on user satisfaction in recommender system point to that the user gladly provides feedback as long as the process is engaging and the benefits are clear[36].

3.4 User interaction with recommender systems

When it comes to what makes a successful recommender system most research has been devoted to the efficiency and accuracy of the underlaying algorithms, but some studies address the user experience as a factor of the overall success of the system[40, 36, 39]. After studying user satisfaction in commercial recommender systems, all running a collaborative filter engine behind the scenes, Swearingen and Sinha found that users expressed a high level of overall satisfaction. However, users did not like all the systems equally for reasons that was directly related to the various interfaces. In light of this they proposed a set of design suggestions.

Firstly, systems that collocated some basic information about the item being rated on the same page greatly increased the satiation in comparison to those system that only included title. User had difficulties making decisions without additional information. A guideline is therefore to always provide easy access to information that better can asses the quality of the recommendation.
Secondly, prediction of user liking as part of the recommendation can simplify the users decision making and provide a intuitive ordering on the screen, but prediction must be used with casuion. For example, a user might lose confidence in a system that predicted a high degree of liking for an item he or she hates. A system would need to have a very high degree of accuracy for users to benefit from this feature.
Thirdly, it's vital to preventing dead end situations were the user can't reach new unexplored sets of recommendations. A well designed system should include easy accessible means to escape these situations.
Fourthly, users like the reasoning of RS to be at least somewhat transparent. Transparency in a system can be said to be to which degree the user is exposed to the inner workings of the particular system. In recommender system this means understand why a certain recommendation was given, and how the user interaction with the system affects the recommendations and the systems perception of the user. A study of users satisfaction in music recommender systems found that the liking was significantly higher for systems that had a transparent interface than those that did not.[36].
When the system collects and interprets information in the background, as often is case with recommender systems, it becomes all the more important to make the reasoning available to the user. Following transparency , a second step is to allow the user to correct reasoning, or make the system scrutable[40]. As recommender systems never will relay on complete information about the user, not allowing allowing the user to easily override or bypass erroneous assumptions are likely to create frustration.

3.5 Explaining recommendations

3.5.1 What is explanations good for?

From the guidelines presented in the previous section we were able to derive some specific requirements on navigational features of user friendly recommender systems

- Easy access to overview information
- No dead end situations

However, transparency and scrutablilty, all though seemingly important, still need some clarifying in terms they can be incorporated into a system. To achieve transparency explanations as part of the recommendations can be one possible solution[40]. Explanations has its roots in HCI research on expert systems[40]. In recommender systems explanations often take the form as short textual or graphical representations of how and why the recommendation was given to the specific user. Accordingly to Herlocker Et Al, using this construct can have the following benefits[16]:

1. **Justification**, as the user understand the reasoning behind the recommendation it's easier to decide how much confidence to place in it.
2. **User involvement**, as the user gets more information the user can apply own knowledge and inference skills in the decision process.
3. **Acceptance**, as the weakness and strength of the system is fully exposed
4. **Education**, as the users though the explanations can learn the system's strengths and weaknesses

However, poorly designed explanations can decrease the user's satisfaction with a system. System designers must therefore carefully choose their explanation strategy in relation to what the system is trying to achieve and the knowledge and needs, and situation of the user[40].

3.5.2 Explanations of collaborative recommendations

They way explanations are design vary depending on the filtering technique used. For collaborative filtering the following model can be used to explain the workings of the algorithm [16]:

1. User enters a profile of ratings
2. The system locates people with similar profiles (neighbors)
3. Neighbors' ratings are combined to form recommendations

Looking at the first step, the user might engage more actively in rating objects if the quality and range of the profile can be inspected[16].
Looking at the next step, a great feature of collaborative systems is that they produce information that can explain to the user how many, and to what degree of closeness the users making up the basis for the recommendations has with the specific user. This can allow the user asses this information as

a metric in order to judge the quality of a recommendation[16].
The final step can be explained by showing information about the people used for finding the recommendation.

Taken together, this information can greatly increasing the users trust in the system and somewhat bridge the sparsity problem we touched on earlier in this chapter[16].

3.6 MobyMapRek - A mobile Tourism applications

We want to end this chapter presenting research on interaction design with a mobile recommender system. MobyRek was a mobile recommender system for restaurants that provided the user with a location aware list of recommended restaurants. The usability evaluation of this system revealed major issues with the interface related to using the list on the mobile phone. To overcome these difficulties another research project was initiated with the goal to wrap the core recommender system with a more user friendly map based interface. The interface was develop to meet the following requirements:

- Recommendations should be shown in a interactive map based interface
- The link between feedback and recommendations should be visible for the user

The result was MobyMapRec. The application used a interactive map to show the recommendations and the position of the user. The system represented each restaurant as markers with varying size and color in the map. The color and size was used to visualize the prediction of how well the recommendation is believed to fit the user. Large, green markers represents great fit. Small red markers naturally represents the other end of the scale. When user inputed ratings, the system would indicate how this change in the user model affected the recommendations in the map by changing their size and color.
The transparent and novel design of MobyMapRec outperformed the initial design in terms of usability metrics as efficiency and user satisfaction. However, the user evaluation of MobyMapRek also revealed that increased user control in terms of means to undo changes in the user model should be included. Further, is was clear that the promising interface of MobyMapRec would greatly benefit form being able to provide the user with a broader set of tourist services such as recommended travel itineraries[5]. We will in the next chapter look at various approaches to creating mobile application that aim at supporting the entire information needs of tourists.

Chapter 4

Personalized mobile tourism applications

We have seen how recommender systems have been successfully applied to give users personalized recommendations. In this chapter we will introduce a number of mobile tourism applications making use of recommender systems to support personalization. We will also look at the related challenges and limitations of these applications from the user perspective. In light of the experiences and limitations we will also briefly introduce our system through some usage scenarios.

4.1 What is personalized mobile tourism applications?

The tourism industry is regarded as one of the biggest sectors in world generating an estimated eleven percent of the global gross domestic product[21]. In light of this, researching how information technology can support tourists evidently becomes important.

As we touched on in chapter 2, location based services on a mobile device provide it's user with information right on the spot. In the tourist domain ease of finding the right information quickly is especially important as the user of such LBS are finding themselves in an unfamiliar environment often with limited time to make decisions. Further, the tourist user is not easily categorized with a specific set of information requirements as this group span from backpackers and business men to package tourists all with different goals and engaging in different activities when visiting a particular city.
Going back to the definition of context given earlier, LBS delivering tourist information should be sensitive not only to the time, location and activity

of the user but also be sensitive to the dimension of the user identity related to personal preferences[42]. Applications that adapts it's behaviour and information to the user's personal preferences can be said to be a personalized application[34, 30]. To support personalization, the design of mobile tourism applications must include some form of information filtering, as well as a way finding the user's preferences. We have seen how recommender system can be used as a tool for this, and personalized tourist applications are characterized by applying this technology to improve the user experience[34, 30].

4.2 Previous work

4.2.1 GUIDE

An early prototype system providing personalized tourist services on a hand held platform is the GUIDE-system developed for visitors of the city of Lancaster[9]. The requirements of the system were based on an in-depth study of tourist needs. This initial research involved both experts on tourism and tourists themselves. The key requirements discovered were:

- Flexibility: Provide sufficient flexibility to enable visitors to explore and learn about the city in their own way and in their own pace
- Context sensitive information: Information presented to the users should be tailored to their personal and environmental context
- Support For Dynamic Information

The resulting implementation was a system providing the user, through a browser based interface, with information on tourist attractions like museums and historical buildings, filtered to meet the user's current information need. GUIDE also featured a map based interface displaying both recommendations and the user's position in the map. In addition, GUDIE allowed users to compose and request recommended tours.
The GUIDE architecture is based on the client - server model. Communication in this model was achieved using cell based WLAN to enable the PDA client to sense geographical position and communicate with the server. The server stored all information about the attractions and applied content based filtering before sending the resulting subset to the PDA. Filtering of information is in the GUDIE architecture based on a model of context distinguishing between personal and environmental context.
Personal context include current location of the user and previously visited places[34]. In addition, personal context also include personal preferences represented as individual user models of features parried with probability

of liking[21]. The environmental context include information about attractions such as features for the content based filtering, as well as links between nearby attractions and opening hours for each attraction.
The user models are created and maintained in a semi automated fashion, requiring the user to specify preferences but also learning through the user's interaction with the system interface. The user evaluation of GUIDE found a high level of acceptance across a wide range of users. However, early versions of guide revealed challenges in relation to its customization strategy in relation to context information. We will come back to this later in this chapter.

4.2.2 CRUMPET

Another interesting prototype system is the CRUMPET system[33]. The goal of this project was creation of a user-friendly mobile services personalized for tourism. The users of CRUMPET can request information and recommendations about tourist attractions, restaurants and tours and get updated information from external content providers tailored to their current situation. CRUMEPET does not support the tour element of GUIDE, but extends their tourist service with the innovative feature of personalized pro-active recommendations . CRUMPET also features a map based interactive interface for user interaction.
CRUMPET is created using a three level architecture with a client tier, middle tier and data access tier. The data access tier integrates several external sources of tourism information, and hides the heterogeneous nature of these services through a interfaces accessible from the middle tier. The middle tier applies geo-coding and adaption of the information from the external providers. The adaption process involves content-based filtering using individual user models. These models are maintained in the same fashion as in GUDIE: The user must specify preferences, but the systems adjust the model thorough analysis of browsing data. The client tier is created with software agent monitoring the user's movement by pooling GPS-information, and automatically contacting the middle tier for pro active recommendations.

The user evaluation of CRUMPET showed that a range of different users would consider paying for the service. The map based interaction model proved to be a useful and intuitive abstraction. Nevertheless, the evaluation of CRUMPET also revealed the need for more functionality especially related additional tools for of content retrieval. Users of CRUMPET could only retrieve information through predefined recommendation request, and a number of users found this insufficient. A straight forward search as proposed as possible solution to this[33].

4.2.3 TIP

Tourist Information Provider(TIP) is a research project with the goal to develop mobile infrastructure for cooperating information services. The TIP core system implemented using an event-based architecture. The system delivers information about sights based on the user's context in terms of location, personal preferences and the users travel history. Cooperating services was identified by the researchers as a key requirement for system providing context sensitive information. Cooperating services allow the user to access and display sight information in different forms e.g through a map display or as list of recommendations, depending on the requirements of the user and limitations of the client device[19].
The limitations and challenges of TIP seems first and foremost to be a result of the state of mobile computing anno 2006: The third party library used for visulation of sights in the map service , supported only basic rendering, and did not allow the designers to make distinctions between categories of sights. Further, the focuses of the researchers was invariable tied to the architectural aspects of personalized mobile tourism applications. The user experience of TIP and the quality of the user interface is therefore difficult to assess.

4.2.4 A Personalized Location-Based Recommendation Services for Tour Planning in Mobile Tourism Applications

In their recently published research on personalized mobile tourism applications Chang et al. argue that means to do travel planning are important for the acceptance of personalized mobile tourism application, as the activity is closely tied to the overall goals of tourists. They go on to defining the process of travel planning as searching, selecting, grouping and sequencing destination related products and services including attractions, accommodations, restaurants, and activities.
Recommended travel plans(or travel itineraries) as well as means to compose personal plans has been part of earlier prototypes system such as the previously mentioned GUIDE-system, however Chang et al. focus on how the process of travel planning can benefit from context awareness.
In relation to this the researchers propose a frame work for a personalized travel-service in a LBS architecture. The framework allow users to request complete tour plans where the content is adopted based on the location, time and preferences of the user. Because of the recommender system used in this framework, users are required to input a set of constrains before any plan can be retrieved. The framework has been implemented in a prototype system that has yet to be evaluated for usability [42]

4.3 Challenges and limitations

4.3.1 Lacking transparency and user control

Our short review of personalized mobile tourism applications show how knowledge of the context can be used in a information system to help a tourist explore an unfamiliar setting without the need for effort in terms of searching. What you get as a user depends on who you are, where you are located and what you are doing. At first glance this seems like the magical recipe for solving the conflict between the demand for information posed by a tourist and the limitations of the hand held device most tourist carry with them. However, using context as filtering properties can introduce looming challenges. The usability evaluation of both GUIDE and CRUMPET revealed some of these. The initial GUIDE-implementation filtered closed attractions, based on assumptions that no one would be interested in visiting a closed sight. This was a wrong assumption. The system became a source of frustration for its users as many found pleasure in studying the architectural facade of famous sights. After closing time, the users could no longer find these sights on the map or read information about them. Based on these findings a set of guidelines for using context was proposed[7]:

1. Do not over-determine the users interaction
2. Need for predictability /consistency of the system
3. The intelligence of the agent responsible for adapting to context needs to be sufficiently flexible to enable users to override the adaption strategy

We have previously touched on that personalized mobile application include user preferences as a element in their notion of context, in order to filter information. The systems we have accounted for, as well the majority of similar system proposed up to the time of writing[21], relies primarily on information about the users found from analysis of implicit interaction data. The data comes from the user's interaction with graphical user interface. All systems keep the resulting user model partly accessible for the user to monitor and adjust, but there is not provided any details on whether the user is capable of using this option for its indented purpose. The reason for hiding the details of the personalization process, as well as relining on implicit data to build and maintain the user model has the evident advantage of minimizing the required user interaction with the cumbersome mobile interface. Going back to the guide lines presented earlier in this section, we believe that a timely question is whether this usage of personal preferences can be said to be coherent with these guidelines.

Filtering on preferences without explaining to the user how this done or how the user can influence the process can cause wrong deductions about the user's situation. This will result in a system that appear both over-deterministic and unpredictable[36]. We have seen how explanations can create transparency in the personalization process to meet with this challenge. Explanation has however not been included or explored in relation to personalized mobile tourism applications[21]. Neither has ways of allowing the user to influence the adaption strategy enforced by the application.

4.3.2 Making use of the user's movement for finding preferences

Fast and accurate acquisition of user's preferences is still an open research problem[30]. The personalized mobile application we have described does not have flexible user interfaces for supporting this task, as they limit their eliciting process mostly to analysis of browsing data. The user is given limited means verify or contribute to this process[21].
Nevertheless, personalized mobile applications have the interesting option to collect and analysis the user's movement in the real world. In light of our previous discussion related to the importance of user control, the reasons for not trying to give meaning to this information seem quite clear. The complexity and potential ambiguity of this information would difficult to handle properly. There is no way for the system to truly know if a user visited a sight because he liked it or for completely different reasons[9]. Because of this, designers personalized mobile tourist application has rejected this as an option from finding user preferences. However exploration of user model deduction in relation to physical movement could greatly benefit mobile recommenders as it would decrease the user interaction with troublesome mobile user interface.

4.4 Introducing our personalized tourism application

We ended the previous section with saying that analysis of the user movement in the real world has a potential for simplifying user interaction and facilitate fast and accurate learning of user preferences in a mobile recommender system. This can perhaps seem rather contradictive with the emphasis we put on transparency and user control as important properties of a personalized mobile tourism application. We argue however that these are not mutely exclusive system properties, but rather can work well together. Imagine an application that managed to make the personalization process transparent to user by constructs in the user interface. The user can understand how

and why recommendations are given. The user also understands how feedback influences the user model. We believe that such an application would generate incentive for the user to provide direct feedback, as the benefits of this would be clear.
Imagine further that the application can monitor the user's movement, so that it knows when the user has visited a tourist service that is relevant for adjusting the user model of preferences. Let us for example say that the user has visited a cathedral.
Instead of doing any probability calculation or other advanced analysis for finding out whether the user liked cathedral, the application simply asks the user for his opinion about the place. We believe that if this request is done in non-obtrusive way and the opinion is very easily expressed through the user interface, the user will not perceive this form of interaction with system as a burden, but rather gladly provide the underlying recommender system with a stream of reliable feedback as long as the benefits are clear.

4.4.1 Goals and requirements

The goal of this project is to design this system (From now we will refer to this application as "Mobile Tourist Service Recommender"(MTSR)).

We also want MTSR to incorporate the best practice and experiences drawn form the research we have described in the last two chapters. More specifically, we will try meet the following high level goals and requirements:

Allow tourists to easily find information on- and physically locate points of interests

- Requirement 1 (R1): The system must have pre-defined requests for typical tourist services as part of the user interface
- Requirement 2 (R2): The system must allow the user to search freely after information
- Requirement 3 (R3): The system must provide the user with the distance and route to each service from the users current location
- Requirement 4 (R4): The system must allow the user to request and managed complete travel itineraries
- Requirement 5 (R5): The system must provide the user with automatic/proactive recommendations

Facilitates transparency and user control in the personalization process

- Requirement 6 (R6): The system must provide explanations of the personalization process to the user

- Requirement 7 (R7): The system must allow the user to override the adaption strategy taken by the system

Gather information on the user's preferences in way that is non-obtrusive, accurate and requires little effort from user

- Requirement 8 (R8): The user must be able to control the personal preferences used to create recommendations
- Requirement 9 (R9): The system must allow the user to express his opinion about the services presented
- Requirement 10 (R10): The system must automatically provide the user with an overview of places the user has visited

4.4.2 Usage scenarios

To further illustrate how we believe MTSR can used by tourists, we will now present some usage scenarios. These scenarios form the basis for the user evaluation of MTSR we will present in chapter 8.

Scenario 1 - Receiving and evaluating information

Tony is a tourist walking around in an unfamiliar city. He is running MTSR on his mobile phone. Suddenly his phone gives a light buzz. When looking at the screen of the phone, Tony can see that he has received some recommendations for things to do close to his current position (R5). One of the places is a museum of modern art.
Tony sees from the description of the museum that it might be interesting. He then consults the explanation of why the recommendation was sent to him, and he learns that the museum has received high average rating from people that also have liked his favourite museums in other cities (R6). This was all the persuading he needed, and decides to go over to the museum. Tony can easily find the way by following the route indicated on the mobile screen (R3).
It turned out that the museum was perfect for Tony. He therefore wants to use MTSR to see if the people that recommended the museum have other places to offer as well. This leads him to information about an old church, located only five hundred meters away (R3). As he about to leave for the church, MTSR asks him if he would like to provide the system with his opinion about the museum he has just visited (R8,R9,R10). Tony gladly does this, as he enjoyed it very much.

Scenario 2 - Actively searching for information

Meadow is a backpacker visiting an unfamiliar city. She has just arrived by train and would like to find a place to eat dinner. She starts up MTSR on her cell phone and requests a place to eat close to her current position. Meadow don't want the hassle of going through many restaurant options, so he chose to receive only restaurants that are recommended for her personally(R1).The system presents her with a couple of options, and she picks the closest one. Meadow then enjoys a long dinner. When she is finished eating, it's starting to get late. As most backpackers, she has not booked her accommodation upfront, and she now realizes that she needs to find a hostel to stay for the night. The hostel must be close, as she doesn't want to go far with her heavy backpack. She starts up MTSR again, and requests a place to sleep maximum 2 km from her current position. The system could not find any recommended places within this range. Meadow does another search, but this time not only for recommended places (R7).
Safely in her room Meadow decides that she want to read some information about an old fortress she visited some weeks back. MTSR is set to her current destination, but she quickly changes to the focus over to the city the fortress is located in. She then do specific search to locate information about it (R7,R2).

Scenario 3 - Using the travel plan

Silvio and his wife Gabriella is married couple enjoying a holiday cruise up the Norwegian coastline. This morning they float into an unfamiliar city for an entire day stop over. During breakfast Silvio starts up MTSR on his cell phone and request a itinerary for the day (R4). MTSR presents him with a set of activities as well as places to eat lunch and dinner. The system has recommended a cathedral and two museums. For lunch the system believes they will enjoy light seafood, and for dinner they are recommended a steak house. Silvio is pleased with the recommended plan, but he does not want to eat steak for dinner. He quickly finds another restaurant recommendation, and places this into the plan (R4).

Part II

Design science

Chapter 5

Research method

In this chapter we describe the methods we have used for evaluating MTSR . The result from the evaluation can be found in chapter 8.

5.1 Design science

5.1.1 What is design science?

The goal of information system research is further knowledge that aids in the productive application of information technology to human organizations and their management. Hevner et al. argue that gaining such knowledge involves two complementary but distinct paradigms, that is behaviour science and design science[18]. Behaviour science has its root in natural science. This branch of IS research uses analysis of empirical data to provide theories and explanations to organizational phenomenon. The ultimate goal is to find truth. The goal of design science, which has its roots in engineering, is to through the design processes create something, an artifact, that fill a need that not yet has been explored or to find a more efficient solution to a problem already investigated. In this context creation can be given a broad interpretation ranging from software to informal langue descriptions. The two paradigms complement each other in the sense that new theories can be found from studying an innovative artefact that represent utility in an organization. Similarly theories can be incorporated into the design of new innovative products to verify their utility.
The work presented in this thesis has its basis in the design science paradigm.

5.1.2 How should design science be conducted?

The danger of a design-science research approach is putting too much emphasis on the design artefacts, without considering the usefulness of the artefact in real organizational settings. To meet with this challenge Hevener et.al has proposed seven guidelines for conducting design science with scientific contributions. We will describe how we made use of these guidelines

- **Guideline 1** states that design science must produce a viable artefact. Our artefact is the "Mobile Tourist Service Recommender",and design and implementation of this this instantiation will be descried in chapter 6 and 7.
- **Guideline 2** requires the problem to be a relevant and important business problem. The tourism industry is regarded as one of the biggest sectors in the world. We did in Part 1 explain why supporting the vast number of tourist with information services is a relevant business problem
- **Guideline 3** stresses the need to use well-executed evaluation methods. We have evaluated MSRT with a group of students using methods form usability engineering and ethnography. We will provide details on this in the next section.
- **Guideline 4** calls for the design-science research to provide a clear and verifiable contribution. The background in chapter 4 lists similar existing services for mobile tourism, but there is a clear lack of applications that emphasize user control and transparency.
- **Guideline 5** states that the research should be based on the use of rigorous methods. In this project, the methods for construction and evaluation are chosen based on a study of previous research presented in Part 1
- **Guideline 6** argues that the research must be seen as an iterative process, where you continuously search for an effective solution to the problem. The goal of this thesis is not solve the problem of information support in the tourist domain, but rather use to gain a better understanding of how the optimal solution might look like. As we are using theory as a heuristic, the result of our research process will be presented as design suggestions that might guide future iteration.
- **Guideline 7** stresses the importance of proper communication of research to both technology-oriented as well as management-oriented audiences. We are in this thesis describing our artefact from both a user perspective (chapter 5) and a technical perspective (chapter 6). In

addition, Part 1 of this thesis serves as broker, as we link the business problem information support in the tourist domain with possible technical solutions.

5.2 Evaluation methods

The third guideline for conducting design science stresses the importance of evaluating the artefact produced. We will in this section give short description of the specific methods we have used for this, but first will be clarify an important distinction between research methods.

5.2.1 Qualitative versus quantities methods

Design science can be evaluated with both qualitative and quantitative research methods. Quantitative research methods have its roots from natural science, and are characterized by how numbers serves as strong scientific evidence. This class of research methods has traditionally been applied in positivistic research, where statistical analysis of numbers has been used to draw meaning form samples generated by methods surveys and closed laboratory experiments.
Qualitative research methods comes from social science as techniques for gaining rich insight into a phenomena entangled within a social context. The goal is often not to derive precise theories but to create understanding of phenomena. As the phenomena under study is inseparable for the social context it exist in qualitative research usually requires the researcher to interact directly within context using methods such as interviews and observation as tool for gaining insight.
In this thesis we sat out to investigate how an artefact supporting tourists with personalized information should be designed to prove utility in this domain. In relation to this question, quantitative methods like technology acceptance models could provide us with valuable insight.

The time constrains on this project has however not allowed to involve the large group of users that are necessary for providing scientific sounds result with these method.
Instead we have chosen a qualitative approach were we observe a small group of users while solving tourist tasks with MSTR in a realistic environment (We will come back to the specifics of this experiment in chapter 8). By qualitatively assessing the results from this experiment we hope to gain a better understanding of the utility of MSRT and the feasibility of our approach that will allow us to make valid assessments in relation to our research questions.

5.2.2 Usability testing with "the Think-loud"-protocol

A product should be easy and pleasant to use for the people that apply it as tool for accomplishing their goals. When testing for usability the goal is to find out if the design has these qualities. Usability testing encompasses a range of different methods that can be used at different stages in the product life cycle.
A commonly used method for conducting usability testing against a operational system is the "think-aloud protocol" .In this method the users are asked to express their thoughts (what they, think,do or feel) about the application while they are performing a set of specified tasks. This gives usability engineers information about how the user interface matches the natural human way of thinking and acting, and highlights the features and processes to be improved. Results from such tests are usually quite reliable and close to what would be experienced by users in a real-world environment. In large scale software project the results from this kind of user testing is often videotaped and analyzed by interaction designers and psychologists. Nielsen argues however that the benefits can be great from developers themselves taking the role of user experts and swapping the video tape with a note book.

5.2.3 Interview and observation

Observation is a research method with its roots in ethnography. The method has been adopted by information systems researchers, as it is a very useful method for finding out people are doing, not only what they say they are doing[15]. The qualitative interview can complete observation, as the researchers through the interview make direct inquiries about the user's perception, feelings and thoughts towards an it-artefact. Further, the interview allow the researches to expand on new issues that emerge in the context of the dialog, thus gaining insight that otherwise would be hidden[25]. Nevertheless, the interview can be problematic as a tool for finding empirical evidence. It is important to recognize the interview as social process where both sides of the tables (both the interviewer and interviewee) can affect the result. The interviewer does not mealy collect knowledge that exists independent of the social setting her she is currently in. New knowledge can be created on the spot, and the way the interviewer formulates his or her questions, as well as the general attitude and behaviour of the researchers can strongly influence the result.

Chapter 6

The Mobile Tourist Service Recommender

In this chapter we will describe the user interface of our personalized mobile tourist applications.

6.1 Introducing MTSR

The user interface of MTSR has been developed for smart phones running the Android operative system. The interface makes use of an interactive map as the primary way of displaying information. The user can interact with the map using the touch screen of the phone. The user can zoom the map by pushing zoom-buttons that appear when the map is touched. The user can also pan the map by holding down a finger on the screen and dragging it in the preferred direction.

The interface is divided in three main windows: "Find"-window, "Travel plan"-window and "Pending ratings"-window. The "Find"-window contain means to search for points of interest (POIs). The "Travel plan"-window contains an overview of the POIs currently in the user's plan, and means to request an automatic plan. The "Pending ratings"-window contains a list of places the user has visited. The user can switch between these windows by pushing the labels shown in figure 7.1. The user's position is indicated on the map as the red circle shown in figure 7.1.

6.2 Context awareness in MTSR

6.2.1 What is context in MTSR?

We have in MTSR chosen to classify the context properties used by the system as the following:

- Location: Geographical location of the user
- Field of interest: The area currently shown on the users map
- Time: The time of day at the user's location
- Preferences: The set of opinions expressed by the user
- Visiting history: The set of places the user has physically visited

6.2.2 Adaption strategy

We have used the following strategies for adapting the user interface and information sent to the user in relation to the notion of context stated above

- The map centres to match the location
- When the user searches for places of interest the result is filtered on preferences and field of interest
- When the user requests a travel plan the result is filtered on preferences, location and time
- The system sends automatic recommendations to the user based on location and preferences
- The system sends automatic rating requests to the user based on location and visiting history

6.3 The user interface of MTSR

We did in Chapter 4 formulate the following system requirements for MTSR

Req.number	Description
R1	The system must have pre-defined requests for typical tourist services as part of the user interface
R2	The system must allow the user to search freely after tourist services
R3	The system must provide the user with the distance,route and description of each service
R4	The system must allow the user to request and managed complete travel itineraries
R5	The system must provide the user with automatic/pro-active recommendations
R6	The system must provide explanations of the personalization process to the user
R7	The system must allow the user to override the adaption strategy
R8	The user must be able to control the personal preferences used to create recommendations
R9	The system must allow the user to express his opinion about the services presented
R10	The system must automatically provide the user with an overview of places the user has visited

We will now show how these are reflected in the user interface of the application. We will do this with basis in some typical user tasks and activities that can be accomplished using the system, these are

- Receiving pro-active recommendations
- Evaluate information received from the system
- Searching for information
- Providing feedback
- Using the travel plan

6.3.1 Receiving pro-active recommendations

- **Requirement 5**: The system must provide the user with automatic/pro-active recommendations

Pro-active recommendations has been formulated as a system requirement because it allow the tourist to retrieve recommendation close to their current location without having to search. This is favourable for two main reasons

1. It can be annoying/difficult to search in outdoor environment using a small screen

2. Without knowing that there is something interesting nearby, it can be difficult to search for it

Pro-active recommendation has therefore been part of earlier prototype systems, such as TIP and Crumpet[19, 33] (referred to in TIP as "browse by walking"'). This research include models and details on interaction design[20], but there exist little information how to include the feature as part of the user interface.
Following general usability guide lines we have tried to make automatic recommendations as non-obtrusive as possible from the user's point of view. We have ensured that the user can never get the recommendation for same POI twice and the frequency between recommendations it at least 10 minutes. In addition, the recommendation will disappear from the user interface if the user has not interacted with it within in three minutes. We have however included a light vibration when new recommendations are received, as the user is likely to carry the phone out of sight.

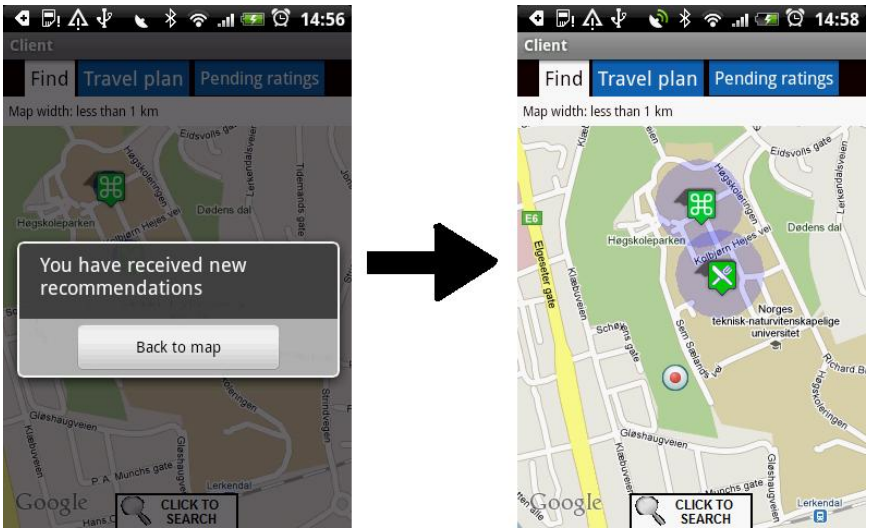


Figure 6.1: The user has recvied pro-active recommendations and the phone makes light vibration

6.3.2 Evaluate information received from the system

- **Requirement 3**: The system must provide the user with the distance,route and description of each service
- **Requirement 6**: The system must provide explanations of the personalization process to the user

After receiving pro-active recommendations, or any information on places of interest, the user will typically seek answers questions like[8]:

- Is this something I would like to do?
- How much time and effort would it take me to get there?

To answer these questions the user must be able asses the subjective value of the current POIs on the map as well as the distance and route from his current position.

Is this something I would like to do?

In addition a textual description of each POI we have in the user interface of MTSR made use of three constructs that we hope can help the user chose to answer this question. These are

- Illustrations of services with a colour indicating it's predicted subjective value in relation to the user
- Explanations
- A listing of "top rated"-places based on correlation with other users

Firstly, we have included illustrations of each tourist service shown in the map instead of the common approach of plain markers. These are given a colour ranging from red to green based on the recommender systems prediction of user liking. We have in chapter 3 describe how predictions can be used in recommender system interfaces to help the user in the process of assessing relevance and probability of liking. Ricci et.al has taken this idea into mobile recommenders in the design of MobyMapRec[5]. The colour schemes contributed strongly to efficiency and satisfaction of this system, as it provided the user with a way compeering and evaluating the resulting recommendations directly by looking at map. We believe combining this approach with easily recognizable icons bring much more information into the map without complicating the user interface.

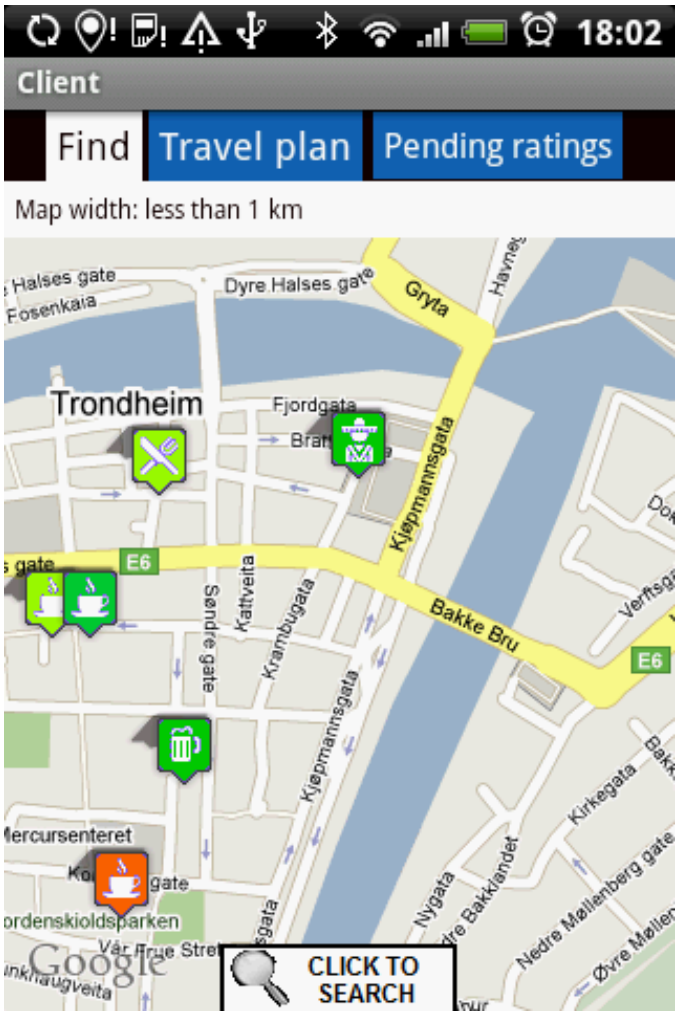


Figure 6.2: The user has retrieved relatively large set of services and can potentially reject recommendations directly based on the provided image and prediction

The second construct we have introduced to support the user in the process of evaluating recommendations is explanations. The explanations contains, as pictured in figure 6.3, two main components:

- The average rating as stars (between 0-5)
- A textual description explaining the group of people that make up the average rating

We believe that a simple star-scale can be useful as overall metric of the quality of the recommendation. Nevertheless, we argue that the subjective element of preferences can be so large in the tourism domain that the user can benefit from additional means for quality assessment.

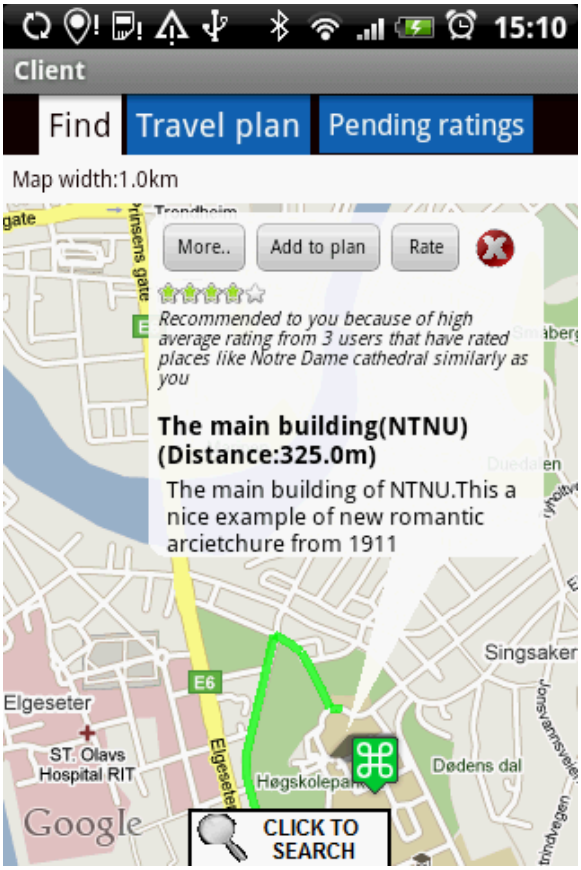


Figure 6.3: The user has touched the icon representing the main building of NTNU campus

We have chosen an approach where we provide the user with information about similarity in taste between the user and the group the average rating is based on. We will provide details on how this is done in the next chapter. For

An example of the explanation technique used in MTSR is shown in the figure 6.3. The user is presented with a recommendation for the "The main building of NTNU". It has received a very high average rating from a group of people, but because of the sparsity problem, which we touched on chapter, this might not be a accurate prediction. The user can turn to the most influential places in terms of the correlation between him and the group, in this case only the Notre Dame cathedral, to give more meaning to the average rating. The third construct can be found by accessing the "More info"-button available in the dialog shown in figure 6.3 .

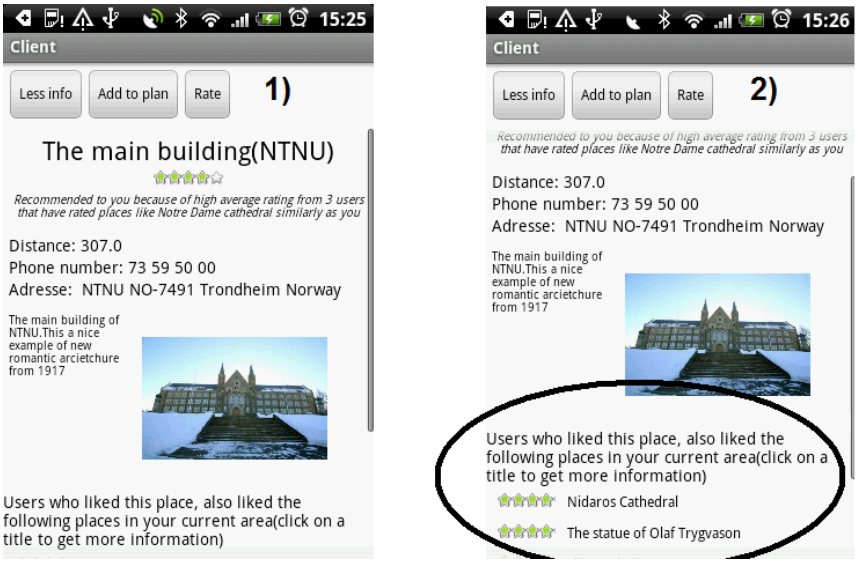


Figure 6.4: 1) : The user has pushed the "more info" , 2) User has used his finger to scroll

The results from this are shown in figure 6.4. In this screen shot we can see that the user can explore "top rated"-places of the people that have liked the specific point of interest. Browsing this list the user can learn more about the users that have provided rating, as well finding new places to seek out. By pushing one of the items on the screen, the application will automatically locate this new place on the map.
These constructs is also meant to contribute to the overall goal of a transparent system interface. In the textual explanation we have tried to expose the underlying filtering algorithm, in a way that high lights the importance of rating in relation to the personalization process. The list of "top rated"-

items is meant to further high light the importance of ratings in relation to the system perception of preferences.

How much time and effort would it take me to get there?

Going back to this question, we can see from figure 6.3 that the system provides the user with the distance to each POI as part of the dialog box that appear when the users touches a icon in the map. The walking route is automatically drawn as green path from the user location to the tip of dialog(that represents the location of the service).

6.3.3 Searching for information

- **Requirement 1**: The system must have pre-defined requests for typical tourist services as part of the user interface
- **Requirement 2**: The system must allow the user to search freely after tourist services
- **Requirement 3**: The system must allow the user to override the adaption strategy

According to the research we presented in Part 1, tourist are for the most part looking for places to eat, sleep or places to experience tourist attractions (typically sight ,museums etc). To be useful a personalized mobile tourism application must provide essay access to recommendations that fall within these categories[8]. To meet this requirement, but at the same limit the amount of screen space taken up by means to request information, we introduce in MSRT a construct we have called "the sliding search window". The user can slide this window up on to the map by pushing (or dragging) the handle labelled "click here to search".

Our predefined search requests are "Eat", "Sleep" and "Experience" and the user can do a personalized search by pressing one of these labels and hitting search. The search window will then disappear, and the result will be shown in the map. A free/custom search can be done by clicking the search field and typing in a custom search string.

The field of interest can be changed by pan and zoom, and the user can bypass filtering on personal preferences by deselecting the option field labelled "Only stuff recommended for me". The result form this search is every match within the current area covered by the map. An example of this type of search is shown in figure 6.6

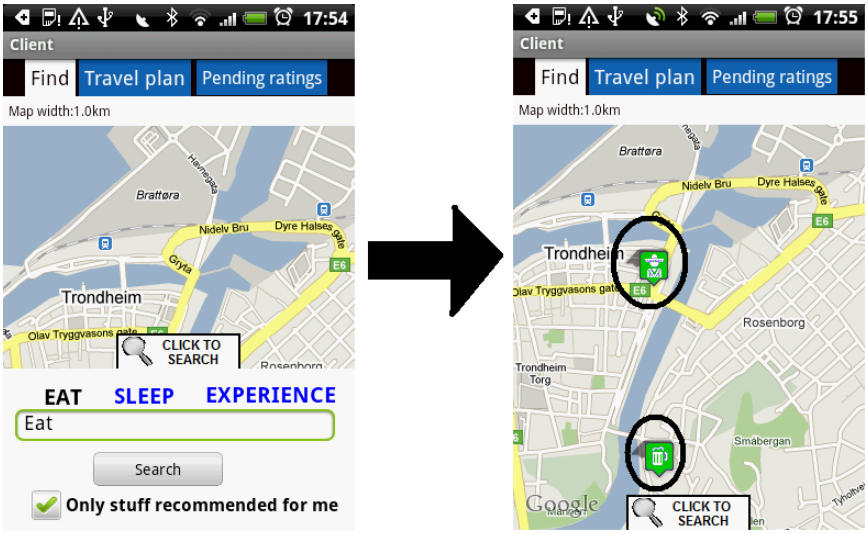


Figure 6.5: The user is making a personalized search by using the predefined "eat" request. The result is a Mexican restaurant

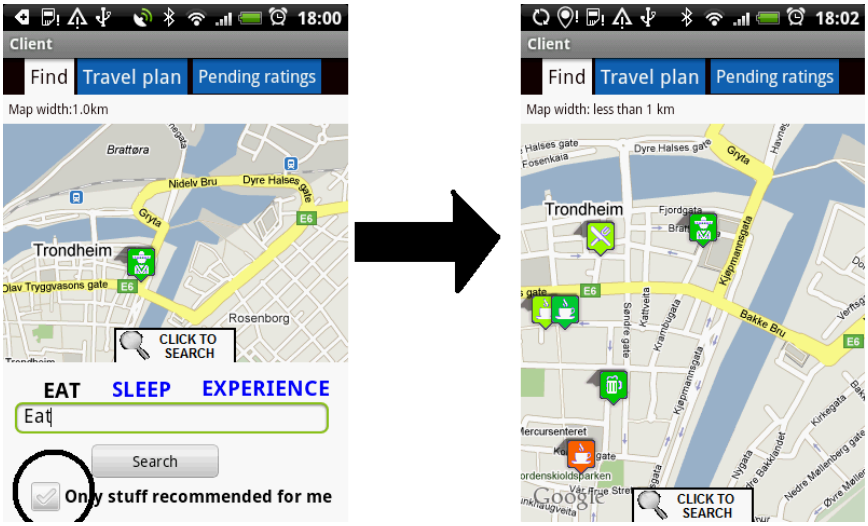


Figure 6.6: The user is by passing personalization, this yields a lager result set

6.3.4 Providing feedback

- **Requirement 8**: The user must be able to control the personal preferences used to create recommendations
- **Requirement 9**: The system must allow the user to express his opinion about the services presented

- **Requirement 10**: The system must automatically provide the user with an overview of places the user has visited

In MTSR the user can only change and control the under laying user model by providing feedback. We could have exposed the full model and allowed the user to adjust it explicitly, but the challenge of presenting this information in a intelligible way proved very difficult on the mobile screen. Further, we believe it is unlikely that user would bother with maintaining and updating this list of preferences if it is to comprehensive . Instead we relaying on that partly exposing the user model through explanations will make the user understand how the feedback given to the system will affect the recommendations retrieved.
We have focused on making the process of giving feedback as simple as possible, thus differencing us from the system listed in chapter 4 as these relayed on exposing the full (content based) model. The user can open the dialog of a place in the map, and hit the "rate"-button. This will present the user with the screen on the left in figure 6.7 .The user can then apply his finger to select the preferred number of stars.

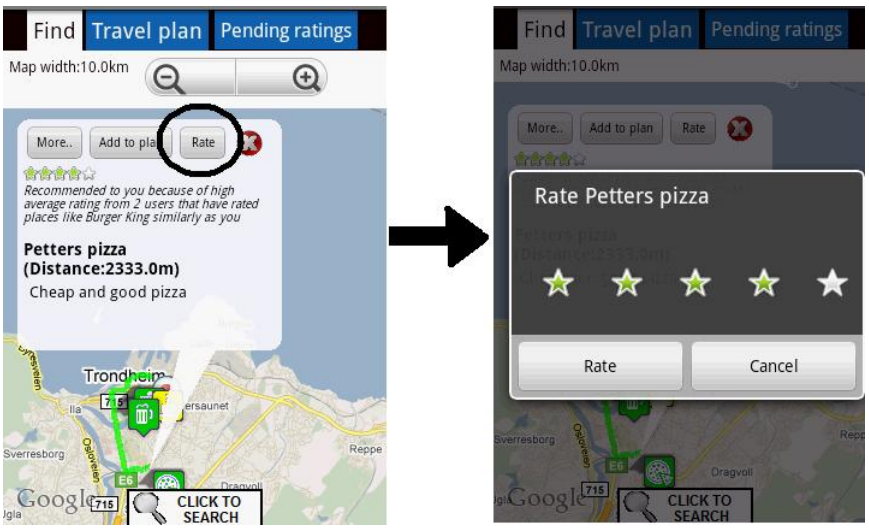


Figure 6.7: The user has pushed the rate button after opening the icon representing Petters Pizza

The second way the user can provide feedback to the system is by accessing the "Pending ratings"-window. In this window the system keeps a list of all the places the users has visited during the day. Each time the system has detected a new visit, it alerts the user by a short vibration and a animation that high lights the number of ratings currently not been checked by the user.
This implementation allow the user to let the system build up rating requests, so that they can be handled at point where the user has some time to spare.

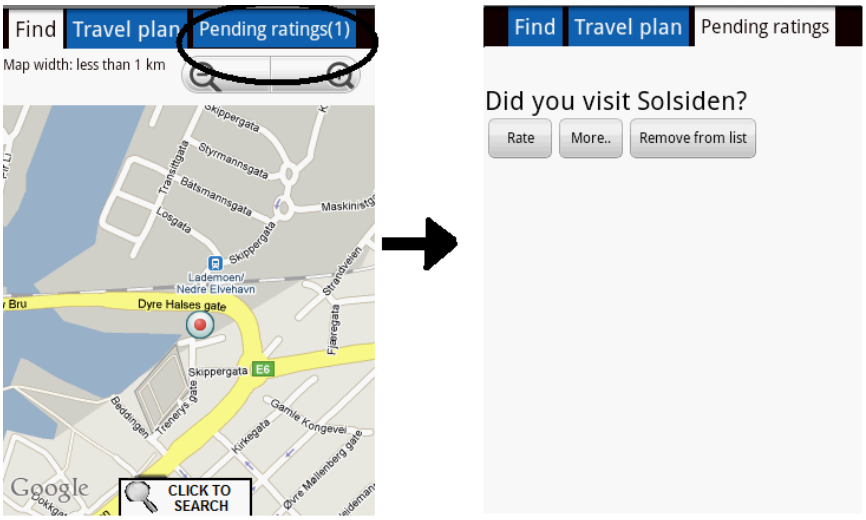


Figure 6.8: The user access the "Pending rating"-window after an notification

6.3.5 Using the travel plan

- **Requirement 4**: The system must allow the user to request and manage travel itineraries

From our study of previous system this requirement has been brought out as an important feature for two main reasons

- It can provide the user with much travel information quickly
- It allow the user to save recommendations/information on POIs between sessions with application

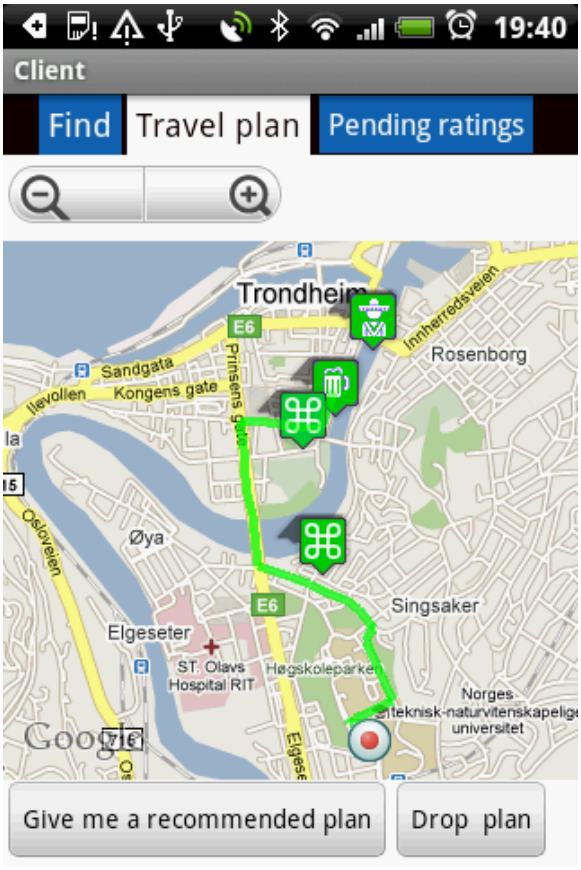


Figure 6.9: The travel plan window

As for the user model control, we argue that previous approaches to personalized travel plans require too much information from the user. In both CRUMPET and the system by Chang et al. using this feature require the user to input a considerable amount of information before plans can

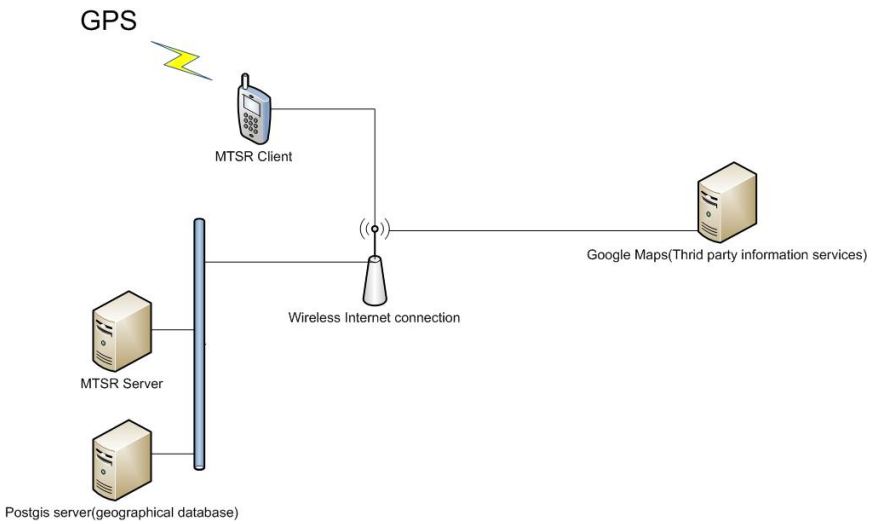
be retrieved[43, 32].
Again we have tried to make the processes of retrieving the plan as simple as possible. The user has only on enter the "Travel Plan Window" and push the button labeled "give me a recommended plan". The system will then use the context properties to find a set of activities as well as places to eat lunch and dinner(if the time allows it). The user can not directly by pass the adaption strategy in relation to this feature, as we believe this would complicate the interface more than it could benefit the user. The user can however add and drop places to the plan by pushing the icons and "add to plan" or "remove from plan" buttons in the dialog box. New items must be found through the "Find window". As the means for finding information here go beyond the adopting strategy, we believe it functions as leverage to the lack of user control in relation to the travel plan.

Chapter 7

Design and implementation details

In this chapter we described in detail how the server and client component in the MTSR architecture are designed and how they function together to enable the services we discussed in the previous chapter.

7.1 Architecture of MTSR system



As shown the figure above we have chosen architecture based on a client - server model. The services delivered by the system is both

- **push** services as pro-active recommendations and pending rating request and

- **pull** services as the various information requests that can be made by the client

7.2 The MTSR client

7.2.1 Choice of platform

The client application has been developed for smart phones running Android. Android is an operating system and software stack for mobile devices that includes middle ware and key applications, and uses a modified version of the Linux kernel. For application development on android a comprehensive collection of libraries and tools is freely available in the Android Software Development Kit(SDK)[12]. The SDK has many advantages as an platform for rapid development of location based services, these include:

1. Native support for Java SE
2. Dedicated libraries for developing advance map applications
3. Convenient interfaces to positioning hardware as GPS
4. Framework for handling events from the touch screen

7.2.2 Important components of the Android SDK

The Activity class

An activity presents a visual user interface for one focused endeavor the user can undertake. Each activity is given a default window to draw in. Typically, the window fills the screen, but it might be smaller than the screen and float on top of other windows[12]. We have developed three main activities in our applications. These are the "Find"-activity, the "Travel plan"-activity and the "Pending ratings"-activity. When evoked, activities run as single threads in a Linux process. To communicate between two activities remote procedure calls must be used.
In addition to serving as container for the visual user interface of the application, the activity can be responsible for handling the user interaction with the touch screen. The Android framework allow activities to receive simple notifications each time the user interact with the screen. These notifications contain information about which pixels and component of the interface that was touched.

The service class

A service does not have a visual user interface, but rather runs in the background for an indefinite period of time. For example, a service might play background music as the user attends to other matters, or it might fetch data over the network or calculate something and provide the result to activities that need it[12].

The View class

A View is the basic component of the user interface in Android applications. The graphical content in a activity is represented as a hierarchy of views. The various views can be defined either in Java-code or in XML. The view hierarchy is loaded into the the activity at initiation. However,new views can be added at later stages in the activity life cycle by so called inflation. In this project we have made much use of a tool called "Droid Draw" for interface design. This tool allow designers compose view-trees throug a drag-and-drop interface. "Droid draw" out puts a xml-file than can later be exported into the android project[13].

To create interesting graphical effects, such as animations, the draw-method found in each View-class can easily be customized. The Android SDK also provide a framework for animation, but we found this limiting when working with map applications.

The Location manager class

This class control the access to various system services related to location. Other components can sign up as listeners to the manager for receiving location related information. The class provide a convenient interface to the underlying GPS-implementation. Application designers need only to provide the location manager with information on how often, fast, or far the user has to move in order to get an update on the geographical location of the device in their logical components .

7.2.3 How does the MSRT client work?

The tasks of the MSRT client is to

- Display information
- Allow the user to request information
- Keep the server updated on the user current context

The following high level class diagram illustrated the the workings of the client.

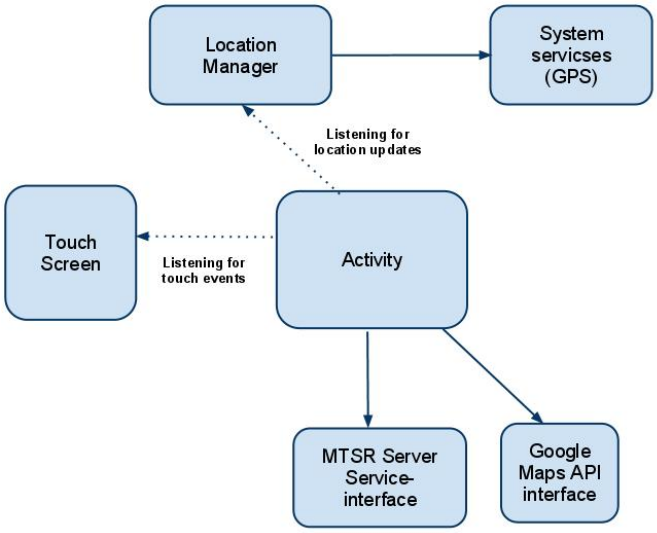


Figure 7.1: Functional architecture

Display information

For displaying maps on the cell phone screen we have extended the MapActivity-class. This class provide developers with a "out of the box" interactive Google Map. The Android SDK includes a Java-implementation of the standard Java Script Google Maps API. The android implementation is rather extensive, but we had to fetch the the walking routes described in the previous chapter directly from the online map services provided by Google[14]. To add information to the map we created custom implementations of the Android SDK Overlay-class. This class makes it simple to put basic icons on top the map.
We have made extensive use of overlays to make the map in MTSR interesting in terms of animation and the colored icons. The base icons come from the "Google Maps Icon project". The goal of this project is to visualize all common services to make the interaction with google maps more user friendly. The project has at of the time of writing a collection of more than 900 free icons[24].

Allow the user to request information

The various activities serve as controllers of the user interaction. For example, when the user push the "Search"-button the "Find"-activity calls the corresponding method in the MTSR server interface. This interface contain methods for retrieving recommendations, sending feedback and context updates as well as managing the travel plan. The MTSR interface is an implementation of the Service-class. When the result comes back from the MTSR server, it is interpreted by the activity that made the request. The final result is then drawn on the map.

Keep the server updated on the user current context

For allowing the client to keep the server updated on the user current location without having the user to explicitly input this information, the various activities are signed up as listeners to the location manager. The activities retrieve notifications each time the user has moved 30 meters or more than 5 minutes has passed since the last update. When the location manager provide a update, the current activity sends a context update to the server by accessing the corresponding method in the MTSR server interface.

7.3 Introducing the MTSR server

The MSRT server is implemented as a three layered web application running on the Apache Tomcat application server.

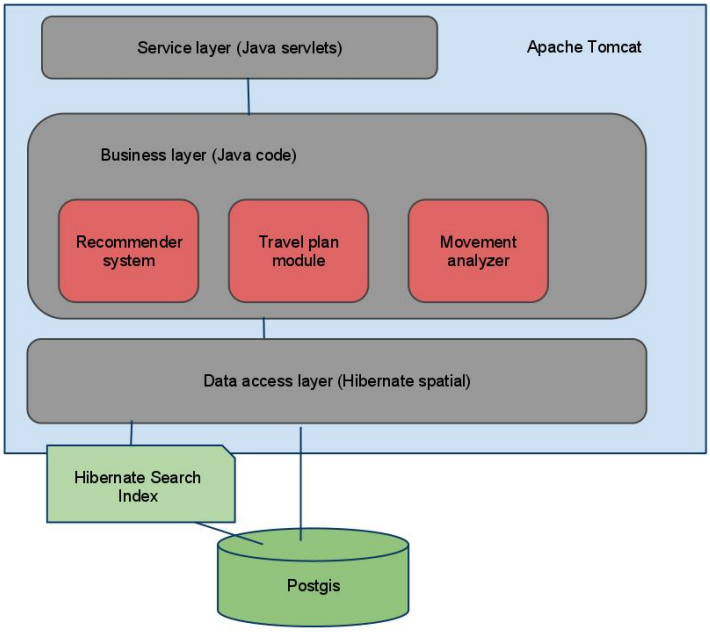


Figure 7.2: Functional architecture

The layers has the following tasks and responsibilities

- **The service layer** is responsible for calling methods and aggregating the resulting data from the business layer before on requests from the client
- **The business layer** implements the business logic needed in MTSR. More specifically this layer holds a recommender system, logic for composing travel plans as well as a module for analyzing the user movement. The business layer stores and retrieves geographical data from the data access.
- **The data access layer** acts like a interface for the underlying spatial data base, and provides the business layer with Java-objects that represents tables in the database.

Before going into the details of each layer we will provide a short description of the third party components and framework we used in the server implementation

7.3.1 Third party components and frameworks

- **Servlets** is a class part of the Java Enterprise Software Development Kit that can respond to Http-requests when deployed in web server environment. We have used servlets to implement the service layer. Having an http-interface to the server gave us flexibility as most devices are capable of using this protocol [2]

- **Apache Tomcat** is an open source web server. We have used Tomcat as a container for the MTSR server, as the platfrom has mechanism for handling servelets[10].

- **Postgis** is an open source spatial relational database. We have used Postgis to store and manipulate geographical information [3] .

- **Hibernate spatial** is object-relational mapping framework for mapping an object-oriented domain model to a spatial database. We have used hibernate to simplify our implementation[37]

- **The JTS Topology Suite(JTS)** is an API of 2D spatial object and functions such as intersection and distance calculation. We used JTS together with hibernate to allow us to work efficiently with spatial objects[1].

- **Hibernate Search** is an object-index solution for full text search engines. It provides an easy to use framework for mapping an object-oriented domain model to the indexes created by a traditional search oriented architecture[35].

7.4 Data access layer

The data access layer provide a interface for storing and receiving Java-objects in the following domain model. Each class and association map to a table in the relational database.

7.4.1 Domain model

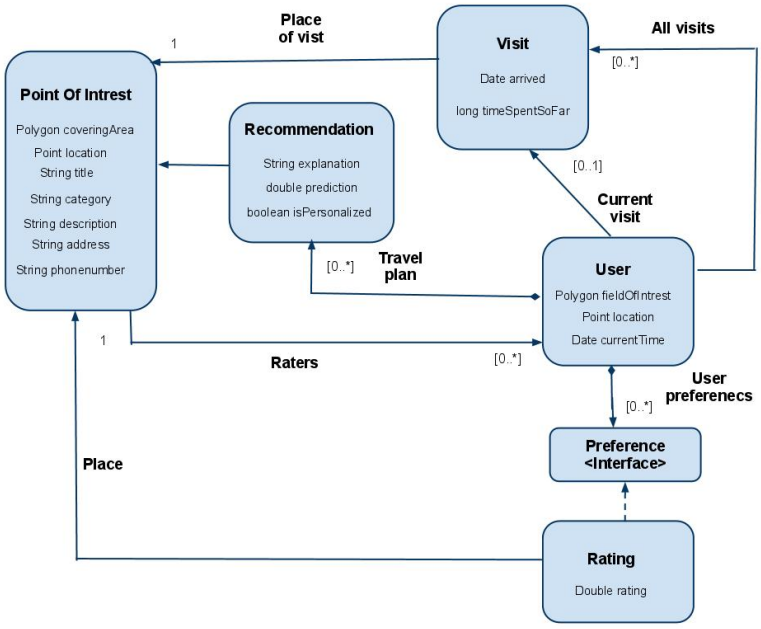


Figure 7.3: The classes used in MTSR

- **Point of interest** represent a tourist service. The location-attribute is a JTS class called Point which holds the physical location of the POI. The CoveringArea-attribute is an JTS class named Polygon. It contains the area taken up by the particular POI in the real world. Each POI is also associated with a set of users. These are the people that have provided their opinion about the POI.

- **User** represents the mobile user in MTSR. It holds the users location,time and field of interest. These are mapped by JTS point, Java Date and JTS polygon respectively. The user is associated with a set of preferences, that make up the user model. The user has also is associated to set of recommendations that make up the current travel plan

of the user. In MTSR, the user can also be associated with the POI he is currently visiting. This is modeled by the current visit association.

- **Visit** represents a physical visit made by the user at a POI. The the date and time of day of the visit is represented by a Java Date-object. The duration of the visit is stored as a "long" in the timeSpenSoFar attribute of the model.

7.4.2 Data access objects

For storing and retrieving the objects in the domain model we have wrapped the hibernate sessions- and transaction management by a data access objects(DAO)[23]. The DAOs provides a abstract interface independent of underlyding data source and ORM. We have designed this so that future project easily can extend the the architecture, as we believe that MTSR may need tourist service information from external content providers. The same interface can be used even though the basic spatial database is changed for external resources. This would only require new implementions of the DAOs. To illustrate how the business layer can make us of the data access layer we will now list parts of the abstract interface used by the POI-data access object, as we will refer to this later in this chapter.

- *findPOIs(Geometry area,String searchString)*, this method finds all the POIs within the given JTS-Geometry that matches the search string. The area will typically be the polygon representation of the user field of interest.

- *findPOIsContaining(Point p)*, this method finds the POIs that currently contains the specific point. This method is typically used to find out if the user is standing at a POI

7.5 The business layer

The business layer is the core of the MTSR-system as it holds the logic for creating recommendation, movement analysis and the scheduling needed for providing travel plans. The methods in this layers is evoked on request from the service layer, and uses the data access objects of the data access layer to retrieve and store the information necessary to provide these services. We wanted this layer to encompass a modular design, so that future project easily can extend the functionality. Each of the following modules therefore implements an abstract interface, that allow us to swap implementations easily

- Recommender system
- Movement analyzer
- Travel plan module

7.5.1 Recommender system

The recommender system module has an abstract interface that consist of the following to methods

- *createRecommendations(List <PointOfIntrest> pois,User user,boolean doFilter)*, this method create a list of recommendation based on the input list of POIs. If the variable doFilter is true, only a small set of highly recommended places is returned. The method allow the other parts of the application to retrieve recommendations.
- *handleFeedback(User user, PointOfIntrest place, double feedback)*

How does the system work to provide recommendations?

The implementation of the abstract interface used in this project is a basic implementation of collaborative filtering using the Pearson product-moment correlation coefficient as similarity metric[6].

$$r = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}}.$$

Figure 7.4: The Pearson product-moment correlation

In this formula the ratings from two users that are going to be compared are denoted as the vectors X and Y respectively. The resulting correlation metric is a value in between -1 and 1 representing the linear dependency between the two sets of rating.

We will now provide a step-by-step description of how our recommender system work to provide recommendations after being evoked by the *createRecommendations*-metohd. We will refer to the user in the method declaration as User A.

1. Fetch the people that have provided rating to the places in the pois-list (these are automatically fetched due to the "raters" association from POI to User)

2. Do correlation calculation between these people and User A.
3. Drop users that have low correlation.
4. Calculate average rating for each POI from the users left in the process
5. Select only the POIs with the highest average rating
6. Create the recommendations by using the resulting POIs.
7. Insert explanation and prediction for each recommendation.

This implementation is only created to fill the basic need for recommender system in MTSR. The other modules in the business layer is depended on a implementation of the abstract interface we listed at the beginning of this section to provide their services. Further, we needed a operational recommender system to create a realistic user experience in the field experiment we will present in the next chapter. As of this, our contributions in this project is not connected to the accuracy and efficiency of this implementation, but we hope that further projects will evaluate our approach, and come up with another implementation that is designed and evaluated against readable metrics like precision and recall[6].

How is the explanations created?

An important part of this project is however to observe how a transparent user interface to a recommender system is perceived by users. When it comes to how these are created, we must go back to the steps in the previous section. Before sending the result from step 7 we do the following for each POI to create an explanation of why it got picked out.
For each POI left do :

1. Find the people that have provided rating for the POI
2. Create hash table on the form <POI, number>, where the number represents how many users that share an high rating for the particular POI
3. Fill the hash table by going through each users POIs and increment the number if two POIs are the same (this information is found through personal prefernce association in the domain model)
4. Do a look up in the table for each POI in User A's set of ratings where User A has provided a high rating

5. Pick the POIs in the hash table that have the most users
6. Use these POIs in the explanation

If the algorithm does not find any common POIs between the User A and the group, it uses the POI with the most users independent of User a. In any case, the algorithm does not pick more than three POIs to use in the explanation as it would take up to much screen space on the client device. We believe this process represent and simple, but effective way of conveying the correlation within a group of users.

7.5.2 Movement analyzer

The task of the movement analyzer is to

- Find out if the user should retrieve pro-active recommendations
- Find out if the user should retrieve pending rating requests

This module also implements an abstract interface consisting of the following two methods

- *checkForProactiveRecommendations(User u)*
- *checkForPendingRatings(User u)*

Find out if the user should retrieve pending rating requests

As we described in our account for the MTSR Client: Context updates are sent to the server when the user has moved thirty meters or the time since last update is more than five minutes.
The call sequence leadening to a pending rating can be illustrated by the following diagram:

After the update has reached the analyzer, it checks whether the user's new location is within in a place of interest. This is done by calling the method in the POI-data access object. At this point it's important to note that each place of interest is associated with a geographical polygon, representing the space taken up by the place in real world. Simple spatial mathematics can then be used to calculate whether the user's current location is within the particular place.
If the user is within, this could mean that the user has found a new tourist service. The next step taken by the analyzer is to control this new place against the last known place the user has visited. Going back to the class-diagram we can see that the user is tied to this POI through the current visit

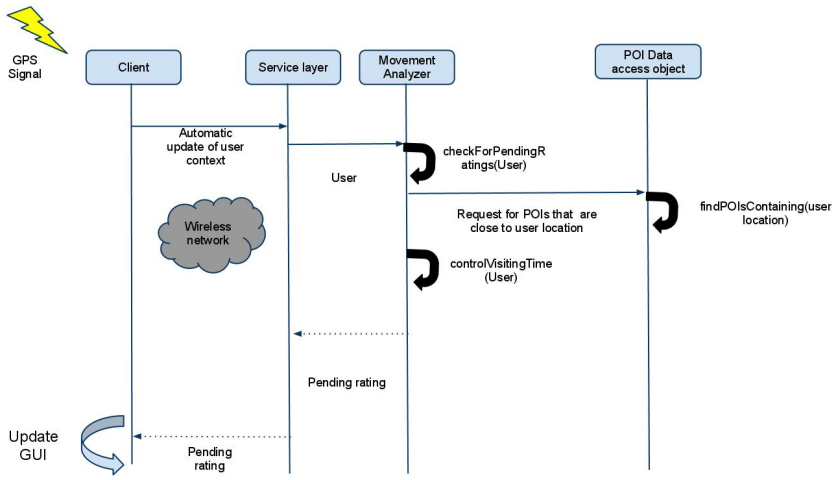


Figure 7.5: The movement analyzer finds a pending rating

association. If these are the same POIs the analyzer calculates the total time the user has spent at the POI. If this time exceed thirty minutes, we believe this yield a good indications of that the user has made a actual visit to the POI. In this case the analyzer sends a pending rating back to the user.
If the places are the same, but the visiting time is less than 30 minutes the analyzer only updates the visiting time.
In the cases were the last known visit and the new visit is different places, the analyzer tries to find out if the user stayed at the last known visit more than 30 minutes by looking at the time since last update. Next the analyzer will store the new POI as the last known visit.

Find out if the user should retrieve pro-active recommendations

The call sequence leading to a pro-active recommendation is similar and is illustrated in figure 7.6.

Upon receiving an context update the analyzer first checks whether the user has moved a considerable amount since last update. If the user has moved, the movement analyzer then creates a circular spatial area with center at the users spatial position. The the length of the radius is the indicates the maximum distance from the user that potential POIs can be. The analyzer then calls the POI-data access object using the newly created circle as the geographical parameter.
The analyzer then checks whether these POIs has been sent to the user at an earlier time.
If not, the next step is collaborative filtering, that outputs a set of recommendations if the POIs was not removed in this process. The final result is

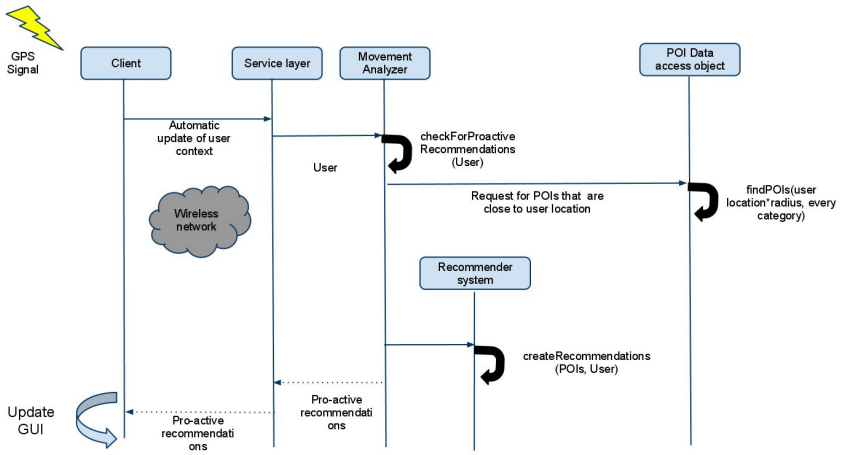


Figure 7.6: The movement analyzer finds a pro-active recommendation

sent bakc as pro-active recommendations back to the service layer.

Limitations

- The techniques we have presented here are wasteful in terms of network bandwidth and computer resources as most of the time no information will be sent back to the user. The scheme is there for no feasible in communication networks with expensive data transfer.
- The geographical location used by the algorithms must come from a very accurate positing component. We found that the GPS-sensor in our HTC Hero Smart phone provided sufficient accuracy, but we have not done any testing with other devices
- The pending rating algorithm require a polygon model of the area taken up by a POI. This information could be challenges to provide if external content providers were used in the architecture.

7.5.3 Travel plan module

The core of the travel plan module is an implementation of the process for providing recommended itineraries proposed by Chien-Chih and Hsiao-Ping[43].

The tour planing process

The process for creating travel plans are depictured in figure 7.7

```
1
2
3  Set temp_time=User.time
4
5  Set temp_location=User.location
6
7  while loop
8      if temp_time is between 11am and 1400 and luch status = no
9      or temp_time is between 1600pm and 2100pm and dinner status=no
10
11         potentialPOIs = PointOfIntrestDAO.findPOIs( temp_location*radius, "eat")
12         restuarant recommendation = RecommenderSystem.createRecommendations(potentialPOI,User,true)
13         put restuarant in plan
14         temp_time = temp_time+suggested_eating_time
15         temp_location = location of restuarant recommendation
16
17     if temp_time<2200 pm and number_visited_activites<7
18
19         potentialPOIs = PointOfIntrestDAO.findPOIs( temp_location*radius, "experince")
20         activity recommendation = RecommenderSystem.createRecommendations(potentialPOI,User,true)
21         activity recommendation = RecommenderSystem.createRecommendations(potentialPOI,User,true)
22         put activity in plan
23
24         temp_time = temp_time+suggested_eating_time
25         temp_location = location of activity recommendation
26
27
28 return travel plan
```

Figure 7.7: Tour planning

This process takes the user's current location and time as a starting point. For each iteration the recommender system is called with a set of POIs that are found based on the current location. When the process finds a new recommendation to add to the plan, the location is updated to this POI, and the time is increased by the estimated visiting time.
This process is almost identical to the one presented by Chien-Chih and Hsioa-Ping. We have however removed the dependencies to the content based filtering used in their system. Our travel plan module functions indenpended of the underlaying filtering algorithm. Further, the process require no explicit information from the user.
The process is very simple and it will undoubtedly make mistakes in terms of what is best for the user. However, we believe that it provides an useful enchantment to personalized tourist applications as long as the user is able to correct the mistakes by adding and remove POIs to the plan.

Limitations

- The process takes many assumption (such as when the user will be ready to go to bed)
- The POIs in the plan drift in a certain direction after the recommendations closest to the user has been found. This may cause highly recommended places in the opposite direction of the "drift" to be left out.

Chapter 8

Evaluation

In this chapter we will present the result from our usability test and the interview session we had with a group of users. We will also describe our method for collecting empirical data.

8.1 Our evaluation method

Our evaluation of MTSR is a field experiment carried out in a outdoor environment. The experiment was dived into two parts. The first part was a usability test of the application. This test required the user to complete a set of typical tourist tasks. Our role was for the most part as observers, only intervening when the user appeared to be completely stuck. Up front we encouraged the users to share their thoughts and doubts while carrying out the tasks, in accordance with the "Think loud" -protocol. We also explicitly prompted the testers for their thoughts when we they appeared to be in doubt.
We had three primary goals for the usability test. Firstly, the test was conducted to get a feel of how easy and pleasant the application is to use. To operationalize "easy" we have used the simple metric of whether the user was able to complete the task or not. To asses "pleasant" we tried to observe the feelings expressed by the test subjects while carrying out the tasks. As both these metrics are qualitatively assessed, we will give thoroughly description of the basis of our assessment in the following sections.
Secondly, the test was used to feedback and suggestions on how we could improve the design of MTSR. The testing was conducted in the course of a week. We therefore had some time to alter our design when large usability issues were discovered. We will come back to the details of this in our description of the result from the various tests.
The third goal of the usability test was to get a feel for how MTSR might

function in real usage situation. Most of the tasks the users were required to complete did not have any predermintedcompletion criteria, but rather called to for the user to subjectively decide when they had found a recommendation they were pleased with.
The second part of the evaluation was a interview session. Each interview was lasted approximately 40 minutes. The goal of the interviews, which were semi structured, was to find out if the user had understood the personalization process. We also used the interviews to expand on the users perception and feelings towards aspects of the system that was not directly included in usability test: Throughout the experiment the user received automatic rating requests after visiting places. We did not explicitly make the user aware of this, so we could explore the users initial reaction to this feature. The interviews allowed us to gain further insight on the user's attitude towards this feature. Extensive note taking was used during the interview sessions, and all conversations was record for further analysis.

8.2 The test group and environment

The test group consisted of nine students. Four of these had background in computer science, while the other four were students of electrical engineering, micro biology and economics respectively. Neither of the testers had any experience using smart phones with touch screen or knowledge of recommender systems.
The first six tests were carried out on the Norwegian University of Science and Technology-campus. The campus was chosen because it represented predicable and safe environment in terms of wireless internet connection. As the first six tests were successfully completed we decided to change the test environment. The last three last users in our evaluation carried out the usability test in down town Trondheim. The "Wireless Trondheim project" provides this area of the city (pretty) reliable wireless internet connection. Before conducting the test we had placed about hundred point of interest in the data base, and distributed ratings. We also sat the time to receive pending to five minutes, so that we were sure that all the testers should receive at least on pending rating.

User	Sex	Background	Environment of the experiment
User 1(U1)	Female	Microbiology	NTNU Campus
User 2 (U2)	Male	Computer Science	NTNU Campus
User 3 (U3)	Male	Economics	NTNU Campus
User 4 (U4)	Male	Electrical engineering	NTNU Campus
User 5(U5)	Male	Electrical engineering	NTNU Campus
User 6(U6)	Male	Computer Science	NTNU Campus
User 7(U7)	Male	Electrical engineering	City centre
User 8(U8)	Female	Computer Science	City centre
User 9(U9)	Male	Computer Science	City centre

8.3 Result from usability testing

8.3.1 Scenario 1

The first scenario was created to illustrate how tourist can make use of the information retrieved as automatic recommendation. The scenario was presented to the test subjects as the following list of tasks

1. Rate 10 places
2. Chose the closest recommendation and find out why it was recommendation for you
3. Find the address of this particular place
4. Use the information available on the particular place to find a new place in your current area recommended by users with the same ratting pattern as you
5. Locate the new sight physically

Firstly, in order to provide recommendations the social filtering algorithm needs a set of ratings from the user. To build this initial profile and provide flying start for the test subject, each user had to provide a rating between zero and five stars for ten famous landmarks and restaurants in relation to in which degree the user would like to visit the place, or similar places, in the future. We used landmarks such as the Eiffel Tower and Gran Canyon, as we assumed these would be known to all our test subjects.
This initial task was carried out in a separate application, designed only for this purpose. A screen shot from the application is shown below:

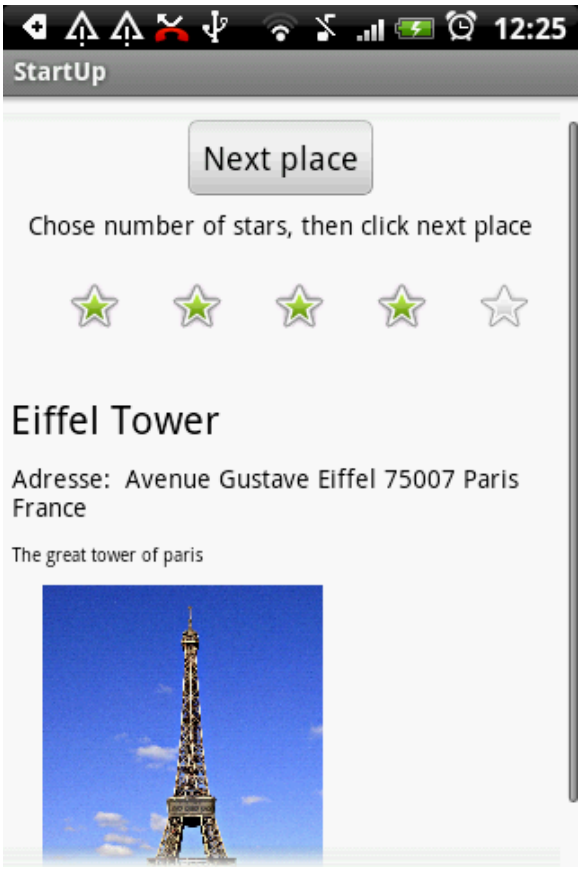


Figure 8.1: The user has recvied pro-active recommendations and the phone makes light vibration

Eight of the nine testers completed this task, and seemed to enjoy the process. Several users provided some anecdotal remarks of past holidays upon receiving familiar places to rate. There was initially some confusion in terms of if the user should rate a place he had not visited in the past. After it was made clear that the rating should be provided on general term that they might enjoy similar places, all but User 1 was able to provide ratings. User 1 did not complete the test due to repeatable network issues. As she had limited time we had skip this task. She ended up using an existing profile.
After completing this exercise the testers were asked to start up the MTSR-client in order to receive some recommendations. All the users quickly got some recommendations sent to their phone. All the test subjects understood when a recommendation was received. Several users commented about the result on map right away, and comments like

> "I guess it wants me to eat junk" *-user 9*

referring to a icon picturing hamburger and soda, were common.
After receiving the recommendations the users were required to use the touch screen to access information on the closest recommendation to their current position. All the testers identified the icon representing their position on the map, and most of them chose the nearest recommendation based on a visually comparing the distance from this icon to the various icons representing places on the map. A few referred to the numeric value in meters provided in the dialog boxes.
All the users assumed that they should push the icons on the map to find more information. There was however issues with getting the dialog box up on the screen. It proved challenging to put the correct amount on pressure to trigger dialog box. The main problem seemed to be that the user tried to use their nail rather than the whole surface of their finger. There were also some initial problems with the dialog box, once on the map, partly being drawn outside the screen. The user then had to pan the map in order to see rating, explanation and option buttons. This caused especially user 5 troubles as he thought the part of the dialog box he could see on the screen was all the information available. We chose to help him to get through the task. Nevertheless, after the initial trial and error, all the users seemed to easily get more information from the icons. Neither of the testers had any troubles with finding the address.
After finding the address the users were required to locate the list of recommendations from other users. This task was successfully completed by all. Several testers commented on this as a nice feature, but we got some requests for better ordering of the places. Users felt that the places in the list should categorized by type(sight, restaurant and hotel).
As a final task in this scenario the users were asked to locate the place they

just had found information on, by following the route provided by the system. This test revealed that the routes provided not always were optimal as small walk ways and steps were not were included. The testers were familiar with the campus and the city centre, and therefore tended not to follow the application naively. Short cuts were taken, and several users told that they would feel sorry for tourist that chose to base their route merely by the one indicated by the system.

8.3.2 Test scenario 2 - Actively search for information

In the second scenario we had put together tasks to illustrate how we believe a tourist will use MTSR to search for information.

1. Find the name of a restaurant that is recommended for you by using the search-feature.
2. Find the name of a hotel that is located maximally 2km for where you are standing using the search-feature.
3. Find a restaurant named "Dronningens Kebab" located in Oslo using the search-feature.

In the first task we wanted the users to find information on restaurants not yet received as an automatic recommendation. Before conducting the test we believed that this task would be very simple, but our first tests revealed problems with the "Sliding Search Window". All the users sought out the find-tab right away, but the users were also required to locate the handle at the bottom of the map in order to get the search window up on the map. In the three first tests we used the standard android handle icon for the purpose of indicating that the window was hiding outside the screen. Neither User 1,User 2 or User 3 was able to get window up without our help, as they simply did not notice it on the screen. We therefore decided to switch the handle icon to the magnifying glass in a black square pictured in chapter 5, before conducting more tests. After this change the problems with locating the search window disappeared. As for the search itself, the predefined eat, sleep and experience buttons seemed as intuitive constructs. Comments like,

> "Ok, it's already set to eat. I guess I just have to push search.- *User 2*"

were very common in relation to this.

Four users did not get any result on the first search, but three of these users quickly applied the "zoom out"-button to increase their search field.

User 7 had some problems locating the zoom-buttons. He had not paid any attention to them earlier tasks, and as they disappear once the map no longer is being focused it took close to 30 seconds before he managed to complete the task.

In the next task the users were required to find a hotel maximally 2km form their current position. We had deliberately removed all ratings from hotels in the area of the field experiments, so that searching for recommended hotel would yield no result. The user was therefore required to deselect "recommended for me" in order to get a result. The first three users had problems with this, as they were not aware of the option field in the first place. They found it eventually without our intervention, but they were undoubtedly confused about how this would influence their search. In light of this we decided to make some changes.
We made the "Only stuff recommended for me" bold and in a larger font. We also included a dialog messages explaining how selecting and deselecting the "Recommended for me" would influence the result from the search. This seemed to solve the problem, as the rest of the users could complete the task without any problems. There were no issues with setting the correct map boundary to only include hotels within 2 km radius.
The goal for the final task in this scenario was to investigate, if and how quickly users we able to search for information far beyond their current location by having them to locate a specific restaurant 350 kilometres away. Faced with this task all the test subjects expect user 6 started to look for short cuts for getting the map of Oslo. User 6 started to zoom-out and pan right and away, and found the restaurant by typing in "dronningen" in the search field. The other eventually followed a similar strategy. The subjects were unanimous in that they felt that this was cumbersome approach.

8.3.3 Test Scenario 3 - Using the travel plan

This scenario required the testers to get familiar with the functionality related to the travel plan through the following tasks:

1. Request a new travel plan
2. Read information about each of the places in the plan and based on this find two places you want to remove from the plan
3. Remove these places from plan
4. Find two new places you can add to the plan that you think you will like better than the ones you just removed

All the users managed to find a new plan without any problems. When browsing for information on places the touch screen still caused some prob-

lems for a few users, but we witnessed a significant improvement from the first the first scenario. The users were able to quickly go through elements in the map. When picking candidates for removal the user seemed to rely on different information. The overall rating was used by all for assessment, but only User 3, User 5 and User 9 seemed pay any notice to the explanations and similarities with the ratters this average was based on. Neither of the users had any problem with removing point of interest once they had decided.
The next step was to find new places to put into the plan. The users generally expressed that they were unsure where to find new information to the plan. All, except User 1 and User 3, eventually began using the "find-window" for this without our help. After learning that new information must be found from the "Find"-window, the users were easily able to find new content to their plan.

8.4 Result from interview sessions and general observation

8.4.1 Observations

During the course of the experiment all the users in our evaluation received one, or more, automatic request for rating. All noticed the pending rating while working in either the "Find" or "Travel plan"-window. Upon accessing the "Pending rating"-window most of the user expressed some sort of theory of why they had received it. Comments like the one below were common in relation to this:

> Ah, It knows that I went there -*User 8*

While other users seemed to believe that the pending ratings came from that they had browsed more information on the particular place in the map. All the users noticed the rating requests and chose to rate them.
The usability test did not require the user to explicitly make us of the rating buttons in the dialog boxes, but most of the users did this anyway. The rating button was usually applied upon finding a restaurant the user had a strong opinion from previous experience.
When it comes to pro-active recommendations, all the test subjects also received several of these during the experiment. The buzzing seemed to be an effective way of notifying about incoming recommendations, although user 9 initially thought it was an incoming text message. The general comment was that this was useful feature, but several of the testers pointed out that it should be able to be turned off.

8.4.2 The interview sessions

In the interviews all the users said that they enjoyed using the application, and that it would be a useful tool for them if they were visiting an unfamiliar city. The interactive map, combined with quick access to places with high average rating was the most highly proclaimed features. Nevertheless, several users also pointed out that one hour was too little time to get the required feel of the application to really have an opinion.
When asked what kind of information presented by system that was important when composing the travel plan the subjects were unanimous in that average rating was a useful assessment metric. Only User 3, User 5, User 7 directly referred to the relative average rating of similar users as valuable information in the process of picking places to visit. Not unexpected the same users expressed a mental model of the how the system worked in order to provide recommendations very similarly to the conceptual algorithm of collaborative filtering, like the one given by User 3:

> I guess it finds recommendations from the users that have rated similarly as me -*User 3*

On the other hand several users gave a mental model closer to content based filtering. These believed that the system used their ratings from the initial task of "Rate 10 places" to elicit their preferences for historical buildings, fast food and other features. Some user expressed a model in between, thinking that the "Rate 10 places" provided the system with a static model of their preferences that it used to compare with other users.
When we asked the user if they could affect the recommendations from the system, the user who believed in a static model gave large importance to initial process of rating 10 places. Many did however point to the ongoing process of rating new places as way of guiding the system to provide useful recommendations. In cases where the users referred to the rating process as a tool for influencing the system, we pushed on how often the user would make us of this if they were using MTSR on daily basis. In relation this several commented at they would do this frequently as long as it only required a push of a button. At this point many referred to the pending ratings window as a convenient construct. User 5 described how he would make us of it like this:

> I would use it[the list of pending ratings] during a break,while having coffee or something, to review the places I've been to...I guess only would rate the places that were really bad or really good - *User 2*

This kind of presumed usage seemed to be shared by half of the users in the test group. Nevertheless, the users that shared the static view of the recommendations process expressed that they most likely would never used it when pressed on their thoughts around the "Pending rating"-window.
When we asked about suggestion on how to improve our design all of the users, not surprisingly given the result of our usability test, commented that specific searches were difficult to carry out. As we mentioned in our account for "Scenario 3", the users wanted the system to suggest possible geographical locations where their searches would yield a result. Further, they wanted the map automatically adjust to the search result.
User 3 gave shared some interesting ideas on how he believed the user experience could be improved in relation to this. He suggested that the search field should suggest search strings by auto completing the user input so that user would not have to use the awkward virtual key board to any large extent. The auto completion should also include possible geographical matches in a hierarchical fashion. The match on top of the auto complete list should be closest place to the area currently shown on the map.
Another interesting suggestion was in relation to the "Sliding Search window". Users commentated that it would be very useful to be able to search only top ratted places. This should be reflected in a separate option field. A specific suggestion was to use three options: "Recommended for me", "Top rated" and "Everything".

Chapter 9

Discussion

9.1 Research questions

In the introduction to this thesis we asked how a personalized mobile tourism application should be designed in order to:

1. Allow tourists to easily find information on- and physically locate points of interests?
2. Facilitate transparency and user control?
3. Gather information on the user's preferences in way that is non-obtrusive, accurate and requires little effort from user?

9.1.1 Allow tourists to easily find information on- and physically locate points of interests?

We have developed a personalized mobile tourist application that included the following features to allow the users to easily find information and physically locate points of interest:

- Pre-defined information requests for eating,sleeping and experiencing as well the option to search freely
- A pro-active recommendations
- A interactive map to display and request information
- Meaningful illustrations of services on the map
- Explanation and average ratings as part of the recommendations of services as well as route and distance to each service

- Means to request and manage a travel plan

The design incorporating these features has been evaluated and the results is presented in the previous chapter. The evaluation seemed to show that a small test group, primarily consisting of technology students, were able easily solve typical tourist task with MTSR, meaning that they could complete the task they were presented with.
Nevertheless, our evaluation also revealed some issues with the design of MTSR. The most profound seemed to be the search field. All of the test users were struggling to type in and execute searches that deviated from the predefined requests.
We got several comments and suggestions on how to improve this aspect of the system. The most interesting and feasible was, in our eyes, the suggestion to include context aware auto completion as part of the search field. We believe this feature would greatly improve the efficiency and user satisfaction in relation to information retrieval. In relation to pro-active recommendation we believe that future implementations should allow the user to set the frequency, as well be able to turn the service off.

9.1.2 Facilitate transparency and user control?

The second overall design goal for MSRT was to facilitate transparency and user control thorough its design. The benefits from having a transparent system were accounted for in Part 1. We included the following as part of the user interface to reach this goal:

- Average rating parried with textual explanation of the correlation with other users
- A list of top-rated places from users with high correlation
- A check box to indicate whether the result should be filtered based on preferences
- Means to zoom-and pan the map away from current location

We believe that the user evaluation of MTSR show that these constructs did not seem obtrusive, or in any way hinder the user while carrying out the tasks. On the contrary, all the users seemed to use parts of explanations as an evaluating metric. This was done without us explicitly telling them do make use of it, but naturally occurred when the users were engaged in the open ended parts of the usability test.
However, only a few users expressed a sound model of how the system worked

to provide them with recommendations and understood how they could affect the recommendation process by rating. This can be problematic, as the only way the user can effectively use rating to alter the user model is by understanding the conceptual workings of the underlying filtering engine.
A more focused experiment involving a greater number of users must be conducted to be able to give answer to whether MTSR's approach to transparency truly is beneficial in terms of increased user satisfaction over a system not making use of this. However, as the constructs aimed at facilitating transparency did not seem to hinder the users and that some users gained an understanding of the system by using them; we believe that our approach is worth exploring in future implementations.
To override the adaption strategy the user was able to search for "'everything" rather than the default of only recommenced places. This construct proved simple to use, but based on the feedback we received we believe that an option to search for only top rated places also should be available. This will be easy to include in the interface, as well as in the recommender system module.

9.1.3 Gather information on the user's preferences in way that is non-obtrusive, accurate and requires little effort from user?

We enabled the user to explicitly provide the system with information on his or her preferences through:

- The option to rate each place of interest through an easily accessible "rating"-button
- Accessing a list of places that has been visited (pending ratings)

Our evaluation showed that user easily was able, and willing to provide the system with information through both these channels. All the users ratted actively even though this was not required, or even mentioned as an tasks in the usability test.
We did not tell the user that they would retrieve pending ratings. Neither did we explain how to access and provide opinions through the pending ratings interface. Nevertheless, we experienced that all the users found and made use of this functionality. We received no comments in relation to the feature being perceived as obtrusive, but rather that it represented a convenient way of saving past visits so that feedback could be provided in a timely manner.
We believe that MTSR provides the user with a convenient interface for providing direct feedback on points of interest. As direct feedback is an great advantage when building accurate users models[26], we believe that MSRT

could provide recommender with a reliable stream of feedback.

9.1.4 summary

To summarize, we believe that the design of MTSR is promising. However, it can strongly benefit from the following improvements

- A more user-friendly search, by using a auto completing search field that also give suggestions on geographical areas that will yield result
- Map that move accordingly to the keywords chosen from the search field
- Option to specify search strategy in terms of "search for everything", "search for stuff recommended for me"' and "search for top ratted points of interest"

These improvements should be included in the next development iteration.

9.2 Contributions

We have during in this project developed a frame work for delivering personalized tourist recommendations to a mobile platform . The framework has taken basis in research on location based services for personalized tourist information and interaction design in recommender systems. We have implemented constructs as pro- active recommendations and tour planning in the context of cutting edge smart phone technology, as well as introduced our own concept of pending ratings to simplify the user interaction with the filtering engine.
We tested this system on users, and based on the experiences from suggested improvements for the next iteration of development.

9.3 Further work

9.3.1 User acceptance testing

The evaluation method used in this thesis included few users and lasted for a short period of time. It can not give any valid answerers to whether MTSR would be accepted and used by tourists. Technology acceptance models can be applied to provide answer to such question by quantitative assessment of data from the target users. The so called Mobile services acceptance model

has development with the mobile computing in mind[11], and we believe that conducting an evaluation of MTSR applying this model on real tourists would be the natural next step.

9.3.2 Finding content

In this project we have used a test data base consisting of about 100 points of interests that have been manually inserted in the data base. MTSR will need much, and dynamic content by the useful. The architecture we presented in this thesis should be extended to include external content providers to facilitate this. We have made the data access independent from the logic layer in our framework. Channing the data source should therefore not affect the rest of the application.

9.3.3 Improving the the recommender system

The recommender system used in MTSR should be evaluated for accuracy and efficiency. We believe that the current implementation is not suited for supporting real users. We hope that the modular design of MTSR will enable further projects with the focus of providing accurate recommendations to make use of the MTSR framework, as it will allow for realistic user tests.

Bibliography

[1] JTS Topology Suite 2006. Jts topology suite, August 2006. http://www.vividsolutions.com/jts/jtshome.htm, accessed on 3.6.2010.

[2] Oracle Corporation and/or its affiliates 2010. Java ee, May 2010. http://java.sun.com/javaee/, accessed on 3.6.2010.

[3] Postgis 2010. Postgis, June 2010. http://postgis.refractions.net/, accessed on 3.6.2010.

[4] Gregory D. Abowd, Anind K. Dey, Peter J. Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, London, UK, 1999. Springer-Verlag.

[5] Olga Averjanova, Francesco Ricci, and Quang Nhat Nguyen. Map-based interaction with a conversational mobile recommender system. pages 212–218, 2008.

[6] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[7] Keith Cheverst, Nigel Davies, Keith Mitchell, and Christos Efstratiou. Using context as a crystal ball: Rewards and pitfalls. *Personal Ubiquitous Comput.*, 5(1):8–11, 2001.

[8] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Developing a context-aware electronic tourist guide: some issues and experiences. pages 17–24, 2000.

[9] Keith Cheverst, Nigel Davies, Keith Mitchell, Adrian Friday, and Christos Efstratiou. Developing a context-aware electronic tourist guide: some issues and experiences. pages 17–24, 2000.

[10] The Apache Software Foundation. Tomcat, June 2010. http://tomcat.apache.org/, accessed on 11.6.2010.

[11] Shang Gao, John Krogstie, and Per Anton Grans?ther. Mobile services acceptance model. *Hybrid Information Technology, International Conference on*, 0:446–453, 2008.

[12] Google. Android sdk, May 2010. `http://developer.android.com/sdk/index.html, accessed on 3.6.2010`.

[13] Google. Droiddraw, May 2010. `http://www.droiddraw.org/, accessed on 3.6.2010`.

[14] Google. Google maps api, May 2010. `http://code.google.com/apis/maps/, accessed on 3.6.2010`.

[15] Christian Heath, Paul Luff, and Guildford Cambridge. Collaboration and control: Crisis management and multimedia technology in london underground line control rooms. *Computer Supported Cooperative Work*, 1:69–94, 1992.

[16] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. Explaining collaborative filtering recommendations. pages 241–250, 2000.

[17] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, 2004.

[18] A. R. Hevner, S. T. March, J. Park, and S. Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–106, 2004.

[19] Annika Hinze and George Buchanan. The challenge of creating cooperating mobile services: experiences and lessons learned. pages 207–215, 2006.

[20] Annika Hinze, Petra Malik, and Robi Malik. Interaction design for a mobile context-aware system using discrete event modelling. pages 257–266, 2006.

[21] Katerina Kabassi. Personalizing recommendations for tourists. *Telematics and Informatics*, 27(1):51–66, February 2010.

[22] Axel Küpper. *Location-Based Services: Fundamentals and Operation*. Wiley, October 2005.

[23] 2001-2002 Sun Microsystems. Java ee, June 2002. `Core J2EE Patterns - Data Access Object, accessed on 3.6.2010`.

[24] Nicolas Mollet. Google maps icon project, May 2010. `http://code.google.com/p/google-maps-icons/, accessed on 3.6.2010`.

[25] Michael D. Myers and Michael Newman. The qualitative interview in is research: Examining the craft. *Inf. Organ.*, 17(1):2–26, 2007.

[26] David M. Nichols. Implicit rating and filtering. pages 31–36, 1998.

[27] Puntip Pattaraintakorn, Gregory M. Zaverucha, and Nick Cercone. Web based health recommender system using rough sets, survival analysis and rule-based expert systems. In *RSFDGrC '07: Proceedings of the 11th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing*, pages 491–499, Berlin, Heidelberg, 2007. Springer-Verlag.

[28] Ulrich Rabanser and Francesco Ricci. Recommender systems: Do they have a viable business model in e-tourism? pages 160–171, 2005.

[29] Al Mamunur Rashid, Istvan Albert, Dan Cosley, Shyong K. Lam, Sean M. McNee, Joseph A. Konstan, and John Riedl. Getting to know you: learning new user preferences in recommender systems. pages 127–134, 2002.

[30] Francesco Ricci. Mobile recommender systems. 2010.

[31] J. Ben Schafer, Joseph Konstan, and John Riedi. Recommender systems in e-commerce. pages 158–166, 1999.

[32] B. Schmidt-Belz, S. Poslad, and A. Zipf. Creation of user-friendly mobile tourism services. pages 36–42, 2001.

[33] Barbara Schmidt-Belz, Heimo Laamanen B, Stefan Poslad C, and Alexander Zipf. Location-based mobile tourist services - first user experinces. 2003.

[34] W. Schwinger, Ch. Grün, B. Pröll, W. Retschitzegger, A. Schauerhuber, Technische Universität Wien, and Wit Wissenschafterinnenkolleg Internettechnologien. Context-awareness in mobile tourism guides – a comprehensive survey. 2005.

[35] Hibernate Search. Hibernate search, June 2010. `http://www.hibernate.org/subprojects/search.html, accessed on 11.6.2010`.

[36] Rashmi Sinha and Kirsten Swearingen. The role of transparency in recommender systems. In *CHI '02: CHI '02 extended abstracts on Human factors in computing systems*, pages 830–831, New York, NY, USA, 2002. ACM.

[37] Hibernate Spatial. Hibernate spatial, June 2010. `http://www.hibernatespatial.org/, accessed on 3.6.2010`.

[38] Stefan Steiniger, Moritz Neun, and Alistair Edwardes. Foundations of location based services lesson 1 cartouche 1- lecture notes on lbs, v. 1.0. 2005.

[39] Kirsten Swearingen and Rashmi Sinha. Interaction for recommender systems. 2002.

[40] Nava Tintarev and Judith Masthoff. A survey of explanations in recommender systems. pages 801–810, 2007.

[41] Kirsi Virrantaus, Henry Tirri, Jari Veijalainen, Jouni Markkula, Artem Katanosov, Artem Garmash, and Vagan Terziyan. Developing gis-supported location-based services. *Web Information Systems Engineering, International Conference on*, 2:0066, 2001.

[42] Chien-Chih Yu and Hsiao-Ping Chang. Personalized location-based recommendation services for tour planning in mobile tourism applications. pages 38–49, 2009.

[43] Chien-Chih Yu and Hsiao-Ping Chang. Personalized location-based recommendation services for tour planning in mobile tourism applications. pages 38–49, 2009.