Context-Aware Places of Interest Recommendations for Mobile Users

Linas Baltrunas, Bernd Ludwig, Stefan Peer, and Francesco Ricci

Free University of Bozen-Bolzano, Piazza Domenicani 3, 39100, Bolzano, Italy
{lbaltrunas,bernd.ludwig,fricci}@unibz.it

Abstract. Recommender Systems (RSs) are software tools and techniques providing suggestions for items, such as movies, CDs, or travels, to be of use to a user. In general, a recommendation can be more compelling and useful if the context of the user is known. For instance, in a travel recommender, the season of the travel, or the group composition, or the motivation of the travel are all important contextual factors that, as a traveler normally does, should be taken into account by a system to generate more relevant recommendations. In this paper we show how a context-aware mobile recommender system for places of interest (POIs) selection can generate more effective recommendations than those produced by a non context-aware version, i.e., those normally provided to the city visitors by the local tourist office. Here we mainly focus on the HCI solutions and in particular in the explanation of the recommendations that are perceived by the user as an important element of the graphical interface.

1 Introduction

Recommender Systems (RSs) are software tools and techniques providing suggestions for items to be of use to a user [13]. It is a matter of fact that more compelling and useful recommendations can be identified if the context of the user is known [8] [9]. Here we adopt the definition of context provided by [8]: Context is "any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves". For instance, in a travel recommender, the season of the travel or the distance to a Place of Interest (POI) are important contextual conditions that should be considered before suggesting (or not) a tourist to visit that POI.

For this reason, context-aware recommender systems (CARSs) are gaining more and more attention [2], and in particular in the tourism domain [1] [16] [7] [3] [6] [12]. But their supposed advantage and the true impact of the context on the user acceptance of the recommendations must be assessed. Moreover, in order to adapt the recommendations to the user's context one must first elicit all the potential contextual factors that may (qualitatively) influence a travel decision. This knowledge can be obtained by referring to the vast consumer behavior literature, especially in the tourism domain [14]. But this list of factors can only be used as a starting point. In a necessary second step the quantitative dependency of the user preferences – his ratings for items – from each single contextual factor must be modeled. This dependency

A. Marcus (Ed.): Design, User Experience, and Usability, Pt I, HCII 2011, LNCS 6769, pp. 531–540, 2011.

model can be built by acquiring, for a user population, explicit evaluations for some of the POIs under several possible values of these contextual conditions. So for instance, one should acquire the evaluations of a museum, or the likelihood to visit such POI, when the users are traveling with or without children. In this example, "traveling with children" expresses a contextual condition that may be relevant for deciding if the visit to the museum is appropriate or not.

Such data is difficult to obtain because it requires a substantial user effort, since the users must provide evaluations (ratings) of items after having experienced them in several different contextual conditions. For instance, referring to the previous example, one should collect evaluations for a museum if visited with or without children. Hence, it is important to design suitable methodologies for simplifying this data acquisition step still offering enough knowledge about the users' preferences to generate effective recommendations. In a previous paper [5] we described an effective methodology for detecting the quantitative dependency of the user preferences from contextual conditions. The proposed methodology relies on a web based survey application where the users are requested to imagine a contextual condition and then to evaluate a POI taking into account the effect of the context on their decisions, i.e., if the context increases or decreases (or does not impact) the likelihood to select such a POI. With this data we have built a predictive linear model that generates for each contextual situation and POI combination a prediction of the relevance of this POI in that condition. This model is the core computational component of ReRex, a mobile iPhone application that enables the user to: a) select the contextual factors that the system must take into account in the generation of the recommendations; and b) browse the context-aware recommendations identified and displayed by the system.

The system was developed by addressing the previously mentioned challenges, in three steps: first the relevance of several important contextual factors were measured; then in-context ratings were acquired for a population of users; and finally a mobile application (ReRex), running on an iPhone, was designed, implemented and tested in a live user experiment. The first two steps of this process are not described here for lack of space; the reader can find a description of the methodology in [4]. Here we focus on the third step and we illustrate the results of a live user experiment where ReRex was compared to a similar system variant that supports an analogous HCI, but does not take into account the user context for the generation and the explanation of the recommendations. In this case the recommendations are sorted by their average rating among a population of visitors, which is the sorting usually adopted by the local tourist office when offering suggestions to the tourists. An important interaction aspect of ReRex is its explanation functionality (see [15] for a survey on the role of explanations in RSs). Using the above-mentioned predictive linear model, ReRex is capable to identify the most influential contextual condition for each particular recommendation request, and can therefore justify its proposed recommendations by mentioning this condition in the recommendation description. So, for instance, ReRex can detect that the visit to a museum is recommended because the day is rainy and visiting a museum is a good activity for a family in that situation. The live user experiment showed that ReRex's context-awareness can largely improve the system effectiveness and that the users prefer the context-aware variant with respect to the non context-aware one.

The rest of the paper is organized as follows. Fist we illustrate the ReRex user interface and in particular its context-aware recommendation functionality (Section 2). Then, in Section 3, we describe the evaluation methodology, based on a within group comparison of ReRex with a variant, obtained by removing its context-awareness capability. We report there the results of this evaluation and, finally in Section 4 we briefly summarize the outcome of this research and we mention some future work.

2 The ReRex User Interface

In a typical interaction with ReRex the user initially establishes the context of the visit. Using the system GUI the user can enable and/or set the values of context variables representing important contextual factors. The user can switch on/off some of these variables, e.g., the "Temperature" or "Weather" (see Figure 1, left). When one of these variables is switched on the recommender system will take into account their current values in the recommendation generation process. We observe that some of these values are automatically obtained from an external weather forecast service, e.g., the temperature or the weather conditions. Whereas for some other variables the user is supposed to enter a value (with an appropriate screen – not shown here for lack of space). For instance, the user can specify the group composition of the current travel (here called "companion"). In both cases the recommendations are adapted accordingly.

The full set of contextual variables considered in ReRex, their values, and whether they are automatically collected or not is provided in the following:

- Distance to POI (automatic): far away, near by;
- Temperature (automatic): hot, warm, cold;
- Weather (automatic): sunny, cloudy, clear sky, rainy, snowing;
- Season (automatic): spring, summer, autumn, winter;
- Companion (manual): alone, friends, family, partner, children;
- Time day (automatic): morning, afternoon, night;
- Weekday (automatic): working day, weekend;
- Crowdedness (manual): crowded, not crowded, empty;
- Familiarity (manual): new to city, returning visitor, citizen of the city;
- Mood (manual): happy, sad, active, lazy;
- Budget (manual): budget traveler, price for quality, high spender;
- Travel length (manual): half day, one day, more than a day;
- Means of transport (manual): car, bicycle, pedestrian, public transport;
- Travel goal (manual): visiting friends, business, religion, health care, social event, education, cultural, scenic/landscape, hedonistic/fun, activity/sport.

After the user has entered the specification of the contextual situation (see Figure 1, left) the system can be requested to provide recommendations. A short number of suggestions, namely six, are the provided (see Figure 1, right). If the user is not happy with these suggestions he can request "more" of them. This is achieved by

touching the corresponding button on the top right of the screen. The user is then requested to specify the type of POIs he is looking for (e.g., castle or art museum or else) and finally more POIs of that type are recommended.

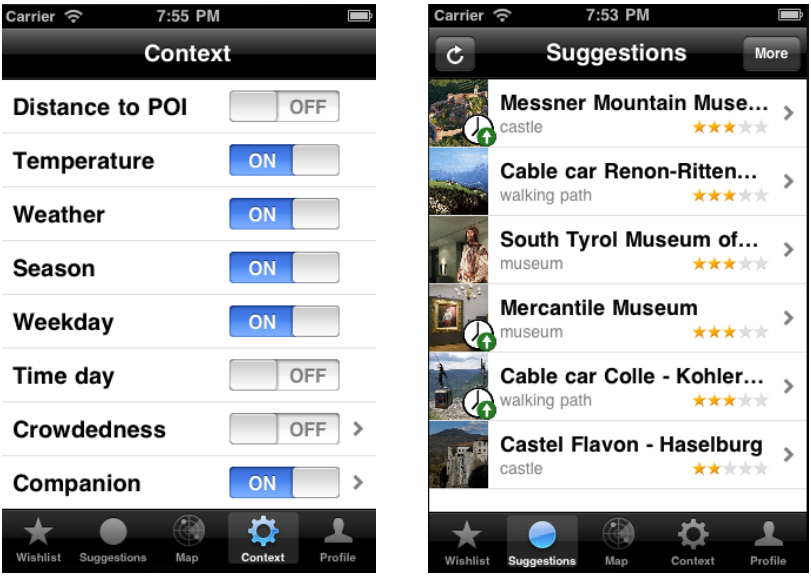

Fig. 1. ReRex screen for context management (left); the user can set (unset) the contextual conditions that the system must take into account in the recommendation generation. The second screen (right) shows a small set of ranked list of POIs recommendations.

In the suggestion list the user can touch any of these suggestions to access a more detailed description of the POI (see Figure 2, left). It is worth noting that some of these suggestions are marked with an icon showing a small clock and a green arrow. This means that these recommendations are particularly suited for the current context of the request as it was previously acquired. In the description of these recommendations (Figure 2 left) there is an explanation sentence like "This place is good to visit even if it is cold today". This refers to the contextual condition that was largely responsible for predicting the top rank for this item. Note, that cold weather condition could even decrease the rank of some items, i.e., their relevance for the current context. However, some items become more attractive than others (indoor museum in our case) if the weather is bad. The other items, i.e., those not marked with the clock icon, are suited as well for the current contextual situation. But we decided not to explain their relationship with the context to highlight and better differentiate those marked with the clock icon from the rest. This can be considered as a persuasive usage of the contextual information.

We have identified custom explanation messages for all the possible 54 contextual conditions listed previously. We note that even if more that one contextual condition holds in the current recommendation session, and all of them are actually used in the computation of the predicted score of each recommendation, nevertheless the system

exploits only one of them for the explanation. The contextual condition that is used in the explanation is the most influential one as estimated by the predictive model used by the recommender to predict the relevance (rating) of items in the current context. This design choice is motivated by a simplicity reason; we hypothesized that a single statement would be easily understood by the users and ultimately would produce the best effect on them. Naturally this issue, and more in general a better explanation functionality could be implemented in a future version of the system. In fact, as it will be illustrated in the next section, the quality of these canned explanations are not perceived by the users as strikingly good, indicating that a better explanation message could be generated.

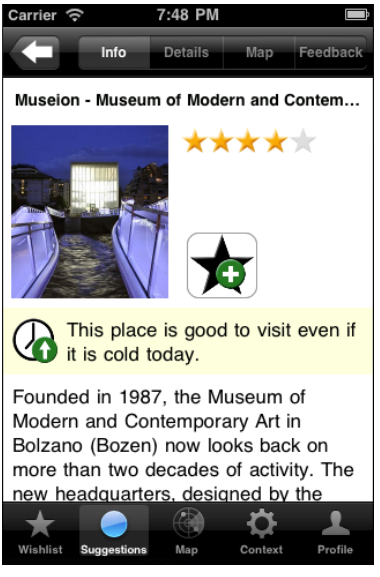

Fig. 2. ReRex screen for explanation visualization (left); the user can browse details of the recommended POI and read the explanation for the recommendation. The second screen (right) shows the position of the recommended POIs on a map.

Some additional functions have been implemented to enable the user to better exploit the system. The first one is called "wishlist" and enables a user to select any recommended POI and add it to a personal wish list. This is activated by touching the star like icon with a green plus shown in the details view of a POI (Figure 2, left). The user can also browse his wish list by touching the "wishlist" button on the bottom left of any screen. The wish list screen is very similar to the recommendations list visualization screen but has an important feature; if the contextual conditions are changing, either because the user is entering new values (e.g., he specifies that he feels "active" now), or because some "automatic" factor is changing (e.g., the weather), ReRex automatically computes a new ranked list and if a POI present in the wish list has increased (decreased) its predicted relevance score the system shows this information with a green upward (red downward) arrow, similarly to those used in the

"suggestions" screen (in Figure 1 right). The motivations of these changes are shown in a detailed view of the POI as in Figure 3. This is a useful and compelling characteristic of ReRex that was highly appreciated by the users.

The last function is a typical one for a recommender system, namely entering a rating for a POI. This is available by touching the "feedback" button (top right) that is shown in the POI details screen (Figure 2, left). If the user touches this button a new screen opens letting the user to enter a context-aware rating for the item. This information is sent to the server and will be exploited in the next recommendation requests.

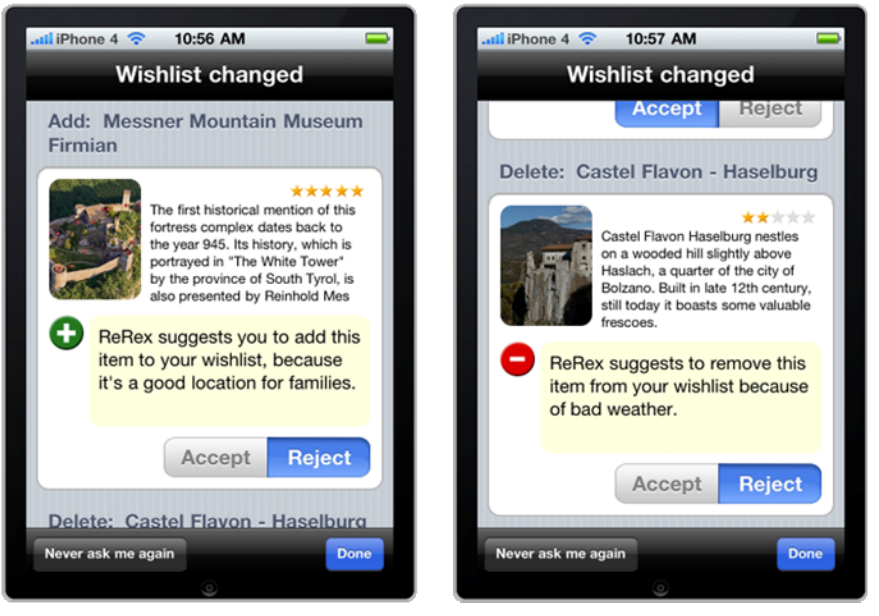

Fig. 3. ReRex screens for visualizing the effect of a contextual situation change in the wish list of the user

We conclude this section by briefly describing the software architecture of ReRex. It is a client/server application where the iPhone client is just managing the human/computer interaction. All the recommendation requests are sent to a server that stores the current user model (the user ratings) and the current request context and computes the recommendations. The client/server protocol is based on a custom designed set of XML files as described in [11]. The recommendations are computed by first computing the predicted relevance score for all the items, depending on the item and the context description. Then the items are sorted according to the predicted score and sent to the client. The complete description of the predictive model is provided in [4].

3 Experimental Evaluation

In order to measure the effectiveness of this approach we developed two variants of our ReRex mobile recommender system. The first is that described previously, the second variant is not context-aware, i.e., there is no possibility for the user to specify the current context, and the UI screen shown in Figure 1, left, has been removed. This variant does not offer any explanation for the recommendation, so the screen shown in Figure 2, left, does not display any explanation message, but it provides the other information about the POI exactly in the same way. The recommendations offered by this second variant are sorted according to the item average rating that we collected previously using a web-based interface (mentioned briefly in the Introduction). Hence, the recommendations produced by this second variant are essentially similar to those that the local Tourist Office makes to a visitor without any special knowledge of the context of the visit.

Using these two variants we could measure if the proposed context-aware system can improve the user perceived relevance of the recommendations with respect to those generated by a non context-aware model. To achieve this goal the test participants, 20 in total, tried out both variants of the system (within groups experimental model), in a random order, and executed, supported by each system, similar but different tasks. For instance, in one of these two tasks the user was instructed to:

> *"Imagine you are living in Bolzano. Suppose that it is a cold, rainy Wednesday and you are alone. Select a single point of interest of your choice and add it to your wish list. Then, suppose that your parents are coming to visit you. If needed, revise your previous selection and add another point of interest to your wish list."*

A second similar, but different task was assigned when the user was asked to try the second variant; hence each user tried both variants and each variant was used to solve one (randomly assigned) of these tasks. After the user completed the assigned task using one system, he was requested to fill out a usability questionnaire including the following statements:

- Q1: It was simple to use this system.
- Q2: The interface of this system is pleasant.
- Q3: The organization of information provided by the system is clear.
- Q4: It was easy to find the information I needed.
- Q5: The system was effective in helping me complete the scenario.
- Q6: It was easy to learn to use this system.
- Q7: Overall, I am satisfied with this system.
- Q8: I like using this system.
- Q9: This system has all the functions and capabilities I expect it to have.
- Q10: I am satisfied with the suggested points of interest.
- Q11: I can effectively find interesting suggestions when using this system.
- Q12: I understood the benefit of using the contextual conditions.
- Q14: I am satisfied with the provided contextual explanations.
- Q13: It was easy to specify the desired contextual conditions.
- Q15: I believe that the contextual explanations are useful.
- Q16: The contextual explanations provided by this system are clear.

The questionnaire was filled out again after the user tried the second system. Statements 12-16 were provided only when the questionnaire was filled out after testing the context-aware version. The user could express a level of agreement to the statement ranging from -2 (strongly disagree) to 2 (strongly agree). These questions were extracted, and slightly adapted to the scope of our investigation, from the IBM Computer System Usability Questionnaire [10]. Moreover, at the end of the interaction with the two variants we asked two additional questions:

- Q 17: which system the user preferred.
- Q 18: which system suggested more appropriate points of interest.

The average response to the first eleven questions, which were asked after the user tested both variants are shown in Figure 4. There is a clear preference of the users for the context-aware variant. However we must note that not all the observed differences are statistically significant (t-test); those significant are 5, 7, 9-11.

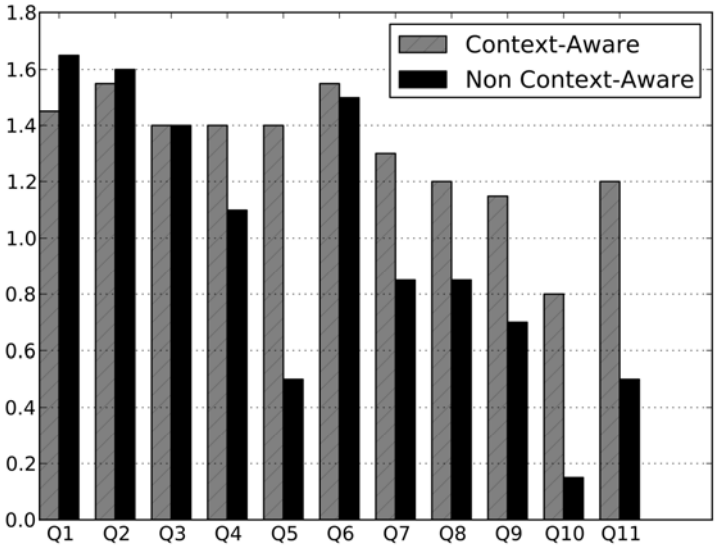

Fig. 4. Average of responses for the first 11 questions of the usability questionnaire. The response values could range from -2 (strongly disagree) to +2 (strongly agree).

We may observe that both systems are considered easy to use and their interface is nicely organized, but the context-aware system is:

- more effective in assisting the user for completing the task (Q5);
- it produces overall higher satisfaction (Q7);
- it is more complete, in term of functionality (Q9);
- it suggests POIs that more largely satisfy the user (Q10);
- it suggests more interesting POIs (Q11).

Regarding questions Q12-Q16, which were asked only after the context-aware variant was tested, the average results are: Q12 1.30; Q13 1.10; Q14 1.05; Q15 1.50; Q16 1.20. So, also with respect to the specific context-aware aspects the users were globally positive. We note that among these aspects the one that it is most interesting is the explanation support. Paradoxically the quality of the explanations scores low (Q14 1.05), but its importance scores higher (Q15 1.50). This indicates that the recommendation explanation is a very important component, and the user is particularly sensible to the quality of these explanations; but the current formulation of the system explanations can be further improved, as these are simple canned texts and no special natural language generation techniques have been used.

Finally, when the users were requested to directly compare the two variants (Q17 and Q18), 85% of the users preferred the context-aware version, while 95% of the users considered the context-aware recommendations more appropriate. In conclusion this evaluation, even if limited in the number of testers, provided a clear evidence that the context-aware recommendations produced by the system were more effective than those produced by the non context-aware version, i.e., the recommendations normally provided to visitors of the city of Bolzano by the tourist office.

4 Conclusions and Future Work

In this paper we have illustrated the importance of exploiting a traveler's contextual conditions when recommending POIs. The proposed mobile application offers to the user context-aware recommendations that are justified and explained by referring explicitly to the contextual situation in which the user will experience them. We have shown that the proposed system can offer effective context-aware explanations that are generated by identifying the contextual conditions that show the largest influence on the predicted relevance score (rating) of the recommended items. In a live user study we have compared a context-aware version and a non context-aware one. We have shown that the user acceptance and satisfaction is larger for the context-aware version and that the users prefer this version compared to another, with a very similar user interface, which does not consider the request context.

As future work we want to address two important issues, personalization and a better explanation functionality. Regarding the first point, we observe that ReRex does not personalize the recommendation, i.e., the recommendations are not based on the domain specific preferences of the user. The system ignores if the user likes more museums of sport activities, and the recommendations do not reflect these preferences. In fact, we do not ask the user to express this type of preferences, and this could also be perceived as an advantage, since it simplifies the human computer interaction. In a future work we want to better understand the role of personalization and contextualization, comparing a context-aware, but not personalized system, as that described in this paper, with a system that generates personalized recommendations, and possibly even combines personalization with contextualization. The second issue was mentioned already in the paper and refers to the measured low user satisfaction for the generated explanations. We want to improve the quality of the explanations exploiting advanced natural language processing techniques to better adapt the explanation to the type of the recommended item and using more information extracted from the predictive model.

References

1. Abowd, G., Atkeson, C., Hong, J., Long, S., Kooper, R., Pinkerton, M.: Cyberguide: a Mobile Context-Aware Tour Guide. Wireless Networks 3(5), 421–433 (1997)
2. Adomavicius, G., Tuzhilin, A.: Context-Aware Recommender Systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, V. (eds.) Recommender Systems Handbook, pp. 217–256. Springer, Heidelberg (2011)
3. Ahn, H., Kim, K.-j., Han, I.: Mobile Advertisement Recommender System Using Collaborative Filtering: MAR-CF. Technical Report of The Korea Society of Management Information Systems (2006)
4. Baltrunas, L.: Context-Aware Collaborative Filtering Recommender Systems. PhD Thesis Free University of Bozen-Bolzano (2011)
5. Baltrunas, L., Ricci, F., Ludwig, B.: Context Relevance Assessment for Recommender Systems. In: Proceedings of the 2011 International Conference on Intelligent User Interfaces. ACM Press, New York (2011)
6. Bellotti, V., Begole, J.B., Chi, E.H., Ducheneaut, N., Fang, J., Isaacs, E., King, T.H., Newman, M.W., Partridge, K., Price, B., Rasmussen, P., Roberts, M., Schiano, D.J., Walendowski, A.: Activity-Based Serendipitous Recommendations with the Magitti Mobile Leisure Guide. In: Proceedings of the 2008 Conference on Human Factors in Computing Systems, CHI 2008, pp. 1157–1166. ACM Press, New York (2008)
7. Cena, F., Console, L., Gena, C., Goy, A., Levi, G., Modeo, S., Torre, I.: Integrating Heterogeneous Adaptation Techniques to Build a Flexible and Usable Mobile Tourist Guide. AI Communcation 19(4), 369–384 (2006)
8. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5(1), 4–7 (2001)
9. Dourish, P.: What We Talk about When We Talk about Context. Personal and Ubiquitous Computing 8(1), 19–30 (2004)
10. Lewis, J.R.: IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. International Journal of Human Computer Interaction 7(1), 57–78 (1995)
11. Peer, S.: Real-Time Context-Aware Recommendations for Mobile Users. Bachelor of Science in Applied Computer Science Thesis. Free University of Bozen-Bolzano (2010)
12. Ricci, F.: Mobile recommender systems. Journal of Information Technology and Tourism 12(3) (2011) (in press)
13. Ricci, F., Rokach, L., Shapira, B.: Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, F., Shapira, L., Kantor, B. (eds.) Recommender Systems Handbook, pp. 1–35. Springer, Heidelberg (2011)
14. Swarbrooke, J., Horner, S.: Consumer Behaviour in Tourism, 2nd edn. Butterworth-Heinemann (2006)
15. Tintarev, N., Masthoff, J.: Designing and Evaluating Explanations for Recommender Systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, V. (eds.) Recommender Systems Handbook, pp. 479–510. Springer, Heidelberg (2011)
16. van Setten, M., Pokraev, S., Koolwaaij, J.: Context-Aware Recommendations in the Mobile Tourist Application COMPASS. In: Adapive Hypermedia 2004, pp. 235–244. Springer, Heidelberg (2004)