Knowledge-based Interactive Selling of Financial Services with FSAdvisor

Alexander Felfernig[1,2], **Alfred Kiener**[3]
[1]ConfigWorks GmbH, Lakeside B01, A-9020 Klagenfurt, Austria, www.configworks.com
[2]University Klagenfurt, Computer Science and Manufacturing,
Universitätsstrasse 65-67, A-9020 Klagenfurt, Austria, www.ifi.uni-klu.ac.at
alexander.felfernig@uni-klu.ac.at
[3] Bausparkasse Wüstenrot AG, Alpenstrasse 70, 5033 Salzburg, Austria, www.wuestenrot.at
alfred.kiener@wuestenrot.at

Abstract

In this paper we describe the knowledge-based recommender application *FSAdvisor* (Financial Services Advisor) which assists sales representatives in determining personalized financial service portfolios for their customers. Commercially introduced in 2003, *FSAdvisor* is licensed to a number of major financial service providers in Austria. It supports the dialog between a sales representative and a customer by guaranteeing the consistency and appropriateness of proposed solutions, identifying additional selling opportunities and by providing intelligent explanations for solutions. In the financial services domain (especially in the retail sector) sales representatives can differ greatly in their expertise and level of knowledge. Therefore financial service providers ask for tools effectively supporting sales representatives in the dialog with the customer. Knowledge-based recommender approaches meet these requirements by allowing an intuitive and flexible mapping of marketing and sales knowledge to the representation of a recommender knowledge base. In *FSAdvisor* we integrate model-based diagnosis, constraint satisfaction and personalization thus supporting customer-oriented sales dialogs. A graphical development environment enables the implementation of financial service knowledge bases for non-programmers which leads to significant reductions of development and maintenance costs.

Task Description

Due to the increasing complexity of product assortments and high cost pressure, one of the major challenges of today's retail banking is to improve the efficiency and effectiveness of sales processes. Financial service advisory is a knowledge-intensive task which in many cases overwhelms sales representatives thus leading to low quality results for the customer. Therefore financial service providers ask for tools supporting sales representatives in the dialog with the customer. *FSAdvisor* is a knowledge-based recommender application (Burke 2000; Ardissono *et al.* 2003) helping sales representatives by simulating the behaviour of sales experts. Financial service providers currently applying *FSAdvisor* dispose of a product assortment of about 100 (partly configurable) products which cover different areas of interest (e.g. investment decisions, financing, pension, life insurance or business and property insurance). Beside a number of ongoing projects *FSAdvisor* has already been deployed for two major financial service providers in Austria.[1] In the context of such projects the following challenges have to be tackled:

- Quality of solutions. Since the product assortment is quite manifold, many representatives focus on selling a restricted set of products causing sub-optimal offers. The goal here is to identify a portfolio which fits to the wishes, needs and financial restrictions of the customer (e.g. retirement planning decisions are crucial for the customer) and conforms to the sales strategy of the company.
- Error reduction. Some products are bound to conditions which must be fulfilled by customers (e.g. certain building loan contracts can only be offered to customers with an age under 20 years). Since the ratio between the time spent with the customer and the number of sold products is important for a representative's productivity, the reduction of infeasible offers is a very important issue.
- Effective dissemination of product knowledge. The earlier a new product becomes known to sales representatives the better. Generally, the capability of selling new products is bound to the participation in training courses. The goal here is to reduce the time a representative needs to get familiar with new products.
- Customer-oriented sales. Due to a restricted knowledge about the product assortment, representatives often prefer a product-oriented advisory approach. The goal here is to automatically provide questions and explanations focusing on the customer's wishes and needs, i.e. supporting a customer-oriented sales dialog.
- Documentation. Due to regulations of the European Union, financial service providers are forced to improve the documentation of advisory sessions. Intelligent reporting is required which includes explanations as to why certain products were offered to a customer.
- Cross selling. In many cases cross-selling opportunities (e.g. a married sole wage earner with two children, taking out a loan, is also a candidate for a risk insurance) are neglected because the representative focuses on products

[1]*Wüstenrot* building and loan association (www.wuestenrot.at) and the *Hypo-Alpe-Adria* bank (www.hypo-alpe-adria.com).

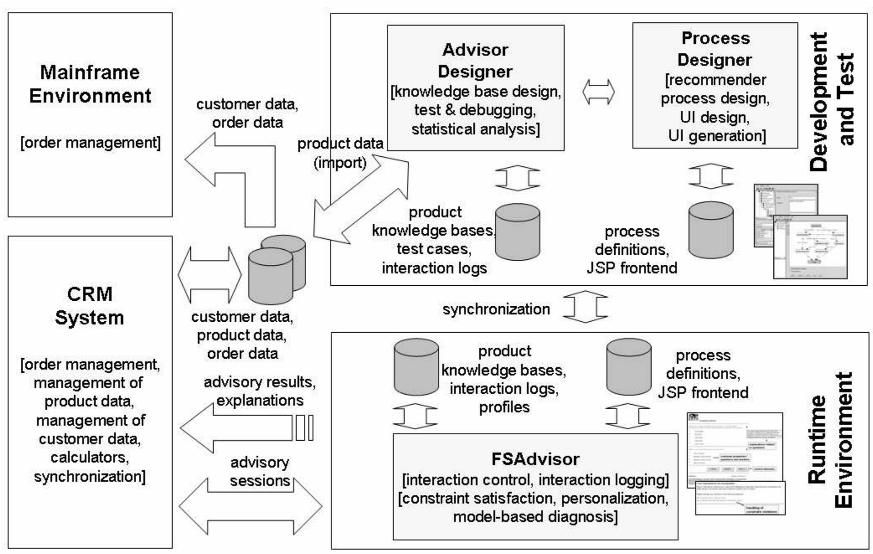

Figure 1: Overall architecture.

he knows about. Automated support for identifying cross-selling opportunities is needed in this context.

- Knowledge acquisition and maintenance. A crucial factor for a successful application of recommenders is the effective acquisition and maintenance of sales knowledge. Domain experts (generally non-programmers) must be able to handle financial service knowledge bases.

In this context *FSAdvisor* is used for the following purposes:

- checking the consistency of customer requirements (e.g. *expected return rates*, *willingness to take risks*) and (in the case of inconsistencies) supporting a corresponding error handling.
- matching customer requirements to product properties (e.g. *type of fund*).
- diagnosing and repairing a set of inconsistent customer requirements.
- explaining/documenting the calculated solutions in order to increase the customer's confidence in a given solution.

A knowledge-based approach (see Section *Uses of AI Technology*) was chosen for the following reasons:

- Financial service advisory is a complex task with a large number of constraints and possible solutions. In an integrated recommender application there exist about 1-2 million solution alternatives and about 300-400 constraints. In this context, knowledge-based approaches can significantly reduce efforts related to advisor development and maintenance.
- A customer's taste is not of primary concern in the financial services domain. Recommendations must be correct and explainable, i.e. *Collaborative Filtering* (Herlocker *et al.* 2004) or *Content-based Filtering* (Burke 2002) approaches are not the best choices.
- Intelligent explanation, debugging, and repair mechanisms as well as automated test case generation are based on model-based knowledge representations, i.e. deep knowledge about the application domain must be available (which is not the case when applying *Collaborative Filtering* or *Content-based Filtering* approaches).
- In many cases financial service providers want to develop advisors autonomously, i.e. representation formalisms are needed which support the development of recommender knowledge bases for non-programmers.

Summarizing, the key innovative contributions of the work presented in this paper are the following:

- Improvements of sales dialogs in the financial services domain through intelligent customer-oriented sales dialogs which are enabled by innovative technologies from the area of model-based reasoning and personalization.
- A graphical development and test environment which enables rapid prototyping and makes the management of knowledge bases feasible for non-programmers.

Application Description

FSAdvisor. *FSAdvisor* provides comprehensive assistance for sales representatives selling financial services to customers by supporting guided and personalized dialogs related to different wishes and life phases of customers. Advisors are applied in order to improve the dialog between sales representatives and customers, i.e. sales representatives interact with advisors when talking with the customer or use the advisor in order to prepare a sales dialog. Further application scenarios for financial service advisors are interactive training courses for new employees and advisors directly deployed on the homepage of a financial service provider where they are primarily used to pre-inform customers about the existing product assortment in order to relieve sales representatives from routine advisory jobs. In many cases, the integration of financial service advisory into existing Customer Relationship Management (CRM) environments is a

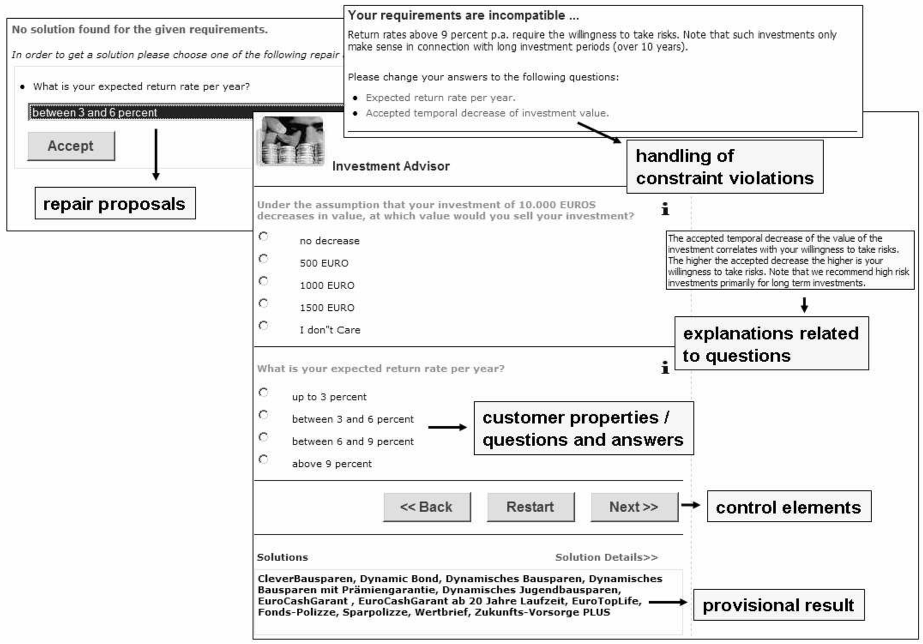

Figure 2: Example user interface.

major customer requirement. The architecture depicted in Figure 1 shows the integration of advisory services into the software environment (*CRM System* and *Mainframe Environment*) of a financial service provider. The CRM system acts as front-end for sales representatives. It can be a local installation on notebooks of field representatives or an application provided by a central server. The major task of the CRM system is the administration of customer data (address data, list of products, marital status, personal goals, etc.). The CRM system is linked to financial service advisors related to different topics such as investment, pension, financing, or life insurance (see Figure 2). Results from advisory sessions are returned to the CRM system where the corresponding contracts and offers are stored, i.e. the CRM system acts as a central administration unit for advisory results. Sales representatives synchronize their datasets weekly with a central mainframe or other server system.

Typically, an advisory session starts with an analysis of the major customer requirements which depend on the current life phase and the personal goals, wishes and financial restrictions of the customer (see Figure 3). Within this context *FSAdvisor* is used for matching customer life phases and personal goals (e.g. *starting to build a house in three years*, *to provide for one's old age* or *to provide for one's children*) to a corresponding set of selling opportunities (e.g. *medium-term investments for building a house*, *long-term investments for closing the pension gap*). Figure 2 depicts an example input unit of an investment advisor which corresponds to one state of a recommender process definition (see e.g. Figure 6). Depending on the selections, the customer is forwarded to the next input unit. In the case of contradictory answers provided by customers, a corresponding constraint handling interface is activated, if no solution can be found, a set of possible repair actions is proposed. Having completed this initial analysis phase (the result is a set of topics/selling opportunities to be discussed with the customer), one or more detailed advisory sessions can be started (e.g. an investment advisor is used to identify an investment solution which fits to customer preferences related to the dimensions profit, availability and risk). Having com-

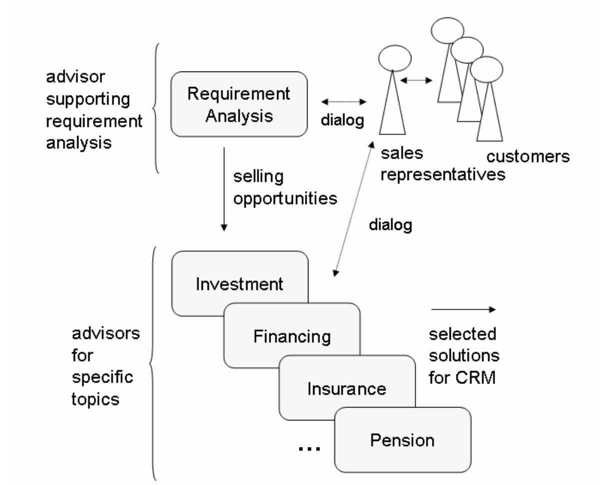

Figure 3: Types of advisors.

pleted such an advisory session the customer decides which solutions she wants to buy. Selected solutions are returned to the CRM system which manages customer orders. Answers given by customers during advisory sessions are aggregated and stored in a customer profile (the construction of customer profiles and further AI technologies integrated in *FSAdvisor* are discussed in *Uses of AI Technology*).

Development & Test Environment

Recommender knowledge bases (product knowledge base including process definition) are developed and maintained using a *Development and Test* environment (*Advisor Designer* and *Process Designer*). Having completed development and test, advisors are automatically generated and made available for sales representatives.

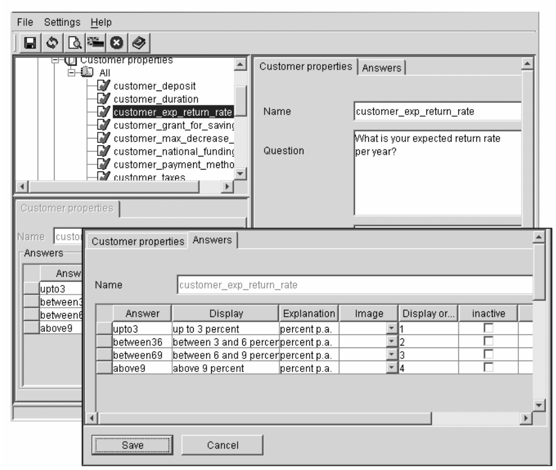

Figure 4: Definition of customer properties.

Advisor Designer *Advisor Designer* is a graphical development environment for recommender applications (advisors). It is based on Java Web Start technology which provides a browser-independent architecture for deploying Java-2 based applications on a client. The supported concepts are based on long-term AI research in the area of knowledge-based configuration and personalization (Felfernig *et al.* 2003; 2004; Ardissono *et al.* 2003; Friedrich 2004). Within *Advisor Designer* a set of recommender applications can be designed and maintained in parallel. Multilingual applications are supported, i.e. each knowledge base can be maintained for different languages. The relevant set of product- and customer properties[2] is identified and transformed into a corresponding formal representation, i.e. a *recommender knowledge base* (Soininen *et al.* 1998; Felfernig *et al.* 2003) is defined. Such a knowledge base consists of:

- product properties, i.e. a structural description of the provided set of products (e.g. *length of life insurance policies*, *premiums of life insurance policies* and links to additional product documentation).
- customer properties (see Figure 4), i.e. a description of the possible set of customer requirements (e.g. within the scope of an investment advisory process the question *under the assumption that your investment of 10.000 EUROS decreases in value, at which value would you sell your investment*? is related to the *willingness to take risks*).
- a set of constraints (see e.g. Figure 5) restricting the combinations of customer requirements and product properties, e.g. *return rates above 9 percent p.a. require the willingness to take risk*s, *customers with an age over 55 must not receive a recommendation of a pension product*.

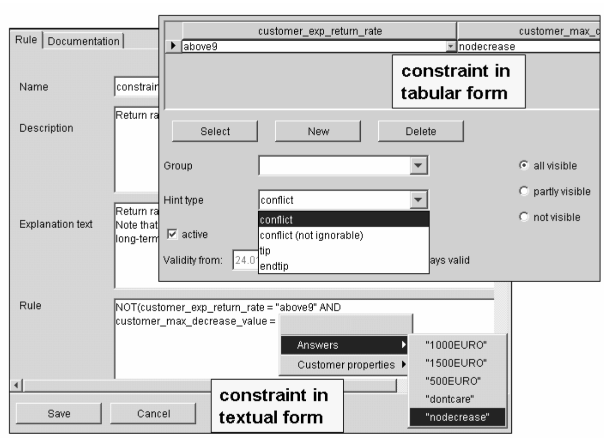

Figure 5: Constraints (textual and graphical representation).

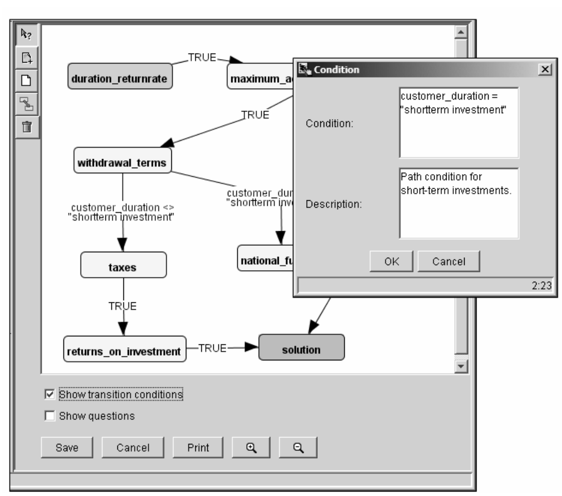

Figure 6: Process design (simplified).

Process Designer A *recommender process* represents personalized navigation paths defining the way the system adapts its dialog style to the knowledge level and interests of the customer. Interactive applications typically have a finite number of states, where state transitions are triggered by user interactions. Process definitions are based on a predicate-augmented finite state recognizer (PFSR) (van Noord & Gerdemann 2001) (constraints describe transitions between different states of a recommender process) which represents allowed navigation paths within an advisor (see Figure 6). Given the definition of a layout template, a recommender knowledge base can be automatically (no programming is needed in this context) translated into an executable advisor (see e.g. Figure 2).

[2]Data-types are Integer, Float, String and Enumeration.

Testing & Debugging Environment The increasing size and complexity of knowledge-based recommender applications makes quality assurance a critical task (Preece, Talbot, & Vignollet 1997). Figure 7 depicts the basic process for validating solutions calculated by *FSAdvisor*. Process definitions are the basis for automatically generating test cases. Solutions (results calculated by the knowledge base) for generated test cases are presented to the domain expert who decides on their validity (*Result Validation*). Correct results are marked as *checked* by the domain expert, faulty results are used by a diagnosis component (*Knowledge Base Design&Debugging*) for identifying faulty constraints in the knowledge base (Felfernig *et al.* 2004). Test cases deemed as correct by the domain expert are used for regression tests (Fleischanderl 2002). Test case generation follows a path-

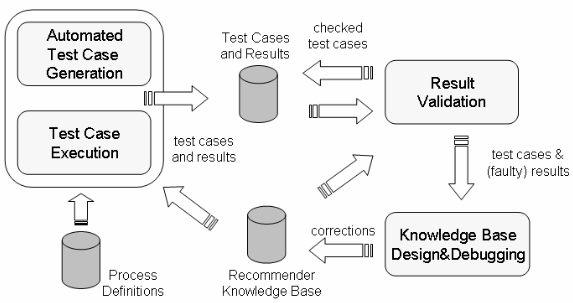

Figure 7: Validation process.

oriented approach which allows a high degree of coverage (Edvardson 1999). The disposable time for testing is restricted, consequently mechanisms are provided which reduce the amount of tests. Our experiences show that domain experts agree with accepting efforts related to the inspection of test results since solution quality is of serious concern. Calculating a complete set of test cases, which includes all possible transitions of a process definition, is only feasible for small and strongly constrained recommendation tasks. The following approaches reduce the number of test cases:

- Equivalence partitioning. Large variable domains can be split up into a set of equivalence classes out of which we can select a representative subset of test cases. A persons age can be split up into a set of equivalence classes, e.g. the age under 13, between 13 and 16 years, etc. Depending on the equivalence class, different legal regulations restrict the set of possible solutions.
- Certified Constraints. Test cases including combinations of customer requirements which are inconsistent with the knowledge base can be neglected by certifying the corresponding constraints as valid, e.g. the constraint *return rates above 9 percent p.a. require the willingness to take risk*s. If this constraint is certified, we can neglect all test cases with the corresponding assignment combinations.
- Variables with no effects. In some situations the advisor poses questions which have no influence on the solution (marketing questions, where no constraints are defined on the corresponding variable), e.g. when recommending pension products, the customer can be asked to make a decision concerning *returns on investment* (*singular*, *annuity payment*). Since pension products allow a decision to be taken at the end of the investment period, the customers answer doesn't have any effect on the solution.
- Random selections. Confronted with large variable domains and lengthy processes, random selections (e.g. path selection or assignment selection, i.e. reduction of a variable domain using a statistical distribution) are a means to reduce the set of test cases.

The complete set of possible test cases for a recommender knowledge base with 20 customer properties (variables) with a domain of cardinality 5 would comprise about 5^{20} test cases which is definitely infeasible for a domain expert. Reducing the input space to 20 possible paths each path defined by 7 variables and 5 possible values per variable reduces the number of potential test cases to 1.5 mio which is still unfeasible. By applying additional restrictions (equivalence partitioning, certified constraints etc.) we can reduce the number of test cases from 1.5 mio to about 500.

Uses of AI Technology

Concepts implemented in *FSAdvisor* are based on long-term AI research in the area of knowledge-based configuration and personalization (Felfernig *et al.* 2003; 2004; Ardissono *et al.* 2003; Friedrich 2004).

Selected Recommender Technology

In contrast to *Collaborative Filtering* (Herlocker *et al.* 2004) and *Content-based Filtering* (Burke 2002) approaches, *Knowledge-based Recommender Systems* (Burke 2000; Ardissono *et al.* 2003) exploit deep knowledge about the product domain in order to determine solutions fitting to the customer's wishes and needs. Using such an approach, the relationship between customer requirements and financial services can be explicitly modelled in a recommender knowledge base (Felfernig *et al.* 2003). Such model-based representations are an excellent basis for applying model-based diagnosis and testing techniques.

Constraint Satisfaction

Constraint Satisfaction Approach. *FSAdvisor* is based on constraint satisfaction problem solving (Tsang 1993). A Constraint Satisfaction Problem (CSP) (C,V,D) (Tsang 1993) is defined by a set V of variables x_i, a set C of constraints c_j and a set D of domains d_i which defines for each variable the set of possible values. A CSP is solved if there exists a set of instantiations of the variables x_1, x_2, ..., x_n s.t. all constraints contained in C are satisfied. A *recommendation task* can be defined as a CSP (C, V_{SRS}, V_{PROD}, D_{SRS}, D_{PROD}), where V is additionally divided into V_{SRS} (set of variables describing customer requirements) and V_{PROD} (set of variables describing product properties). The constraint solver tries to find a solution for a given recommendation task. If no solution can be found, constraints with a priority > 0 (0 is the highest priority) are relaxed starting with constraints with lowest priority. If nothing but non-relaxable constraints (priority = 0) remain, a repair mechanism is activated. In addition to constraints, *FSAdvisor* supports *tips*, i.e. constraints representing e.g. cross-selling opportunities which are shown to the customer without interrupting the recommender process (in contrast to constraints, where an additional constraint violation handling dialog is started - see e.g. Figure 2). If a customer is risk-averse and interested in long-term investments, a tip could be: *long-term investments reduce risks, i.e. allow higher return rates than short-term investments without taking high risks.*

Diagnosis and Repair of Requirements. If the result set is empty, conventional recommender applications tell the user that no solution was found. *FSAdvisor* supports the calculation of repair actions for customer requirements (a minimal set of changes allowing the calculation of a solution). If Σ = {$x_1 = a_1$, $x_2 = a_2$, ..., $x_n = a_n$} is a set of customer requirements ($\Sigma \cup C$ has no solution), a repair is a minimal set of changes to Σ (resulting in Σ') s.t. Σ' $\cup C$ has a solution. The computation of repair actions is based on the Hitting Set algorithm (Reiter 1987) (see Figure 2 for the representation of repair alternatives). Model-based diagnosis of customer requirements has been introduced in (Felfernig *et al.* 2004), state-of-the-art constraint reasoners (Junker 2004) provide conflict detection but do not support the calculation of minimal repair actions.

Automated Test Case Generation. Automated test case generation is based on the definition and execution of a constraint satisfaction problem (Tsang 1993). For this purpose a (complete) set of possible paths through a recommender process is determined. For each path a CSP is generated and executed - identified solutions correspond to test cases. Generally, this set of test cases has to be reduced in order to be manageable (see Section *Development & Test Environment*). Test case generation is extensively discussed in the Software Engineering community (Edvardson 1999) but (with a few exceptions, e.g. (Preece, Talbot, & Vignollet 1997)) neglected in knowledge-based systems development.

Personalization

Customer Profiles. Due to the heterogeneity of users, *FSAdvisor* includes mechanisms allowing the adaptation of the dialog style to the user's skills and needs (Ardissono *et al.* 2003). 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 which relate user answers to abstract dimensions (Ardissono *et al.* 2003) such as *preparedness to take risks* or *interest in high profits* (dimensions describing the users interests) and *knowledge about funds, etc.* (dimensions describing the users knowledge about the domain). Initial values for the customer profile are collected in a requirements analysis phase where best-matching stereotypes are applied to complete profiles. Foundations for the personalization concepts implemented in *FSAdvisor* are presented in (Ardissono *et al.* 2003).

Personalized Dialog Style. Customers have different approaches to specify their requirements which range from the direct specification of product parameters (e.g. a certain savings account, running for 3 years) to a very general specification of their personal goals (e.g. financing their children's education). An adaptation of the interaction style contributes to an improved approximation to the behavior of a human sales expert. Depending on the profile information and a set of answers provided by the customer, the following concepts support the personalization of the dialog style:

- Alternative formulation of questions, e.g. questions posed to experts can be differentiated from those posed to customers with less knowledge about financial services.
- Rule-based formulation of default-answers, e.g. in an investment advisory session where the goal of the customer is to *put money by for a rainy day* the default answer to a question related to the *maximum accepted decrease in value* is *no decrease in value accepted*.
- Alternative explanations for constraint violations, e.g. if the customer is a novice, a very general explanation about changes in the pension law is given, more detailed information can be included for experts.

Personalized Repair Proposals. If no solution can be found for a given set of customer requirements, *FSAdvisor* proposes a minimal set of possible repair actions which allow the calculation of a solution. Different customer properties (variables) have an assigned priority which indicates the importance of the variable for the customer. The lower the priority of the variable the higher the probability is that the variable is considered as focus of repair actions, e.g. if the type of *returns on investment* (*reinvestment*, *dividend output*) is unimportant for a customer, this property is primarily considered as a potential candidate for repair actions, i.e. repair actions are adapted to the customer's preferences. The repair mechanisms are based on model-based diagnosis concepts (Felfernig *et al.* 2004) which are definitely not integrated in similar systems such as (Junker 2004).

Personalized Ordering of Solutions. A solution for a given recommendation task is a set (portfolio) of financial services. The order of solutions should strictly correspond to the degree a solution contributes to the wishes of a customer. *FSAdvisor* supports multi-attribute object rating (Ardissono *et al.* 2003), where each solution is evaluated w.r.t. to a predefined set of abstract dimensions. *Profit*, *availability* and *risk* are examples for such abstract dimensions. Depending on the weighting of the dimensions for a specific customer (e.g. a customer is strongly interested in products with a high return rate) the set of solutions is ordered using the formula $g(x) = \sum_{i=1}^{n} e_i s_i(x)$, where n denotes the number of dimensions, $g(x)$ represents the utility of one solution x, e_i represents the customer's interest in dimension i, and s_i is the contribution of solution x to dimension i.

Personalized Solution Presentation. For each solution a corresponding set of explanations is calculated. The generation of explanations is based on the concepts presented in (Friedrich 2004). Furthermore *solution-specific explanations* are supported, e.g. if the customer is strongly interested in high return rates and a solution shows a remarkable return rate, this fact is explicitly mentioned when the solution is presented to the customer.

Knowledge Acquisition

Advisor Designer and *Process Designer* allow the design and maintenance of recommender knowledge bases for non-programmers. In many cases constraints are defined within a specific *context*, e.g. constraints related to customers interested in long-term investments. Having defined a context *long-term investments*, the condition *customer_duration_of_investment = longterm* can be omitted when defining constraints related to that context. Contexts are defined using the environment for the textual definition of constraints (see Figure 5).

Effective debugging support for the implementation of recommender knowledge bases is a critical issue for a successful development of recommender applications. In *FSAdvisor* we have implemented model-based diagnosis algorithms (Reiter 1987; Felfernig *et al.* 2004) supporting the identification of minimal sources of inconsistencies in recommender knowledge bases. Similar to the diagnosis and repair of customer requirements, we apply model-based diagnosis techniques in order to identify a minimal set of constraints $\in C$ which - when deleted from the recommender knowledge base - allow consistency restoration.

Application Use and Payoff

Application. *FSAdvisor* is installed for 150 sales representatives of the Hypo-Alpe-Adria bank since July 2003 and for 1400 sales representatives of the Wüstenrot building and loan association since June 2004. The motivation for the application of knowledge-based recommender technologies was to improve the efficiency and effectiveness of sales processes in terms of solution quality (substituting a product-oriented advisory approach with a customer-oriented one), error reduction (checking the feasibility of calculated solutions at the customer side), and intelligent documentation of advisory sessions.

Maintenance. The *development and maintenance* of recommender knowledge bases is conducted by domain experts, i.e. the provided concepts for knowledge base development and test have shown to be applicable for non-programmers (after a three-day course and a first project in which they were accompanied by an engineer experienced in advisor development). In the *Wüstenrot* case 3 domain experts are responsible for the development and maintenance of knowledge bases (investment & financing, pension & life insurance, property insurances). Changes are conducted within the development and test environment, where a set of test cases is used for regression testing. Having successfully completed the test phase, the new version of the knowledge base is synchronized with the runtime environment.

Product Assortment. The sales volume of Wüstenrot in 2004 was about 345.000 products (185.000 building loan contracts, 50.000 personal insurances, 100.000 property insurances, and 10.000 other products). Wüstenrot as well as the Hypo-Alpe-Adria bank had the strategy to deploy *FSAdvisor* for Austrian sales organizations and in a second step to deploy the application to subsidiary companies which will be the focus of follow-up projects.

Evaluation by Sales Representatives. The Wüstenrot CRM system (ADAP - Aussendienstmitarbeiter-Arbeitsplatz) is installed for 1.400 sales representatives. On the average a sales representative sells about $60 - 70$ products per year (highly performing sales experts sell up to 500 products per year). The advisors implemented for ADAP have been evaluated by sales representatives (experts as well as less experienced representatives) from different sales organizations in Austria. The interviewees were agreeing on the quality of the calculated solutions and the design of the advisory dialogs. Automated generation of intelligent summaries of advisory sessions, error-free solutions and cross-selling support are the major motivations for applying *FSAdvisor*, where all those aspects were regarded as important by less experienced sales representatives. Due to legal regulations of the EU, sales representatives of financial service providers and assurance companies are forced to a transparent documentation of advisory processes - this was the major motivation for experienced sales representatives.

Time Savings. *Time savings* related to the application of the advisors can amount up to $30 - 50\%$ per advisory session. This reduction is achieved by generating advisory summaries and using customer answers and results from the advisory process to automatically generate offers.

Quality of Solutions. *FSAdvisor* knowledge bases are developed and tested by marketing and sales experts. Sales representatives can rely on the solutions calculated by the financial advisor and can provide the customer with qualified explanations. 100% error-free offers are provided to the customer. Added value is provided by intelligent explanations for calculated solutions which are automatically generated and used as starting point for future advisory sessions.

Other Applications. Although financial services is our leading application domain, a set of additional applications have been implemented on the basis of the recommender technologies presented in this paper, e.g. the digital camera advisor PIXLA which was implemented for the largest Austrian online product platform (www.geizhals.at). PIXLA is deployed since November 2003 and exhibits about 10.000 successful advisory sessions per month. Users of www.geizhals.at were interviewed before and after the introduction of PIXLA. The major result of the study was a statistically significant increase of customer satisfaction (related to dimensions such as easiness to find products etc.).

Application Development and Deployment

Depending on the complexity of the product assortment and the areas to be supported by recommender functionality, the overall efforts related to a customer project are between 1.5 man months (single advisor) and 15 man months (complete integration of an advisor suite into the customer's CRM system, i.e. construction of the knowledge base and integration of *FSAdvisor* in the customers CRM environment). In most cases the strategy of the customer is to seamlessly integrate *FSAdvisor* on the technical level as well as on the organizational level, i.e. on the one hand *FSAdvisor* is an integrative part of the CRM system, on the other hand domain experts must be capable of developing and maintaining financial service knowledge bases autonomously. In the first phase of a customer project domain experts as well as technical experts (e.g. CRM system programmers) are introduced into the concepts behind *FSAdvisor*, prototypical knowledge bases for selected application areas are developed and potential additional requirements related to *FSAdvisor* are identified. In the following expert groups are established who are responsible for the development of advisors, where *FSAdvisor* provides a set of reference knowledge bases which can be adapted for the customer-specific product assortment. Before the official deployment of the knowledge bases for sales representatives, two quality assurance cycles must be passed. In this context the test environment is a very helpful tool which contributed to shorter feedback cycle times between sales representatives and domain experts.

Learnings

On the organizational level a major precondition for the successful implementation of an advisor project are accompanying marketing activities. In order to be able to use the new service, the attention of the customer must be directed to the new service. On the technical level a crucial success factor for advisor projects is that non-programmers are enabled to implement and maintain knowledge bases, i.e. the knowledge acquisition bottleneck must be reduced as much as possible by a development environment supporting *graphical design and debugging of knowledge bases*. *Rapid prototyping* is a very useful concept in the context of recommender application development since domain experts directly see the effects of changes to the recommender knowledge bases (constraints, explanations, figures, etc.). The correctness of solutions plays a vital role for the acceptance of the system by sales representatives applying the system while communicating with the customer. Therefore, *automated test case generation* mechanisms are needed which support the effective validation of knowledge bases.

Conclusion

FSAdvisor is a knowledge-based recommender application including as set of innovative AI technologies (model-based reasoning and personalization) supporting customer-oriented sales dialogs for sales representatives. A graphical development environment and automated test data generation allow the effective implementation and maintenance of recommender knowledge bases for non-programmers. This has been shown by the autonomous development of financial service advisors by domain experts. Future work will focus on the analysis of advisory sessions with the goal to identify customer requirements not supported by the offered product assortment and to automatically localize parts of a recommender knowledge base which are responsible for unsuccessful advisory sessions.

Acknowledgements

The authors would like to thank the Bausparkasse Wüstenrot AG and the Hypo-Alpe-Adria bank who contributed to the development of *FSAdvisor*.

References

Ardissono, L.; Felfernig, A.; Friedrich, G.; Jannach, D.; Petrone, G.; Schaefer, R.; and Zanker, M. 2003. A Framework for the development of personalized, distributed web-based configuration systems. *AI Magazine* 24(3):93–108.

Burke, R. 2000. Knowledge-based Recommender Systems. *Encyclopedia of Library & Information Syst.* 69(32).

Burke, R. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 4:331–370.

Edvardson, J. 1999. A Survey on Automatic Test Data Generation. In 2^{nd} *Conference on Computer Science and Engineering CCSSe'99*.

Felfernig, A.; Friedrich, G.; Jannach, D.; Stumptner, M.; and Zanker, M. 2003. Configuration knowledge representations for Semantic Web applications. *AI Engineering Design, Analysis and Manufacturing Journal* 17:31–50.

Felfernig, A.; Friedrich, G.; Jannach, D.; and Stumptner, M. 2004. Consistency-based Diagnosis of Configuration Knowledge Bases. *AI Journal* 2(152):213–234.

Fleischanderl, G. 2002. Suggestions from the software engineering practice for applying consistency-based diagnosis to configuration knowledge bases. In 13^{th} *International Workshop on Principles of Diagnosis (DX-02)*.

Friedrich, G. 2004. Elimination of Spurious Explanations. In 16^{th} *European Conference on Artificial Intelligence (ECAI'04)*, 813–817.

Herlocker, J. L.; Konstan, J. A.; Terveen, L. G.; and Riedl, J. T. 2004. Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems* 22(1):5–53.

Junker, U. 2004. QUICKXPLAIN: Preferred Explanations and Relaxations for Over-Constrained Problems. In *19th National Conference on AI (AAAI04)*, 167–172.

Preece, A.; Talbot, S.; and Vignollet, L. 1997. Evaluation of Verification Tools for Knowledge-Based Systems. *International Journal of Human-Computer Studies* 47:629–658.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial Intelligence* 23(1):57–95.

Soininen, T.; Tiihonen, J.; Maenistoe, T.; and Sulonen, R. 1998. Towards a General Ontology of Configuration. *AI Engineering Design Analysis and Manufacturing Journal, Special Issue: Configuration Design* 12(4):357–372.

Tsang, E. 1993. *Foundations of Constraint Satisfaction*. London: Academic Press.

van Noord, G., and Gerdemann, D. 2001. Finite State Transducers with Predicates and Identities. *Grammars* 4(3):263–286.