Case-based destination recommendations over an XML data repository

Francesco Ricci[1] and Hannes Werthner[1,2]

[1] eCommerce and Tourism Research Laboratory
ITC-irst
via Sommarive 18
38050 Povo, Italy
[2] University of Trento
via Inama
30100 Trento, Italy
{ricci,werthner}@itc.it

Abstract. This paper describes the objectives and the general architecture of an intelligent recommendation system aimed at supporting a leisure traveler in the task of selecting a tourist destination. The system enables the user to identify his own personalized destination by aggregating elementary items (locations, services and activities). Case-Based Reasoning techniques enable the user to browse a repository of past travels and make possible the relative ranking of the elementary items included in a recommendation. The system integrates data and information originating from external, already existent, tourist portals exploiting an XML-based information server and data mapping techniques. Moreover, user information needs are supported by a XML-query component that combines OLAP techniques and similarity-based retrieval.

1 Introduction

The term "tourist destination" refers to a complex concept. First of all, its spatial extension is known to be a function of the traveler distance from the destination. For instance, Italy is a destination for a Japanese tourist but not for a European traveler who will focus on the Alps rather than on a historical city or something else. Moreover, destination is not a mere geographical entity; it may be a wish or a collection of activities or experiences. Modeling this vague concept and the decision process that leads different users to their preferred destination is still an open and challenging research problem.

Current web-based systems can present, with multimedia content, an archive of "pre-defined" tourist destinations but they are not able to interact with the user satisfying his information needs, helping the user to express some possibly vague interests and finally delivering a personalized recommendation that collects a goal location and a set of additional services like lodging, car rental or cultural events.

Recommendation systems provide advice to users about products they might be interested in. Burke [5] distinguishes three types of recommendation systems: collaborative- or social-filtering; content-based and knowledge-based.

Amazon.com is a very popular example of eCommerce site that exploits a collaborative-filtering approach. At Amazon, data about a customer purchasing history are stored and book recommendations are compiled picking up books in the purchasing history of other customers with similar purchasing patterns.

News Dude [2], which is an example of content-based recommendation system, observes what online news stories the user has read and not read and learns to present the user with articles he may want to read. Content-based systems are classifier systems based on machine learning research [20].

The third type of recommendation system uses knowledge about users and the products to build up a recommendation. Knowledge may be expressed in the form of a detailed user model, a model of the selection and suggestion process and a rich description of the items to be suggested. These systems often integrate both collaborative- and content-based filtering techniques [14] and may be "conversational" [1, 9]. In contrast to a classical recommendation systems, which reply to a user query with a ranked set of results, conversational systems mimic a real dialogue between the "inquirer" and the "advisor" to solve a user need.

Furthermore, knowledge-based recommendation systems may be based on Case-Based Reasoning. Case-Based Reasoning (CBR) is a problem solving methodology that faces a new problem by first retrieving a past, already solved similar case, and then reusing that case for solving the current problem. In a CBR recommendation system [5, 9] a set of suggested items is retrieved from the case base by searching for cases similar to a probe case described by the user. A case in the memory may represent an item previously suggested or the union of an item and the user to whom the item was suggested (his profile data and the feedback given to the item suggestion).

In a CBR recommendation system the effectiveness of the recommendation is based on: the ability to match user preferences with item description; the tools used to explain the rationale for the match and to enforce the validity of the suggestion; the function provided for navigating the information space.

CBR is largely based on the notion and implementation of case similarity and similarity based retrieval from the case base. Case bases have been traditionally implemented using ad-hoc data structures loaded in memory at system start up. However, integration of CBR with RDBMS is becoming more and more important for many reasons, e.g., scalability; enterprise application integration; reuse of previous data. An initial set of works have therefore addressed this issue [18, 5, 9, 16]. The main technical problem consists in the compilation of a similarity-based request into SQL, which does not support natively this type of query. This problem has been studied in the data base community [7, 17] but it is still practically open in high dimensional spaces.

In this paper we propose a knowledge-based CBR recommendation system that addresses the following technical objectives:

- **XML data storage.** Cases are represented as semi-structured documents and are collected in an XML data store. In this setting, an XML query language [8] replaces SQL. Users view data in a mediated (simplified) schema and access data using a query by example interface. Standardized terminology for tourist applications are used (hotel descriptions, user profile, etc.).
- **Support for query refining.** The proposed system is highly interactive, that is, a query result may be criticized and refined in a mixed initiative approach. For instance, when the user wants to relax or tighten a constraint, the system suggests the most reasonable relaxation or the minimal changes in query constraints that will yield a result with a manageable size.
- **Scoring with local similarity metrics.** Local similarity functions re-rank a feature relevance (weight) based on the feature value [15]. Therefore, it is possible to increase a feature relevance under certain conditions (user input) and decrease that relevance in other situations. This behaviour is also called "context dependent", and better deals with typical context data such as: weather condition, time of travel, etc.

In this paper we describe the user needs and the basic layout of the proposed approach. Section 2 describes the working hypothesis for the destination model. Section 3 describes the system context and the logical architecture. From that description it emerges our main current concern, i.e., the query management component, that is discussed in Section 4. Finally, Section 5 comments briefly the current state of work.

2 Destination Modeling and Selection

A tourist destination may be modeled from different perspectives [19]. DMOs (Destination Management Organizations), for instance, promote and reshape a geographical destination managing a network of actors (service providers, ...) that contribute in providing the information content of a destination and finally depicting that in a tourist web site.

From the opposite side potential travelers (users of a web based information system) try to match this offered representation with their needs: geographical location, activities to perform in the location, services or events to consume, budget and timing constraints. The exact definition of a user-perceived destination is not clear. We believe that the elicitation of this rather vague concept calls for the adoption of an information tool, that is here described.

The tool must be initialized with a possibly simple model of the destination and must support the user in reshaping the destination aggregating simple elementary components. In a second step, having at disposal a historical repository of recorded human/computer interactions, it will be possible to build a new more sophisticated model as an evolution of the initial one.

Figure 1 shows the basic model as a UML class diagram [3]. The central building block is the TravelAsset class (TA in short). This is described by a LocationArea, an Activity performed, that may optionally consume a ServiceEvent

during a certain Timing. A LocationArea may have relations of spatial inclusion/containment with other LocationAreas. A ServiceEvent can be further specialized in a number of different sub-types, for instance: a lodging service, a climbing school, a recital, an exhibition. Analogously, Activity is the root of a hierarchy of traveler activities as: climbing, visiting, wandering, attending.

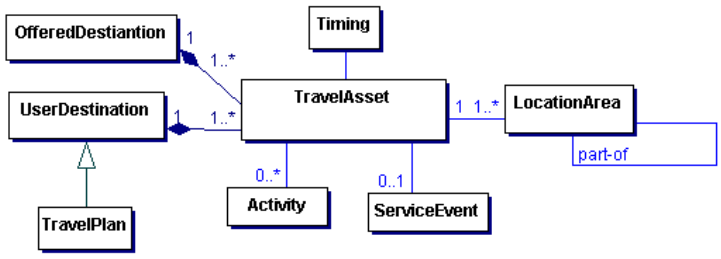

Fig. 1. Destination main model.

The second strong modeling assumption can be phrased by saying that a UserDestination is a user defined aggregation of those building blocks, i.e., a set of TravelAssets. In the same way, but possibly in a different way, the OfferedDestination is another aggregation of the same building blocks. The elicitation of the potential mismatch between the two destination types is one of the goal of our project. Finally, a TravelPlan is a UserDestination, i.e. a selected destination along with a set of service and activities bundled together in a coherent plan.

We note that the idea of representing a tourism service as an aggregation or elementary building blocks is shared by an IFITT initiative, the Reference Model Special Interest Group (RMSIG) [10]. Our model can be mapped to the RMSIG one and will reuse the basic components of that reference schema.

This paper mainly describes how the user is supported in the selection of the TravelAsset blocks from a catalogue and how this process leads to the constitution of a collection of "cases" that we call also TravelPlans. A TravelPlan bundles together some TAs, it is the unique entry point for all the information linked to the travel and a user history of TravelPlans provide the "fuel" for scoring the personalized recommendations (see Section 4).

Summing up, the user has two main modalities to obtain a destination recommendation:

1. **Direct Travel Asset Selection and Aggregation.** This is the initial modality, it enables the user to query a repository of TravelAssets. This is supported with OLAP techniques [6] and similarity based search and scoring (see Section 4).
2. **Case-Based.** In this second modality the repository of past suggested TravelPlans is searched, using the same tools quoted above, but here the user is

looking for travels made by other "similar" users. This will be described in a forthcoming paper.

We now describe the architecture of the system and the query component.

3 System Architecture

In this Section we describe the overall logical structure of the system. The main function of the system is enabling the user (leisure traveler) to access a repository of TravelAsset objects, querying this repository and building up a travel plan. Figure 2 shows the main components:

- **User Interface.** This module, together with the Dialogue Management and the Tourist Decision Management, is the principal responsible for user input analysis and content presentation. Focusing on the innovative aspects, the UI is based on a model of the decision process, extracted from a initial set of cases, that records real human/human interactions. The model determines the dialogue flow, i.e. the set of screens delivered to the user's browser, the user gestures that a screen can generate and the state machine that, given a screen and a gesture, outputs the next screen. The initial model will be successively refined using the logs of the interactions.
- **CBR Query Management.** This component processes queries posed by the user to the information server. It is used to both interact with the case base of Travel Plans and to select and filter TAs, whose descriptions originate from external tourist portals. This component is described in the next Section.
- **Presentation Personalization.** Information content is contained in the data sources and must be presented to the user in accordance to the user profile information. Profile data are collected during interaction, e.g. at the user registration or when constraints are specified in a query. The derived model (XML-based) enables the online personalization of the presentation using Natural Language automatic generation techniques [12].
- **Travel Plan Management.** These functions enable the user to store, retrieve, update and share his history of travels. Moreover, a travel plan can be exploited, during the travel, to further suggest new activities or comment on a TA that the user has in his plan and will probably consume. The user has the possibility to control this process, exposing or shading information contained in his travel plan.

The data originate from external systems and are integrated in the information server tier. Here the TA objects are stored (cached in materialized views) or mapped to the external repository using adapters. Moreover, all the Travel Plans are stored locally along with User Profiles and the similarity measures to score the searched objects (TAs or TravelPlans) in accordance to user's preferences.

The information server implements a mediator-based architecture [11]. The mediated schema is implemented as a set of Document Type Descriptions (DTDs)

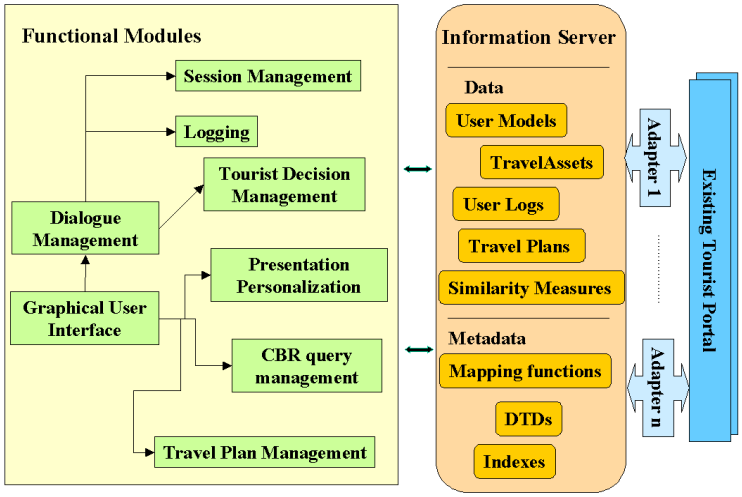

Fig. 2. System architecture.

[4] and a set of mapping functions between these DTDs and the original schemas used in the source repositories (or in the materialized views stored in the information server). The DTDs are the user's exported view over the data contained in the information server and in the external sources. They represent the bridge between the user's requests and the target data, shadowing the details of data distribution and data management (e.g. XML vs. relational tables or local vs. external).

4 Querying the Case Base

The "CBR query management" component collects input data from the user, builds a query on the mediated schema and then compiles a query plan that may involve both local and remote data. The results collected are then ranked using a similarity metrics that measures the "distance" between an item and the "ideal" solution. This pattern is used for two types of queries:

- **Range Scored Query.** In this case the query constraints are ranges of allowed values for a set of features and must be satisfied in the result set items. For instance: *Select all TA where Activity="lodging" and Service.type = "hotel" and hotel.cost < 60 and Location="Rome"*. Or, *Select all TA where Activity="canoeing" and Location.type="high-mountain or deep-canyon"*. In processing these queries, the system first retrieves all the TAs that satisfy the constraints and then these are scored by measuring the distance between a selected TA and an "ideal" TA that is contained in a past TravelPlan or that can be "synthesized" from the User Profile.
- **Nearest Neighbor Query.** In this second case the system does not try to match all the constraints but gives some result looking in a "neighbor"

of the partially defined TA which is specified in the query. The scoring is performed with the same approach as above.

Moreover, the query process is not linear, the user interacts with the system by looking at an initial result set and by refining the query that returned that result. The proposed approach is illustrated in Figure 3. Let us consider a typical scenario:

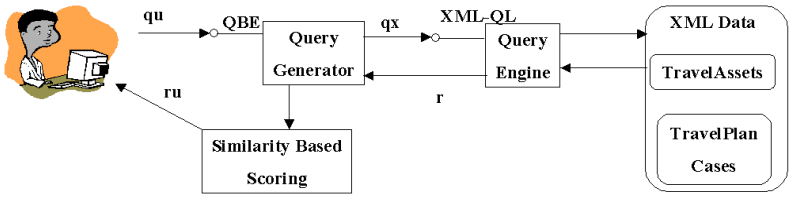

Fig. 3. Query processing.

1. The user enters a query q_u in a simplified query language, e.g., based on examples (QBE). The user does not have to know the query language used for the real data access.
2. The Query Generator, according to data mapping information, translates the user query q_u in the language used by the Query Engine, q_x, and send it to the engine. In this example the XML repository is accessed using the XML-QL query language [8].
3. The engine executes the query on the data specified and returns the result set r to the Query Generator.
4. The Query Generator analyzes the result. If it is "reasonable" (for instance the result size is less than a given $MaxN$) the result is scored by the "Similarity Based Scoring" component and shown to the user (r_u). Otherwise the result r and the original query (q_u, q_x) are used to relax or tighten a query constraint and to produce a new query (q'_u, q'_x). We emphasize that both a new user query q'_u and the corresponding translation into XML-QL, q'_x, are generated. That enables the user to understand what the system has done for him. The user may decide to accept the proposed changes or to further modify the query.
5. The new query q'_x is executed and a new result r' is produced. The interaction is now again at the stage described above and could loop until a break condition is satisfied.

We note that even if a Nearest Neighbor Query is issued by the user this is not executed directly on the target data set, but it is performed on the result set of an intermediary range query. This is a common approach used to deal with the computational complexity inherent to nearest neighbor queries to databases and XML data sets as well [7, 17]. We also believe that from the user point

of view (especially for TravelAsset selection) it is preferable to clearly separate what is "logically" satisfied in the result set from what is only partially matched in the result. We therefore believe that the user will first issue and refine a query that will constraint the result set with a set of "required" features and then will score the result using some "optional" features. For instance, if he is looking for an hotel, first he will select the hotels having a certain maximum price and certain required facilities (e.g. no-smoking room and AC) and then he will score the selected candidates by measuring the distance from an "ideal" hotel (e.g. the cheapest).

The known limitation of this approach is that a range selection may end up with no result at all. In our approach that could happen but in this case the system is able to suggest the "culprit". For instance, if the user issues the following query: *Select all TA where Activity="lodging" and Service.type = "hotel" and hotel.cost < 60 and Location="Rome"*. Should the result set be void, the system "looks" in a neighbor of the data selected by the query and determines that the culprit is the price range. It then suggests two repair options: increase the price bound to 100; or expand the location to an area surrounding Rome.

This function is enabled by: (a) segmentation of the information space in buckets; (b) indexing of the buckets. The system uses an indexing technique called bitmap indexes [13, 6] that have been introduced in OLAP systems. Bitmap indexes make possible to compute the size of a result set without actually executing the query and they can speedup multi-dimensional access to the data as needed by the example query shown above.

As a final issue we discuss how a set of results may be ranked using the case base, i.e., the library of Travel Plans that have been built and stored.

Let us a assume that a TA can be described as a point in $X = \prod_{i=1}^{n} X_i$ [1]. Each X_i is the space of possible values for the i-th feature. For instance, $X_i = \mathbb{R}$ for real features, like the price of a room or the latitude of a location. There might be also Boolean features ($X_i = \{true, false\}$) or other discrete symbolic features like the type of the "cusine" (e.g., $X_i = \{italian, french, arab, indian\}$).

A simple Range Scored Query may be expressed as a list of features and for each feature a set of allowed values (range): $x_1 \in R_1 \wedge \ldots \wedge x_k \in R_k$, where $R_i \subseteq X_i$. If $x = (x_1, \ldots, x_n)$ is a TA that satisfies the query range constraints then the score for x is computed as follow:

$$d_{min}(x) = \min_{y \in PTA} \sum_{i=1}^{n} w_i d_i(x_i, y_i),$$
$$Score(x) = 1/(1 + d_{min}(x)).$$

where PTA is the set of all TAs that appear in some past Travel Plan of the user, $0 \leq w_i \leq 1$ is a weight used to balance feature relative relevance, and d_i is the metric used for distance computation on each feature dimension (the details

[1] For the sake of simplicity we are limiting the discussion to linear data structures. What follows can be generalized to semi-structured data types (XML).

of d_i definition are omitted here). The above formulas imply that, the closer is x to some TA in a past travel, the better it scores.

The PTA set is empty before an history of interaction is collected and a case base is built. Therefore, at the beginning PTA will be identical for all users and will contain a set of ideal prototypes that represent generally optimal cases (for instance the hotel with best "quality/cost" ratio).

The features' relative importance in the scoring are determined by the weights w_i. Initially, i.e. when PTA is empty, we will use $w_i = 1$ if the i-th feature was specified in the query, and 0 otherwise. This incorporates a simple assumption, that is, the features specified as constraint matter to the user. After a case base was collected the weights can be more finely tuned to reflect user preferences using a methodology described elsewhere [15].

5 Conclusion

This paper describes a work in progress at a newly established research center on eCommerce and Tourism (http://ectrl.itc.it). The proposed approach will be implemented and validated on an operational system to be integrated in a tourist web portal. We are currently designing the proposed system and defining the interfaces with the external data providers.

The main result of the proposed approach is a comprehensive middleware for issuing personalized recommendation to a leisure traveler in finding a geographical destination and aggregating elementary services or activities to be consumed. It is based on the principle that suggestion effectiveness relies on a combination of factors like: appropriate destination modeling, data retrieval and filtering with both sharp and approximate matching, scoring using personal preferences stored in a base of previous cases.

A recommendation system based on that middleware will help in better understanding the user needs and in particular the possible mismatch between offered destination packages and user's wishes. Moreover, the case base of TravelPlans, which is an output of the user interaction with the system, will enable additional advanced functions like: TravelPlan composition with TravelAssets contained in other similar TravelPlans or support during the travel.

Acknowledgements

This project is partially funded by CARITRO foundation. We would like to thank E. Not and D. Fesenmaier for their feedback and comments.

References

[1] D. Aha and L. Breslow. Refining conversational case libraries. In *Case-Based Reasoning Research and Development, Proceedings of the 2nd International Conference on Case-Based Reasoning (ICCBR-97)*. Springer-Verlag, 1997.

[2] D. Billsus and M. Pazzani. A hybrid user model for news story classification. In *Proceedings of the Seventh International Conference on User Modeling, UM '99*, Banff, Canada, 1999.

[3] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language User Guide*. Object Technology Series. Addison Wesley Longman, Reading, Mass., 1999.

[4] T. Bray, J. Paoli, C. M. Sperberg-McQueen, and E. Maler. Extensible markup language (xml) 1.0 (second edition). W3C Recommendation 6 October 2000, 2000.

[5] R. Burke. *Encyclopedia of Library and Information Science*, chapter Knowledge-based Recommender Systems. ., 2000. To appear.

[6] S. Chaudhuri and U. Dayal. An overview of data warehousing and olap technology. *SIGMOD Record*, 26(1):65–74, 1997.

[7] S. Chaudhuri and L. Gravano. Evaluating top-k selection queries. In *Proceedings of the 25th VLDB Conference*, Edinburgh, Scotland, 1999.

[8] S. Cluet, A. Deutsch, D. Florescu, A. Levy, D. Maier, J. McHugh, J. Robie, D. Suciu, and J. Widom. Xml query languages: Experiences and exemplars. http://www.w3.org/1999/09/ql/docs/xquery.html.

[9] M. H. Göker and C. A. Thomson. Personalized conversational case-based recommendation. In *Advances in case-based reasoning: 5th European workshop, EWCBR-2000, Trento, Italy, September 6–9, 2000: proceedings*. Springer-Verlag Inc., 2000.

[10] W. Höpken. Reference model of an electronic tourims market. In *Information and Communication Technologies in Tourism 2000 - Proceedings of the International Conference (ENTER2000)*, pages 265–274. Springer, 2000.

[11] A. Levy. More on data management for xml. http://www.cs.washington.edu/homes/alon/widom-response.html, 1999.

[12] E. Not and M. Zancanaro. The macronode approach: mediating between adaptive and dynamic hypermedia. In *Proceedings of the International Conference on Adaptive Hypermedia and Adaptive Web-based Systems, AH'2000*, Trento, Italy, 2000.

[13] P. O'Neil and D. Quass. Encoded bitmap indexing for data warehouses. In *Proceedings of SIGMOD '97*, Tucson, Arizona, USA, 1997.

[14] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13:393–408, 1999.

[15] F. Ricci and P. Avesani. Data compression and local metrics for nearest neighbor classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(4):380–384, 1999.

[16] J. Schumaker and R. Bergman. An efficient approach to similarity-based retrieval on top of a relational database. In *Advances in case-based reasoning: 5th European workshop, EWCBR-2000, Trento, Italy, September 6–9, 2000: proceedings*. Springer-Verlag Inc., 2000.

[17] T. Seidl and H. Kriegel. Optimal multi-step k-nearest neighbor search. In *Proceedings of the ACM SIGMOD Conference*, pages 154–165, Seattle, WA, June 1998.

[18] H. K. H. Shimazu and A. Shibata. Retreiving cases from relational data-base: Another stride towards corporate-wide case-base systems. In *IJCAI-93*, 1993.

[19] H. Werthner and S. Klein. *Information Technology and Tourism - A Challenging Relationship*. Springer-Verlag, 1999.

[20] I. H. Witten and E. Frank. *Data Mining*. Morgan Kaufmann Publisher, 2000.