KNOWLEDGE-BASED RECOMMENDER TECHNOLOGIES FOR MARKETING AND SALES

ALEXANDER FELFERNIG, ERICH TEPPAN

Business Informatics and Application Systems, University Klagenfurt, Universitaetsstrasse 65-67, Klagenfurt, A-9020, Austria
{alexander.felfernig, erich.teppan}@uni-klu.ac.at
http://www.uni − klu.ac.at

BARTOSZ GULA

Cognitive Psychology Unit, University Klagenfurt, Universitaetsstrasse 65-67, Klagenfurt, A-9020, Austria
bartosz.gula@uni-klu.ac.at

Recommender applications support decision-making processes by helping online customers to identify products more effectively. Recommendation problems have a long history as a successful application area of Artificial Intelligence (AI) and the interest in recommender applications has dramatically increased due to the demand for personalization technologies by large and successful e-Commerce environments. Knowledge-based recommender applications are especially useful for improving the accessibility of complex products such as financial services or computers. Such products demand a more profound knowledge from customers than simple products such as CDs or movies. In this paper we focus on a discussion of AI technologies needed for the development of knowledge-based recommender applications. In this context, we report experiences from commercial projects and present the results of a study which investigated key factors influencing the acceptance of knowledge-based recommender technologies by end-users.

Keywords: Knowledge-based Recommender Technologies; User Acceptance of Recommender Technologies; Deployed Applications.

1. Introduction

Recommender applications support online customers in the effective identification of products and services suiting their wishes and needs. These applications are of particular importance for increasing the accessibility of product assortments for users not having a detailed product domain knowledge. Application areas for recommender technologies range from the recommendation of financial services [10] to the personalized provision of news [2,23]. An overview of different recommender applications can be found, e.g., in [19,25]. Compact overviews of different technological approaches to the implementation of recommender applications can be found in [1,4,5,24,28]. There are three main approaches to the implementation of recommender applications. First, *collaborative filtering* [14,24,26] stores preferences of a large set of customers. Assuming that human preferences are correlated, recommendations

given to a customer are derived from preferences of customers with similar interests. If two customers have bought similar books in the past and have rated those books in a similar way, books (with a positive rating) read by only one of them are recommended to the other one. Second, *content-based filtering* [21] uses preferences of a *specific* customer to infer recommendations. In this context, products are described by keywords (categories) stored in a profile in the case that a customer buys a product. The next time, the customer interacts with the recommender application, stored preferences from previous sessions are used for offering additional products which are assigned to similar categories. Finally, *knowledge-based recommender applications* (advisors) [4,10,16,29] exploit deep knowledge about the product domain in order to determine recommendations. When selling, for example, investment portfolios, recommendations (solutions) must conform to legal regulations and suit a customer's financial restrictions as well as his/her wishes and needs. Compared to simple products such as books, movies or CDs, such products are much more in the need of information and mechanisms alleviating their accessibility for customers without detailed product domain knowledge. Primarily, knowledge-based advisors provide the formalisms needed in this context.

The remainder of this paper is organized as follows. In Section 2 we introduce the architecture and major technologies implemented in Koba4MS[a], a domain-independent environment designed for the development of knowledge-based recommender applications. In Section 3 we report experiences from successfully deployed recommender applications and discuss effects knowledge-based recommender technologies have on the behavior of customers interacting with the recommender application. Finally, in Section 4 we provide a discussion of related work.

2. Koba4MS Environment

2.1. *Architecture*

Knowledge-based advisors exploit deep product domain knowledge in order to determine solutions which suit the wishes and needs of a customer. Two basic aspects have to be considered when implementing a knowledge-based recommender application. First, the relevant product, marketing and sales knowledge has to be acquired and transformed into a formal representation, i.e., a *recommender knowledge base* [10] has to be defined. Such a knowledge base consists of a formal description of the relevant set of products, possible customer requirements and constraints defining allowed combinations of customer requirements and product properties. Second, a *recommender process* [11] has to be defined which represents personalized navigation paths through a recommender application. Both, knowledge base and recommender process design are supported on a graphical level in the Koba4MS environment. Figure 1 depicts the overall architecture of the Koba4MS environment. Recom-

[a]Koba4MS (*Knowledge-based Advisors for Marketing and Sales*, FFG-808479) is a research version of a commercially available recommender environment (see www.configworks.com).

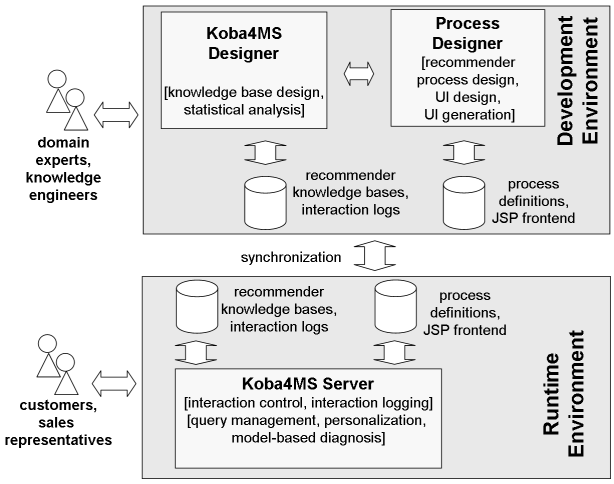


Fig. 1. Koba4MS architecture.

mender knowledge bases and process definitions are designed and maintained on a graphical level using the *development environment* (*Koba4MS Designer* and *Process Designer*) and are stored in an underlying relational database. The resulting graphical models can be automatically translated into an executable recommender application (Java Server Pages). The recommender application is made available for customers (e.g., via online-stores) and sales representatives (e.g., via intranet applications or installations on notebooks of sales representatives), where *Koba4MS Server* supports the execution of advisory sessions (*runtime environment*). Based on given user inputs, the server determines and executes a personalized dialogue flow, triggers the computation of results and determines explanations as to why a product suits the needs and wishes of a customer.

2.2. *Recommender Knowledge Base*

The first step when building a knowledge-based recommender application is the construction of a *recommender knowledge base* which consists of two sets of variables representing *customer properties* and *product properties* (V_C, V_{PROD}) and three sets of corresponding constraints which represent three different types of restrictions on the combination of customer requirements and products (C_C, C_F, C_{PROD}). A simplified example of a financial services knowledge base is depicted in Figure 2. We will now discuss the major parts of such a knowledge base in more detail.

Customer properties. Customer properties (V_C) describe possible customer requirements related to a product assortment. *Customer requirements* are instantiations of customer properties. In the financial services domain, *willingness to take risks* (wr_c[*low*, *medium*, *high*]) is an example of such a property and $wr_c = low$ is an example of a customer requirement. Further examples of customer properties

Customer Properties (V_C):
/* level of expertise */
kl_c[expert, average, beginner]
/* willingness to take risks */
wr_c[low, medium, high]
/* duration of investment */
id_c[shortterm, mediumterm, longterm]
/* advisory wanted? */
aw_c[yes, no]
/* direct product search */
ds_c[savings, bonds, stockfunds, singleshares]
/* type of low risk investment */
sl_c[savings, bonds]
/* availability of funds? */
av_c[yes,no]
/* type of high risk investment */
sh_c[stockfunds, singleshares]

Product Properties (V_{PROD}):
/* product name */
$name_p$[text]
/* expected return rate */
er_p[1..40]
/* risk rate of product */
ri_p[low, medium, high]
/* minimum investment period */
$mniv_p$ [1..14]
/* product type */
$type_p$ [savings, bonds, stockfunds, singleshares]
/* financial institute */
$inst_p$(text)

Compatibility Constraints(C_C):
CC_1: $\neg(wr_c = \text{high} \wedge id_c = \text{shortterm})$
CC_2: $\neg(kl_c = \text{beginner} \wedge wr_c = \text{high})$
CC_3: $\neg(sl_c = \text{bonds} \wedge id_c = \text{shortterm})$
... *further compatibility constraints*

Filter Constraints(C_F):
CF_1: $id_c = \text{shortterm} \Rightarrow mniv_p < 3$
CF_2: $id_c = \text{mediumterm} \Rightarrow mniv_p >= 3 \wedge mniv_p < 6$
CF_3: $id_c = \text{longterm} \Rightarrow mniv_p >= 6$
CF_4: $wr_c = \text{low} \Rightarrow ri_p = \text{low}$
CF_5: $wr_c = \text{medium} \Rightarrow ri_p = \text{medium} \vee ri_p = \text{low}$
CF_6: $wr_c = \text{high} \Rightarrow ri_p = \text{high} \vee ri_p = \text{medium} \vee ri_p = \text{low}$
CF_7: $kl_c = \text{beginner} \Rightarrow ri_p <> \text{high}$
CF_8: $sl_c = \text{savings} \Rightarrow type_p = \text{savings}$
CF_9: $sl_c = \text{bonds} \Rightarrow type_p = \text{bonds}$
CF_{10}: $ds_c = \text{savings} \Rightarrow type_p = \text{savings}$
... *further filter constraints*

Allowed instantiations of Product Properties(C_{PROD}):
/* product 1 */
CP_1: $name_p = \text{savings1} \wedge er_p = 3 \wedge ri_p = \text{low} \wedge mniv_p = 1 \wedge type_p = \text{savings} \wedge inst_p = \text{A} \vee$ ...
/* product 2 */
CP_2: $name_p = \text{bonds2} \wedge er_p = 5 \wedge ri_p = \text{medium} \wedge mniv_p = 5 \wedge type_p = \text{bonds} \wedge inst_p = \text{B} \vee$ ...
/* product 3 */
CP_3: $name_p = \text{stock3} \wedge er_p = 9 \wedge ri_p = \text{high} \wedge mniv_p = 10 \wedge type_p = \text{stockfunds} \wedge inst_p = \text{B}$
... *further product instances*

Fig. 2. Example recommender knowledge base.

are the *intended duration of investment* (id_c[*shortterm*, *mediumterm*, *longterm*]), the *knowledge level of a customer* (kl_c[*expert*, *average*, *beginner*]), or the requested *product type* (sl_c[*savings*, *bonds*]) (for low risk investments).

Product properties. Product properties (V_{PROD}) are a description of the properties of a given set of products in the form of finite domain variables. Product properties in the financial services domain are, e.g., the *minimal investment period* ($mniv_p$[1..14]), the *product type* ($type_p$[*savings*, *bonds*, *stockfunds*, *singleshares*]), or the *expected return rate* (er_p[1..40]).

Compatibility constraints. Compatibility constraints (C_C) are restricting the possible combinations of customer requirements, e.g., *if a customer has little knowledge about financial services, no high risk products should be preferred by the customer*, i.e., CC_2: $\neg(kl_c = \textit{beginner} \wedge wr_c = \textit{high})$. Confronted with such incompatible requirements, the recommender application indicates the incompatibility and requests an adaptation of the given preferences. On the one hand, incompatibility constraints can be represented on the *textual level*. On the other hand, such constraints are represented in the form of *incompatibility tables* (tables representing

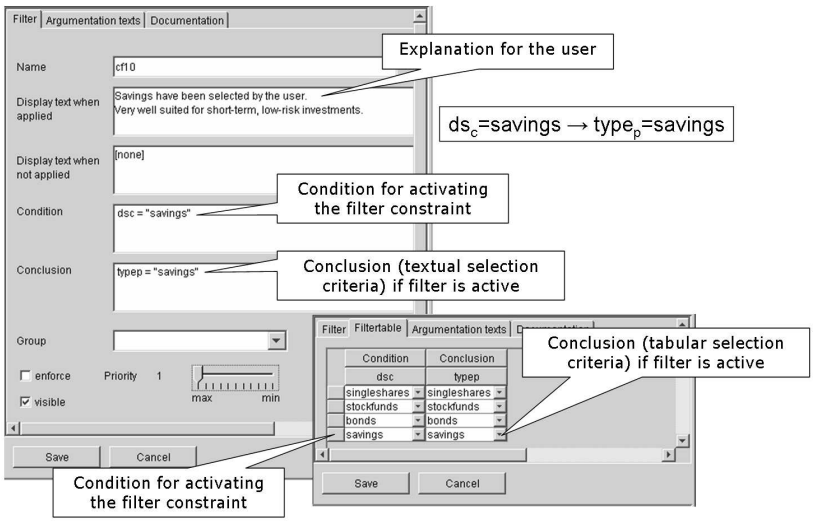


Fig. 3. Filter constraints: textual and graphical representation.

not allowed combinations of customer requirements) which is the preferred representation used by domain experts designing knowledge bases (see, e.g., [10]).

Filter constraints. Filter constraints (C_F) define the relationship between customer requirements and an available product assortment. Examples of filter constraints are: *customers without experiences in the financial services domain should not receive recommendations which include high risk products*, i.e., CF_7: $kl_c = beginner \Rightarrow ri_p <> high$ or *if the customer strongly prefers savings, the corresponding recommendation should include savings*, i.e., CF_{10}: $ds_c = savings \Rightarrow type_p = savings$. Figure 3 depicts an example of the representation of filter constraints in the Koba4MS environment (filter constraint CF_{10} of Figure 2) where CF_{10} is represented on the *textual* as well as on the *graphical level*. Using the tabular representation, an arbitrary number of condition and conclusion variables can be added.

Product instances. Allowed instantiations of product properties can be interpreted as constraints (C_{PROD}) which define restrictions on the possible instantiations of variables in V_{PROD}, e.g., the constraint CP_2: $name_p = bonds2 \wedge er_p = 5 \wedge ri_p = medium \wedge mniv_p = 5 \wedge type_p = bonds \wedge inst_p = B$ specifies a product of type *bonds* of the financial services provider B with the name *bonds2*, an expected return rate of 5%, a *medium* risk rate, and a minimum investment period of 5 *years*.

Product comparisons. Comparison rules (see Figure 4) specify which argumentations are used to explain differences between products which are part of a recommendation result, e.g., if the risk rate of the selected product (product 1) is lower than the risk rate of another product (product 2) part of the recommendation result, then the comparison component should display the explanation *the*

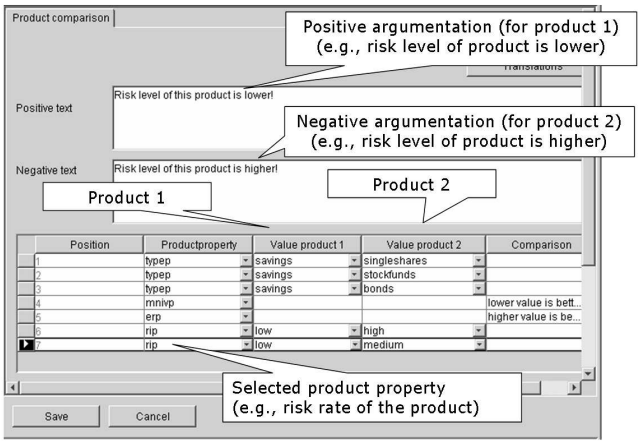


Fig. 4. Defining rules for product comparison.

risk level of this product is higher (if product 1 is selected as reference product the explanation is given for all other products with a medium risk level).

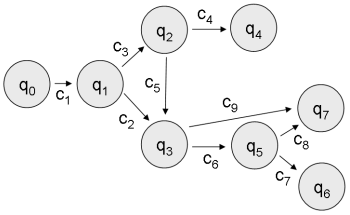
2.3. *Recommender Process Definition*

In order to be able to adapt the dialog style to a customers preferences and level of product domain knowledge, we have to provide mechanisms which allow the definition of the intended (personalized) behavior of the recommender user interface. A recommender user interface can be described by a finite state automaton, where state transitions are triggered by requirements imposed by customers. Such automata are based on the formalism of *predicate-based finite state automata* ($PFSA$) [11], where constraints specify transition conditions between different states.

Definition 1 ($PFSA$). We define a Predicate-based Finite State Automaton (recognizer) ($PFSA$) to be a 6-tuple (Q, Σ, Π, E, S, F), where

- $Q = \{q_1, q_2, ..., q_m\}$ is a finite set of states, where $var(q_i) = \{x_i\}$ is a finite domain variable assigned to q_i, $prec(q_i) = \{\phi_1, \phi_2, ..., \phi_n\}$ is the set of preconditions of q_i ($\phi_\alpha = \{c_{\alpha 1}, c_{\alpha 2}, ..., c_{\alpha o}\} \subseteq \Pi$), $postc(q_i) = \{\psi_1, \psi_2, ..., \psi_p\}$ is the set of postconditions of q_i ($\psi_\beta = \{c_{\beta 1}, c_{\beta 2}, ..., c_{\beta q}\} \subseteq \Pi$), and $dom(x_i) = \{x_i{=}d_{i1}, x_i{=}d_{i2}, ..., x_i{=}d_{ik}\}$ denotes the set of possible assignments of x_i, i.e. the domain of x_i.
- $\Sigma = \{x_i = d_{ij} \mid x_i \in var(q_i), (x_i = d_{ij}) \in dom(x_i)\}$ is a finite set of variable assignments (input symbols), the input alphabet.
- $\Pi = \{c_1, c_2, ..., c_r\}$ is a condition set restricting the set of accepted words.
- E is a finite set of transitions $\subseteq Q \times \Pi \times Q$.
- $S \subseteq Q$ is a set of start states.
- $F \subseteq Q$ is a set of final states. □

Figure 5 contains a $PFSA$ definition for our example financial services recommender knowledge base depicted in Figure 2. Following this definition, customers can specify requirements (input values) for the defined set of customer properties. Depending on the input of the customer, the automaton changes its state, e.g., an *expert* (c_3) *who isn't interested in financial advisory* (c_4) is forwarded to the state q_4 by the transitions (q_0, c_1, q_1), (q_1, c_3, q_2), (q_2, c_4, q_4). The recommender process definition of Figure 5 can be automatically translated into a corresponding recommender application. This approach allows rapid prototyping development processes for knowledge-based advisors [11]. Note that for reasons of effective knowledge acquisition support recommender process definitions are represented on a graphical level within the Koba4MS development environment (see, e.g., [11]).



$Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$.
$var(q_0) = wr_c$. $var(q_1) = kl_c$. $var(q_2) = aw_c$.
$var(q_3) = id_c$. $var(q_4) = ds_c$. $var(q_5) = av_c$.
$var(q_6) = sh_c$. $var(q_7) = sl_c$.
$dom(wr_c) = \{wr_c$=*low*, wr_c=*medium*, wr_c=high$\}$.
$dom(kl_c) = \{kl_c$=*beginner*, kl_c=*average*, kl_c=*expert*$\}$.
$dom(aw_c) = \{aw_c$=*yes*, aw_c=*no*$\}$.
$dom(id_c) = \{id_c$=*shortterm*, id_c=*mediumterm*, id_c=*longterm*$\}$.
$dom(ds_c) = \{ds_c$=*savings*, ds_c=*bonds*, ds_c=*stockfunds*, ds_c=*singleshares*$\}$.
$dom(av_c) = \{av_c$=*yes*, av_c=*no*$\}$.
$dom(sh_c) = \{sh_c$=*stockfunds*, sh_c=*singleshares*$\}$.
$dom(sl_c) = \{sl_c$=*savings*, sl_c=*bonds*$\}$.

$prec(q_0) = \{\{true\}\}$. $prec(q_1) = \{\{c_1\}\}$.
$prec(q_2) = \{\{c_1, c_3\}\}$.
$prec(q_3) = \{\{c_1, c_2\}, \{c_1, c_3, c_5\}\}$.
... *further preconditions*

$postc(q_7) = \{\{true\}\}$.
$postc(q_6) = \{\{true\}\}$.
$postc(q_5) = \{\{c_7\}, \{c_8\}\}$.
$postc(q_4) = \{\{true\}\}$.
$postc(q_3) = \{\{c_6, c_7\}, \{c_6, c_8\}, \{c_9\}\}$.
... *further postconditions*

$\Sigma = \{kl_c$=*beginner*, kl_c=*average*, kl_c=*expert*, aw_c=*yes*, aw_c=*no*, ..., sl_c=*savings*, sl_c=*bonds*$\}$.
$\Pi = \{$
c_1: true, c_2: kl_c = beginner,
c_3: kl_c <> beginner, c_4: aw_c = no,
c_5: aw_c = yes,
c_6: id_c <> shortterm $\wedge$ kl_c <> beginner,
c_7: av_c = yes, c_8: av_c = no,
c_9: id_c = shortterm$\}$.
$E = \{(q_0, \{c_1\}, q_1), (q_1, \{c_3\}, q_2),$
$(q_1, \{c_2\}, q_3), (q_2, \{c_4\}, q_4),$
$(q_2, \{c_5\}, q_3), (q_3, \{c_6\}, q_5),$
$(q_3, \{c_9\}, q_7), (q_5, \{c_7\}, q_6),$
$(q_5, \{c_8\}, q_7)\}$.
$S = \{q_0\}$. $F = \{q_4, q_6, q_7\}$.

Fig. 5. Example recommender process definition ($PFSA$).

2.4. *Calculating Recommendations*

We denote the task of identifying products for a customer as *recommendation task*.

Definition 2 (Recommendation Task). A recommendation task is defined by (V_C, V_{PROD}, $C_F \cup C_C \cup C_{PROD} \cup C_R$), where V_C is a set of variables representing possible customer requirements and V_{PROD} is a set of variables describing product properties. C_{PROD} is a set of constraints describing available product in-

stances, C_C is a set of constraints describing possible combinations of customer requirements (compatibility constraints) and C_F is a set of constraints describing the relationship between customer requirements and available products (also called filter constraints). Finally, C_R is a set of concrete customer requirements (represented as unary constraints). □

Example 1 (Recommendation Task). In addition to the recommender knowledge base (V_C, V_{PROD}, $C_F \cup C_C \cup C_{PROD}$) of Figure 2, C_R=$\{wr_c = low$, $kl_c = beginner$, $id_c = shortterm$, $sl_c = savings\}$ is a set of requirements. □

Based on the given definition of a recommendation task, we can introduce the notion of a solution (*consistent recommendation*) for a recommendation task.

Definition 3 (Consistent Recommendation). An assignment of the variables in V_{PROD} is denoted as consistent recommendation for a recommendation task (V_C, V_{PROD}, $C_F \cup C_C \cup C_{PROD} \cup C_R$) iff each variable in $V_C \cup V_{PROD}$ has an assignment which is consistent with $C_F \cup C_C \cup C_{PROD} \cup C_R$. □

Example 2 (Consistent Recommendation). A consistent recommendation (result) for the recommendation task defined in Example 1 is, e.g., $name_p = savings1$, $er_p = 3$, $ri_p = low$, $mniv_p = 1$, $type_p = savings$, $inst_p = A\}$. □

For the calculation of solutions we have developed a *relational query-based approach*, in which a given set of customer requirements makes a conjunctive query. Such a query is composed from the consequent part of those filter constraints whose condition is consistent with the given set of customer requirements (*active* filter constraints), e.g., the consequent part of CF_7: $kl_c = beginner \Rightarrow ri_p <> high$ is translated into the expression $ri_p <> high$ as part of the corresponding conjunctive query. Accordingly, V_{PROD} is represented by a set of table attribute definitions (the product table) and C_{PROD} is represented by tuples whose values represent instantiations of the attributes defined in V_{PROD}. Furthermore, customer properties (V_C) are represented as input variables where the compatibility (C_C) of the corresponding instantiations is ensured by a consistency checker. The execution of the conjunctive query on a product table results in a set of recommendations which are presented to the customer. For the given customer requirements (C_R) of Example 1, the set $\{CF_1, CF_4, CF_7, CF_8\}$ represents active filter constraints. The consequent parts of those constraints make a conjunctive query of the form $\{mniv_p < 3 \wedge ri_p = \text{low} \wedge ri_p <> \text{high} \wedge type_p = \text{savings}\}$. For our example knowledge base of Figure 2, this query results in the single recommendation of Example 2 $\{name_p = savings1, er_p = 3, ri_p = low, mniv_p = 1, type_p = savings, inst_p = A\}$.

Repair of Customer Requirements. If the result set of a query is empty (no solution could be found), conventional recommender applications tell the user (customer) that no solution was found, i.e., no clear explanation for the reasons for such a situation is given. Simply reporting retrieval failures (no product fulfils all requirements) without making further suggestions how to recover from such a situation is not acceptable [13,18]. Therefore, our goal is to find a set of possible compromises that are presented to the customer who can choose the most acceptable alternative. Koba4MS supports the calculation of repair actions for customer

requirements (a minimal set of changes allowing the calculation of a solution). If $C_R = \{x1_c = a_1, x2_c = a_2, ..., xn_c = a_n\}$ is a set of customer requirements and the recommendation task (V_C, V_{PROD}, $C_F \cup C_C \cup C_{PROD} \cup C_R$) doesn't have a solution, a repair is a minimal set of changes to C_R (resulting in C'_R) s.t. (V_C, V_{PROD}, $C_F \cup C_C \cup C_{PROD} \cup C'_R$) has a solution. The computation of repair actions is based on the Hitting Set algorithm [8,22] which exploits minimal conflict sets [17] in order to determine minimal diagnoses and corresponding repair actions.

A simple example of the calculation of repair actions is depicted in Figure 6. In this example, $C_R \cup C_C$ has no solution since $\{CR_1, CR_2\} \cup C_C$ and $\{CR_1, CR_3\} \cup C_C$ are inconsistent and therefore both $\{CR_1, CR_2\}$ and $\{CR_1, CR_3\}$ induce a conflict [17] with the given compatibility constraints. Conforming with the hitting set algorithm [22], we have to resolve each of the given conflicts. A minimal repair for C_R (resulting in C'_R) is to change the requirement related to the willingness to take risks, i.e., wr_c = *medium* which makes $C'_R \cup C_C$ consistent ($C'_R = \{wr_c =$ *medium*, $id_c =$ *shortterm*, $kl_c =$ *beginner*$\}$).

Recommender Knowledge Base (relevant parts) ***Compatibility Constraints***(C_C): CC_1: $\neg(wr_c = \text{high} \wedge id_c = \text{shortterm})$ CC_2: $\neg(kl_c = \text{beginner} \wedge wr_c = \text{high})$ CC_3: $\neg(sl_c = \text{bonds} \wedge id_c = \text{shortterm})$		
⇳ inconsistent ($\{CR_1, CR_2\}$, $\{CR_1, CR_3\}$)		⇳ consistent
Customer Requirements (C_R) CR_1: wr_c = high CR_2: id_c = shortterm CR_3: kl_c = beginner	⟹ ***repair action*** $\{CR'_1\}$	***Customer Requirements*** (C'_R) CR'_1: wr_c = medium CR_2: id_c = shortterm CR_3: kl_c = beginner

Fig. 6. Example: repair of customer requirements.

Explanation of Solutions. For each product part of a solution calculated by a recommender application, a set of immediate explanations [12] is calculated, i.e., a set of explanations which are derived from those filter constraints which are responsible for the selection of a product (see, e.g., the filter constraint of Figure 3). An explanation related our example filter constraint CF_7: $kl_c =$ *beginner* $\Rightarrow ri_p <>$ *high* could be *this product assures adequate return rates with a lower level of related risks*. Note that explanations are directly assigned to filter constraints.

User Modeling. Due to the heterogeneity of users, the Koba4MS environment includes mechanisms allowing the adaptation of the dialog style to the users skills and needs [3]. The user interface relies on the management of a user model that describes capabilities and preferences of individual customers. Some of these properties are directly provided by the user (e.g. age, nationality, personal goals, or self-estimates such as knowledge about financial services), other properties are derived using personalization rules and scoring mechanisms [3,10] which relate user

answers to abstract dimensions such as *preparedness to take risks* or *interest in high profits* (dimensions describing the users interests).

3. Evaluation

3.1. *Example Application*

Koba4MS technologies have been applied in a number of commercial projects (see, e.g., [9,10]). Figure 7 depicts example screenshots of an investment advisor implemented for the Hypo Alpe-Adria-Bank in Austria (www.hypo-alpe-adria.at). First, a set of questions is posed to a customer, i.e., preferences are elicited (a). The corresponding answers provided by the customer (customer requirements) serve as input for the calculation of a solution. In the case that no solution can be found by the recommender application, the calculation of repair alternatives is activated (b). After having selected one of the alternatives, the customer can continue the recommendation session. Finally, a set of alternative investment proposals is determined and presented to the customer (in our case, two portfolios have been identified) (c). For each alternative, a corresponding set of explanations is calculated, as to why a certain product suits the wishes and needs of a customer (d). Finally, product comparisons provide basic mechanisms to compare different products which are part of a recommendation result (e) where differences between the selected (reference) product an other products are clearly indicated (the definition of comparison rules is shown in the simple example of Figure 4).

A number of additional applications have been implemented on the basis of Koba4MS, e.g., financial service recommenders for the Wuestenrot and the Fundamenta building and loan association (www.wuestenrot.at, www.fundamenta.hu), recommenders for www.quelle.at, one of the leading online selling environments in Austria, the digital camera advisor for www.geizhals.at, the largest price comparison platform in Austria, and the recommender application which supports students at the Klagenfurt University (www.uni-klu.ac.at) in the identification of additional financial support opportunities (e.g., grants). Experiences from two selected projects will be discussed in the following subsection.

3.2. *Experiences from Projects*

Financial services advisor. In the case of the *Wuestenrot building and loan association*, financial service advisors have been developed with the goal to support sales representatives in the dialog with the customer. The recommenders have been integrated with an existing Customer Relationship Management (CRM) environment and are available for 1.400 sales representatives [10]. The motivation for the development of financial service recommender applications was twofold. First, *time efforts* related to the preparation, conduction and completion of sales dialogues should be reduced. Second, an *automated documentation of advisory sessions* should be supported in order to take into account regulations of the European Union [30] related

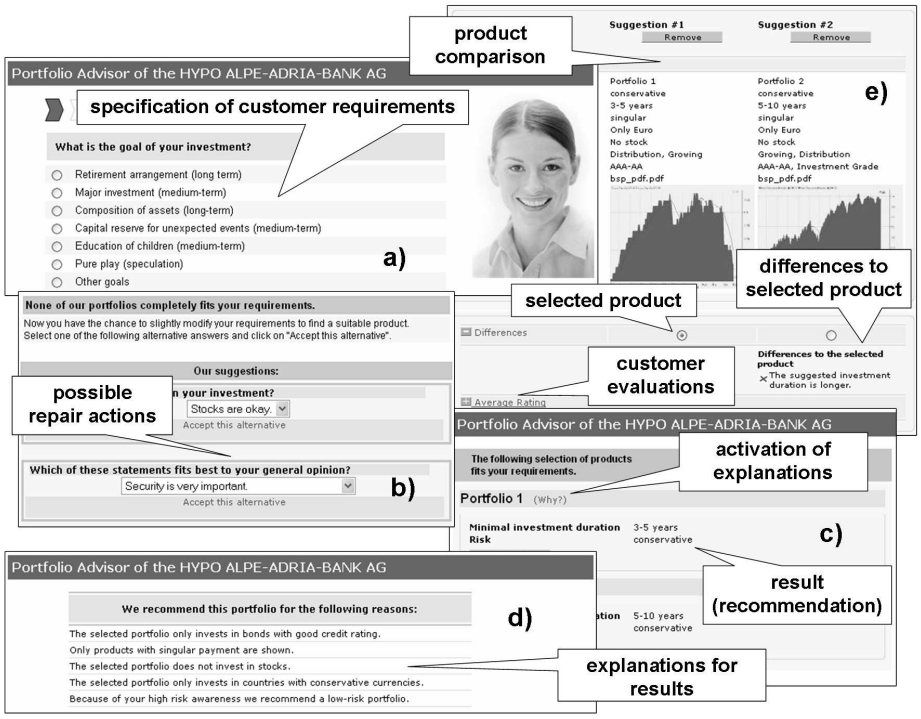


Fig. 7. Example financial services recommender application (investment advisor).

to the documentation of financial service advisory dialogs. With the goal to get a picture of how users apply the new technology and which impacts this technology has on sales processes, we have interviewed sales representatives of the Wuestenrot building and loan association (n=52). Summarizing, the major results of the evaluation of the questionnaire were the following [9,10]:

- *Time savings*: On an average, interviewees specified the reduction of time efforts related to financial services advisory with 11.73 minutes per advisory session (SD (standard deviation) = 7.47), where the reductions are explained by the automated generation of advisory protocols and available summaries of previous dialogues at the beginning of a new advisory session. This corresponds to about 30.0% reduction of time efforts in the start phase and the final phase of an advisory session. Assuming that an average sales representative conducts about 170 advisory sessions per year, this results in time savings of about 33 hours per year.
- *Usefulness of recommender functionalities*: in the case of Wuestenrot the majority of interviewees were judging the provided recommendation functionalities as very useful. Most of the sales representatives reported to use

Koba4MS functionalities throughout the sales dialogue or for the generation of documentations for completed advisory dialogues. Each such documentation includes a summary of customer preferences, a listing of recommended products, and an argumentation as to why the recommended products fit to the wishes and needs of the customer.

- *Importance for new representatives*: Most of the sales representatives definitely agreed on the potential of knowledge-based recommender technologies to provide e-learning support. Due to this feedback, a corresponding project has already been initiated which exploits recommender technologies in order to support learning processes for new sales representatives. The software will be applied in the context of sales courses.

Financial support advisor. Apart from user studies in the financial services domain [9,10] we have evaluated the impacts of a financial support recommender application developed for students at the Klagenfurt University in Austria. On an average, about 150 students per month use the services of the financial support advisor for the identification of additional financial support opportunities. Our evaluation of the advisor consisted of n=1.271 online users of www.uni-klu.ac.at, the homepage of the Klagenfurt University. An announced lottery ensured that participants identified themselves with their genuine names and addresses and no duplicate questionnaires were counted. The sample consisted of arbitrary online users of www.uni-klu.ac.at, who did not necessarily know the recommender application (36% if the participants already knew and 12% actively applied the advisor). The major results of this study were the following:

- *Increase of domain knowledge*: of those interviewees who actively applied the recommender application, 69.8% reported a significant increase of domain knowledge in the area of financial support for students as a direct consequence of having interacted with the advisor.
- *Additional applications*: of those interviewees who actively applied the recommender application, 19.4% applied for additional financial support as a direct consequence of having interacted with the advisor.
- *Time savings*: on an average, interviewees specified the overall time savings caused by the application of the advisor with 61.93 minutes per advisory session ($SD = 27.8$). Consequently, students invest less time to get informed about additional financial support opportunities. Furthermore, members of the students council invest less time in routine advisory tasks.

3.3. *Empirical Findings regarding User Acceptance*

In this section we focus on the presentation of the results of a user study (n=116) which investigated explicit and implicit feedback of online users to various interaction mechanisms supported by knowledge-based recommender applications. The findings of the study show interesting patterns of consumer buying behaviour when

interacting with knowledge-based recommender applications. In particular, there exist specific relationships between the type of supported interaction mechanisms and the attitude of the user w.r.t. the recommender application. In the study we analyzed the degree to which concepts such as explanations, repair actions, and product comparisons influence the attitudes of online users towards knowledge-based recommender technologies. In the scenario of the study the participants had to decide which online provider they would select for their home internet connection. To promote this decision, 8 different versions of an *Internet Provider* recommender application have been implemented. The participants of the study had to use such a recommender application to identify the provider which best suits their needs and to place a fictitious order. Each participant was randomly assigned to one version of the implemented recommender applications (an overview of the provided versions of recommender applications is given in Table 1). Before and after interacting with the advisor, participants had to fill out an online questionnaire (see Table 2a, 2b). Participation was voluntary and a small remuneration was offered. We were interested in the frequency, participants used a recommender application to order products or as an additional information source (Table 2a-1). Self-rated knowledge and interest in the domain of internet connection providers (Table 2a-4,5) was assessed on a 10-point scale before interacting with the recommender application. After solving the task of virtually buying a connection from an Internet Provider, the participants had to answer follow-up questions as well assessed on a 10-point scale (Table 2b) except 2b-10 where a probability estimate had to be provided. Additional variables have been extracted from interaction logs (Table 2c). The inclusion of the variables depicted in Table 2 is based on a set of hypotheses which are outlined in the following together with the corresponding exploratory results. The participants of the user study were randomly assigned to one of the Internet Provider advisors shown in Table 1. If a participant was confronted with the advisor version (a) or (b) and answered the question related to his/her expertise with *expert* than he/she was forwarded to a path in the recommender process which was designed for the advisory of beginners (and vice-versa) - we denote this as *switched expertise*. This manipulation was used to test the hypothesis that a dialog design fitting to the knowledge level of the participants leads to a higher satisfaction with the recommender application. Note that *positive explanations* provide a justification as to why a product fits to a certain customer, whereas *negative explanations* provide a justification for the relaxation of certain filter constraints. *Product comparisons* where supported in two different ways: first, comparisons had to be explicitly activated by participants, second, the result page was automatically substituted by the product comparison page. Finally, a *pure product list*, i.e., product selection without any advisory support, was implemented by automatically navigating to the result page and displaying all available products.

We have tested 116 participants with a mean age of $\bar{x}$= 28.7 SD (standard deviation) = 9.78 (33,6% female). 42.2% were recruited from the Klagenfurt University and 57.8% were non-students. Explanations were used by 29.2% of the participants,

Advisor versions
(a) switched expertise, positively formulated explanations, with product comparisons.
(b) switched expertise, without explanations, without product comparisons.
(c) positively formulated explanations, without product comparisons.
(d) negatively formulated explanations, without product comparisons.
(e) positively formulated explanations, with product comparisons.
(f) without explanations, with product comparisons.
(g) pure list of products (without any recommendation functionalities).
(h) positively formulated explanations, with product comparisons (automatically activated).

Table 1. Different versions of *Internet Provider* advisors.

(a) Questions posed before advisor has been started
1. previous usage (for buying purposes, as an information source)
2. satisfaction with recommendation processes (advisory support) up to now
3. trust in recommended products up to now (products suit personal needs)
4. knowledge in the Internet Provider domain
5. interest in the domain of Internet Providers
(b) Questions posed after completion of advisory session
1. knowledge in the Internet Provider domain
2. interest in the domain of Internet Providers
3. satisfaction with the recommendation process (advisory support)
4. satisfaction with the recommended products
5. trust in the recommended products (products suit personal needs)
6. correspondence between recommendations and expectations
7. importance of explanations
8. competence of recommender application
9. helpfulness of repair actions
10. willingness to buy a product
(c) Data derived from interaction log
1. session duration
2. number of visited web pages
3. number of inspected explanations
4. number of activated product comparisons
5. number of clicks on product details
6. number of activations of repair actions

Table 2. Variables assessed in the study.

repair actions have been triggered in 6.9% of the cases. Finally, a product comparison was used by 32.8% of the participants.[b] To assess the significance of correlations and differences, non-parametric tests were used [15]. Because the assessed variables were either ordinal-scaled or violated the assumptions of normal distribution or homogeneity of variance (visited pages, session duration), the Mann-Whitney U-Test was used to compare two groups and the Kruskal-Wallis-H Test to assess differences between more than two groups. In the following, only significant results are reported, with α set to 0.05 for all subsequent tests. The corresponding z-values are provided to show the size of the effects.

There were clear differences between the eight versions of recommender appli-

[b]Note that the relative frequencies refer to participants who had the possibility to use the corresponding feature (explanations, repairs, product comparisons).

cations. The most positive ratings related to trust in the recommended products (Table 2b-5) and satisfaction with the recommendation process (Table 2b-3) were provided by participants interacting with the versions (e) and (h), i.e., advisor versions with positively formulated explanations and a product comparison functionality. Let us now consider the relationship between the features in the different advisor versions and the participants´ impressions in more detail.

Recommender application vs. pure product list. We have found recommender applications to be more advantageous with respect to most of the assessed variables (see Table 2b). Participants using a recommender application were significantly more satisfied with the recommendation process ($z = -3.872$; $p < 0.001$) (Table 2b-3) and had a significant increase in satisfaction due to the interaction with the Internet Provider advisor ($z = -2.938$; $p < 0.01$) (Table 2a-2, 2b-3). Participants' trust in that the application recommended the optimal solution was higher for those interacting with the recommender application compared to those confronted with a pure product list ($z = -3.325$; $p = 0.001$) (Table 2b-5). Furthermore, participants stated that the final recommendation better fitted to their expectations than when they were confronted with a simple product list ($z = -3.872$; $p = 0.001$) (Table 2b-6). Most interestingly, the increase of subjective product domain knowledge due to the interaction was higher when participants interacted with a recommender application ($z = -2.069$; $p = 0.04$) (Table 2a-4, 2b-1). The estimated (subjective) probability to buy a product in a purchase situation was higher for those interacting with a recommender application than for those interacting with a pure product list ($z = -2.1$; $p < 0.01$). Actually, this mean probability was only $p = 0.19$ for participants confronted with a product list, suggesting that these participants estimated a real purchase of the selected product as rather unlikely.

Effects of providing explanations. The perceived correspondence between recommended products and expectations (Table 2b-6) as well as the perceived competence of the recommender application (Table 2b-8) were rated higher by participants provided with the *possibility to use* explanations ($z = -3.228$; $p < 0.01$ and $z = -1.966$; $p < 0.05$). Most importantly, these participants´ trust in recommended products clearly increased due to the interaction process ($z = -2{,}816$; $p < 0.01$) (comparing pre- to post-test, Table 2a-3, 2b-5). There is a tendency that providing explanations leads to more satisfaction with the recommendation process ($z = -1.544$; $p = 0.06$) (Table 2b-3). However, as hypothesized before the study, the increase in the rated knowledge from pre- to post-test did not differ significantly between both groups (Table 2a-4, 2b-1). Participants who have *actively* (!) inspected explanations express a higher correspondence between expected and recommended products ($z = -2.176$; $p = 0.01$) (Table 2b-6) and an increased interest in the product domain when comparing pre- to post-test ($z = -1.769$; $p < 0.05$) (Table 2a-5, 2b-2). Participants who inspected explanations and had experience with applying recommender applications, showed a tendency to rate the importance of explanations higher (Table 2b-7). They showed more trust in the recommended products (Table 2b-5) and stated a higher interest in the product domain (Table 2b-2). This

suggests that a certain degree of familiarity with recommender applications is necessary in order to optimally exploit explanations.

Exploring variables that may potentially influence the actual use of explanations, it was found that experience correlated with the degree of explanation use. Participants already having experience with recommender applications were more likely to use an explanation ($r = 0.23$; $p < 0.05$) (Table 2c-3). Interpreting interaction processes with advisors as processes of preference construction, as described by [20], we assume that explanations influence preferences by adjusting the expectations of customers. This influence may be simply due to the fact that an explanation contains product features to which customers are primed. As argued in [20], priming of features causes customers to focus attention to those features and thus possibly to compare the recommended products with their expectations mainly along the primed features. This provides an explanation as to why the perceived correspondence between recommended and expected products and trust is higher when providing explanations.

Effects of product comparisons. Participants using recommender applications supporting product comparisons were more satisfied with the recommendation process ($z = -2.186$; $p = 0.03$) (Table 2b-3) and the recommended products ($z = -1.991$; $p < 0.05$) (Table 2b-4) than participants using advisors without product comparison support. Furthermore, participants using advisors with product comparisons showed a significant higher trust in the recommended products ($z = -2.308$; $p = 0.02$) (Table 2b-5). Product comparison functionality leads to a higher perceived competence of the recommender application ($z = -1.954$; $p < 0.05$) (Table 2b-8). Interacting with advisors supporting product comparisons leads to a clear increase in trust ($z = 3.016$; $p < 0.01$) (Table 2a-3, 2b-5) and interest in the product domain (Internet Providers) ($z = 1.885$; $p < 0.05$) (Table 2a-5, 2b-2). Interestingly, these positive effects seem to be due to the offer of comparisons and not to their usage since only 32,8% of the participants actually used them.

Those participants who actually used product comparisons, were more satisfied with the recommendation process ($z = 2.175$; $p = 0.03$) (Table 2b-3). Positive effects due to the possibility of using a product comparison were even accentuated for those participants who already had experiences with recommender applications. They were more satisfied with the suggested products ($z = 2.233$; $p = 0.03$) (Table 2b-4) and established more trust ($z = -3.658$; $p < 0.001$) (Table 2b-5). Furthermore, product comparisons combined with existing experiences leads to a higher perceived competence of the advisor ($z = 1.940$; $p < 0.05$) (Table 2b-8).

The multitude of positive influences that product comparisons offer (especially the increase in satisfaction) can be explained by the lower mental workload when products and product features are visually clearly presented to enable an evaluation of the recommended product set. Interestingly, taken together with the results on the explanation feature, some suggestions for the optimal design of product comparisons can be made. First, as already suggested by [7] it is useful for customers to visually highlight feature (settings) in the result that vary between the products

(e.g., different color or font size). Also, assuming that a customers product evaluation will be rather based on features that she/he was primed to in the course of the interaction process through questions or an explanation feature, it should aid her/his purchase decision when primed features are highlighted as well. These implications will be tested in a follow-up study.

Effects of repair actions.[c] If we compare the participants who triggered repair actions (due to their inconsistent specifications) to those who did not trigger repair actions, we find that the first group stated to have less knowledge in the product domain ($z = -1.801$; $p < 0.05$) (Table 2a-4) and that they rarely used recommender applications before ($z = -1.645$; $p < 0.05$) (Table 2a-1). This is plausible since participants with higher product domain knowledge and more experience with recommender applications will have more realistic expectations regarding product features and costs and they will provide information to an advisor that will most likely generate a set of recommended products, which makes a repair action dispensable. Thus, participants who used repair actions indicated an increase in product domain knowledge ($z = -1.730$; $p < 0.05$) (Table 2a-4, 2b-1) and rated repair actions as more useful ($z = -2.978$; $p < 0.01$) (Table 2b-9).

Effects of *switched expertise*. Participants who received switched versions showed less satisfaction with the recommendation processes ($z = -1{,}790$; $p < 0{,}05$) (Table 2b-3) and provided a lower rating for the competence of the advisor ($z = -2{,}997$; $p < 0{,}01$) (Table 2b-8). They regarded the helpfulness of repair actions as lower ($z = -2{,}379$; $p < 0{,}01$) (Table 2b-9) compared to participants not confronted with the switched expertise scenario. This may be interpreted as an indicator of lower interest in recommender applications that fail to put questions that appropriately incorporate the expertise or knowledge level of the customer.

Willingness to buy a product. We examined which of the assessed variables show a significant correlation with the willingness to buy a product. The highest correlation has been detected between the willingness to buy (Table 2b-10) and trust in the recommended products ($r = 0.60$; $p < 0.01$) (Table 2b-5).[d] Furthermore, the higher the fit between the suggested products and the expectations of the participants (Table 2b-6), the higher was the willingness to buy the recommended product ($r=0.54$, $p < 0.01$). Another interesting relationship exists between the perceived competence of the recommender application (Table 2b-8) and the willingness to buy ($r = 0.49$, $p < 0.01$) (Table 2b-10).

4. Related Work

Recommender Technologies. In contrast to collaborative filtering [14,24,26] and content-based filtering [21] approaches, knowledge-based recommendation [4,10,16,29]

[c]In the present study only 6.9% of the participants triggered repair actions. For this reason we combined the data with a sample from a pilot study.

[d]For the computation of correlation measures, the Spearman correlation r for ordinal scale variables was used.

exploits deep knowledge about the product domain in order to determine solutions suiting the customers wishes and needs. Using such an approach, the relationship between customer requirements and products is explicitly modelled in an underlying knowledge base. Thus ramp-up problems [4] are avoided since recommendations are directly derived from user preferences identified within the scope of the requirements elicitation phase. The main reason for the choice of a knowledge-based recommendation approach stems from the requirements of domains such as financial services where deep product knowledge is needed in order to retrieve and explain solutions. [16] embed product information and explanations into multimedia-enhanced product demonstrations where recommendation technologies are used to increase the accessibility of the provided product descriptions. Using such representations, basic recommendation technologies are additionally equipped with a level supporting the visualization of calculated results. [29] focus on the integration of conversational natural language interfaces with the goal of reducing system-user interactions. A study in the restaurant domain [29] clearly indicates significant reductions in efforts related to the identification of products (in terms of a reduced number of interactions as well as reduced interaction times). Natural language interaction as well as visualization of results are currently not integrated in the Koba4MS environment but are within the scope of future work. Compared to other existing knowledge-based recommender approaches [4,16,29], Koba4MS includes model-based diagnosis [8,22] concepts allowing the calculation of repair actions in the case that no solution can be found and provides a graphical development environment which makes the development of recommender applications feasible for domain experts [10].

User Acceptance of Recommender Technologies. [27] evaluates navigational needs of users when interacting with recommender applications. A study is presented which reports results from an experiment where participants had to interact with recommender applications providing two different types of products (digital cameras and jackets offered in a digital store). It has been shown that different types of products trigger different navigational needs. The major factors influencing the navigational behaviour is the product type, e.g., compared to digital camera shoppers, jacket shoppers spent significant less time investigating individual products. The study of [27] focused on the analysis of different navigational patterns depending on the underlying product assortment. The results presented in this paper report experiences related to the application of basic recommender technologies in online buying situations. The investigation of differences related to different product domains is within the scope of future work. [20] analysed the impact of personalized decision guides to different aspects of online buying situations. An interesting result of the study was that consumers choices are mostly driven by primary attributes that had been included in the recommendation process which clearly indicated the influence of personalized decision guides on consumer preferences. Compared to the work presented in this paper, [20] did not investigate effects related to the application of knowledge-based recommender technologies such as explanations of calculated results or repair actions. Furthermore, no detailed anal-

ysis has been done on psychological aspects of online buying situations such as trust, subjective perceived increase of domain knowledge, or the probability to buy a product. [6] analyse different dimensions of the users perception of a recommender agents trustworthiness. The major dimensions of trust which are discussed in [6] are systems features such as explanation of recommendation results, trustworthiness of the agent in terms of, e.g., competence and finally trusting intentions such as intention to buy or intention to return to the recommender agent. Where the results are comparable, the study presented in [6] confirms the results of our study (explanations are positively correlated with a user's trust and well-organized recommendations are more effective than a simple list of suggestions).

5. Summary and Future Work

We have presented the Koba4MS environment for the development of knowledge-based recommender applications. Koba4MS is based on innovative AI technologies which provide an intuitive access to complex products for customers as well as for sales representatives. Innovative technologies are crucial for successfully deploying recommender applications in commercial environments. However, a deeper understanding of the effects of these technologies can make recommender applications even more successful. A step towards this direction has been shown in this paper by analyzing the effects of mechanisms such as explanations, repair actions or product comparisons on a user's overall acceptance of the recommender application. The major direction of future work is the *integration of psychological theories from the area of consumer buying behavior* into design processes of knowledge-based recommender applications. A corresponding project has already been started.

References

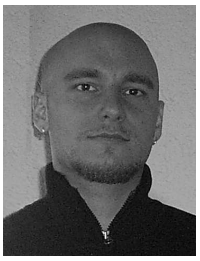
1. G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.
2. L. Ardissono, L. Console, and I. Torre. An adaptive system for the personalized access to news. *AI Communications*, 14(3):129–147, 2001.
3. L. Ardissono, A. Felfernig, G. Friedrich, D. Jannach, G. Petrone, R. Schäfer, and M. Zanker. A Framework for the development of personalized, distributed web-based configuration systems. *AI Magazine*, 24(3):93–108, 2003.
4. R. Burke. Knowledge-based Recommender Systems. *Encyclopedia of Library and Information Systems*, 69(32), 2000.
5. R. Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
6. L. Chen and P. Pu. Trust Building in Recommender Agents. In *International Workshop on Web Personalization, Recommender Systems and Intelligent User Interfaces (WPRSIUI05)*, pages 135–145, Reading, UK, 2005.
7. W. Edwards and B. Fasolo. Decision technology. *Annual Review of Psychology*, (52):581–606, 2001.
8. A. Felfernig, G. Friedrich, D. Jannach, and M. Stumptner. Consistency-based Diagnosis of Configuration Knowledge Bases. *AI Journal*, 2(152):213–234, 2004.

9. A. Felfernig, K. Isak, and C. Russ. Knowledge-based Recommendation: Technologies and Experiences from Projects. In *17th European Conference on Artificial Intelligence (ECAI06)*, page to appear, Riva del Garda, Italy, 2006.
10. A. Felfernig and A. Kiener. Knowledge-based Interactive Selling of Financial Services using FSAdvisor. In *17th Innovative Applications of Artificial Intelligence Conference (IAAI'05)*, pages 1475–1482, Pittsburgh, Pennsylvania, 2005.
11. A. Felfernig and K. Shchekotykhin. Debugging User Interface Descriptions of Knowledge-based Recommender Applications. In C. Paris and C. Sidner, editors, *ACM International Conference on Intelligent User Interfaces (IUI'06)*, pages 234–241, Sydney, Australia, 2006.
12. G. Friedrich. Elimination of Spurious Explanations. In G. Müller and K. Lin, editors, *16th European Conference on AI (ECAI 2004)*, pages 813–817, Valencia, Spain, 2004.
13. P. Godfrey. Minimization in Cooperative Response to Failing Database Queries. *International Journal of Cooperative Information Systems*, 6(2):95–149, 1997.
14. J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. on Inf. Systems*, 22(1):5–53, 2004.
15. M. Hollander and D.A. Wolfe. *Nonparametric Statistical Methods*. Wiley, 1999.
16. B. Jiang, W. Wang, and I. Benbasat. Multimedia-Based Interactive Advising Technology for Online Consumer Decision Support. *Comm. of the ACM*, 48(9):93–98, 2005.
17. U. Junker. QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. *19th National Conf. on AI (AAAI04)*, pages 167–172, 2004.
18. D. McSherry. Maximally Successful Relaxations of Unsuccessful Queries. In *15th Conference on Artificial Intelligence and Cognitive Science*, pages 127–136, 2004.
19. M. Montaner, B. Lopez, and J. De la Rose. A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review*, 19:285–330, 2003.
20. K.B. Murray and G. Häubl. *Processes of Preference Construction in Agent-Assisted Online Shopping, in: Online Consumer Psychnology*. Lawrence Erlbaum, 2005.
21. M. Pazzani and D. Billsus. Learning and Revising User Profiles: The Identification of Interesting Web Sites. *Machine Learning*, (27):313–331, 1997.
22. R. Reiter. A theory of diagnosis from first principles. *AI Journal*, 23(1):57–95, 1987.
23. P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *ACM Conf. on Computer Supported Cooperative Work*, pages 175–186, 1994.
24. B. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *10th Intl. WWW Conference*, pages 285–295, 2001.
25. J. Schafer, J. Konstan, and J. Riedl. Electronic Commerce Recommender Applications. *Journal of Data Mining and Knowledge Discovery*, 5(1/2):115–152, 2000.
26. B. Smyth, E. Balfe, O. Boydell, K. Bradley, P. Briggs, M. Coyle, and J. Freyne. A Live User Evaluation of Collaborative Web Search. In *19th International Joint Conference on AI*, pages 1419–1424, Edinburgh, Scotland, 2005.
27. S. Spiekermann. Product Context in EC Websites: How Consumer Uncertainty and Purchase Risk Drive Navigational Needs. In *ACM Conference on E-Commernce 2004 (EC04)*, pages 200–207, 2004.
28. L. Terveen and W. Hill. *Beyond recommender systems: Helping people help each other, HCI in the New Millennium*. Addison Wesley, 2001.
29. C. Thompson, M. Göker, and P. Langley. A Personalized System for Conversational Recommendations. *Journal of Artificial Intelligence Research*, 21:393–428, 2004.
30. European Union. *Richtline 2002/92/EG des Europäischen Parlaments und des Rates vom 9. Dezember 2002 über Versicherungsvermittlung*. Amtsblatt der EU, 2002.

Biographical Sketch and Photo

Alexander Felfernig Alexander Felfernig received a PhD degree in Computer Science from Klagenfurt University, Austria, in 2001. He is co-founder of ConfigWorks, a provider of knowledge-based recommender technologies.

Bartosz Gula Bartosz Gula received the M.Sc. degree in Psychology from University of Berlin, Germany, in 2003. Currently, he is working as scientific researcher at Klagenfurt University.



Erich Teppan Erich Teppan received the M.Sc. degree in Computer Science from Klagenfurt University, Austria, in 2005. Currently, he is working as scientific researcher at Klagenfurt University.