A User-Oriented Model-Driven Requirements Elicitation Process based on User Modeling

Han Liu[12], Chao Li[1], Jizhe Wang[1], Qing Wang[1], Mingshu Li[1]

[1]Itechs. Lab, Institute of Software, Chinese Academy of Sciences Beijing, 100080, China P.R
[2]Department of Computer Science, University of Toronto Toronto, M5S 3G4, Canada.

Abstract As software is becoming more and more interweaved with people, organizations, and social systems, the users we face are becoming more and more complex in all aspects. On the other hand, user participation is largely ignored in traditional requirements elicitation methods (including Model-Driven Requirements Elicitation (MDRE) methods). In this paper, we adopt user modeling techniques into requirements elicitation, specifically MDRE, enhancing MDRE process into a user-oriented one, and at the same time personalizing the requirements reuse pattern of MDRE. Our approach facilitates user collaboration and interaction in MDRE and establishes personalized requirements reuse in MDRE, consequently offers better participation experiences for users.

1 Introduction

In many surveys conducted so far such as [STANDISH 95, 99], lack of user input took the most responsibilities for software project failure. Nowadays, software systems, instead of being attachments of hardware systems, are becoming more and more interweaved with people, organizations, and social systems. Consequently, the users we face are becoming more and more:

- complex in organizational structure, workflow relationships, and dynamic interactions
- hierarchical in organizational roles
- specialized in knowledge body
- fuzzy-headed in what the software system they need will look like

All these bring new challenges in how to elicit user input correctly and completely. We argue that attention should be paid to consider how to carry out effective collaborations and interactions with users from the very beginning of the software life cycle, that is, the requirements elicitation (RE) phase [Li 00].

In the RE state of art, the analysts typically drive RE process; users are just passively involved in. This may partly due to the neglect of user participation in traditional RE approaches. Additionally, the requirements reuse approaches appeared in literature, such as the operational approach [Darimont 97] and the analogical approach [Massonet 97], did not provide user-oriented practice guide for dealing interactions with users.

To cooperate with users with the characteristics mentioned above, we identified two possible relevant vacancies in RE arena due to the methodological overlook of user participation:

1. *Absence of user-oriented RE process*: most existed RE approaches do without the recognition of internal structures of users, which partly results in ad hoc user participation.
2. *Absence of adaptive/personalized requirements reuse*: in complex social systems, users in different organizational positions with different backgrounds possess different knowledge body and interests. On the other hand, today's overload of information often makes information systems ineffective. What of importance is to deliver 'right knowledge' to 'right person'. In RE process, requirements reuse promises many benefits. However, we should not promise these benefits, in turn, to our users while only relying on the experiences of analysts to perform such personalization.

In this paper, we adopt user modeling techniques into RE, specifically Model-Driven Requirements Elicitation method (MDRE), enhancing MDRE process into a user-oriented one, and at the same time personalizing the requirements reuse pattern of MDRE. We utilize stereotyped user modeling [Rich 89] and user-relation modeling to provide workflow management of user RE process, and employ user preference modeling to support personalized requirements reuse.

The paper is organized as follow. In Section 2 and 3, MDRE and user modeling techniques are briefly introduced and summarized, respectively. In Section 4, our approach combining these two techniques is elaborated in detail with three subsection, namely 4.1 overall user-oriented process model, 4.2 MDRE workflow definition language and 4.3 implementing techniques for personalized requirements reuse. The first subsection 4.1 gives an overview of our approach and the corresponding process. The next two subsections 4.2 and 4.3 present the technical details and the choices we made in designing and implementing MDRE workflow management and personalized requirements reuse. In Section 5, conclusions are made and future work of our research is introduced.

2 Model-Driven Requirements Elicitation (MDRE)

MDRE originates from the ideas of knowledge representation, that from the cognitive perspective, the world is considered as instances of some abstract concepts. Examples of such concepts include entity, action, condition, event and so on. In 1986, this concept-based idea was introduced into requirements elicitation and modeling with the pioneer work of S. Greenspan [Greenspan 86] in RML (Requirements Modeling Language). Up-to-date, several important MDRE methods have been proposed in the literature, such as goal-directed requirements elicitation methods: KAOS [Dardenne 93], I* [Chung 00] and Scenario-based method: CREWS [Maiden 98].

MDRE method is based on a three-layer model architecture. The three models are meta model, domain model and instance model respectively. Meta model, which is composed of abstract concepts and their relations, is domain independent, and can be regarded as the oracle of MDRE method, which conceptually drives the MDRE process. Domain model, an instance of meta model in certain domain, captures common characteristics of entities and relations in the domain of interest. Instance model, in turn an instance of domain model, is a model of entities and relations for a specific software system and its environment. The abstract level decreases from meta model to instance model. Fig. 1 illustrates the three models with a purchase operation in instance level.

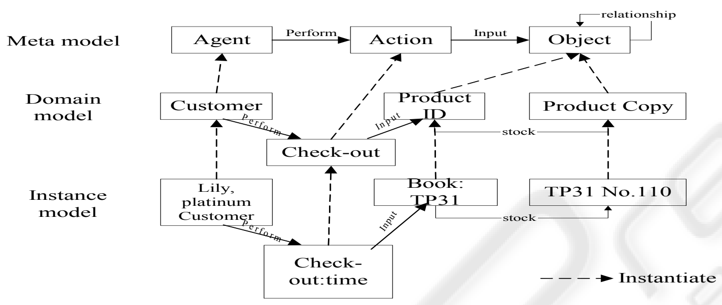

Fig. 1 three-layer architecture of MDRE

We can see that certain MDRE method distinguishes itself by the constituents of its meta model, which determines the way of thinking of analysts who perform RE activities. We can say, most RE methods contain a meta model, whether explicitly or implicitly. For example, in structured analysis, meta model includes data and process as concepts; while in object-oriented paradigm, the concepts in its meta model include object, message and so on. A MDRE method, besides a meta model which dictates what to elicit, contains an elicitation strategy as another important component which defines how to elicit. The strategy describes specific steps on how to find instances of concepts defined in meta-model in problem domain. In goal-directed method, elicitation strategy kicks off on finding the instances of the concept 'goal', or say goal refinement. In scenario-based method, elicitation strategy advises recognizing scenarios first, which in fact belong to the instance model, and then acquire instances in domain-model through abstraction from scenarios.

3 User Modeling Techniques

User modeling is usually tracked back to the end of 70's last century [Perrault 78]. After decades, especially in recent years when the importance of user adaptive systems is highly recognized, numerous application systems are developed based on user modeling techniques.

User Modeling techniques concern mainly on modeling of user's category, property, plan and preference in order that software systems could adaptively fit the needs of different users. Traditional modeling techniques can be used in modeling user category and property. It's more difficult to model user's plan because of the combinatorial explosion to determine user’s goal from casual user action. In the modeling of user preference, users' behavior (interaction with software systems) is the main source to infer the interests of users. At first, natural language processing is the most frequent used technique in collecting user information. Now many alternative information sources are identified and are more effective than natural language processing, e.g., user's interactions with GUI, hit sequence of hypertext links and/or the user’s queries with databases.

User modeling has been successfully applied in many application areas, including library reservation systems, user context-sensitive help systems, user-sensitive navigation systems, adaptive websites and user sensitive filters.

4 Incorporating User Modeling Techniques into MDRE

Our approach offers two improvements to MDRE process by incorporating user modeling techniques: 1) setup workflow management to facilitate user collaboration and interaction in MDRE; 2) establish personalized requirements reuse in MDRE.

In complex user environment, static user relations have strong effect on the RE workflow between users. So recognize user models containing user relations before enacting RE activities can facilitate both multi-users RE workflow management and the integration of the elicitation strategy of MDRE method into multi-user environment. In our approach, we use stereotyped user modeling and user-relation modeling to capture the underlying user models in multi-user environment. These user models, combined with the elicitation strategy of MDRE method, are used to help define MDRE workflow before the enactment of RE activities. Finally, a workflow engine, dispatching and maintaining tasks among multi-users, supports the enactment of predefined MDRE workflow.

As mentioned above, providing adaptive/personalized services to users is important in reusing domain knowledge in complex user environment. In our approach, we provide a context-based mechanism for implementing personalized requirements reuse because we believe the exact knowledge a user needed is determined by some contextual factors. We use three contextual factors among others for deciding what information is more relevant to users in context. The three contextual factors are 1) user-specific behaviors, 2) user stereotype, and 3) user’s task at hand.

4.1 Overall User-Oriented Process Model

Fig. 2 illustrated the overall process model of our user-oriented approach of integrating user modeling techniques and MDRE. The MDRE process consists of four steps and is accompanied by a domain knowledge base. We can see in Fig. 2 that the MDRE method is a plug-in component, thus the architecture is flexible enough for adopting new MDRE methods. Before we go to details of the four steps, we briefly introduce the components of the domain knowledge base of their contents and usages.

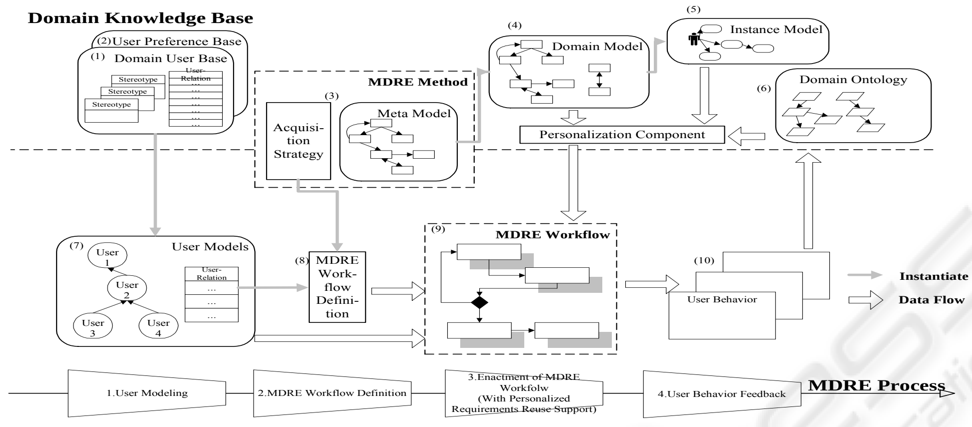

Fig. 2 Overall User-Oriented Process Model

Domain User Base: Domain User Base ((1) in Fig. 2) contains user stereotypes and possible user-relations in domain of interest.

User Preference Base: User Preference Base ((2) in Fig. 2) contains user preferences data of all stereotyped and non-stereotyped users.

Domain Model: Domain model ((4) in Fig. 2) is the second level model of the MDRE three-layer architecture.

Instance Model: Instance model ((5) in Fig. 2) is the third level of the MDRE three-layered architecture.

Domain Ontology: Domain ontology ((6) in Fig. 2) contains the factual ontology in domain of interest, namely the concepts and their semantic relationships.

Now is the illustration about how the process works:

1. *User Modeling: Define User Models by instantiating and tailoring the domain user base ((7) in Fig. 2)*
2. *MDRE workflow definition: define MDRE workflow in comply with user-relations in user models and elicitation strategy of MDRE method ((8) in Fig. 2)*
3. *Enactment of MDRE Workflow (with Personalized Requirements Reuse Support): the actual RE activities with the support of workflow engine and domain knowledge base*
4. *User Behavior Feedback: record and analyze user behavior in RE activities to accomplish evolution process of domain knowledge base*

4.2 MDRE Workflow Definition Language

The workflow definition language we used contains only a small subset of modeling elements of traditional process modeling languages (such as SLANG, SPADE in [Armenise 93]). In this small language, process, task, role are three basic modeling elements:
Role: an abstract user assigned with some tasks.
Task: the atomic unit of the process. Task must be associated with some role(s). To indicate the status of the task, it can carry a return value of Boolean or Integer type. We define 4 temporal relations between tasks: Begin-Begin (BB), Begin-End (BE), End-Begin (EB), and End-End (EE). For example, if $Task_1$ and $Task_2$ have BB relation, then the start point of $task_2$ must be later than the start point of $task_1$.
Process: the network of tasks and their relations.

The language ignores the input/output of both tasks and processes in order to ease the access of documents and work products among users. A potential shortcoming of this choice is users have to take the responsibilities of maintaining the consistency between documents and work products. Luckily, it won't be a tough problem if using some configuration tools. In the workflow definition language, role is abstract user. Specific user must be filled in during instantiation. For example in a subordinate relation, there are only two roles: superior and junior. In workflow enactment, these abstract roles must be binding to some concrete users.

4.3 Implementation Techniques for Personalized Requirements Reuse

The personalized requirements reuse support is based on three contextual factors: 1) user-specific behaviors, 2) user stereotype, and 3) user's task at hand. The techniques engaged in implementing context-based personalization are: stereotype-based collaborative filtering, ontology-based semantic search, rule-based user preference revision. The implementation framework using this technique is illustrated in Fig. 5. Context is in the left box, containing user behaviors, task description stereotype and corresponding user preference. As we can see in Fig. 5, user can get two kinds of knowledge support, task-specific knowledge and preference knowledge. Task-specific knowledge is derived from task description, while preference knowledge comes from user stereotype and user behavior.

The process of generating preference knowledge:

1. Retrieve all preference records of past users with the same stereotype; perform collaborative filtering [herlocker 99] to predict the user's domain preference based on those history records and the user's initial profile.
2. With the user preference, perform ontology-based semantic search on domain model to retrieve most relevant items.
3. In the RE activities, user behaviors are recorded, with which we can use rule-based revision to dynamically adjust the user's preference to count in user-specific behavior contextual factor.

The process of generating task-specific knowledge:

1. Acquire task property vector by performing ontology-based semantic search on task description
2. With the task property vector, perform ontology-based semantic search on domain model to retrieve most task-relevant items.

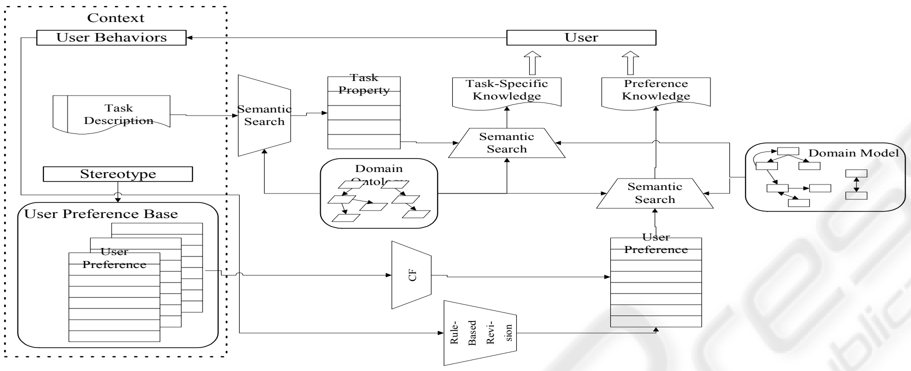

Fig. 5 the Implementation Framework of Personalized Requirements Reuse

Before we introduce the techniques in detail, some structures are first introduced as follow:

User preference vector: the user preference vector keeps the mapping from a user (u_i) to some concepts in domain ontology, which measures the users interests of those concepts. The first row in Table 1 is the concepts c_i, the second row is the relative preference score $s_r(c_i)$ of u_i to each concept. The range is from 0 to 100.

Table 1 User Preference Vector

C_1	c_2	c_3	•	•	c_n
$S_r(c_1)$	$S_r(c_2)$	$S_r(c_3)$	•	•	$S_r(c_n)$

Task property vector: Task property vector shows whether a concept is relevant to current task. If so, the score of the concept equals to 100, otherwise the score is 0. Similar to the user preference vector, the first row in Table 2 is the concepts c_i, the second row is the relative preference score $s_t(c_i)$ of current task to each concept.

Table 2 Table of task property

c_1	c_2	c_3	•	•	c_n
$S_t(c_1)$	$S_t(c_2)$	$S_t(c_3)$	•	•	$S_t(c_n)$

Domain ontology: domain ontology contains all the factual ontology in the domain of interest, namely the concepts and the their relations in domain of interest. Three semantic relationships are selected out: synonymy, generic, and specific semantic relation.

1. Predict user preference with collaborative filtering

If a user u_a 's preference $p_{a,j}$ to a concept c_j is not available in the initial user profile, we can predict it with the following formula:

$$p_{a,j} = \bar{v}_a + \kappa \sum_{i=1}^{n} w(a,j)(v_{i,j} - \bar{v}_i) \quad ①$$

In formula ①, all the user preference records of the same stereotype in the user preference base are collaboratively considered to predict u_a's preference. $v_{i,j}$ represents u_i's preference to a domain concept c_j. κ is a normalization factor. v_i is the average value user u_i to all the domain concepts, it can be computed with the following formula:

$$\bar{v}_i = \frac{1}{|I_i|} \sum_{j \in I_i} v_{i,j} \quad ②$$

In ②, I is the set composed of all the domain concepts to which u_a's preference is not Φ. In ①, w(a,i) is the similarity degree between user u_a and u_i. We computes this similarity with Pearson coefficient [Resnick 94], the formula is:

$$w(a,i) = \frac{\sum_j (v_{a,j} - \bar{v}_a)(v_{i,j} - \bar{v}_i)}{\sqrt{\sum_j (v_{a,j} - \bar{v}_a)^2 \sum_j (v_{i,j} - \bar{v}_i)^2}} \quad ③$$

2. Ontology-based Semantic search

Ontology-based semantic search computes the correlative degree between each item in domain model and domain concepts in domain ontology, then assigns a score to each item based on user preference vector. The score indicates a user's possible interests concerning a particular item.

Most domain concepts are not isolated. each has some semantic relations with others (In domain ontology, such relations include synonymy, generic, and specific semantic relation. For each concept c_i, so called "semantic family" is composed of all concepts which have either relations with it). We believe when considering the correlation between an item in domain model and a particular concept in domain ontology, it's better and more natural to count in not only the concept ontology itself but also its semantic family. Table 3 is an example of a concept c_i's semantic family.

Table 3 concept c_i's semantic family.

c_{i1}	c_{i2}	c_{i3}	•	•	c_{ik}
$h(c_{i1})$	$h(c_{i2})$	$h(c_{i3})$	•	•	$h(c_{ik})$
$w_{c_1c_1}$	$w_{c_1c_2}$	$w_{c_1c_3}$	•	•	$w_{c_1c_k}$

In table 3, the first row is the concepts in the semantic family of concept c_i. The second row denotes the scores of every correlative type, namely synonymy, generic and specific. The third row is the weights of correlation between concept c_{ij} and c_i. All these values are stored in domain ontology.

Before the algorithm, we first introduce some definitions:

G_0 = {c_i | $1 \le i \le n_c$, (n_c is the number of domain concepts in user preference vector)
G_{c1} = {b | b is the domain concept which has synonymy relation with concept c}
G_{c2} = {b | b is the domain concept which has generic relation with concept c }
G_{c3} = {b | b is the domain concept which has specific relation with concept c }
I = {c | c is relevant with item i}

$k_{c,j}$ is the number of keywords in concept c which are matched in $item_i$. For example, if key(c) is the set of keywords in concept c and key($item_i$) is the set of keywords in $item_i$, then $k_{c,i} = |key(c) \cap key(item_i)|$.

For $item_i$, we compute its score when regarded as preference knowledge and task-specific knowledge (denoted as $h_r(item_i)$ and $h_t(item_i)$) with the following two formulas respectively:

$$h_r(item_i) = \sum_{c \in I \cap G_0} \{s_r(c) \cdot k_{ci} + \kappa \sum_{l=1}^{3} \left[\sum_{c' \in I \cap G_{cl}} s_i(c') \cdot w_i(c') \cdot k_{c'i} \right] \} \quad ④$$

$$h_t(item_i) = \sum_{c \in I \cap G_0} \{s_t(c) \cdot k_{ci} + \kappa \sum_{l=1}^{3} \left[\sum_{c' \in I \cap G_{cl}} s_i(c') \cdot w_i(c') \cdot k_{c'i} \right] \} \quad ⑤$$

In ④ and ⑤, the first part of the equation computes the relevant scores of $item_i$ to concept c, the second part computes the relevant scores of $item_i$ to the semantic family of concept c. k is a normalization factor.

3. Rule-based user preference revision

When dynamically adjusting user preference, we need to compute two ratios for a particular domain concept c_i. The first one is the percentage the score of c_i to the sum of all domain concepts' preference scores: $T_1 = s_r(c_i) \Big/ \sum_{j=1}^{n_c} s_r(c_j)$ (n_c is the number of domain concepts in user preference vector). The second one is the percentage of actual accessing times of each domain concept to the sum of the times all the concepts are actually accessed within a period of user RE activities (multiple domain concepts might be involved in one access of an item): $T_2 = \sum_{j=1}^{n_i} s_{ci}(item_i) \Big/ \sum_{i=1}^{n_c} \sum_{j=1}^{n_i} s_{ci}(item_i)$ （n_i is the total number of the domain concepts' actual accessed times.）. The function $s_{ci}(item_i)$ is defined as follow:

if (key(ci)∩key(itemi)$\neq \Phi$) $s_{ci}(item_i) = 1$ else $s_{ci}(item_i) = 0$;

Define $\Delta T = T_2 - T_1$. User preference revision rules are defined as follow:

Rule1. $|\Delta T| \le 0.1$, the predictive result is proper and we need not adjust it this time.

Rule2. $|\Delta T| > 0.1$, c_i should be adjusted. The revision formula is $s'_r(c_i) = (1 + \lambda \bullet \Delta T) s_r(c_i)$, in which λ is the domino effect zooming factor and could be adjusted based on practical experience.

5 Conclusions and Future work

In this paper, we introduced a novel RE approach, which, by incorporating user modeling techniques, generates better user participation experiences. To be more specific, the MDRE workflow management based on user models automates and facilitates MDRE activities in multi-user environment; the personalized requirements reuse based on context promotes the effectiveness and the efficiency of domain knowledge reuse by providing users appropriate knowledge in appropriate occasion.

Finally, we list out some future work of this research: refine the prototype CASE tools we developed so far to further demonstrate its use. Integrate existed requirements engineering tools or languages such as UML. Build the structure of domain ontology into domain model to support user-friendly query and provide stronger reasoning capability of the domain knowledge base.

References

[Ardissono 01]L. Ardissono, A. Goy, etc. "A Software Architecture for Dynamically Generated Adaptive Web Stores". Proc. 17th Joint Conf. on Artificial Intelligence. 2001.

[Armenise 93] P. Armenise, S. Bandinelli, C. Ghezzi, A. Morzenti. "A Survey and Assessment of Software Process Representation Formalisms". GOODSTEP TR NO.015, 1993.

[Brachman 85] R. Brachman, H. Levesque (Eds). "Readings in Knowledge Representation". Morgan Kaufmann. 1985.

[Broadvision 00] http://www.broadvision.com/

[Chung 00] L. Chung, B. Nixon, E. Yu, J. Mylopoulos. "Non-Funcitonal Requirements in Software Engineering". Kluwer Academic Publisher. 2000.

[Dardenne 93] A. Dardenne, A. Lamsweerde, S. Fickas. "Goal-Directed Requirements Acquisition". Science of Computer Programming. Vol. 20. 1993.

[Darimont 97] R. Darimont, J. Souquikres. "Reusing Operational Requirements: A Process-Oriented Approach". 3rd IEEE Intl. Symp. on Requirements Engineering. 1997.

[Fink 00] J. Fink, A. Kobsa. "A Review and Analysis of Commercial User Modeling Servers for Personalization on the WWW". User Modeling and User-Adapted Interaction. 10:209-249. 2000.

[Greenspan 86] S. Greenspan, A. Borgida, J. Mylopoulos. "A Requirements Modeling

Language and its Logic". On Knowledge-Based Management Systems. M. Brodie, J. Mylopoulos (Eds). Springer-Verlag. 1986.

[Herlocker 99] J. Herlocker, J. Konstan, A. Borchers, J. Riedl. "An Algorithmic Framework for Performing Collaborative Filtering". Proc. of Intl. Conf. on Research and Development in Information Retrieval. 1999.

[Li 00] M. Li. "A New Methodology for User-Driven Domain-Specific Application Software Development". Journal of Software (China). Nov. 2000.

[Maiden 98] N. Maiden. "CREWS-SAVRE: Scenarios for Acquiring and Validating Requirements". Automated Software Engineering. Vol. 5, No.4. 1998.

[Massonet 97] P. Massonet, A. Lamsweerde. "Analogical Reuse of Requirements Frameworks". 3rd IEEE Intl. Symp. on

[Nuseibeh 00] B. Nuseibeh. "Requirements Engineering: A Roadmap". ICSE Future of Software Engineering Track. 2000.

[Orwant 91] J. Orwant. "The Doppelganger User Modeling System". Proc. of the IJCAI Workshop W4: Agent Modeling for Intelligent Interaction. 1991.

[Perrault 78] R. Perrault, F. Allen, R. Cohen. "Speech acts as a basis for understanding dialogue coherence". Report 78-5, Department of Computer Science. University of Toronto. 1978.

[Raskutti 97] B. Ruskutti, A. Beitz, B. Ward. "A Feature-Based Approach to Recommending Selections based on Past Preferences". User Modeling and User-Adapted Interaction. Vol. 7. 1997.

[STANDISH 95] Standish Group. "CHAOS". STANDISH GROUP REPORT.95

[STANDISH 99] Standish Group. "CHAOS: A Recipe to Success". STANDISH GROUP 1999