17 MobyRek: A Conversational Recommender System for On-the-move Travellers

Francesco Ricci and Quang Nhat Nguyen

1. Introduction

Although many e-commerce websites support travel information search and in particular travel planning, most of them simply let users search (e.g. with keywords or with query forms) through their electronic catalogues of products. In fact, e-commerce travel and tourism websites often contain huge quantities of travel items with different characteristics and types. Hence, a user's search request often returns a potentially overwhelming set of options, causing an 'information overload' (Maes, 1994). This problem has at least three causes. First, some users may not have enough knowledge to express their needs in accordance with the system language and interface, i.e. to define a query to be processed by the system. Second, the preferences, which are collected at the time of a user's request, are typically a subset of the user's real 'needs and wants', because users (especially mobile users) usually do not like to input data. Third, users often receive poor support in analysing search results, in comparing products and in bundling final choices.

Both leisure and business travellers need system support throughout all travel stages: from pre-travel planning to the on-the-move support during the travel, and even when the travel is finished (Ricci, 2002). We have developed two systems, NutKing and MobyRek, which cooperate to support travellers through their full travel life cycle. 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 then adding additional products related to the selected destinations (accommodations, activities, events). AQ1

MobyRek, which is described in this chapter, deals with the on-tour stage. On-tour support is needed by travellers when they are on the move to, or during the stay in, their selected destination. In the on-tour stage, travellers typically use mobile devices to search for desired travel products, or to complement their pre-travel plan (which is already built before their leave, using the NutKing system). MobyRek is based on the assumption that the complementary products should conform to what have already been selected in the pre-travel stage. Moreover, we hypothesize that when a traveller is on the move the time span and cognitive effort spent to find his or her desired products should be minimized using appropriate methodologies.

In the MobyRek system the decision support is provided to travellers through personalized recommendations. Given a traveller's request, MobyRek produces those travel product recommendations that are personalized to the traveller in that particular situation. To minimize the user's effort, MobyRek does not require the user to formulate a precise and complete query at the time of the request, but involves the user in a dialogue (i.e. a conversation), which interleaves system's recommendation with user's critique. The ideas of 'recommendation by proposing' and 'similarity-based query revision' have been introduced in previous research (Burke, 2000; Shimazu, 2001; McGinty and Smyth, 2002). The basic idea is that critique-based elicitation of user preferences (i.e. by interleaving elicitation with recommendation) seems to be more effective in pushing the users to the elicitation of their needs while keeping the interaction alive. Usually, users communicate their needs and preferences when they are convinced that they will benefit from that. Hence, for instance, the request of formulating a precise and complete query right from the beginning of the interaction may not be practical, especially for mobile users.

AQ2

Our approach in MobyRek is innovative in two aspects: the way the first proposals are computed and the interaction supported. In a recommendation session, the closer the initial proposal is to the user needs, the higher is the chance that the recommendation is accepted. In user–system interaction, the simpler it is to receive feedback or critiques to the proposal, the more likely the user will interact with the system to improve the recommendation. We propose to exploit different knowledge sources of users' data in building the initial representation of users' preferences. When a user is in a place, which is specified by precise space–time coordinates, MobyRek first selects all the restaurants that are in a given distance range, and then sorts these restaurants according to their similarity to previous restaurant choices of the user (or of similar users). This approach tries to initially offer to the user restaurants similar to those the traveller normally chooses. The system then enables the users to browse the 'proposals', encouraging them to choose or to criticize each option. A typical critique is, for instance: 'I am interested in this restaurant, but it is a bit too expensive.' Depending on the type of critique (i.e. feedback), the critique is automatically incorporated into the system's representation of the users' preferences as either a must-have requirement or an optional preference. Hence, if the users indicate that a restaurant is too expensive, the system discards those with costs above the criticized restaurant. Whereas, if the users say that they would prefer to pay with credit card, the system uses this preference to order those that accept this payment method first in the recommendation list.

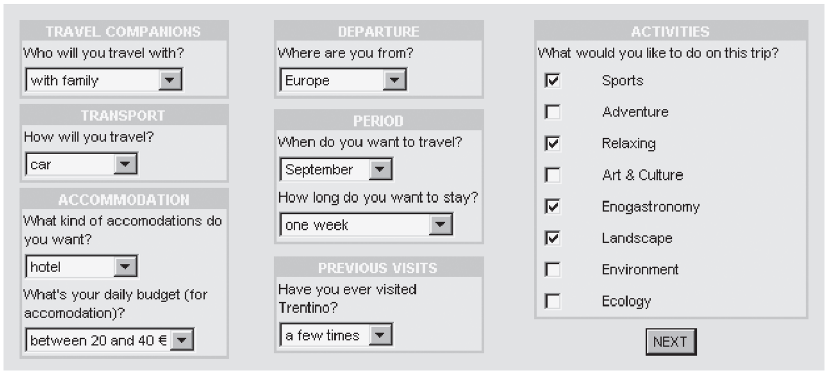
2. Pre-travel Support

The traveller initially defines some trip characteristics and personal interests, such as the travel party, the available budget and means of transportation. NutKing uses these features: (i) to identify similar trips built by other users; and (ii) to set some default constraints in successive query forms. After this initial step, the traveller starts bundling his or her trip by searching for the travel products recommended by the system. The system allows the user to issue a query (with a simple query-by-example form) and retrieves the desired products (see Fig. 17.1). If the query fails because no products satisfy the user's query, the system proposes alternative query relaxations, which, if applied, would retrieve a suitable result set. Conversely, when the query retrieves too many products, NutKing asks the user to provide some additional constraints to narrow the result list. The retrieved products, which satisfy the user's (explicit) constraints, are then sorted and presented to the user, ordering the products most similar to those selected by other users who have expressed similar general travel wishes (see Fig. 17.2).

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 travellers. Indeed, the mobile context imposes a number of peculiar constraints and requirements (Passani, 2002):

- the limitation of mobile devices (e.g. small screen size, limited computation capability);
- the graphical user interface and interaction supported (e.g. compact layout, limited input modality);
- the behaviour of mobile users (e.g. like to input less, but to receive results quickly); and
- the external environment impacts (e.g. noise, interruptions, light).

A recommender system designed to support on-the-move travellers should take into consideration, among other things, the following characteristics:



TRAVEL COMPANIONS
Who will you travel with? with family

TRANSPORT
How will you travel? car

ACCOMMODATION
What kind of accomodations do you want? hotel
What's your daily budget (for accomodation)? between 20 and 40 €

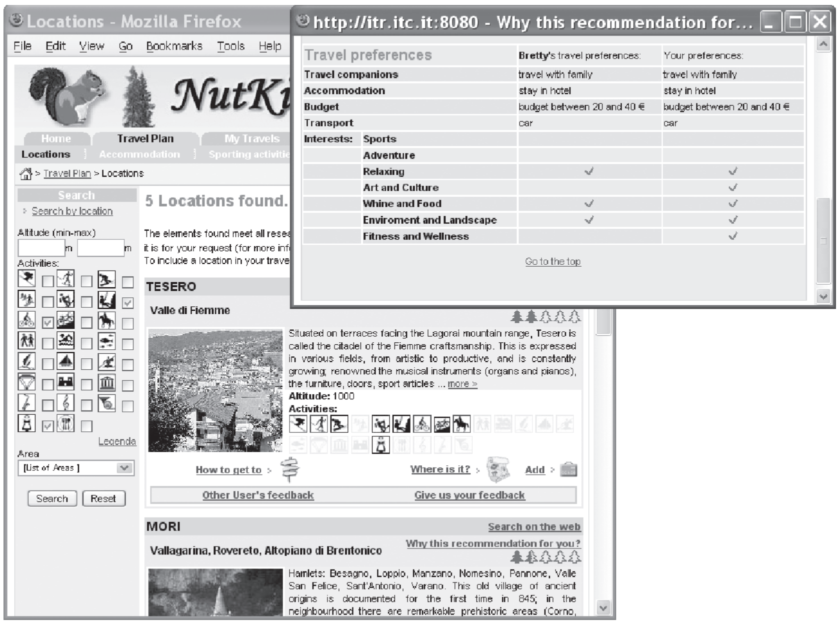
DEPARTURE
Where are you from? Europe

PERIOD
When do you want to travel? September
How long do you want to stay? one week

PREVIOUS VISITS
Have you ever visited Trentino? a few times

ACTIVITIES
What would you like to do on this trip?
- ✓ Sports
- ☐ Adventure
- ✓ Relaxing
- ☐ Art & Culture
- ✓ Enogastronomy
- ✓ Landscape
- ☐ Environment
- ☐ Ecology

Fig. 17.1. NutKing interface to define the trip characteristics.



5 Locations found.

The elements found meet all rese
it is for your request (for more inf
To include a location in your trave

TESERO

Valle di Fiemme

Situated on terraces facing the Lagorai mountain range, Tesero is called the citadel of the Fiemme craftsmanship. This is expressed in various fields, from artistic to productive, and is constantly growing; renowned the musical instruments (organs and pianos), the furniture, doors, sport articles ... more >

Altitude: 1000

Activities:

MORI

Vallagarina, Rovereto, Altopiano di Brentonico

Hamlets: Besagno, Loppio, Manzano, Nomesino, Pannone, Valle San Felice, Sant'Antonio, Varano. This old village of ancient origins is documented for the first time in 845; in the neighbourhood there are remarkable prehistoric areas (Corno,

Travel preferences		Bretty's travel preferences:	Your preferences:
Travel companions		travel with family	travel with family
Accommodation		stay in hotel	stay in hotel
Budget		budget between 20 and 40 €	budget between 20 and 40 €
Transport		car	car
Interests:	Sports		
	Adventure		
	Relaxing	✓	✓
	Art and Culture		✓
	Whine and Food	✓	✓
	Enviroment and Landscape	✓	✓
	Fitness and Wellness		✓

Fig. 17.2. List of recommended products.

- Mobile users do not like to input lots of data; rather, they prefer answering to Yes/No questions with one click.
- A mobile user's interaction session should be kept (very) short in time.
- Mobile travellers should receive a useful recommendation within about 2–3 recommendation cycles.
- 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, MobyRek, provide an integrated solution to support travel decision choices.

3. On-tour Support

3.1 User preferences representation

A travel product is represented as a vector of feature values $x = (x_1, \ldots, x_n)$, where a feature value x_i may be numeric, nominal or symbol-set. Hereafter, we shall illustrate system functionality using a restaurant recommendation as an example. For the sake of simplicity, we represent a restaurant with six features (in reality, MobyRek describes a restaurant with 15 features): Name (nominal), Type (symbol-set), Location (nominal), MaxCost (numeric), OpeningDays (symbol-set) and Characteristics (symbol-set). Hence, the restaurant x = ('Pizzeria Ristorante al Vesuvio', {pizzeria}, 'Trento', 10, {1,2,3,4,6,7}, {parking, animals allowed}) has name x_1 = 'Pizzeria Ristorante al Vesuvio', type x_2 = {pizzeria}, location x_3 = 'Trento', maximum cost $x_4 = 10$, opening days x_5 = {1,2,3,4,6,7} and characteristics x_6 = {parking, animals allowed}.

To produce recommendations personalized to a particular user, recommender systems need a representation of the user's preferences. Preferences vary from user to user; and even the same user, in different situations (sessions), may have different preferences. A user's preferences are represented as a composite query containing three components: logical query, favourite pattern and feature importance weights vector.

- The logical query (Q_L) models must-have conditions that need to be independently satisfied by any of the products recommended. The logical query is constructed by a conjunction of logical constraints:

  $$Q_L = c_1 \cdots c_m$$

  where c_j is a constraint on a feature. Each feature type has a corresponding constraint representation. A constraint deals with only one feature, and a feature appears in only one constraint.
- The favourite pattern (p) models wish conditions that are expected to match as much as possible the products recommended. Differently from 'must-have' conditions, wish conditions allow trade-offs to be made. The preferred pattern is represented in the same vector space in which travel products are present:

  $$p = (p_1, \ldots, p_n)$$

  where p_i is a preference value for the ith feature; and x_i and p_i belong to the same feature type, $\forall_i = 1 \cdots n$. A value p_i may be unknown (denoted as '?') to indicate that the MobyRek system does not know about the user's preference on the ith feature. Such unknown values are, therefore, ignored in the similarity computation.
- The feature importance weights (w) model how much each feature is important with respect to the others:

  $$w = (w_1, \ldots, w_n)$$

  where w_i ($\in$ [0,1]) is the importance weight of the ith feature.

For example, the representation of a user's preferences $< Q_L = (x_3 = \text{'Trento'}) \wedge (x_5 \supseteq \{7,1\})$, p = (?,{spaghetteria},?,?,?,?), w = (0, 0.6, 0, 0.4, 0, 0)> indicates that the user is interested in only those restaurants in Trento that are open on Saturday and Sunday, and he or she prefers spaghetti restaurants to the others. The user considers the feature 'Type' as most important, the cost feature as second most important and the remaining aspects as unimportant.

When a traveller is on the move and starts searching for a desired travel product, MobyRek builds an initial representation of the user's preferences exploiting different knowledge sources of the user's data (i.e. the past on-tour product selections, the pre-travel plan, the spatial–temporal constraints and the initial preferences explicitly specified). The system's initial representation of the user's preferences (i.e. the initial query) is refined as the user proceeds with the dialogue, where MobyRek proposes candidate products and the user criticizes them or accepts one.

3.2 On-tour recommendation process

In our application scenario, on-tour support is offered when a traveller who has possibly built a pre-travel plan before leaving is at the selected destination (or on the move towards it). On-tour support is provided by the MobyRek system in cooperation with the pre-travel planning aid system (NutKing). The cooperation between these two systems allows the pre-travel information (in terms of knowledge of the user's decisions) to be exploited in the process of providing on-tour support.

An on-tour recommendation session starts when an on-the-move traveller requests the MobyRek system to find some desired travel products and ends when the traveller either selects a travel product or gives up the current session with no product selected. The recommendation process evolves in cycles. At each recommendation cycle, the system's representation of the user's preferences is used to produce a set of recommended products that are presented to the user. MobyRek models the on-tour recommendation process as in Fig. 17.3.

A recommendation session is initiated by the traveller's request for a generic product recommendation (e.g. 'I need a restaurant for lunch'). The

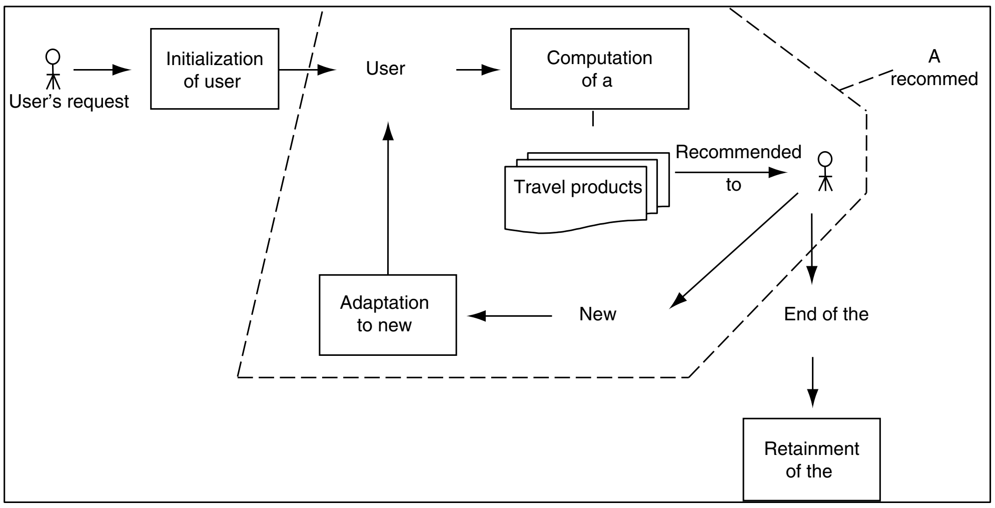


Fig. 17.3. The on-tour recommendation process.

MobyRek system builds an initial representation of the user's preferences without querying the user and is aimed at minimizing the user input (effort). In MobyRek, the logical query (Q_L) is initialized using the spatial–temporal constraints detected from the traveller's position and current time. The preferred pattern (Q_S) is initialized as a blank pattern (i.e. the preference value for every feature is unknown). The feature importance weights (w) are initialized in such a way that all the features are considered equally important.

The MobyRek system uses the initial representation of the user's preferences to produce the first recommendation set. In the computation of a recommendation, those products not satisfying the logical query (Q_L) are first excluded, and only those Q_L-satisfying products are ranked according to their similarity to the favourite pattern (p) taking into account the feature importance weights (w). Those (Q_L-satisfying) products most similar to (p) appear at the top of the ranked list; in the case where two products score the same, the least expensive product appears first in the ranked list. Only the k best products in the ranked list are shown to the user as recommendation result for the current cycle. The cut-off value (k) is a system parameter which is determined so that the recommendation list fits the screen size of the traveller's mobile device. Hence, the traveller can easily and quickly consider the recommendation list.

With a recommendation list, the user is supposed to browse the details of these products. In response, the user can execute one of the three actions: selection, critique or quit. A *selection* action is done when the user is satisfied with one of the recommended products. This selected product is added to the travel notes, and the current session ends successfully. A *critique* action is done when the user is somewhat interested in one of the recommended products, but unsatisfied with one (or more) feature(s) of this product. By criticizing the interested product, the user exposes preference on the unsatisfactory features. The MobyRek system uses such critiques to refine the previously incorrect representation of the user's preferences. Based on the refined representation, a new recommendation set that is expected to be closer to the user's real needs is produced. MobyRek's adaptation to a critique depends on the type of that critique and on the type of the feature criticized.

A *quit* action is made when no recommended products satisfy the user and he or she does not want to proceed with the dialogue. The session terminates with a failure. When an on-tour recommendation session finishes, either successfully or with a failure, it is retained as a case for future references. In this way, past recommendation sessions can be exploited by the system in building the initial representation of users' preferences. We are in the process of addressing this problem (i.e. building the system's initial representation of users' preferences) by exploiting different available knowledge sources of users' data (Nguyen and Ricci, 2004).

When the user criticizes a recommended product, it means that the criticized product interests the user but lacks some features (e.g. parking or live music) or some particular one is unsatisfactory (e.g. the price is too high). In our model, a user critique is supposed to express 'must-have' or 'wish-to-have' conditions, thereby indicating that the feature is either a non-compensatory or a compensatory decision criterion (Edwards and Fasolo, 2001). Non-compensatory criteria are such conditions that should be satisfied completely and independently; whereas compensatory ones are such conditions that would be satisfied to some extent (i.e. not necessarily to be completely satisfied) and one can be traded off against another. Non-compensatory criteria are encoded in MobyRek as logical constraints in Q_L, whereas compensatory ones are encoded in the favourite pattern (p) and the feature importance weights (w).

An important issue that should be considered and verified carefully is the number of critiques per cycle. In our approach, the user can express only one critique per cycle. In other words, after the user has criticized one product, with respect to one feature, the system acquires this input and recomputes the recommendation list. One may argue that the system should allow the user to express all the critiques before revising the user's preferences representation and producing a new recommendation set. Our design choice (i.e. to acquire only one critique per cycle) is motivated by some characteristics of the user behaviour and the mobile context. First, at the time of criticizing, the user usually does not know perfectly the distribution of products available in the catalogue; hence, as new products are recommended to the user, these may change his or her mind and suggest new preferences. In many real recommendation sessions, it is not surprising to observe that a user starting with some preferences ends with (very) different ones. Second, users are usually not good at making multi-objective decisions. In fact, users usually find it very difficult to take decisions that involve more than one feature where they must consider simultaneously both: (i) trade-offs between different values of the features and (ii) the probability that alternative outcomes occur. Third, mobile users typically do not like to have to input a lot before seeing something interesting. In particular, users only make explicit their preferences when they are convinced that they will immediately benefit from that. Hence, a mobile recommender system should play a more active and interactive role in the user–system dialogues, rather than wait (i.e. with no response) until the (mobile) user makes explicit all his preferences. Because of these characteristics, the user can criticize only one feature of a single product; after that, MobyRek immediately incorporates the critique to produce a new recommendation set. The adaptation method proposed here can be described as the instance-to-instance learning mode in the machine learning domain (as opposed to the batch learning mode).

More formally, the user can express his or her preference regarding a feature of a criticized product using the following functions:

F1 – Positive critique on a nominal feature. The user states that he or she likes the value l_i of the ith feature of a product $l = (l_1, \ldots, l_n)$. Then the new value l_i is assigned for the ith element in the favourite pattern (p):

$$p_i = l_i$$

F2 – Positive critique on a numeric feature: want less. The user states that overall he or she 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 the logical query (Q_L):

$$c_i \equiv (x_i \leq l_i - \delta)$$

where δ (>0) is an adjustment factor that allows to retrieve other products rather than the current one.

F3 – Positive critique on a symbol-set feature. The user states that he or she likes some values (Values_set_i) of the *i*th symbol-set feature (e.g. the feature 'Characteristics'), and wants to see some more products having these values for the *i*th feature. So the following constraint is included in Q_L:

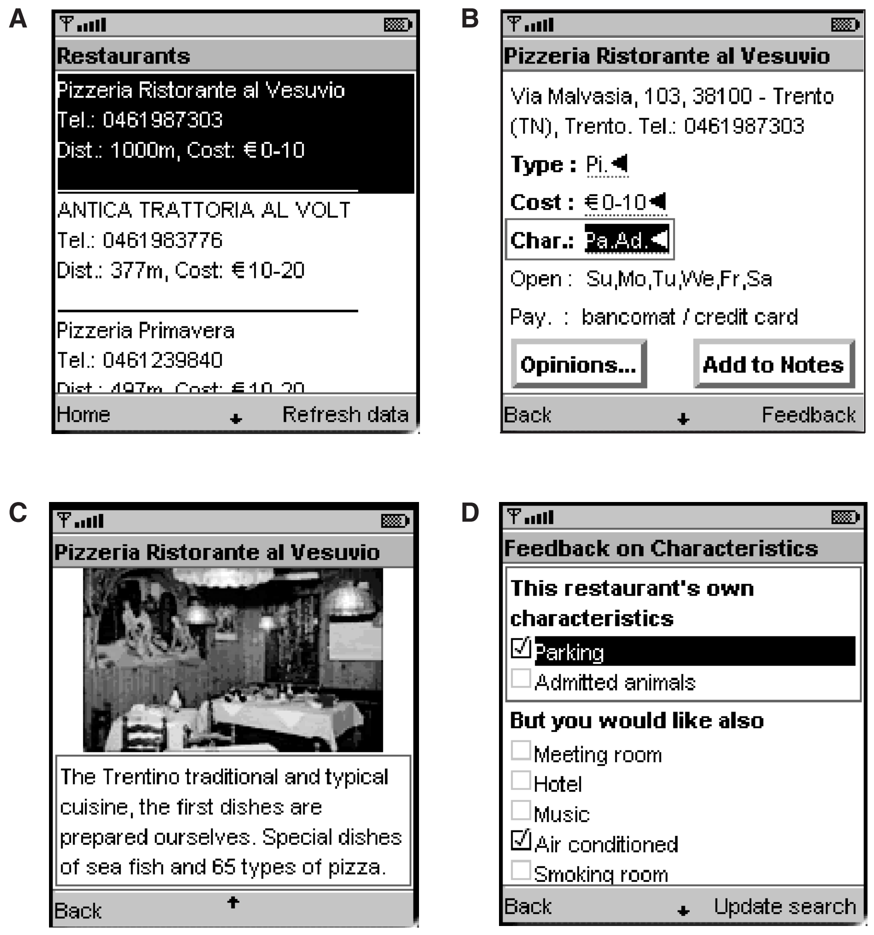
$$c_i \equiv (x_i \supseteq \text{Values\_set}_i).$$

The method of eliciting user preferences through critiques has two advantages. First, preferences are explicitly stated by the user and, hence, are much more reliable than those implicitly collected (e.g. by mining the user's navigation or browsing). Second, the user effort required in the user–system interaction is not as high as that required by some other methods of eliciting user preferences such as through interviews or early rating.

4. User Interface

We assume that an on-the-move traveller accesses MobyRek to look for a restaurant. MobyRek, as described above, exploits the traveller's current time and position (achieved via Global Positioning System (GPS) services) to initialize the representation of the traveller's preferences (i.e. speaking more precisely, to initialize the logical query (Q_L)). The initial representation of the traveller's preferences is used to compute an initial recommendation list (as shown in Fig. 17.4A). Then the traveller can view detailed information about the listed restaurants. Figure 17.4B shows the attributes of the 'Pizzeria Ristorante al Vesuvio' restaurant, and Fig. 17.4C shows the restaurant's brief description and customers' opinions. If the traveller accepts this recommendation, he or she can add it to the travel notes (a convenient container of all his or her selections); otherwise, he or she can criticize one recommendation (Fig. 17.4D).

In this example, the traveller criticizes the feature 'Characteristics', by checking the two characteristics 'Parking' and 'Air-conditioned' to indicate that he or she is interested in these characteristics (F3 critique in Section 3). Note that only the first characteristic ('Parking') is available at the viewed restaurant, whereas the second ('Air-conditioned') is missing. The critique is then exploited (i.e. incorporated in the representation of the traveller's


A

Restaurants

Pizzeria Ristorante al Vesuvio
Tel.: 0461987303
Dist.: 1000m, Cost: €0-10

ANTICA TRATTORIA AL VOLT
Tel.: 0461983776
Dist.: 377m, Cost: €10-20

Pizzeria Primavera
Tel.: 0461239840
Dist.: 497m, Cost: €10-20

Home Refresh data

B

Pizzeria Ristorante al Vesuvio

Via Malvasia, 103, 38100 - Trento (TN), Trento. Tel.: 0461987303

Type : Pi.

Cost : €0-10

Char.: Pa,Ad.

Open : Su,Mo,Tu,We,Fr,Sa

Pay. : bancomat / credit card

Opinions... Add to Notes

Back Feedback

C

Pizzeria Ristorante al Vesuvio

The Trentino traditional and typical cuisine, the first dishes are prepared ourselves. Special dishes of sea fish and 65 types of pizza.

Back

D

Feedback on Characteristics

This restaurant's own characteristics

- [✓] Parking
- [ ] Admitted animals

But you would like also

- [ ] Meeting room
- [ ] Hotel
- [ ] Music
- [✓] Air conditioned
- [ ] Smoking room

Back Update search

Fig. 17.4. MobyRek's graphical user interface.

preferences) when the traveller clicks on the button 'Update search'. In reply, a new recommendation list with three products is computed where the requested characteristics, if available, are present.

5. Evaluation

The evaluation involved six test users with no experience in applications running on WAP- or Java-enabled mobile phones. The test procedure, which each test user was asked to follow, consists of three phases. In the first phase (i.e. *training*), the test user is introduced to the simulator usage (e.g. the meanings of the buttons, the meaning of the contextual commands and how to execute them). Next, they are introduced to the system functionality and usage through a sample of a recommendation that demonstrates how the prototypical system works. The test users familiarize themselves with the simulator before solving a real task. We note that using the simulator is slightly more difficult than using a real mobile 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 to some number keys of the PC keyboard. This causes some difficulties. To test the prototype, test users associate icon buttons on the simulator interface (i.e. on the PC screen) with keys of the PC keyboard; however, on a real mobile phone, the user presses physical buttons on the phone's keypad.

In the second phase (i.e. *testing*), the test users are asked to think about the attributes of the desired restaurant and then to try to use the system to find one such. If the test users can find the restaurant, they are then asked to add it to their travel notes, and to open these notes to look at that restaurant.

In the third phase (i.e. *evaluating*), the test users are asked to complete a usability survey. The survey form consists of two parts: one for the test users' subjective evaluation of the system's performance; the other for their brief comment on the difficulties they face and on the desirable extensions to, and improvements of, the next system version. The test users are asked to state their agreement or disagreement on a 7-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 have found the restaurant that satisfies my needs.
6. I have 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 directly from the Post-Study System Usability Questionnaire (PSSUQ) following Lewis (1995); the remainder were added to obtain an evaluation on the new functionality specific to the on-tour travel recommendation setting. The results of the experiment are shown in Table 17.1.

All the test users stated that they were able to find such a restaurant that matches their needs (s_5). Moreover, all of them rated the found restaurant at the highest rating score. In addition, all the test users were able to complete the predefined task scenario (s_8). Finally, almost all the test users indicated that they would definitely use the proposed on-tour recommendation service if it is available on their mobile phone (s_9).

On-the-move travellers, who use some kind of mobile devices, need not only to find their desired travel products but also to find them quickly (i.e.

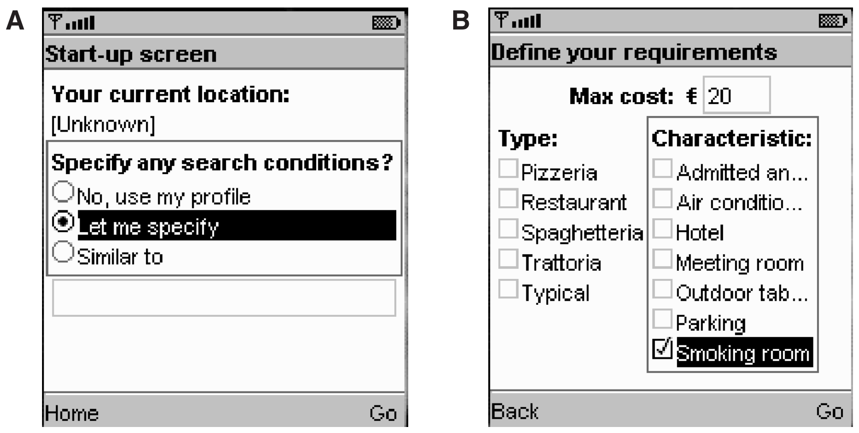
Table 17.1. The experiment results.

Statement	User						Average	Standard deviation
	u1	u2	u3	u4	u5	u6		
s_1	4	1	1	3	1	3	2.17	1.33
s_2	3	1	3	2	4	3	2.67	1.03
s_3	6	1	1	1	2	3	2.33	1.97
s_4	5	2	1	1	2	4	2.50	1.64
s_5	1	1	1	1	1	1	1.00	0.00
s_6	1	2	7	1	3	1	2.50	2.35
s_7	2	2	3	3	4	2	2.67	0.82
s_8	1	1	1	1	1	1	1.00	0.00
s_9	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

after a few recommendation cycles). By mining the log file (which records the test users' recommendation sessions), we saw that all the test users had found their desired restaurant within three recommendation cycles. It should be noted that a critique required only 2–3 button clicks. Therefore, being able to find the desired restaurant within three recommendation cycles (i.e. maximum of three critiques) is a convincing result.

However, the experiment results also show some critical issues that are extracted from their comments. First, some of the test users found it difficult to use the on-screen commands, which are embedded in the screen. In the traditional web interface, users execute a command by first moving their pointing device (e.g. mouse) to that command, and then activating it. As shown in Fig. 17.4B, to execute an on-screen command, users have to navigate through several display objects (using the device's 'go-down' button) and then activate the command (using the device's 'select' button). Second, some of the test users preferred to initialize the search by explicitly specifying some preferences (see Fig. 17.5). In a traditional web interface, users usually state their conditions before the system's retrieval. The first prototype of MobyRek, which was used in the experiment, automatically recommends candidate products (i.e. those cheapest amongst the restaurants close to the users' position). Hence, the first recommendation set is produced without any consultation with the users. This manner of producing the first recommendation set was not recommended by some testers who wanted to have more control on the search initialization process.

Some of the test users also preferred a longer (even the full) list of recommended products at each recommendation cycle. This seems to disprove the hypothesis that mobile users prefer to reduce the time and cognitive effort spent to fulfil a task. There could be two explanations for this decision. First, the search's goals and the acceptable trade-offs vary from user to user. Some users simply search for a 'good-enough' item while some others want to find the 'optimal' solution. Small recommendation sets could be acceptable for the


A

Start-up screen

Your current location:
[Unknown]

Specify any search conditions?
- ○ No, use my profile
- [✓ Let me specify]
- ○ Similar to

Home Go

B

Define your requirements

Max cost: € 20

Type:	Characteristic:
☐ Pizzeria	☐ Admitted an...
☐ Restaurant	☐ Air conditio...
☐ Spaghetteria	☐ Hotel
☐ Trattoria	☐ Meeting room
☐ Typical	☐ Outdoor tab...
	☐ Parking
	☑ Smoking room

Back Go

Fig. 17.5. Preferences initialization.

first class of users, but not for the second one. Second, some test users might be too familiar with traditional web interfaces that offer the possibility to browse complete lists of items. Therefore, they were surprised to find such a different interface. Third, all of the test users asked for a map-based navigation support to reach their selected restaurant from their position. This is a missing function in the prototype that should be supported in a next version of the system.

6. Conclusion and Future Work

In this chapter we have shown that MobyRek provides a usable approach to recommend tourism products to travellers when they are on the move to (or in the stay at) the selected destination. However, a number of open issues will need to be tackled in future work. First, we would like to perform a more extensive user evaluation: exploiting a real mobile phone to evaluate the system, and defining an appropriate methodology to assess the accuracy of the system in recommending products. Second, we would like to refine the system's initialization of the user preferences representation. In this initialization step, the system should exploit the knowledge about the user's preferences contained in past recommendation sessions, as well as the user's information and preferences that are specific to the current session and available at the time of the request. Third, we would like to improve the capability of the system to capture the users' preferences. In fact, the users' 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). A final issue we expect to tackle to improve the recommendations is an analysis of the relationships amongst the critiqued items in the same interaction session.

Chapter Summary

Users of travel and tourism websites often experience difficulty in selecting desired travel products. This difficulty is especially true for on-the-move travellers who use some kind of mobile devices to browse travel products information repositories. On the one hand, travellers are overwhelmed by a huge number of options to consider. On the other hand, travellers lack system support in filtering information and comparing amongst candidate products. Given the inherent limitations of the mobile usage context, mobile travellers need system support in making travel decision choices. In this article we present a computational approach for providing travel product recommendations to on-the-move travellers. This system employs a dialogue approach whereby a set of candidate products are proposed and the user is asked to critique the recommended products. Users' critiques elicited through dialogues are incorporated in the system's representation of users' preferences so that the system, step by step, better models users' needs. A prototype that implements the proposed approach is presented and the results of its empirical evaluation are discussed.

Author Queries

AQ1	Au: Please specify whether it is Ricci et al. 2002a, b, c or d.
AQ2	Au: Please specify whether it is Burke 2000a or b.