Context-Aware Recommendations in the Mobile Tourist Application COMPASS

Mark van Setten, Stanislav Pokraev, Johan Koolwaaij

Telematica Instituut, P.O. Box 589, 7500 AN, Enschede, The Netherlands
{Mark.vanSetten, Stanislav.Pokraev, Johan.Koolwaaij}@telin.nl

Abstract. This paper describes the context-aware mobile tourist application COMPASS that adapts its services to the user's needs based on both the user's interests and his current context. In order to provide context-aware recommendations, a recommender system has been integrated with a context-aware application platform. We describe how this integration has been accomplished and how users feel about such an adaptive tourist application.

1 Introduction

With several mobile technologies like mobile data networks (GPRS and UMTS), positioning systems (GPS), mobile phones and personal digital assistants (PDAs) getting more mature, it becomes possible to offer online services to people whenever and wherever they are. Such online services are especially useful for people in places they have never been to before. Apart from business travellers and truck drivers, a large group of such people consists of tourists. Often, tourists do not know their way, nor which restaurants, museums, shops, public services, etcetera are available to them. The number of potential places to visit can be quite overwhelming, especially in touristic regions. Adaptive systems can help a tourist to find places matching his interests and his current situation.

In this paper, we consider two such adaptive systems: recommender systems and context-aware systems. We describe their integration in a mobile tourist application and how users feel about adaptive systems providing context-aware recommendations. We start by introducing context-awareness and recommender systems and how these two types of adaptive systems enhance each other (Sect. 2). This is followed by an overview of our mobile tourist application COMPASS (Sect. 3). Sect. 4 describes the architecture of COMPASS and the underlying platform focussing on the integration of the recommender system and context-awareness system. Sect. 5 discusses the results of a survey on the usefulness of this combination according to possible users. Sect. 6 ends this paper with conclusions on context-aware recommendations.

2 Context-Aware Recommendations

Context is any information that can be used to characterize the situation of any person, place or object that is considered relevant to the interaction between a user and an application, including the user and application themselves [4]. Examples of contextual information are location, time, proximity, user status and network capabilities. A general definition of context-aware systems is given in [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 key goal of context-aware systems is to *provide a user with relevant information and/or services based on his current context*. This goal matches with the goal of recommender systems. Resnick and Varian [10] define recommender systems as systems that use opinions of a community of users to help individuals in that community more effectively identify content of interest from a potentially overwhelming set of choices. However, recommender systems do not only have to incorporate the opinions of other users, but may also use other methods, such as content-based reasoning. For this reason, we define recommender systems as systems capable of helping people to *quickly and easily find their way through large amounts of information by determining what is of interest to a user* [14]. Both context-aware systems and recommender systems are used *to provide users with relevant information and/or services*; the first based on the user's context; the second based on the user's interests. Therefore, the next logical step is to combine these two systems.

Context and interests can be used as hard or soft criteria in the selection of relevant services. Hard criteria limit the set of available services; those services that do not match a hard criterion are discarded from the set. Soft criteria are used to order the set of selected services or to present a relevance score to the user for each selected service. For example, location, by far the most exploited context factor, can be used to select only the services within a certain distance from the user (hard criterion); location can also be used to decrease the predicted relevance of a service the further away that service is located from the user (soft criterion). In recommender systems, the interests of a user are mostly used as soft criteria where the predicted level of interest is presented as a score, using for example a number of stars. However, interests can also be used as hard criteria by only selecting services that match the users' interests. In our application COMPASS, location is used as a hard criterion to select relevant services that are close to the user; the predicted interest of the user is used as a soft criterion, just like some other contextual factors (see Sect. 4.4).

3 The COMPASS Application

COMPASS is an acronym for COntext-aware Mobile Personal ASSistant and is an application that serves a tourist with information and services (ranging from buildings to buddies) needed in his specific context that are interesting to him given his goal for that moment. For example, a tourist expressing an interest in history and architecture is served with information about nearby monuments built before 1890. A tourist

expressing the wish to find a place for the night gets a list of hotels and campsites in and around town that match his preferences for accommodations.

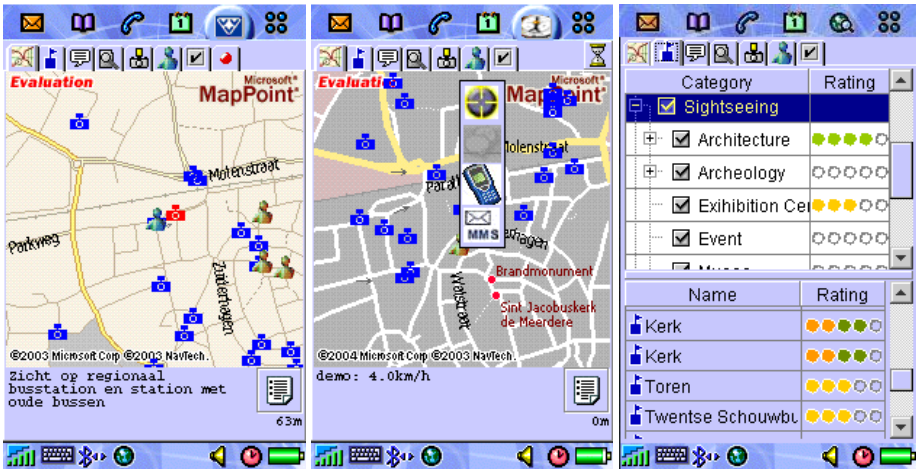

Fig. 1. Screenshots of the COMPASS application: objects near the user on the map, interacting with services offered by objects and a list of objects near the user with relevance scores.

After start-up, COMPASS shows the user a map of his current location. The location is either obtained from the mobile network or from other devices such as GPS receivers. Depending on the user's profile and goal, a selection of nearby buildings, buddies and other objects is shown on the map and in a list. The map and the objects shown are updated when the user moves or his profile or goal changes. Other context changes might also force the map to change. For example, an increase in the user's speed by starting to drive in a car causes the map to zoom out automatically as the user's notion of nearness can be defined by what he can reach in a certain amount of time. Clicking on objects on the map usually means interacting with services provided by that object (see middle image of Fig. 1), e.g. calling a buddy, reserving a table at a restaurant, or booking tickets for a show.

The application is built upon the WASP platform (see Sect. 4.2) that provides generic supporting services such as a context manager and service registry. The platform and system are based on web services technology combined with semantic web technology. Web services are believed to help the integration of diverse applications while the semantic web promises to increase the "intelligence" of the web, enabling richer discovery, information integration, navigation and automation of tasks. However, the current web services technologies, based on WSDL and UDDI, do not provide a means for building semantic context-aware services. Therefore we extended existing web service standards with additional semantics to enable more intelligent, semantic retrieval of services while taking into account the contextual information associated with the user's request.

The platform is open, which means that third parties can easily integrate their information and services with the platform; these services can then transparently be found and used by the population of COMPASS users. For example, an organization

that owns a collection of digitized old postcards wrapped its database with postcards as an internet-accessible web service, published the web service in the public service registry of the platform and related the web service's interface to the registry's ontology. The net effect is that all COMPASS users with an interest in such postcards are now able to view postcards depicting objects near their location instantaneously. Depending on the visualisation, they see a map of their environment with icons indicating the location depicted on the old postcards (see the left image in Fig. 1) or a thumbnail list of the postcards. Clicking on an icon displays the postcard, the date of the picture and a short description. This way, it is quite easy to recall the atmosphere of early times while walking through a street or neighbourhood.

The COMPASS application accomplishes this functionality by querying the service registry for search services that are bound to deliver objects related to the user's context. The underlying platform retrieves services matching the hard criteria of the user's context and goal. For example, for someone located in Enschede and looking for sightseeing attractions it delivers search services for museums, landmarks, architectural buildings, etc. Next, the relevant search services are queried to retrieve the objects matching the context's hard criteria, e.g. to be within a certain radius from the location of the user. The retrieved objects are then sent to the recommendation engine which scores each object based on the soft criteria, such as the user's interests and context. The retrieved objects and scores are then displayed on the map and in the list of objects (see Fig. 1). The openness of the platform underlying the COMPASS application makes it easily applicable in other domains as well.

3.1 Related work

There are related research projects in the tourist domain using adaptive systems. The Intrigue system [2] is an interactive agenda offered by a tourist information server that assists the user in creating a personalised tour along tourist attractions. This research focuses on planning and scheduling a personalised tour taking into account the location of each tourist attraction and the interests' of the user. Console et al. [3] created a prototype system called MastroCARonte, which provides personalised services that adapt to the user and his context onboard cars. This research focuses on the effects of having such adaptive systems onboard cars.

The research focus of the COMPASS system is on the open platform, which allows easy creation of context-aware personalised applications and the services that are part of such a platform, including a service registry, a context manager and a recommendation engine. The next section discusses this open platform with a focus on the context manager and recommendation engine.

4 System Architecture

In the discussion of the architecture of the WASP platform underlying COMPASS and the architecture of the COMPASS application itself, we focus on the retrieval of services taking into account context and user's interests, as the topic of this paper the

integration of context-awareness and recommendations. The overall architecture of the WASP platform and the COMPASS application is shown in Fig. 2.

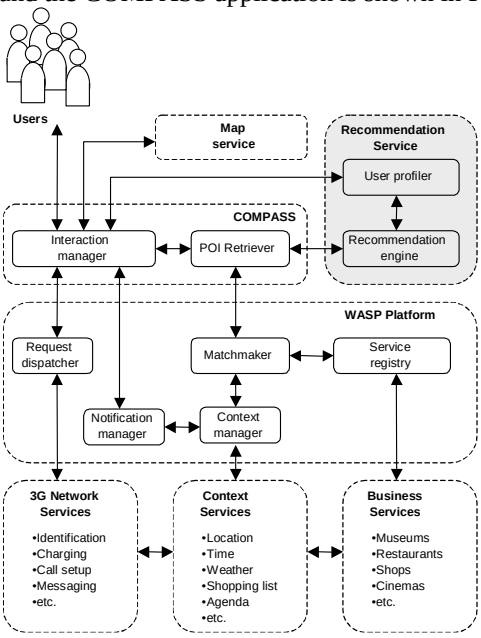

Fig. 2. The overall architecture of the WASP platform and the COMPASS application.

Four main groups can be identified in the architecture: third party services, the WASP platform, the COMPASS application and the recommendation service.

4.1 Third party services

The *3G (GPRS, UMTS) network services* provide network access capabilities, such as user identification, call setup, messaging, charging, etc. These network capabilities are accessible via web services interfaces and offered by mobile network operators.

The *context services* provide information about the context of a user, e.g. the user status (free or busy), his location, etc. Some of this information is obtained from the 3G network via web services. This group includes both services that provide information about the user such as his shopping list or his schedule, as well as services that are independent from the user but which might be relevant when selecting services, e.g. weather or traffic information services.

Business services are those services that offer information and services for an application build on the platform. In the COMPASS application these are businesses that offer so-called points of interest (POI): museums and their catalogues, monumental buildings and historical information associated with them, restaurants and their menus, shops and their current promotions, hotels with reservation services, digitized old postcards, etc.

4.2 WASP platform components

The *Request dispatcher* is a component responsible for forwarding user requests to the appropriate 3G-network platform. This way, users can switch transparently to different network operators or, for instance, use different messaging services.

The *Notification manager* provides functionality for applications to subscribe and receive notifications when the context of a particular user changes. For instance, when a user moves around in a city, his location changes. The notification manager notifies the application about this change and provides the new location of the user. The application can then adapt itself to this change in the user's location.

The *Context manager* retrieves information about the user's context by contacting the appropriate context services (see Sect. 4.1). It is also responsible for aggregating the context or deriving new context based on domain specific rules. For instance, the context manager can infer whether a user walks or drives given the speed of the user and the geographical properties of his location (city street, highway, sea or river, etc) or simply from the fact that his phone is attached to a car kit. The context manager is also responsible to update the notification manager on changes in the context.

The *Service registry* contains information about the services provided by third parties. To improve the semantic of service descriptions we use semantic web technology, notably OWL [8], to create additional annotations of service elements. This way, the platform enables service providers to formally describe their services in detail and to bring those service descriptions in correspondence with existing ontologies. On the other hand, it enables search services to perform a subtle search, by using constraints, relations between concepts, approximate matches and semantically rich queries [9], which delivers a more manageable result set.

The *Matchmaker* uses the service registry to discover the services that match the request received from an application (in this case the POI retriever of COMPASS, see Sect. 4.3). Once services are discovered, based on their types, capabilities and models, the matchmaker component filters out the services that do not match the hard criteria set by the application. To perform this action, the component uses the context ontology and domain-specific rules provided by the application.

4.3 The COMPASS application

The *Interaction manager* is a server side component responsible for finding the most appropriate way to communicate a user's request and assist the interaction of the user and the client side application (on the mobile phone, PDA or other device). For example, if a user clicks on a POI representing a restaurant, the interaction manager can, for example, automatically retrieve the restaurant's menu and present it to the user or prompt to setup a phone call in order for the user to make a reservation.

The *POI retriever* receives a request from the interaction manager when the user context changes or from an action by the user. It creates a search request that is sent to the matchmaker component. After the matchmaker component returns the list of POIs matching the issued request and hard criteria of the user context, the POI retriever sends this list together with the user's identity and the context information to the recommendation service, which assigned scores to each POI indicating the predicted

relevance of the POI for the user (see Sect. 4.4). The POI retriever than sends the list of POIs with scores to the client side application, which displays the POIs.

The COMPASS application also uses external *map service*, such as Microsoft Mappoint [7] for regular maps, a map service providing aerial photographs and a map service providing old cadastral maps. These web services are used to offer dynamic and interactive maps, providing navigation support, etc. COMPASS allows the user to switch between the various types of available maps, while keeping all other functionality, such as displaying POIs on the map and services associated with POIs.

4.4 The recommendation service

The *recommendation engine* uses multiple prediction strategies to predict how interesting each POI is for the user. A prediction strategy selects and/or combines multiple prediction techniques by deciding which prediction techniques are the most suitable to provide a prediction based on the most up-to-date knowledge about the current user, other users, the information for which a prediction is requested, other information items and the system itself [14]. Used prediction methods include social filtering [12], case-based reasoning (CBR) [11], item-item filtering [5] and category learning [13]. For different classes of POIs, different prediction strategies can be defined in the engine. As the semantics of POIs are described by an ontology, the recommendation engine is aware of the class hierarchy of each POI. For example, a Chinese restaurant is an Asian restaurant, which is a restaurant, which is a place to eat or drink, which is a POI. This means that the engine can select a prediction strategy appropriate for each class of POI. If a prediction strategy exists for the actual class of a POI that strategy is chosen, otherwise the engine moves up the class hierarchy until it finds a parent class that has a prediction strategy associated with it. In our hierarchy, POI is the root class, which has a default prediction strategy assigned to it.

For COMPASS, prediction techniques have also been developed that base their predictions on contextual factors; e.g. one technique predicts the relevance based on the time past since the last time the user visited a POI of that class. The more recent the user has been in such a POI, the lower the predicted relevance. This technique has been used in prediction strategies for POIs such as restaurants and museums. The time passed between the last visit and the current time is used as a sort of "linear decay time" for the predicted relevance. This is based on the idea of "Yesterday, I ate in a Greek restaurant, so today I will probably want to eat somewhere else." The rate in which the predicted interest returns to its full strength differs per user per type of POI.

The *user profiler* maintains the profiles of all users. It is used by the recommendation engine to retrieve and store knowledge about users, such as the interests of users and ratings provided by users. The interaction manager can also directly access the profile manager; this way, the interaction manager can also store user preferences or it can retrieve (parts of) the user profile and present it to the user.

The recommendation service is not part of the WASP platform as some prediction techniques are domain dependent or need to be tuned to specific domains, e.g. a similarity function had to be defined for the CBR-based prediction technique that compares two POIs with each other and returns a similarity score. However, the recommendation service is also not part of the COMPASS application; this allows

other applications in the tourist domain to use the same recommendation service. However, the WASP platform contains the generic parts of the recommendation service: prediction techniques for which only the domain specific parts still have to be implemented and a mechanism for defining prediction strategies that combine the various prediction techniques. Each instance of a recommendation service now only needs to implement the domain specific parts, such as the similarity function for case-based reasoning, define the prediction strategies and associate these strategies to the different object classes for which predictions need to be generated.

5 User Experience

So far, this paper has shown the possibilities of the combination of context-awareness and recommender systems and a way to integrate the two. However, it did not address another important question: how useful do users perceive context-aware recommendations? We investigated the usefulness of context-aware recommendations by performing a survey amongst 57 people consisting of 23 females and 34 males. The participants ranged in age from 10 till 70 and had a wide variety of backgrounds and professions.

The survey was an unsupervised online survey. Participants were taken through the usage of context-aware recommendations in COMPASS. They were lead via a scenario and screenshots through the various aspects of recommendations: searching for a type of POI, seeing a list of found POIs near the user's location with relevance scores, providing feedback on the recommendations, seeing the effects of providing feedback, seeing the effects on the recommendations of trying to visit similar POIs after a few days, etc. The survey asked participants both about the perceived usefulness of the context-aware recommendations (quantitatively) and why they felt that it was useful or not (qualitatively). A survey was chosen over real usage in order to provide each user with the same adapted system; large variations in context could have influenced the opinions of users too much for results to be comparable.

To avoid biasing the results with generic interests of participants in certain POIs, we focused the survey on restaurants. We assumed that everybody visits a restaurant every now and then and that most people do not have a positive of negative interest towards restaurants in general; museums for example could have biased the results, as some people do not like to visit museums at all.

Two context factors were included in the survey, namely location and time. Location was implemented by showing only restaurants in the centre of the city in which the users was supposed to be. Hence location was used as a hard criterion. However, time was used as a soft criterion and combined with the predicted interests; time was used to determine the last time the user visited a restaurant of the same type and by temporarily decreasing the predictions based on this time period.

Table 1 shows the quantitative results of the perceived usefulness in using predictions and the added context-aware factor 'last time visited' in the COMPASS application.

Table 1. Usefulness of context-aware recommendations

	Not useful at all				Very useful		
	-2	**-1**	**0**	**1**	**2**	**Average**	**St. Dev.**
Predictions	8.8%	8.8%	19.3%	35.1%	28.1%	0.65	1.232
Time Decay	24.6%	19.3%	14%	31.6%	10.5%	-0.16	1.386

The results show that most people believe that using predictions based on the user's interests is useful. For about half of the participants (50.9%), the addition of 'last time visited' had a negative influence on the perceived usefulness; the difference between the perceived usefulness of only predictions and the perceived usefulness of predictions combined with 'last time visited' was negative (for 36.8% the perceived usefulness stayed the same, and for 12.3% perceived usefulness increased).

We believe there are two possible reasons for the decrease in perceived usefulness. The first is that 'last time visited' is not a good additional context factor; combining other context factors with the predicted interests may increase or not influence the perceived usefulness, such as average costs of eating in the restaurant. However, a lot of the comments made by participants indicated another reason: they were about "the application becomes too intelligent", "I can think and decide for myself" and "it makes the application too complex". This indicates that some people do not like an application like COMPASS to take too many contextual factors into account in recommendations; they want to be able to decide for themselves which factors are important when selecting a POI. This desire for being in control by users is similar to the findings of Alpert et al. [1] in adaptive e-commerce systems.

6 Conclusions

In this paper, we discussed the combination of context-awareness and recommender systems and how this has been applied in the mobile tourist application COMPASS. We discussed how the two fit together and how they have been integrated in the architecture of our platform and application. A user survey indicated how useful people perceive an adaptive tourist service that recommenders points of interests by taking into account the user's interests and contextual factors such as last time visited.

Context-awareness and recommender systems can enhance and complement each other; they both help users in finding relevant and interesting objects, ranging from information and services to points of interests, based on their interests and current context. However, one has to be careful with such adaptive services. Although most people like and see the benefits of recommendations and context-awareness, there are people who may object when systems start to include too many factors in their predictions; they prefer to be able to think and decide for themselves, instead of having systems thinking and deciding for them. A simple solution may be to allow users to specify themselves what type of knowledge about the user or contextual information should be taken into account in recommendations, make the adaptation more scrutable [6]; e.g. "do recommend items based on my interests, location and prices, but do not include last time visited."

Acknowledgements

This research is part of the Freeband project WASP (http://www.freeband.nl), the PhD project Duine (http://duine.telin.nl) and the PhD project Seine at the Telematica Instituut (http://www.telin.nl). The authors like to thank the participants of the survey and Mettina Veenstra, Rogier Brussee and Tom Broens for their helpful comments.

References

1. Alpert, S.R., Karat, J., Karat, C.M., Brodie, C., Vergo, J.G.: User Attitudes Regarding a User-Adaptive eCommerce Web Site. In: User Modeling and User Adaptive Interaction 13, 4 (2003) 373-396
2. Ardissono, L., Goy, A., Giovanna, P., Segnan, S., Torasso, P.: Ubiquitous User Assistance in a Tourist Information Server. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds) Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the second International Conference, AH 2002, Málaga, Spain, May, Springer-Verlag, LNCS 2347 (2002) 14-23
3. Console, L., Gioria, S., Lombardi, I., Surano, V., Torre, I.: Adaptation and Personalization on Board Cars: A Framework and Its Application to Tourist Services. In: De Bra, P., Brusilovsky, P., Conejo, R. (eds) Adaptive Hypermedia and Adaptive Web-Based Systems, Proceedings of the second International Conference, AH 2002, Málaga, Spain, May, Springer-Verlag, LNCS 2347 (2002) 112-121
4. Dey, A. K.: Providing Architectural Support for Building Context-Aware Applications. Ph.D. thesis, College of Computing, Georgia Institute of Technology (2000) Online: http://www.cc.gatech.edu/fce/ctk/pubs/dey-thesis.pdf
5. Herlocker, J., Konstan, J. A.: Content-Independent Task-Focused Recommendation. In: IEEE Internet Computing, 5 (2001) 40-47
6. Kay, J.: A Scrutable User Modelling Shell for User-Adapted Interaction. Basser Department of Computer Science, University of Sydney, Australia (1999) Online: http://www.cs.usyd.edu.au/~judy/Homec/Pubs/thesis.pdf
7. Micosoft Mappoint, http://www.microsoft.com/mappoint
8. OWL, Ontology Web Language, http://www.w3.org/2001/sw/WebOnt
9. Pokraev, S., Koolwaaij, J., Wibbels, M.: Extending UDDI with context-aware features based on semantic service descriptions. The 2003 International Conference on Web Services (ICWS 2003) Las Vegas, USA (2003), 184-190
10. Resnick, P., Varian, H.R.: Recommender Systems. Communications of the ACM 40 (1997) 50-58
11. Riesbeck, C. K., Schank, R.: Inside CBR. Lawrence Erlbaum Associates, Northvale, NJ, USA (1989)
12. Shardanand, U., Maes, P.: Social information filtering: algorithms for automated "Word of Mouth". In: Proceedings of Human factors in computing systems 1995 (New York, USA). ACM (1995) 210-217
13. van Setten, M.: Experiments with a recommendation technique that learns category interests. In: Proceedings of IADIS WWW/Internet, Lisabon, Portugal (2002) 722-725
14. van Setten, M., Veenstra, M., Nijholt, A., van Dijk, B.: Case-Based Reasoning as a Prediction Strategy for Hybrid Recommender Systems. In: Proceedings of the Atlantic Web Intelligence Conference, Cancun, Mexico, Springer-Verlag, LNAI 3034 (2004) 13-22