ADVISOR SUITE – A knowledge-based sales advisory system

Dietmar Jannach [1]

Abstract. In many domains, a large variety of comparable products and services from different manufacturers is available on the market. As a result, customers are increasingly overwhelmed when they have to choose the products that match their personal needs and preferences because the selection process in many cases requires deep technical knowledge about the product features. In the online sales channel, *recommender systems* have already been successfully applied as an additional means for improved customer guidance. Real-world sales advisory, however, can be a knowledge-intensive task, where the customer preferences and the suitable sets of products have to be determined in an interactive dialogue. In this paper we present ADVISOR SUITE, a now-commercialized system for building such intelligent, personalized sales advisory applications. The system consistently follows a knowledge-based approach where all required development activities - from the definition of the user model and the recommendation rules, up to the personalization of the dialogue flow as well as user interface generation - are based on simple conceptual models and declarative knowledge representation. All these tasks are also supported by a set of graphical tools comprehensible for non-programmers, which leads to significant reductions in development and maintenance costs. Experiences from several commercial instantiations of the presented system are discussed in the final sections.

1 INTRODUCTION

In many business sectors, customers can choose from an ever-increasing variety of alternatives for certain products or services as there are more and more manufacturers and service providers competing in today's globalized and open markets. In addition, the service providers themselves tend towards broadening their product catalogues in order to occupy new market niches or to address specific customer groups. Given such a wide range of choices, customers can become increasingly overwhelmed when they have to determine which one of the available products actually matches their specific needs, preferences, or desires. A simple but typical example can be found in the popular sector of *digital cameras*: via the online channel, customers can choose from hundreds of different models from dozens of manufacturers, whereby the information available on the vendor's web-pages mostly covers technical details on the product properties. Only in a few cases the customers' real needs with respect to their intentions or desired functionality is addressed; a means for comparing the alternatives is nearly always missing.

In such situations, many potential online customers will therefore prefer traditional sales channels and consult a sales agent in a local store who will elicit the customer's needs in an advisory dialogue and then propose and explain potentially fitting products and possible alternatives. The goal of the ADVISOR SUITE project and the now commercialized software system was to develop a means to easily transfer such an advisory service to the online channel, i.e., the customer shall be enabled to interact with an online sales assistance system that simulates the behavior of an experienced sales person. We identified different application scenarios for such systems: Analog to the classical sales channel, improved sales assistance generates customer value for the visitors of the web pages of vendors or e-marketplaces. The same advisory application can also be utilized to improve and standardize the internal quality of the sales process in a scenario where the sales representative steps through the dialogues with the customer. Finally, such an application can also be run on self-service terminals in a store, which reduces the workload of the sales assistants.

The presented system differs in several ways from the class of prominent and successful *recommender systems* [15] that base their proposals on collaborative filtering and item and user correlations. When we regard the target domains, the above mentioned systems perform best in domains like books or movies, where recommendations are influenced by factors like *quality* and *taste* and a sufficient number of customer quotes is available. The ADVISOR SUITE system follows a hybrid [4] approach and makes extensive use of explicit expert knowledge such that suitable recommendations can also be made for new customers and products. In addition, personalization in the ADVISOR SUITE system is not limited to the proposals themselves, but also covers dialogue flow and the presentation style. The main costs in a knowledge-based approach, however, arise in the knowledge-acquisition and maintenance phases - in particular, when also the personalization of the presentation is knowledge-driven and the user interface has to be adaptive. The ADVISOR SUITE system is completely based on such declarative knowledge representation and simple conceptual models and therefore comprises a consistent set graphical tools that allow us to minimize development and maintenance times throughout, starting from the acquisition of expert rules up to the definition of personalization rules and user interface parameterisation.

2 KNOWLEDGE-BASED SALES ADVISORY

Figure 1 shows an overview of the major kinds of knowledge used in the ADVISOR SUITE system. The *customer properties* are the variables in the advisory process that represent the current user's characteristics, skills, and interests. The corresponding values are determined by directly questioning the user, by incorporating information from external sources like a CRM[2] system, or by deduction

[1] Institute for Business Informatics and Application Systems, University Klagenfurt, A-9020 Klagenfurt.

[2] Customer Relationship Management

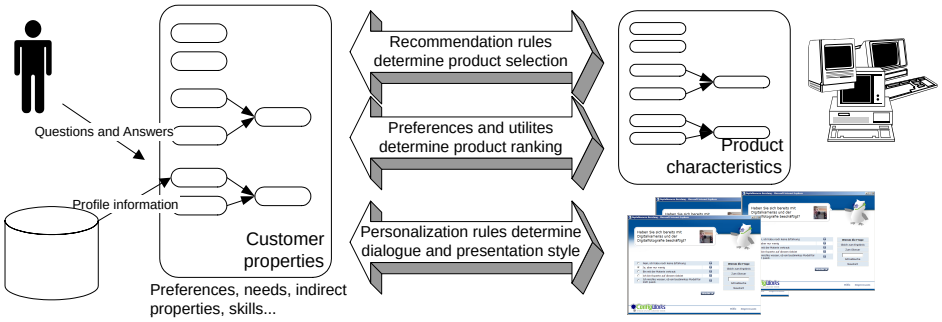

Figure 1. Overview of required knowledge.

from other variables. Similar to Constraint Satisfaction Problems (CSP)[14], each variable has a domain which can be a finite enumeration or a defined domain like real numbers, whereby variables can also have multiple values. Note that for directly acquired properties, each variable corresponds to a question and the allowed values represent possible customer answers. Since we explicitly exploit knowledge about products in the recommendation process, we also describe the product properties in terms of such variables.

The required knowledge for an advisory application is represented in different kinds of rules and constraints. All of this knowledge can be expressed either in a high-level constraint language over the variables of the problem (customer and product properties) or in a tabular representation style. Typical kinds of knowledge are, e.g.,

- *filtering rules*, like "If the customer is interested in no-name products, recommend products from vendor X or Y", or "Recommend products that match the user-defined price limit plus 10%",
- *tabular mappings*, determining the deduction of derived properties which, for instance, a novice user cannot be asked directly,
- *functions* that describe a product's utility with respect to the customer's preferences for ranking the proposals,
- *dialogue flow rules*, like "Proceed to the expert's selection page, if the user's self-estimate on his expertise is high", or
- *rules for the personalization of explanations*, like, "If the customer is a novice, display the following explanation...".

2.1 Product selection and ranking

Filtering. The selection of suitable products in our system is based on *filter constraints* relating customer properties with product characteristics [3]. We have chosen an "if-then"-style representation which is also understandable for non-programmers, i.e., the domain experts. A constraint consists of an expression on arbitrary customer properties (interests and preferences) describing conditions when a filter has to be applied. In the consequent, the corresponding restrictions on the technical product properties are articulated. All constraints in the system are entered using context-aware editors (see a simple example in Figure 2), whereby the system's language comprises relational, arithmetic, and logical operators between variables and constants.

One of the main criticisms of a filtering approach is that there might be situations where all products are filtered out. Therefore, we extended the filtering mechanism with a priority- and preference-based relaxation mechanism. Each filter rule is assigned an initial priority by the domain expert. Note that this is realistic, because in most domains there are both strict recommendation rules as well as "soft" rules reflecting the expert's opinions. At run-time, the recommender engine evaluates the filter constraints at the end of the dialogue. If no product remains, the filters are relaxed incrementally by priority (comparable to the Hierarchical Constraint Satisfaction approach [14]) until a defined threshold of products is reached. At the end, the resulting sets of applied and relaxed filters serve as a means for constructing an *explanation* [8] for the proposal that is presented to the customer. Good explanations are important in sales advisory systems, as they increase the customer's confidence in the recommendation. Therefore, for each rule that is visible for the customer, explanatory texts and personalized variants thereof in natural language can be maintained in the system, which are finally used to construct a set of arguments for the proposal. Note that the initial priorities defined by the domain expert do not necessarily match the preferences of the customer. Therefore, when the explanations are displayed in natural language, the customer can dynamically state his preferences on the priorities of the rules in the sense of [9] or state that individual rules should never be relaxed or never be applied. The reasoning process is then re-started and a new set of products is computed and explained.

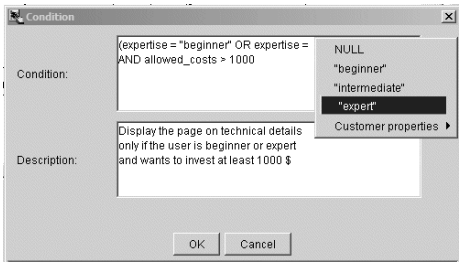

Figure 2. Entering a condition.

Ranking the products. For a personalized ordering of the products, we adopt a utility-based approach relying on MAUT (Multi-Attribute Utility Theory) [1, 12, 16]. Conceptually in between the customer and product properties, we introduce high-level *interest dimensions* like "economy" or "prestige" and define corresponding utility functions. During the advisory dialogue, the system develops an estimate of the customer's relative interests in these dimensions either by direct questions or by indirect reasoning on the basis of other customer properties. Given this information, for each product, we then compute an overall utility and a corresponding ranking that takes the personal interests of the customer into account. Note that the system is also open for the incorporation of other ranking mechanisms, e.g., based on other customers' ratings.

2.2 Personalized dialogues

A trained sales assistant will adapt his interaction style depending on, e.g., the customer's skill level and interests [1, 2] and ask different questions or provide explanations that are adequate for the customer.

In the ADVISOR SUITE system we exploit such expert knowledge in order to be able to build adaptive and personalized online advisory applications.

Personalization is a knowledge-intensive task [10]. Therefore, in order to simplify the process of making the needed knowledge for adaptive dialogues explicit, we developed a *conceptual model* of typical advisory dialogues. In the design of that model two points are important. First, it has to be simple, i.e., it should comprise only a few concepts such that the domain expert is not overwhelmed by a variety of terms. Second, we aimed at defining a model that is already related with the user interface in order to keep the gap between the conceptual model and the final application small. The main concepts of an advisory application are the following.

- An advisory application consists of a set of *dialogue pages*, whereby on each page one or more questions are displayed.
- Each page can have one or more successor pages, i.e., one can define conditions for each possible *transition*, whereby these conditions are again arbitrary expressions on customer properties.
- The whole dialogue can be organized in *phases* which are used to visualize the progress of the dialogue for the customer.
- A dialogue can also comprise *special situations* where the question/answer dialogue is actively interrupted by the advisor. Typical cases are, when the system should give a hint if the customer's answers are inconsistent or when specific information depending on the user properties has to be presented, e.g., for cross-selling purposes or as additional help for customers without deep technical knowledge.
- Finally, a *controller* page dynamically evaluates the transition conditions and manages the page flow.

Based on customer feedback, we did not rely on standard techniques like Petri Nets or UML (Unified Modeling Language) State Transition Diagrams as notation for modeling the application dynamics and user interaction, because terms like "state" or "transition" were harder to comprehend for people with no IT background. Nonetheless, the employed user-oriented notation has a defined semantics that allows us to automate the user interface generation process which is sketched in a later section. Figure 3 shows the dialogue modeling tool of ADVISOR SUITE, where the dialogue flow is modeled graphically.

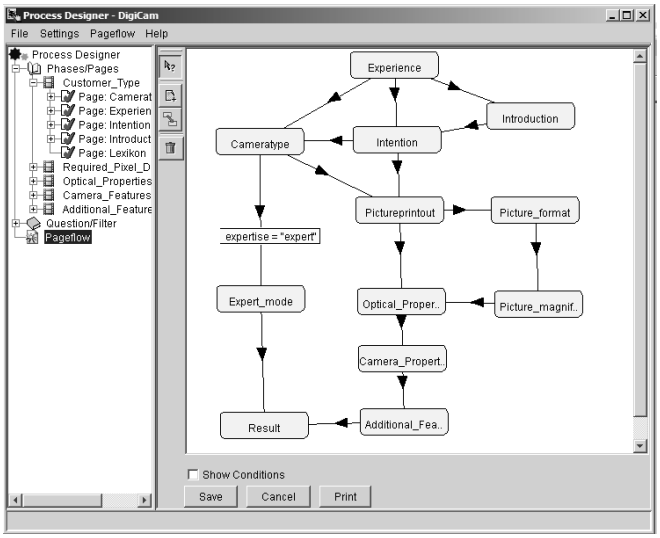

Figure 3. Dialogue modeling tool.

Beside adaptive dialogue flows, the ADVISOR SUITE system also supports the personalization of the content to be displayed. First, multi-lingual versions of the individual pieces of information can be maintained for the different customer groups. This multilinguality feature is also used for defining dialogue versions for different target groups like experts and novice users or for different (e.g., formal or informal) communication styles. In addition, personalized variants for content fragments like explanation texts can be maintained, i.e., conditions over customer properties that determine which version should be selected. At run time, the system evaluates the current customer's properties and displays the appropriate variant, whereby these text fragments can also contain HTML code, which allows us to personalize the presentation style by using different layouts or images.

In general, one of the main benefits of the presented approach lies in the fact that our conceptual approach for dialogue modeling allows us to clearly separate the layers for interaction logic, content selection, as well as presentation style.

3 ARCHITECTURE & WEB USER-INTERFACE

In Figure 4, an overview of the architecture of the ADVISOR SUITE system is given. All of the required knowledge is modeled using graphical tools and stored in a central repository on top of a relational database system. The advisory application itself consists of a set of dynamic web pages (Java Server Pages) that communicate with the *Interaction & Personalization* agent and a central server module that retrieves information from the repository.

User interface (UI) development and maintenance are critical tasks for personalized and adaptive applications, because the individual pages have to be generic, i.e., no changes in the UI should be required if the personalization or recommendation logic in the knowledge base changes. On the other hand, when the user interface construction process is automated and parts of the application are automatically generated, it is fairly important that the design of the resulting web pages can be easily extended and adapted by a web developer without rich programming skills.

In the ADVISOR SUITE system, we follow an approach where the core of the advisory application is automatically constructed using a template mechanism such that the initial version of the application is generated from the repository without any programming. Layout changes can then be carried out by adapting the small templates or by adding HTML code at pre-defined hooks. These changes are not lost when the application is extended afterwards, e.g., when a new dialogue page has to be added. In order to keep the resulting dynamic pages simple, we also make extensive use of *Custom Tags* [7], a standard mechanism to resolve the problem of the mixture of HTML-code and Java-code in dynamic web pages. Figure 5 shows a part of a template consisting of standard HTML and custom tags, where a question and its predefined answers are displayed. This template can be freely extended with HTML code. The internal logic like the choice of the correct language or the transfer of inputs to the interaction agent are hidden from the web developer. The provided tags also pre-load the defaults defined in the knowledge bases or handle "undo"-actions when a backward navigation is done and another path in the dialogue is taken.

4 EXPERIENCES AND APPLICATIONS

Several instances of applications built with ADVISOR SUITE were already developed and deployed for more complex domains like in-

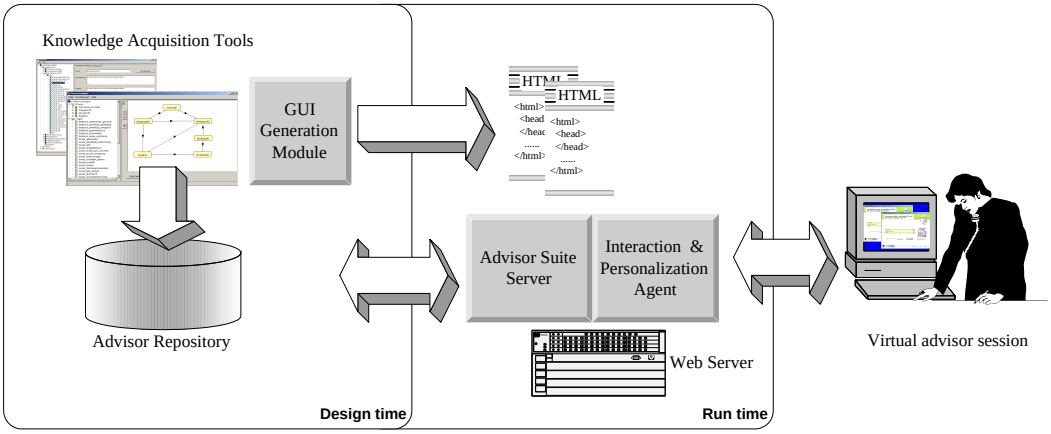

Figure 4. Architecture overview.

```
<form>
  <advisor:question name="$QUESTION_NAME$">
      <advisor:questiontext/><P>
      <advisor:answers>
        <!-- Iterate over the answers.. -->
        ...
      </advisor:answers>
    </advisor:question>
 </form> ...
```

Figure 5. Listing of template code.

Figure 6. A generated dialogue page.

vestments advisory in the financial sector, for ”technical” goods like digital cameras and skis, as well as for ”quality-and-taste” domains like premium cigars. The most important aspect we encountered was that the development times - and the corresponding costs - for the core application were in fact very short when following the declarative, knowledge-based approach. In most cases, the initial knowledge base including the expert rules and the dialogue flow could be defined in a few workshop days. In this initial setup phase, the provided tools were typically not directly used by the domain expert, but rather by a knowledge engineer who discusses and formalizes the expert's knowledge. Later on, however, the maintenance tasks for the knowledge base did require the help of an engineer only in a few cases. Quite interestingly, the size of the knowledge base in most domains remains manageable, i.e., there are around twenty to thirty questions and only a few dozens of expert rules. Even in complex domains like investment advisory, where the set of suitable alternatives is determined by many parameters, the maintenance of the knowledge base was not problematic. For these more complex application domains, we implemented several structuring mechanisms allowed us to build modular knowledge bases which was appreciated by the knowledge engineers and domain experts.

The development time for the final user interface varies with the requirements of the company that provides the advisory application (mostly as add-on to an existing web site). In some cases, we could directly use one of the pre-defined standard layouts developed for ADVISOR SUITE with an animated avatar (see Figure 6), which reduced the UI-programming part to changes in color and style. Our experiences also show that the structure of the generated pages and the templates is simple enough such that the pages can be easily adapted in order to fit the corporate design of the provider company. An important feature that was appreciated by the online users is the possibility to explore different paths in the application, i.e., go forward or backward in the dialogue, revise answers, or change priorities for filters. In general, although the dialogue paths themselves are partially predefined, the usability tests showed that users have to be given certain degrees of freedom in the navigation, for instance to jump to the result page with the answers given so far or by skipping individual phases of the dialogue. Some interaction preferences we identified were the standard ”Question&Answers”- style, product selection on basis of technical parameters, or customers who want to know whether a certain product fits their needs, i.e., they already have a certain product in mind, or customers who want to *compare* a product with different alternatives.

One of the major aspects that influence acceptance of advisory applications by end users lies in the quality, up-to-dateness, and completeness of the underlying product data. In most cases, major parts of the data is already available in electronic form. For instance, if the advisory application runs as add-on for a web shop, the basic data is typically contained in a relational database. For these cases, the ADVISOR SUITE system provides XML-based interfaces for product data transfer and periodic updates or it can be configured to directly access data in an external database. Nonetheless, in some cases the product data had to be enriched manually with additional properties that were not already available. Such updates are typically done on a periodical basis when new products are available. On the other hand, changes in the rules do not occur that often; for technical products, rules that describe ”*what is a high resolution*” or ”*what is a typical memory size*” have to be revised only a few times a year.

From our projects, we also see that customer satisfaction significantly increases when different sorts of additional information can be accessed from within the advisory application like a glossary or a discussion forum, links to vendors and related products, or a free text search facility. For one project, a study about customer satisfaction with more than 1.600 participants showed that around 90% of the users appreciated the quality of the recommendations and explic-

itly asked for the extension of the advisory service to other product groups on the e-Commerce site. From the business perspective, our partners that host the advisory applications report a measurable increase in sales and inquiries over the online channel already a few weeks after online deployment of the system. This increase can be explained by a higher confidence of the customers in their buying decision. As a side effect, when a company runs an advisory application, there is also new information that becomes available for them. Since all the answers of the online users and their navigation behavior can be logged and evaluated online, companies can learn about their customers and their preferences for improved Customer Relationship Management (CRM), for instance *"How many of my online customers consider themselves as experts*?" or *"Which features of a product are important for my customers?*". On the other hand, the information in the logs can also be used to analyze and improve the advisory application itself, e.g., *"Is there a page that is typically skipped by the users?*".

Technically, the complete system is developed using Java and standard Java-based technology and runs on various platforms as a standard web application. Since there may be many parallel users, performance can be a potential bottleneck. As an example, in a project with an advisory system on one of Austria's largest commercial sites with respect to user visits, there were around twenty thousand full advisory sessions within the first days that used our virtual advisor for digital cameras. In the ADVISOR SUITE system, we address this problem by extensive data pre-loading as well as pre-compilation of expert rules, transition conditions, and expressions before deployment.

5 CONCLUSIONS AND FUTURE WORK

Online sales advisory is a promising means for companies to provide added value for their customers and visitors of their web sites. We have presented an integrated knowledge-based approach that combines several techniques both for product selection, personalized ranking, as well as for the rapid development of adaptive and personalized web applications. The knowledge acquisition and maintenance problem is addressed by the provision of adequate tools and mechanisms throughout the development process.

Our future work includes improved tools for offline and online analysis of the interaction data. Although more than eighty percent of around eighty-thousand logged customer sessions in one project were successful, i.e., the customers stepped through the whole dialogue to the result page at least once, we expect that such analysis can give us hints how to shorten the dialogue without losing quality in the solution [11, 13]. Product proposals can then be made more quickly without a long preference elicitation process once the system has obtained a good *estimate* of the user's profile and preferences. Currently, the number of pages in one dialogue path typically ranges from seven to fifteen. Our experience shows that shorter dialogues give the user the impression the system does not ask enough questions about the requirements; in longer dialogues, users often are impatient and simply leave the proposed defaults as input.

We also intend to incorporate collaborative filtering and web mining [5] techniques for that purpose where we can exploit data from other customers, in particular their feedback whether they found that the proposals were adequate. Such feedback can be obtained either by direct questions or by behavior analysis, e.g., did the user click on the detailed fact sheets of the proposed products, did he go back and revise the answers and so forth.

Further work will also be spent on the development of a *learning* agent, that analyzes the current customer's behavior, compares it with similar sessions and correspondingly adapts the logic of the dialogue, proposes other defaults than the predefined ones etc. Although there are systems like described in [6] that mimick a more sophisticated natural language or free-flowing dialogue, these systems typically require significant knowledge acquisition efforts and may also cause user frustration, since users presented with a generic "agent" interface often attribute much more intelligence to the system than is warranted. Other extensions to the current system that improve the customers' confidence in the proposals will be in the area of *product comparison.* We will evaluate and develop techniques that allow us to automatically compute intelligent comparisons (and explanations) based on product characteristics, answering questions like *"In which features do the proposed products differ?*". The explanation of differences and commonalities depend again on the customer's profile and preferences, i.e., we need to analyze whether the differences are important for the current customer or not.

REFERENCES

[1] L. Ardissono, A. Felfernig, G. Friedrich, A. Goy, D. Jannach, G. Petrone, R. Schfer, and M. Zanker, 'A framework for the development of personalized, distributed web-based configuration systems', *AI Magazine*, **24**(3), 93–110, (2003).

[2] L. Ardissono and A. Goy, 'Tailoring the interaction with users in Web stores', *User Modeling and User-Adapted Interaction*, **10**(4), 251–303, (2000).

[3] Derek Bridge, 'Product recommendation systems: A new direction', in *Procs. of the Workshop Programme at the Fourth International Conference on Case-Based Reasoning*, eds., R.Weber and C.G.von Wangenheim, pp. 79–86, (2001).

[4] R. Burke, 'Hybrid recommender systems: Survey and experiments', *User Modeling and User-Adapted Interaction*, **4**, 331–370, (2002).

[5] Y. H. Cho, J. K. Kim, and S. H. Kim, 'A personalized recommender system based on web usage mining and decision tree induction', *User Modeling and User-Adapted Interaction*, **23**, 329–342, (2002).

[6] A. M. Garcia-Serrano, P. Martinezz, and J. Z. Hernandez, 'Using AI techniques to support advanced interaction capabilities in a virtual assistant for e-commerce', *Expert Systems with Applications*, **26**, 413–426, (2004).

[7] J. Goodwill, *Mastering JSP Custom Tags and Tag Libraries*, Wiley Publishers, 2002.

[8] U. Junker, 'Quickxplain: Conflict detection for arbitrary constraint propagation algorithms', in *IJCAI'01 Workshop on Modelling and Solving problems with constraints*, Seattle, WA, (2001).

[9] U. Junker and D. Mailharro, 'Preference programming: Advanced problem solving for configuration', *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **7**(1), 13–29, (2003).

[10] A. Kobsa, J. Koenemann, and W. Pohl, 'Personalized hypermedia presentation techniques for improving online customer relationships', *The Knowledge Engineering Review*, **16**(2), 11–155, (2001).

[11] N. Lesh and O. Etzioni, 'Using plan recognition in human-computer collaboration', in *Seventh Intl. Concerence on User Modeling*, Wien, New York, (1999). Springer Verlag.

[12] G. Linden, S. Hanks, and N. Lesh, 'Interactive assessment of user preference models: The automated travel assistant', in *Sixth Intl. Concerence on User Modeling*, Wien, New York, (1997). Springer Verlag.

[13] D. McSherry, 'Increasing dalogue efficiency in case-based reasoning without loss of solution quality', in *18th Internationl Joint Conference on Artificial Intelligence*, pp. 121–126, Acapulco, Mexico, (August 2003). Morgan Kauffman.

[14] T. Schiex, H. Fargier, and G. Verfaille, 'Valued constraint satisfaction problems: Hard and easy problems', in *Internationl Joint Conference on Artificial Intelligence*, pp. 631–639, Montreal, Canada, (1995).

[15] J. B. Shafer, J. Konstan, and J.Riedl, 'Recommender systems in e-commerce', in *ACM Conference on Electronic Commerce (EC-99)*, pp. 158–166, (2001).

[16] D. von Winterfeldt and W. Edwards, *Decision Analysis and Behavioral Research*, Cambridge University Press, Cambridge, UK, 1986.