Architecture for Context-Aware Pro-Active Recommender System

H. Al Tair, M. J. Zemerly, M. Al-Qutayri
College of Engineering, Khalifa University
Sharjah, PO Box 573, UAE

M. Leida
Etisalat-BT Innovation Centre (EBTIC)
Abu Dhabi, UAE

Abstract

This paper presents architecture of a context-aware pro-active recommender system. The system uses contextual information in order to provide recommendations that are more suitable to the particular individual user. Reduction-based theory has been used in order to be able to use the contextual information besides the user and item components of traditional two dimensional recommender systems. The proposed recommender system provides recommendations pro-actively by using multi-agent technology. The inference engine of the system uses conditional probability and multi-attribute theory in the decision making of what recommendations to be provided to users.

1. Introduction

Advances in information communication technologies continue to affect many aspects of our lives and the way we execute tasks. Paying bills, booking travel packages … etc can all now be done online. The availability of smart phones and other mobile devices is adding flexibility to the execution of such tasks and creating opportunities for interactive applications that meet the demands of mobile users. For example, traditionally, holiday makers try to find suitable travel packages for their trips by visiting a number of travel agencies. The introduction of electronic travel systems improved the search process as an individual can plan the trip on his/her own. The process was further enhanced by incorporating a recommender aspect to the travel system. Such systems have now the capability of making suggestions such as places to visit and best hotel deals and hence reduce the time spent searching the internet to organize a trip. Such systems can be enhanced by using contextual information such as: type of the trip, time, location and user preferences. It will help in making the recommendations for the tourists more suitable and closer to their actual needs. For instance, a user might need to change the trip schedule due to various reasons. Let us assume that the user is supposed to have dinner in Abu Dhabi yet according to the time (dinner time) and his/her current location is Dubai. For some reason the user will change the plan and s/he will decide to stay in Dubai for that night. Obviously, s/he might want to search for restaurants in Dubai then. In such case, it will be helpful if the system can detect such situation and provide the user with a list of restaurants in Dubai (the new location) according to his interests and preferences of restaurants.

On the other hand, a recommender system for users on the move (such as tourists) would like to have a mobile system that would recommend to them places on the spot and pro-actively taking into considerations the user's needs and preferences. However, the pro-activity aspects of these systems is still lacking and in need of improvement. Tourist recommender systems provide recommendations that a tourist might be interested in according to his/her preferences. Yet, the problem is the number of steps the user has to go through in order to get the appropriate recommendations. A lengthy process is obviously not compatible with the expectations of tourists who are on the move and should receive recommendations based on their contextual information without much intervention and in a pro-active way.

In order to achieve pro-activity in the recommender system, agent technology is integrated alongside the use of agents' properties such as pro-activity [1, 2]. Agents can be used to receive and filter information [5], allowing them to work efficiently alongside the recommender system. As the tourism domain contains complex information and the process varies from collecting services from different servers according to a user's context information to providing him/her with the most suitable recommendations, it is important to manage this process in an efficient way. This can be achieved using a multidimensional rating approach as described in [3].

This paper proposes alternative design architecture for a traveler recommender system that will use proactive multi-agents to achieve its objectives. It requires an absolute minimal level of interaction with the end user, as it provides services to the tourists based on their profiles and the contextual information. Following this introduction the paper is organized as follows. Section 2 reviews significant research and commercial related work. Section 3 discusses the requirements of the design and then describes the system design architecture. Section 4 discusses the inference engine of the proposed architecture. Sections 5 and 6 present the

system testing and evaluation. Finally section 6 concludes the paper and highlights the future work of this research.

2. Related Work

A system is context-aware if it was designed and implemented to use a set of information as regard a situation of an object (place or person). The information can be a number of properties of the surrounding environment of this object in a certain situation [6]. Most context-aware systems are implemented for handsets since they allow using information of different situations anytime and anywhere [7]. Context-aware systems are applied widely in different fields and one of them is tourism. It is useful to have a tourist guide at hand and on the move.

The multi-dimensional recommender system described in [3] uses the contextual information to provide recommendations accordingly. Yet, still the pro-activity aspect of the system is missing. This approach can be applied in the tourism domain since it can provide recommendations based on the contextual information. Agent technology has been used for tourism recommender systems that do not consider the contextual information for the purpose of making it more efficient and add properties such as the pro-activity to them. For instance, the multi-agent recommender system in [8] has proved that using rules is the most suitable approach for tourism since different cases could be considered for different users of different interests. Moreover, in [9] a reasoning maintenance system is included in the multi-agent recommender system. This reasoning approach facilitates the agent in their search for the components of travel packages following the user's request for travel recommendations. Another example of the benefits of the multi-agent recommender system for tourism is [10] where it was proved that the multi-agent design can lead to a more scalable recommender system. The system explicitly takes the user's preferences and profile information in order to provide recommendations for tourist attractions that are suitable to that user.

However, these multi-agent recommender systems lack the ability to use the contextual information of the travelers. They do not build profiles of the users, nor do they update them. In addition to that, the pro-activity of these systems remains limited to that specific agents' layer of these systems. The users then have to request for each service. Therefore, in this paper we propose integrating the multi-dimensional approach [3], where contextual information is used, with multi-agent system and apply it for tourism.

3. System Design

Figure 1 shows the context diagram of the system that has the inputs and the outputs. As illustrated, there are four inputs to the system: profile of user, username and password, the trip details, and the selections of recommendations. On the other hand, there are four outputs of the system as well: hotels recommendations, the schedule of the trip which has the restaurants and events recommendations, the maps are generated as well based on the recommendations, and finally the reminders and notifications of the schedule activities.

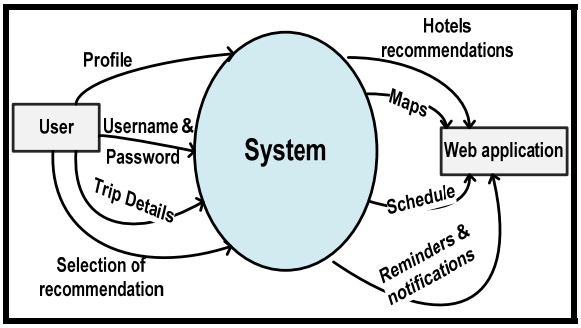

Figure 1. Context Diagram

The system was designed to be user-friendly, easy to use, and able to maintain, and handle different services. The simulation prototype is currently a web-based recommender system developed using JSP (Java Server Pages) and servlets. The language used for the implementation is JAVA as it is suitable for heterogeneous environments. The agents environment used is JADE (Java Agents Development Environment) v3.6. Additionally, the system provides a UAE map (using Google map) to visualize the places to visit that are recommended to the user. Since the system is a multi-agent system, the agents are centralized in the main server and from this server the agents collect information from different sources of information on the web.

The system is designed to fulfill the following functional requirements:

1. Build a user's profile from scratch and/or update it with his/her preferences.
2. Use minimal information from the user.
3. Request missing information that is required for certain recommendations only when absolutely necessary.
4. Detect a user's location frequently.
5. Refine recommendations based on a user's selection or context.

In order to realize the system, the architecture is divided into four layers as illustrated in Figure 2.

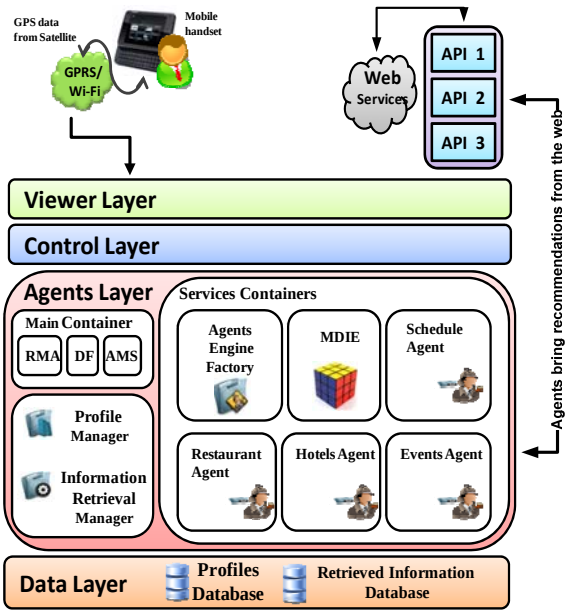

Figure 2. Design Architecture of the System

From bottom to top they are organized as follows:

1. Data layer: this layer is responsible for storing the data such as the user's profile and the information of services that are brought from the web. The retrieved services information database is cleaned frequently. The information is kept for certain time to be re-used in case there is a need to refine recommendations.

2. Agent Layer: this layer is the core of the system and contains the agents' environment. The main part of this layer is the set of agents which are created through using the Agents Engine. Each agent is specialized in performing one task or service provided by the system. As shown in Figure 1, these services range from hotel information, restaurant information or event information. Each agent type will use its own logic which is based on rules. Agents responsible for recommendations use the multidimensional (MD) [11] component to store the ratings of their different recommendations. The *schedule agent* is in charge mainly of taking the recommendations and building up the schedule. Finally, you have the Information Retrieval Manager and the Profile Manager components.

3. Control Layer: This layer facilitates the transition of the information from the user terminal to the agent's layer and vice versa.

4. Viewer Layer: This layer provides a friendly user interface through which the user would interact with the system.

As we pointed out previously there are 3 types of specialized agents: hotels, events/places to visit and restaurants. The hotel agent gathers information about hotels in the UAE (their name, location, stars rating, and the highest price in American dollar) from the services provided by *www.kayak.com*. The events and places to visit agent retrieves information from the services provided by *www.eventful.com*. Due to the limitation in resources providing information about restaurants in the UAE, we have created a table with different restaurants in different cities that were brought from *www.yellowpages.ae*.
In order to serve a large number of users we assume that, since the system is distributed, there are a number of servers which would work as a centralized unit in order to prevent an over-load of the server that contains the agents.

4. Inference Engine

In this section, the techniques that were used for the inference engine of the system are discussed. Figure 3 illustrates the steps that the inference engine of the system works based on. The inference engine of the system consists of three steps. Each one of the steps has inputs and outputs. Outputs of some of the steps can be considered as inputs to others. Each one of the steps is going to be explained in the next section.

4.1 Multi-dimensional Rating Approach

According to [12] the multidimensional rating approach is suitable for dealing with complex information in order to provide suitable specific recommendations. The architecture of the multi-agent recommender system we propose in this paper is based on this approach. Agents provide recommendations of the travel services (hotels, restaurants and events to attend) based on this approach. Each one of the services is considered as an item (product).

The dimensions considered for the recommendations are the *services*; *time* (which is essential since the system is building a profile for the users that evolves over time); and the *users* (which are the final consumers of the system). Each dimension has certain attributes that define it. For example, the restaurant dimension is defined by: the average price of the meal, the location (how far it is from a user's location), and the type of food served. The attributes of the time dimension are: ID and slot of time (morning, afternoon, or evening), while the attributes of the users dimension are: name, date of birth, interest, and marital status.

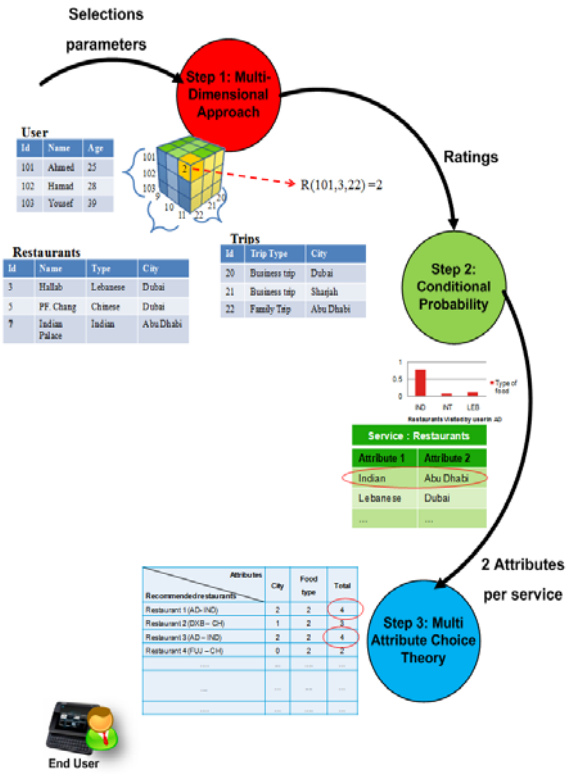

Figure 3. Inference Engine Steps

The combination of attributes within these dimensions defines the space of the recommendations. These are organized in recommendation cube B as it is illustrated in Figure 4. As it is also shown, cube (B) is limited to restaurants recommendations. There are other cubes for the other items (hotels and events). These cubes can be combined all together to formulate the space of recommendations for the whole system as in cube (A) in Figure 2. Cube (B) actually consists of a number of small cubes, similar to cube (C), which contain the rating for a specific recommendation. For instance, user X (X, 25, sports, single) selected to go to a restaurant (affordable price, closer to user's location, Italian food) in the evening. This preference for "Italian food", 'close location' and 'evening' is then stored in the history and profile of the user. The rating can be formulated as the following:

$$R: Users \times Items \times Time \rightarrow Rating \qquad (1)$$

4.2 Conditional Probability

The feedback of the recommendations provided is taken from the users usually explicitly in a way that users would be asked to rate those recommendations [13]. However, since a major requirement of the system is to achieve pro-activity through minimizing interaction with the user, it was decided to make the feedback optional. The user is not obliged to rate a recommendation that was given by the system. As discussed in [2] implicit feedback can be taken from the behaviors of the users, e.g.: selections, duration, and deletion. However, since the interaction between the user and the system will be reduced to the strictly necessary, in our case is mainly reduced to consider as feedback the selection of the recommendations which are proposed to the user.

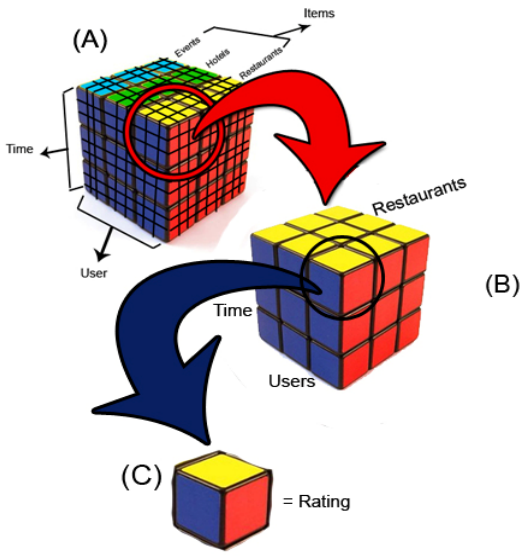

Figure 4. Using Multidimensional Rating Approach [11]

In order to overcome this problem, it was decided to use the conditional probability of a recommended place or type of restaurants for instance, based on the history of the user. The result will be the rating of this recommendation. The conditional probability takes only two dimensions at a time. Since there are 3 dimensions, according to a survey that was conducted during the research to know which of the three dimensions attributes is the most desirable or essential to the users. For example, for the restaurants it was found that it is the type of food. Therefore, the rating R can be calculated from the following where *ea1* is the essential attribute and *a2*, *a3* are the other attributes of the dimension of Item (Restaurant):

$$R = \frac{P(ea1/a2) + P(a3/ea1)}{2} \qquad (2)$$

4.2 Multi-Attribute Theory

Ratings of possible recommendations can be useful to anticipate the future recommendations according to the highest ratings provided by users. There are different approaches for estimating ratings. One of them is the reduction-based approach described in [11]. This approach actually works for

2-dimensions yet it was extended here to work with the multidimensional approach.

Therefore, in order to predict ratings of recommendations, it is assumed that the user already has a history and preferences. Multi-attribute choice theory is used in order to determine the most important attribute to the user and hence give higher weights for such attributes in the recommendations available. Since the important attributes are going to be taken from the history of the user's selections, maximum likelihood is going to be used to identify which attributes are important. The example below explains how to use these attributes: according to an existing history of the user's choices of restaurants, it was found that the user always prefers restaurants closer to his/her location. Also, the type of the food was essential since s/he was choosing mostly the same food type while the price of the food was not as important. Table 1 shows how the multi-attribute choice theory works. Each of the attributes will be assigned a weight (exact preferred =2, was chosen before =1, never =0) based on how close it is to the user's preferences. As shown in this table, the predicted rate of Restaurant1 is 6 and therefore it would be selected for the user as their recommended restaurant.

TABLE 1. MULTI-ATTRIBUTE THEORY

Attributes / Restaurant	Food **type (Italian, Chinese, Mexican)	Price (Expensive, Medium, Acceptable)	Destination** (Far, Medium, Closer)	Total
Restaurant 1	2	2	2	6
Restaurant 2	0	1	1	2
Restaurant 3	2	1	2	5

5. Knowledge Based

Within the agents there are a set of rules which are used by them to model their intelligence. Using the trip type, implicitly the system can differentiate between the different types of recommendations to be provided, beside the techniques explained earlier. Examples of a general rule is shown in the following, where *d* is the day, *t* is the time slot, *hR* is the history of restaurants, and *Rr* is the restaurants recommendations to be brought. :

$$R1: IF((1 \le d \le 5) \wedge (t = 3) \vee (hR \ge 1)) \xrightarrow{THEN} Rr$$

The following rules are examples to be more specific of how different recommendations types can be fired based on different scenarios using the above rule:

- (current trip == Family trip) AND (Current City == Dubai) AND (Time == Afternoon) AND ((History of user = 0) AND (Interest == null))-→ Shopping places in Dubai recommended

- (Current Trip == Business) AND (City == Dubai) AND (Time == evening) AND (Day == Weekday) AND ((History of user == exist) AND (History of Business trip ==Exist) AND (Restaurant= 1)AND(City is Dubai)) -→ Bring recommendations of restaurants in Dubai of same type.

6. Dynamicity of the System

The dynamicity of the system is an important aspect too, taking into account that the users might change their taste suddenly. Therefore, in order to cope with this situation, it is suggested to use a reduction of weights technique. The system normally will bring recommendations mostly according to the weight of the ratings. Yet, when a user rejects a recommendation in favor of another choice, the system will reduce the recommended rating and double the rating of the newly selected choice. This technique ensures the dynamicity in case the taste of the user has changed or simply she/he wants to try something different.

7. System Testing

The testing was conducted to verify and validate the implemented system. The testing covered all the functionalities and services provided by the system with different scenarios. This paper includes a major scenario that will show all functionalities. Evaluation results are analyzed after that.

At first, the registration was tested along with the login. New users must register first in order to use the system. The required information includes creating a username and password while there is additional profile information that users might want to fill in. The system can use this information in order to provide recommendations. In the case that profile is not filled in, the system will still work; it will build the profile from future usage.

After that, a user can fill in their trips basic information including the start and end dates, the type of trip, and the city of stay. This information is considered the minimal and the most essential that the user will be requested to insert before being provided with the recommendations.

Following this form, the user will be asked if s/he would like to check hotels in case they have not booked one. The types of the hotels which are pulled from the web are based on the output of the inference engine. Figure 5 shows a list of hotels that are brought to a new user. The hotels will have different

stars ratings and different prices but the same city of stay. In cases where a history exists in the profile, then a list of recommended hotels will be brought according to the inference engine.

After that, the system will give the option to the user to either create the schedule of the trip manually or the system will automate this process. In case the user wanted the system to create a schedule, it will be created within few seconds. The activities recommended in the schedule are selected based on the inference engine of the system. User profile, history of trips, contextual information, and current trip details are used by the inference engine to bring the most suitable recommendations to the user pro-actively and organize them in a schedule as shown in Figure 6.

For example, the user has previously gone on a business trip. His/her preferences that exist in the history are associated with the business trip type. When the user selects now to go on a family trip, the system will first check whether there are preferences for family trips that belong to this user. If the user's preferences of family trip exist (meaning he has gone on a family trip before) they will be used in order to expect and bring suitable recommendations. Otherwise, it will check the preferences of another trip type which is the business in this case and hence bring recommendations based on them.

The system may also notify the user of different location detections in the reminder. This notification is helpful especially if the user is supposed to be in a city for dinner yet s/he has changed his/her plans. This location detection makes the system aware of any change in the plan. It will also provide the user with the same services (recommendations of restaurants for dinner, for instance) in the new location. Figure 8 shows an example of the notification sent to the user.

Reminders will be sent to the user a couple of hours before each of the activities begin. Users then have the option whether to accept or reject them. In both cases the profile of the user will be updated with the new preferences. In the case of rejection, the system will bring forward other recommendations of the same service. The location of the user is also considered when bringing these recommendations. The GPS in the system is simulated assuming it will be checked frequently especially when reminders are fired. Figure 7 shows an example of a reminder sent to a user.

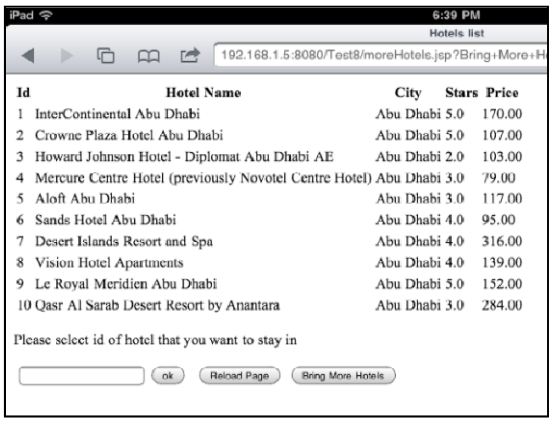

Figure 5. List of recommended hotels

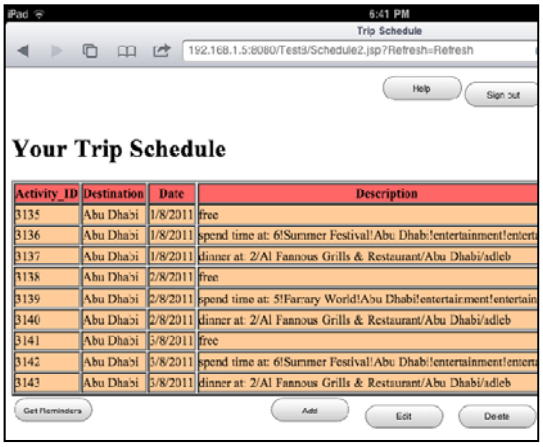

Figure 6. Proposed trip schedule with recommended activities

8. System Evaluation

The evaluation of the system was carried out by 71 evaluators. The number of trips each of the evaluators created varied from 2 to 6. Each trip is considered a test case. There are in total 220 test cases. The main purpose of the evaluation is to show the accuracy of the recommendations and how much the users are satisfied based on their profiles and their contextual information at various types of trips, times and locations. The evaluators were asked to rate the recommendations suggested by the system in each trip, and whether they live up to their expectations.

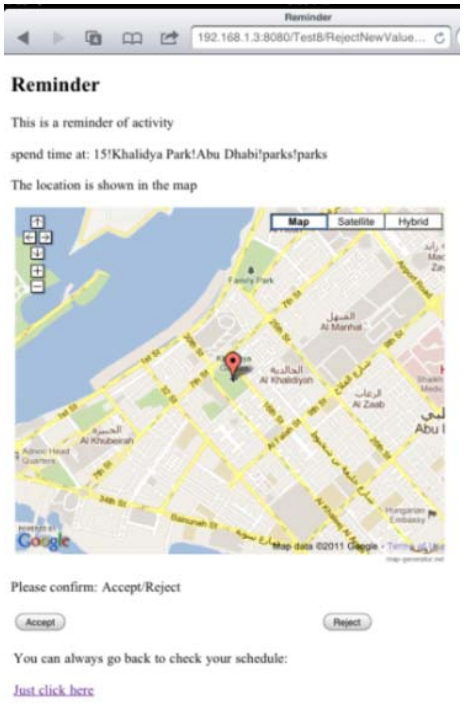

Figure 7. Reminder sent to the user

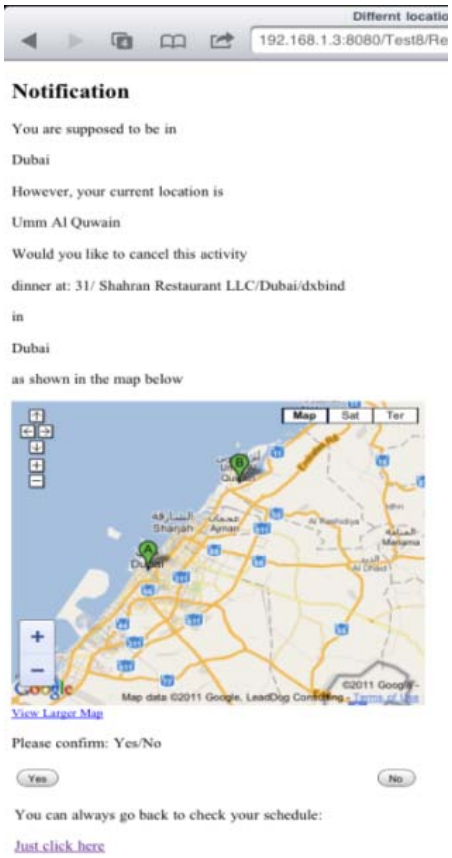

Figure 8. Example of the notification sent to user

The rating scale is from 1 to 5. Figure 9 illustrates all ratings given to the different recommendations services such as hotels, restaurants and events/places. Over 80% of the hotels and restaurant ratings were above average while, 75% of the events/places ratings are also above average. This shows the satisfaction of the users regarding the accuracy of recommendations brought to them.

Based on the evaluation conducted, the system was evaluated whether it satisfies the recommender systems criteria or not that were mentioned in [14]. The criteria and how they have been achieved are explained in Table 2.

In addition to that, the system was evaluated in order to achieve a number of features against other recommender systems. The system features are explained in the following:

- *Least user effort/ interaction:* The complexity of any system can be measured by the number of steps a user has to go through in order to reach the result. The less the steps the easier the system to use.
- *Implicit feedback:* The feedback of the recommendations in any recommender system can be taken either implicitly by watching a user's behavior or explicitly by entering the feedback. Deciding to take the feedback implicitly affects the number of steps as described earlier.
- *Estimating Ratings:* The effectiveness of any recommender system can be measured by its ability to predict the ratings of items provided to users based on their preferences.
- *Using the contextual information:* Examples of the contextual information are type of the trip, time, location and user preferences. Using such information can help providing more accurate and suitable recommendations to users.
- *Pro-activity:* The system can initiate actions and offer recommendations to the user and internally among the agents or the other components of the system.
- *Build/update profile implicitly*: This feature states that the user will have the ability to create his/her schedule. In case the user did not create his/her profile, the system will do it. Also, updating the profiles will be done on behalf of the user.
- *Usability of the system*: The system usability measures how user-friendly the system is and how easy the user interface is.
- *Accuracy of recommendations for every single user:* This indicates whether the system is accurate when sending recommendations to users depending on each user's preferences and context.

- *Portability/Heterogeneity:* This measures whether the system works in heterogeneous environments
- *Create proposed trip schedule automatically:* Some of the systems allow the users to create a trip schedule while others prepare this schedule automatically.
- *Dynamicity of recommendations:* The dynamicity of the recommendations indicates how they change according to user's profiles and context. Recommendations do not remain the same in different context.

As stated earlier, the system can provide recommendations in both user and system generated profiles. It was found from the evaluation that users who fill in their profiles and specify their preferences get what they expected from the recommendations for the first trip. On the other hand, a small number of those who have left their profiles empty did not get recommendations closer to their expectations in their first trip. It was noticed that the more they use the system, the more they become satisfied with the recommendations brought to them.

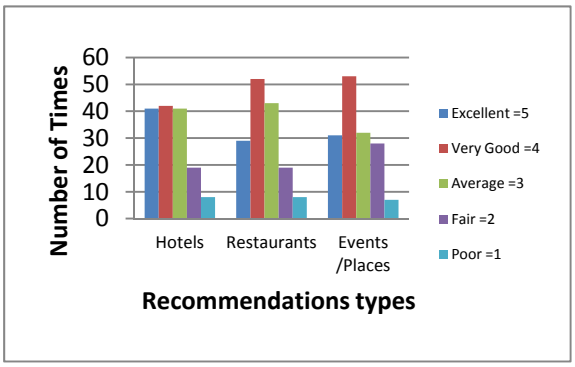

Figure 9. Ratings of recommendations

TABLE 2. RECOMMENDER SYSTEM CRITERIA

Criteria	How it has been achieved
Transparency	The system shows the recommendations from the schedule as reminders and notifications to get confirmations from users. With every modification the schedule is updated. 24% of the users found that these reminders make the system demanding while 69% found it normal and helpful to receive such reminders and notifications.
Scrutability	Users can modify the schedule manually and automatically. 94% of the users found that they can modify their trip schedule with less effort since modification on the schedule is done automatically.
Trust	The majority of the users (94%) agreed that the system learns their preferences with time . 87% of the users prefer to receive recommendations based on their interests, 64% of them prefer to try new things very often, while 57% like to have same food types and visit places of their interest.
Effectiveness	82% of the users have got what they have expected to find of recommendations which are similar to their preferences. Moreover, 93% of the users found that the system brings more accurate and better recommendations with time.
Persuasiveness	51% of the users rated the recommendations brought as exactly how they expected them while 49% as sometimes similar to what they had in mind.
Efficiency	The time they take to be calculated, checked, brought and filled within a schedule is almost 13 seconds at maximum. The time is taken to bring a list of hotels that are according to users' preferences, is about 20 seconds at maximum.
Satisfaction	69% of the users found the system not demanding, and 87% agreed that the system is not noisy as well. Moreover, over half of the users (56%) believed that it requires less interaction than other systems. On the other hand, 80% of the users showed satisfaction about the recommendations.

Additionally, the results describe different responses from the system depending on trip type (business or family) for every user. A user who goes on a business trip will only have free time during the evenings and weekends. Any activity or place to go to that is close to his/her preferences will be recommended during these times only. Restaurants on the other hand are recommended every day. However, when the user selects a family trip, the recommended activities are places that will be suited to all family members such as museums or shopping malls that they can spend all day in and have their meals there too. Certainly the activity types will be selected based on the user profile and interests. The system will firstly make use of preferences s/he had in their last family trip. In case the user has not been

on a family trip previously, the preferences will be checked in general regardless of the trip type.

9. Conclusions and Future Work

Recommender systems are used to filter services and bring forward the most suitable ones for users from a huge amount of information on the internet. Hybrid recommender systems can improve the recommendations provided since they use more than one approach. The multi-dimensional approach of recommender systems allows considering more dimensions in the process of providing more suitable recommendations to users. Contextual information has been proven to be helpful to be considered when providing recommendations based on user's context. Integrating a multi-dimensional recommender system with a multi-agent system will lead to an enhancement of the pro-activity of recommender system especially for the tourism domain. The system proposed in this paper was designed and implemented to bring recommendations for tourists pro-actively and organize them as a trip schedule. The contextual information used are the type of the trip, time, the current location and preferences of the user. The system shows promising results when tested based on various scenarios. Results show the variety of responses of the system in different contexts with/without users' profiles explicitly inferred.

As for future work, we intend to enhance the quality and quantity of services provided to travelers. More contextual information can be considered such as the weather. The pro-active travel assistant should then be able to provide a wider range of services to the user. This can be achieved by using more service provider websites than the one used here. Also, there is a need to automate and update the restaurants table from the state it is in currently. Moreover, the pro-activity and the intelligence of the system will be further enhanced to cover the emergency cases that a traveler might face abroad. Lastly, the evaluation can be enhanced further by applying the methodology of information retrieval experiments [15].

10. References

[1] G. Picco, "Mobile Agents: An Introduction, Microprocessors and Microsystems", Springer, 2001, 25(2): 65-74.

[2] C. G. Harrison, D. M. Chess, and A. Kershenbaum, "Mobile Agents: Are they a good idea?", Technical Report, IBM Research Division, T. J. Watson Research Center, 1995.

[3] A. Felfernig, S. Gordea, D. Jannach, E. Teppan, M. Zanker, "A short Survey of Recommendation Technologies in Travel and Tourism", OEGAI Journal 25 (7),Oesterreichische Gesellschaft fuer Artificial Intelligence, 2007, pp. 17-22.

[4] M. Wooldridge, N. Jennings, "Intelligent Agents: Theory and Practice", The Knowledge Engineering Review, vol. 10, 1995, pp.115-152.

[5] Y. Wei, L. Moreau, and N. Jennings, "Recommender systems: A market-based design", Proc. Second International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS03), Melbourne, Australia, 2003, pp. 600-607.

[6] B. Schilit, N. Adams, and R. Want. "Context-aware computing applications". IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'94), Santa Cruz, CA, US:89-101, 1994.

[7] K. Anind Dey, "Understanding and Using Context". Personal Ubiquitous Computing 5(1): 4-7.2001.

[8] M.N. Huhns, D. Bridgeland, "Multiagent truth maintenance", IEEE Transactions on Systems, Man, and Cybernetics, 1991, 21(6):1437-1445.

[9] F. Lorenzi, A.L.C. Bazzan, M. Abel, "An Architecture for a Multiagent Recommender System in Travel Recommendation Scenarios", ECAI workshop on Recommender Systems, August 2006, Riva del Garda, Italy, pp. 88-91.

[10] A.Casali, A. Von Furth, L. Godo, C. Sierra, "A Tourism Recommender Agent: Form theory to practice", WASI-CACIC 2007, Corrientes, Argentina, 2007, pp 1548-1561.

[11] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, 2005. Incorporating contextual information in recommender systems using a multidimensional approach. JTOIS, 23(1), 103-145.

[12] A. Goy, L. Ardissono, and G. Petrone, "Personalization in e-commerce applications". In The Adaptive Web, Springer, Berlin/Heidelberg, 2007, pp. 485-520.

[13] Z. Wa, "Personalized Tourism Information System in Mobile Commerce", Proc. IEEE International Conference on Management of e-Commerce and e-Government, Washington DC, 2009, pp. 387-391.

[14] N. Tintarev, J. Masthoff: A survey of explanations in recommender systems. In: ICDE Workshop on Recommender Systems & Intelligent User Interface, 2007.

[15] S.E. Robertson, The methodology of information retrieval experiment. In: K. Sparck Jones (ed.), *Information retrieval experiment.* Butterworths, 1981. (pp 9-31).

