

# On-Tour Interactive Travel Recommendations

Quang Nhat Nguyen,  
Dario Cavada, and  
Francesco Ricci

e-Commerce and Tourism Research Laboratory  
ITC-irst, Trento, ITALY  
{quang, dacavada, ricci}@itc.it

## Abstract

Travelers who access the Internet through currently available web information systems often experience difficulty in selecting interesting products. This is especially true for on-the-move travelers who browse information repositories using mobile devices with poor user interfaces. Travelers suffer from an overload of information and options to consider, and they lack system support in filtering information and comparing products. In this paper, we propose an innovative approach to the problem of mapping travelers' needs to a convenient set of travel products or services. In particular, we present an on-the-move restaurant recommender system (mTR) integrated with a pre-travel planning aid system (NutKing). In this approach, firstly, the knowledge contained in a repository of past user choices is exploited to initialize the recommendation process with a set of implicit preferences. Secondly, to minimize user efforts in building a precise search query, we do not require the user to formulate a query at the beginning of the interaction, rather we involve her in a dialogue where the traveler is encouraged to provide critiques and feedback to the system recommendations. These critiques are then incorporated by the system into a new query that tries to better model the user preferences. The approach has been validated empirically by a set of users.

**Keywords:** travel recommender systems; mobile business; empirical evaluation; cooperative query answering.

## 1 Introduction

Nowadays, there are many e-commerce web sites that support travel information search and in particular travel planning. However, most of them simply let users search (e.g., with keywords, or with query forms) through their electronic catalogues of products. This often leaves the users alone in analyzing a potentially overwhelming number of options, causing the well-known "information overload" (Werthner and Klein, 1999). This problem has two sides. Firstly, the user may not have enough knowledge to express her needs in accordance with the system language and interface,

i.e., to define a query to be processed by the system. Secondly, users often receive poor support in analyzing system results, in comparing products, and in bundling final choices. These problems, which have been tackled by research on personalized recommender systems (Kobsa et al., 2001; Schafer et al., 2001), are ubiquitous in the whole travel experience: during the pre-travel stage (planning), while the traveler is on-the-move, and also when the travel is finished (customer support) (Ricci, 2002).

To deal with the full travel cycle, we have developed two systems: NutKing and mITR. NutKing deals with the pre- and post-travel stages. It is a recommender system that combines content-based and collaborative-based filtering methods to support users in building the recommended travel plans (Ricci et al., 2002; 2003). NutKing helps users in selecting one or more destinations to visit and to add additional products related to the selected destinations (accommodations, activities, events).

mITR (Mobile Intelligent Travel Recommender), which is described in this paper, is aimed at on-tour support, and helps a traveler to complement her travel plan by adding new products, such as events or restaurants, when she is at the selected destination or on the move. mITR is based on the assumption that these complementary products should conform to what have already been selected in the pre-travel stage. Moreover, we hypothesize that, when a traveler is on the move, the time span and cognitive effort spent to find interesting products should be minimized using appropriate methodologies.

The approach implemented in mITR to cope with the above requirements, is inspired by previous research (Shimazu, 2001; McGinty and Smyth, 2002; Burke, 2000) that has introduced the ideas of “recommendation by proposing” and “similarity based query revision”. The basic idea is that a user will find it easier to accept (or criticize in feedback) a product recommendation than to formulate a fully specified request listing all her needs. Obviously, the closer the initial proposal is to the user needs, the higher is the chance that the recommendation is accepted. In addition, the simpler it is to get feedback or critiques to the proposal, the more likely the user will interact with the system to improve the recommendation.

Our approach is innovative in two aspects: in the way the first proposals are computed and in the interaction supported. Regarding the first aspect, we propose to exploit the case base of travel plans contained in the NutKing system. When a user is in one place, which is specified by precise space-time coordinates, we select all the restaurants that are in a given distance range, and then we sort them according to their similarity to previous restaurant choices of the user (or of similar users). This approach tries to offer the user the same type of restaurants that she is used to going to. Regarding the second aspect, by interacting with the user we let her browse the proposals and we give her the possibility either to choose one or to criticize one. A typical critique is, for instance: “this restaurant is quite nice, but it is a bit too expensive”. Hence we acquire her feedback regarding the proposals. This feedback is

automatically incorporated into either a new logical filter, or into a condition used in the similarity-based sorting to rank higher those products that satisfy the critique. The logical filters model “must” conditions, whereas the critiques (used only for sorting purposes) model “wish to have”. Hence, if the user says that a restaurant is too expensive, we discard those with average cost above a threshold that is lower than the average cost of the criticized restaurant. Whereas, if the user says that she would prefer to pay with credit card, we just use this preference to put those that accept this payment method first in the result list.

We finally stress that none of the previous approaches quoted above have dealt with the precise requirements of the mobile context of usage.

In the next section, we describe briefly how NutKing is used to build a pre-travel plan. Then we describe the methodology used in mITR to simplify and reduce the length of the recommendation dialogue. Finally, we present the results of the system validation and we propose some future work.

## 2 Pre-travel Support

This section describes a typical human computer interaction in the pre-travel planning stage using the NutKing travel planning aid tool. The traveler initially defines some trip characteristics and personal interests, such as the party travelling, the available budget and the transportation means. NutKing uses these features: a) to identify similar trips built by other users, and b) to set some default constraints in successive query forms. After this initial step, the traveler starts bundling her trip by searching for the travel products offered by the system: destinations, accommodation, events, and activities.

TRAVEL COMPANIONS	DEPARTURE	ACTIVITIES
Who will you travel with? with family	Where are you from? Europe	What would you like to do on this trip? <input checked="" type="checkbox"/> Sports <input type="checkbox"/> Adventure <input checked="" type="checkbox"/> Relaxing <input type="checkbox"/> Art & Culture <input checked="" type="checkbox"/> Enogastronomy <input checked="" type="checkbox"/> Landscape <input type="checkbox"/> Environment <input type="checkbox"/> Ecology
TRANSPORT How will you travel? car	PERIOD When do you want to travel? September How long do you want to stay? one week	
ACCOMMODATION What kind of accommodations do you want? hotel What's your daily budget (for accommodation)? between 20 and 40 €	PREVIOUS VISITS Have you ever visited Trentino? a few times	<input type="button" value="NEXT"/>

Fig. 1. NutKing interface to define the trip characteristics.

The system allows the user to issue a query (with a simple query-by-example form) and retrieves the desired products. If the query fails because no product satisfies the user's query, the system proposes to relax some of the query constraints. Conversely, when the query retrieves too many products for review, NutKing asks the user to provide some additional constraints.

Finally, the retrieved products, which satisfy the (explicit) user constraints, are sorted and presented to the user, putting first the products most similar to those selected by other users having expressed similar general travel wishes (those in Figure 1).

Although NutKing has been validated successfully in the web context for pre-travel planning (Ricci et al., 2003), it cannot be used, as it is, by on-the-move travelers. In fact, the mobile context imposes some peculiar constraints and requirements, such as: the available screen size, the graphical user interface and interaction supported, and finally, user behavior and external environment impacts (e.g., noise, interruptions, light, etc.). A recommender system designed to support on-the-move travelers should take into consideration, among other things, the following characteristics:

- A mobile user's interaction session should be kept very short in time .
- Mobile travelers should receive a useful recommendation within no more than 2-3 recommendation cycles.
- Mobile users do not like to input lots of data. They rather prefer answering to Yes/No questions with one click.
- The preferences are only valid for a specific session, and could be rather different in another situation.

For these reasons, we have designed a completely different recommendation methodology. We see that NutKing contributes to this new methodology in the initialization of the on-the-move recommendation process by providing a set of preferences extracted from the product choices made by users in the past. In this respect, NutKing and the proposed mobile recommender system, mITR, provide an integrated solution to support travel decision choices.

### **3 On-tour Support**

When a traveler who could build a travel plan with NutKing is on-the-move and starts searching for a desired travel product, mITR helps her in building, and iteratively updating, a set of search conditions (query) which capture her needs. Queries are processed by the system to produce a set of recommended products that are presented to the traveler.

The queries processed by mITR contain two components:

- Logical component,  $Q_L$ , contains some logical constraints that *must* be satisfied by the retrieved products;

- Similarity-based component,  $Q_S$ , contains a product pattern (i.e., a partially defined product) that models product features that *could* be satisfied.

The recommendation process evolves in cycles. The user initiates the first cycle by simply asking for a restaurant recommendation. Then, in response to this request, the system builds an initial query defining both the logical and the similarity-based components. It then retrieves a set of candidate restaurants from the catalogue. At this point, the user can either accept one of the recommended restaurants or express a critique about one of them. The last refers to the case in which a restaurant on the whole satisfies the user, but she would also like some additional features (e.g. parking or live music) or some of the features do not match exactly her preferences (e.g., the price is too high). Hence, a new query is built by the system, incorporating the user critique or feedback and a new set of recommended restaurants is proposed.

Here we give some definitions to make the description more formal.

- *Travel products* are represented as vectors of feature values  $x=(x_1, \dots, x_n)$ . The feature value  $x_i$  may be numerical, nominal, symbol-set, or text.

Hereafter, we shall illustrate system functionality using an example of restaurant recommendation. For the sake of simplicity, here we represent a restaurant with five features (in reality, mITR describes a restaurant with 15 features): name (nominal), type (nominal), location (nominal), cost (numerical), and openingDays (symbol set). Hence, the example  $x=(\text{"Pizzeria Ristorante al Vesuvio"}, \text{"pizzeria"}, \text{"Trento"}, 10, \{1,2,3,4,6,7\})$  means that the name is:  $x_1=\text{"Pizzeria Ristorante al Vesuvio"}$ , the type is:  $x_2=\text{"pizzeria"}$ , the location is:  $x_3=\text{"Trento"}$ , the cost is:  $x_4=10$ , and the restaurant is open all days of week except on Thursday:  $x_5=\{1,2,3,4,6,7\}$ .

- *The logical component  $Q_L$*  is a conjunction of constraints. A constraint only relates to one feature, and a feature appears in only one constraint.

$$Q_L = c_1 \wedge \dots \wedge c_m$$

The form of a constraint depends on the type of the constrained feature. For example, a logical query searching for those restaurants in Trento that are open on Saturday and Sunday is written as:  $Q_L=(x_3=\text{"Trento"}) \wedge (x_5 \supseteq \{7,1\})$ .

- *The similarity-based component  $Q_S$*  is represented in the same vector space with a product pattern:

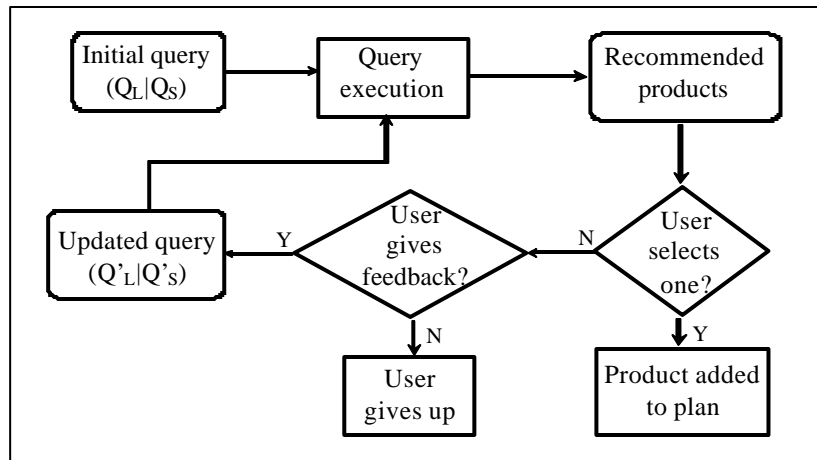
$$Q_S = (p_1, \dots, p_n),$$

where  $p_i$  is a value for the  $i$ -th feature. The similarity between  $Q_S$  and the products in the catalogue is computed according to a heterogeneous Euclidean Overlapping distance metric (Ricci et al., 2002). Note that a feature value  $p_i$  may be undefined (denoted as "?"), and therefore it is ignored in the similarity computation. For example, the similarity-based query  $Q_S=(?, \text{"spaghetteria"}, ?, 15, ?)$  means that only two features ('type' and 'cost') will be used in the similarity computation.

The query ( $Q_L|Q_S$ ) is initialized when the traveler starts searching for a travel product. The logical query ( $Q_L$ ) is initialized with the traveler's current location and time. For example, if the traveler is in Trento and it is Saturday, then the logical query will be initialized as  $Q_L=(x3="Trento")\wedge(x5\supseteq\{7\})$ .

The similarity-based query ( $Q_S$ ) is initialized using the knowledge exploited from the past recommendations contained in mITR and NutKing. The initialization of  $Q_S$  depends on whether the traveler: a) has built a pre-travel plan (by using NutKing), and b) has ever received some recommendations from mITR. In both cases, we use the last accepted recommendation to initialize the similarity-based query, reusing only the general characteristics of the chosen restaurant. If the traveler has neither built a pre-travel plan nor received recommendations from mITR, then all the  $Q_S$  feature values ( $p_i$ ) are undefined ("?").

For instance, if a traveler, last time she used mITR, accepted the recommendation  $x=(\text{"Pizzeria Primavera"}, \text{"pizzeria"}, \text{"Trento"}, 10, \{1,3,4,5,6,7\})$ , then the query  $Q_S$  is initialized as  $Q_S=(?, \text{"pizzeria"}, ?, 10, ?)$ .



**Fig. 2.** The query processing

mITR computes recommendations, by firstly executing  $Q_L$  to identify all the products that match the 'must-have' constraints in  $Q_L$ . Then, all the resulting products are sorted accordingly to their similarity to  $Q_S$ . Finally, the three products most similar to  $Q_S$  are recommended to the traveler. This process is shown in Figure 2.

We now describe how the user critiques are incorporated in a new user query. Because of the mentioned limitations of the mobile environment, in the mITR system,

on each recommendation cycle, the traveler could critique only one feature of a single product.

In the mITR system, we have implemented several types of critiques. However, due to space limitation, here we describe only some examples (cfr (Nguyen et al., 2003) for the complete list).

*F1 - Positive critique on a nominal feature.* The traveler states that she likes the value  $l_i$  of the  $i$ -th feature of a product  $l=(l_1, \dots, l_n)$ . Then the new value  $l_i$  is assigned for the  $i$ -th feature of  $Q_S$ .

$$p_i = l_i$$

*F2 - Positive critique on a numerical feature; want less.* The traveler states that she overall likes the product, but wants to see some alternatives having a smaller value for the  $i$ -th feature. Then the following constraint is included in  $Q_L$ .

$$c_i = (x_i = l_i - d);$$

where  $d (>0)$  is a small adjustment factor that allows to retrieve other products rather than the current one.

*F3 - Positive critique on a symbol-set feature.* The traveler states that she likes the value  $l_i$  of the  $i$ -th symbol-set feature (e.g., the ‘Open days’ feature), and wants to see some more products having similar feature value. So the following constraint is included in  $Q_L$ .

$$c_i = (x_i \supseteq l_i)$$

## 4 User Interface

In this section we illustrate the proposed approach with an interaction example.

We assume that a traveler accesses mITR to look for a restaurant. mITR, as described above, exploits the traveler’s current time and position (achieved via GPS services) and the previous usage history (through NutKing) to initialize the request (query). The query is executed and an initial recommendation list is shown (Figure 3-a). Then, the traveler can view detailed information about the listed restaurants. Snapshot (b) in Figure 3 shows the attributes of the “Pizzeria Ristorante al Vesuvio” restaurant, and snapshot (c) shows the restaurant’s brief description and customers’ opinions. If the traveler accepts this recommendation, she can add it to her travel notes (a convenient container of all her selections); otherwise, she can criticize one recommendation (snapshot (d)).

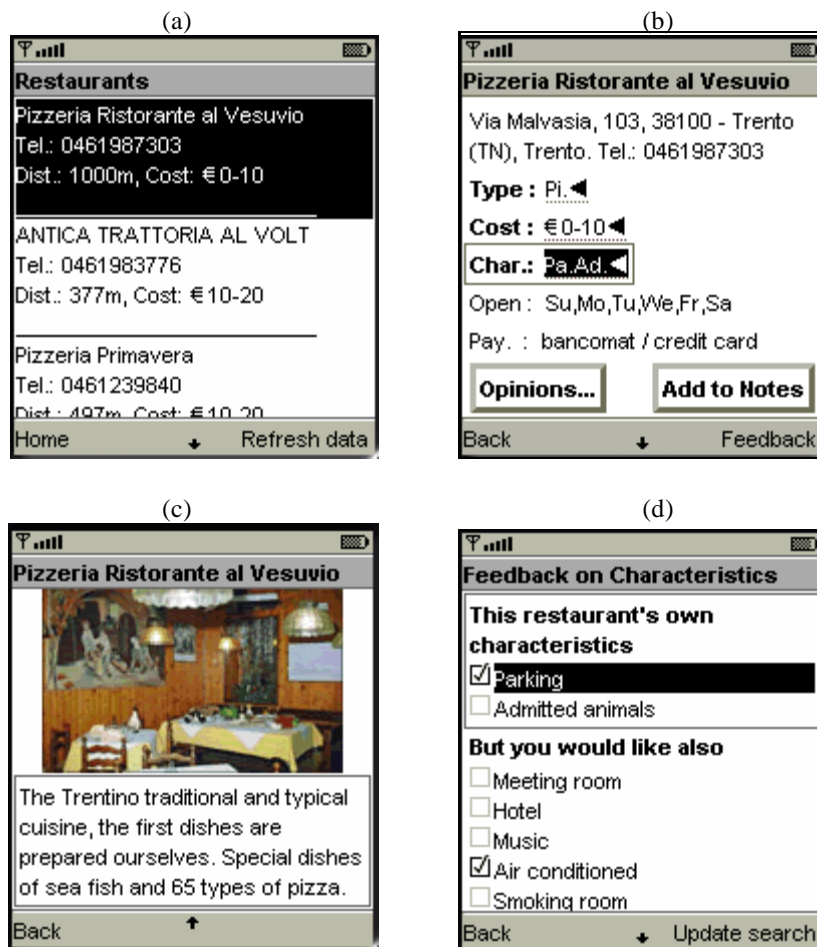


Fig. 3. mITR graphical user interface.

In this example, the traveler checks the “Parking” and “Air conditioned” features, meaning that she is interested in these (F1 critique in Section 3). “Parking” is a feature of the selected restaurant but “Air conditioned” not. These preferences are then exploited, when the traveler clicks on the “Update search” button. If possible, three new recommendations offering these requested features are retrieved.

## 5 Evaluation

We have conducted a usability evaluation to assess whether the proposed prototype provides an effective, efficient and easy-to-use tool (Nielsen, 1993; Chin, 2001). The prototype has been developed using Java2 Micro Edition (<http://java.sun.com/j2me> [September 10, 2003]), which is already included in the newest mobile phones (<http://www.kspar.net/wapImodeJ2me.ppt> [April, 2002]). We have used the latest version (MIDP 2.0) running on a simulator available on a standard PC.

We involved six mobile phone users with no experience in WAP browsing or other Java applications running on mobile phones. In the first step, the users were introduced to system functionality and use. They were asked to familiarize themselves with the simulator before solving a real task. We note that using the simulator is slightly more difficult than using a real phone. For instance, in the simulator, the navigation buttons (up, down, left and right) and the three command buttons (select, left-ok, right-ok) are mapped on the PC numeric keypad. This caused some difficulties. In fact, the users should associate icon buttons on the simulator interface (on the PC screen), with keys on the PC keyboard. On a real phone, obviously, the user clicks on physical buttons on the phone keyboard. We have even tried to let the user click on the screen, using the mouse, but this did not improve the usability much. Then a task was described to the users: "Imagine that you are now in Trento, in Piazza Duomo, at lunch time. Please think about the features of your candidate restaurant and try to locate one with the system. When you have found it, please add it to your travel notes and then open the travel notes to look at that restaurant".

After the training and test stages, we asked the users to fill out a usability survey, in which we asked them to briefly comment on the difficulties they have faced and on the desirable extensions and improvements. The users were asked to state their agreement or disagreement on a seven-point Likert scale regarding the following nine statements:

1. The system was pleasant to use.
2. The organization of information on the system screen was clear.
3. This system has all the functions and capabilities that I expected to have.
4. The information provided by the system was complete.
5. I've found the restaurant that satisfies my needs.
6. I've found the possibility to critique a restaurant and get a new sorting of the offers useful and easy to use.
7. It was simple to use the system.
8. I was able to efficiently complete the tasks and scenarios using this system.
9. If the system were available on my phone I would use it.

Some of the above statements were extracted from the Post-Study System Usability Questionnaire (PSSUQ) (Lewis, 1995), some were added to get an evaluation on the new functionality.

The results of the survey are shown in the following table (1= strongly agree, 7=strongly disagree).

**Table 1.** Experiment result

Statement	user						Average	Std.dev
	u1	u2	u3	u4	u5	u6		
s1	4	1	1	3	1	3	2,17	1,33
s2	3	1	3	2	4	3	2,67	1,03
s3	6	1	1	1	2	3	2,33	1,97
s4	5	2	1	1	2	4	2,50	1,64
s5	1	1	1	1	1	1	1,00	0,00
s6	1	2	7	1	3	1	2,50	2,35
s7	2	2	3	3	4	2	2,67	0,82
s8	1	1	1	1	1	1	1,00	0,00
s9	1	3	1	1	1	1	1,33	0,82
average	2,67	1,56	2,11	1,56	2,11	2,11	2,02	0,42

Users were typically able to find a restaurant that satisfies their needs (s5), they were able to complete the task (s8), and they would definitely use such functionality if available on their mobile phones.

The critical points were made clear by the detailed comments we collected. The most important were:

1. Users found it difficult to use the additional buttons placed on the screen (“Add to Notes” and “Opinions” in Figure 3-b).
2. Even if they liked the approach to re-sort the restaurants according to their critiques, users wanted to initialise the search explicitly stating some preferences.
3. Users did not like getting only three recommendations, but asked to browse the full list.
4. All users asked for an additional map-based navigation support to reach the restaurant from their current position.

The usability problems, such as those related to the buttons (1) usage, or the missing functions (4) will be solved in the next system version. Points (2) and (3) are more interesting. These, on one hand, disprove the original hypothesis that mobile users need systems that reduce the time and cognitive load required to fulfil a task. In fact, we observed that some users want to dedicate time to specify their needs even if the system offers to let them skip this stage at the beginning. They would also like to

browse a longer result list (more than three items). On a more careful analysis, we hypothesised that the motivation of such requests (to specify initially a query form and to browse a longer list of items) is the familiarity of such kind of interfaces. People are used to specify query forms and browse a long list of items, and they are surprised when they find completely different interfaces. This is one interesting topic that we plan to evaluate in a future test, i.e., to compare this system with a variant that allows users to initially specify their queries and to browse then the full list of products which satisfy their conditions. Another explanation is that time and cognitive efforts must be compared with the benefits. Hence, for instance, if the user is engaged and wants to select the “best” offer, she will be likely to accept a longer and more sophisticated interaction. In other words, general usability principles must be grounded on the real user interfaces to finally opt for some precise design choices.

## **6 Conclusion and Future Work**

In conclusion, in this paper we have shown that mITR provides a usable approach to recommend tourism products to a user on-the-move. Nonetheless, a number of open issues will need to be tackled in some future work. Firstly, we would like to perform a more extensive user evaluation: exploiting a precise log of users’ interactions and defining a methodology to assess the accuracy and precision of the proposed recommendations. Secondly, we would like to improve the capability of the system to capture the user’s preferences. In fact, this can be achieved in several ways. Firstly, the user’s needs could change during the interaction session (i.e., from an unclear state to a more refined state) and, therefore, some methods to tackle this “drift” effect must be put in place (Montaner et al., 2002). Secondly, we would like to better use the information about the position (in the result list) of the item selected by the user in a recommendation cycle, and the differences between the proposed items. In fact, the user could compare these, and the system could try to understand the reason for a product preference (McGinty and Smyth, 2002). A final issue we expect to tackle to improve the recommendations is an analysis of the relationships (e.g., similarity) amongst the critiqued items in the same interaction session.

## **References**

- Burke, R. (2000). Knowledge-based recommender systems. In Daily, J. E., Kent, A., & H. Lancour (Eds.), *Encyclopedia of Library and Information Science*, volume 69 (pp. 180-200). Marcel Dekker.
- Chin, N. (2001). Empirical evaluation of user models and user-adapted systems. *User Modeling and User-Adapted Interaction*, 11(1): 181-194.

McGinty, L. & B. Smyth (2002). Deep Dialogue vs Casual Conversation in Recommender Systems. In Ricci, F., & B. Smyth (Eds.), *Proceedings of the Workshop on Personalization in eCommerce, at the 2th International Conference on Adaptive Hypermedia and Web-Based Systems, AH-02* (pp. 80-89). Universidad de Malaga, Malaga, Spain.

Kobsa, A., Koenemann, J., & W. Pohl (2001). Personalised hypermedia presentation techniques for improving online customer relationships. *The knowledge Engineering Review*, 16(2): 111-155.

Lewis, J. R. (1995). IBM computer usability satisfaction questionnaires: psychometric evaluation and instructions for use. *International Journal of Human Computer Interaction*, 7(1): 57-78.

Montaner, M., López, B., & J. L. De La Rosa (2002). Improving case representation and case base maintenance in recommender systems. In Craw, S. and A. Preece (Eds.), *Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002* (pp. 234-248). Aberdeen, Scotland. Springer Verlag

Nguyen, Q. N., Cavada, D., & F. Ricci (2003). Trip@dvice mobile extension in a case-based recommender system. In *Proceedings of the 2nd International Conference on Mobile Business, mBusiness>2003* (pp. 345-356). Vienna, Austria. Austrian Computer Society.

Nielsen, J. (1993). *Usability engineering*. San Francisco. Morgan Kaufmann Publisher.

Ricci, F. (2002). Travel recommender systems. *IEEE Intelligent Systems*, 17(6): 55-57.

Ricci, F., Arslan, B., Mirzadeh, N. & A. Venturini (2002). ITR: a case-based travel advisory system. In Craw, S., & A. Preece (Eds.), *Proceedings of the 6th European Conference on Case Based Reasoning, ECCBR 2002* (pp. 613-627). Aberdeen, Scotland. Springer Verlag

Ricci, F., Venturini, A., Cavada, D., Mirzadeh, N., Blaas, D., & M. Nones (2003). Product recommendation with interactive query management and twofold similarity. In Aamodt, A., Bridge, D., and K. Ashley (Eds.), *Proceedings of the 5th International Conference on Case Based Reasoning, ICCBR 2003* (pp. 479-493). Trondheim, Norway.

Schafer, J. B., Konstan, J. A., & J. Riedl (2001). E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1/2): 115-153.

Shimazu, H. (2001). Expertclerk: Navigating shoppers buying process with the combination of asking and proposing. In Nebel, B. (Eds.), *Proceedings of the 17th International Joint Conference on Artificial Intelligence, IJCAI 2001* (pp.1443-1448). Morgan Kaufmann.

Werthner, H. & S. Klein (1999). *Information Technology and Tourism - A Challenging Relationship*. New York, Springer Verlag Wien.