A Recommender Agent to Support Knowledge Sharing in Virtual Enterprises

Renata S. S. Guizzardi[1], Pablo Gomes Ludermir[2], Diego Sona[1]
souza@itc.it, gomesp@cs.utwente.nl, sona@itc.it
[1]Fondazione Bruno Kessler, Via Sommarive, 18 - 38050 - Povo, Trento, Italy
[2]University of Twente (UT), Computer Science Department
P.O. Box 217, 7500 AE, Enschede, The Netherlands

Abstract. To remain competitive, virtual enterprises depend on effective Knowledge Management (KM). On the other hand, KM is deeply affected by the virtualization of modern organizations. As a result, KM systems need to be reshaped to adapt to these new conditions. This chapter presents KARe, a multi-agent recommender system that supports users sharing knowledge in a peer-to-peer environment. In this way, KARe reflects the intrinsically distributed nature of virtual enterprises. Supporting social interaction, the system allows users to share knowledge through questions and answers. This chapter focuses on KARe's recommendation algorithm, presenting its description and evaluation.

1. Introduction

The evolvement of information technology inaugurated new ways of structuring the organization and executing work. Many organizations became partially or completely virtual, processes gained a more dynamic and distributed nature, and static and hierarchical structures shifted to increasingly adaptable and flexible ones. In this realm, **virtual enterprises** can be defined as *"distributed organizations and teams of people that meet and work together online. Group members rely on support systems to help gather, retrieve, and share relevant knowledge."* (O'Leary, 1997). From this definition one immediately concludes that it is paramount for these organizations to invest money and effort in finding effective solutions for collecting and sharing knowledge.

Focusing on these matters is the Knowledge Management (KM) research area, which deals with the creation, integration and use of knowledge, aiming at improving the performance of individuals and organizations. Advances in this field are mainly motivated by the assumption that organizations should focus on *knowledge assets* (generally maintained by the members of an organization) to remain competitive in the information society's age (Nonaka and Takeuchi, 1995). However, KM practices and systems are also affected by the virtualization of modern organizations. Merali and Davies (2001), for instance, mention three trends that have considerably added to the complexity of KM problems:

- the move towards flexible work practices, resulting in the geographical dispersion of people who would be normally co-located;
- the increasing importance of cross-functional and inter-organizational collaborative work practices;
- the need to provide quick and innovative organizational responses to changes in the environment.

KM systems and practices should consequently be reshaped to adapt to these new conditions. Nevertheless, the current landscape concerning KM systems shows that most initiatives still rely on central repositories and portals, which assume standardized vocabularies, languages, and classification schemes (Liao, 2002). Consequently, employees' lack of trust and motivation often lead to dissatisfaction (Pumajera et al 2003; Merali and Davies, 2001). In other words, workers resist on sharing knowledge, since they do not know who is going to access it and what is going to be done with it.

Workers dissatisfaction many times leads the KM system to be abandoned, while people continue relying on their natural ways of finding knowledge, such as asking for the help of colleagues that are part of their circle of trust. The work described in this chapter aims at improving these natural processes by imitating in a virtual environment, the social processes that are involved in knowledge sharing. Instead of taking a centralized view, we rely on the distributed KM paradigm (Bonifacio et al, 2002), providing autonomous and locally managed knowledge sources organized in a peer-to-peer community. Peer-to-peer technology supports the horizontal relationship between people, seeing them as both consumers and providers of knowledge (Tiwana, 2003). Each peer controls his own personal knowledge artifacts and exchange knowledge with other peers based on, for example, their common interests, roles, expertise, and trust.

In this work, we present KARe (Knowledgeable Agent for Recommendations), a socially aware recommender system that recommends artifacts to organizational members based on their natural language questions. KARe (Guizzardi, 2006) is designed and implemented as a multi-agent system, where agents cooperate to organize and search knowledge artifacts on behalf of their users, interacting in a peer-to-peer network. In order to look for the answers to the users' questions, we propose an algorithm based on information retrieval techniques that associate semantic information to the queries and to the artifacts being searched. This semantic information allows us to decrease the computational complexity of the algorithm, at the same time as providing less noisy results.

The remaining of this chapter is organized as follows: section 2 presents some background information on relevant research areas composing the scope of this work and how they relate to the domain of virtual enterprises; section 3 describes the proposed system; section 4 presents the description and evaluation of the recommendation algorithm; section 5 focuses on the developed prototypes of the KARe system; section 6 discusses some related work; and section 7 finally concludes this chapter.

2. The Scope of this Work

This work is based on developments of four research areas, namely *intelligent agents*, *peer-to-peer communities*, *recommender systems* and *taxonomies*. This section summarizes each area, describing how they are applied in KARe and presenting their connections with the domain of focus of the present book.

2.1. Intelligent Agents

Intelligent agents are frequently proposed as appropriate entities to enable the analysis and design of complex systems, made up of several entities that behave autonomously and interact with each other in order to achieve a common objective (i.e. the system's overall functionality) (Jennings et al., 1998). These characteristics are common to environments of virtual enterprises, which are generally composed of several autonomous agents working on behalf of one or more organizations that must cooperate on the pursuit of common goals (Cole and Gamble, 1997). The social and cognitive characteristics of agents are their main strength, turning them into promising constructs to emulate human interaction and rational behavior. Using them as modeling metaphor enables the analysis of the current social structures embedded in the organization, hence leading to more appropriate system proposals.

Especially in KM scenarios, agents may be found as appropriate building blocks for system development, for providing knowledge both *reactively* and in a *proactive* fashion (Merali and Davies, 2001). In other words, by taking into account the users' preferences and by monitoring their tasks, agents are both able to react to incoming knowledge requests, and to anticipate specific needs of the users, consequently delivering knowledge appropriately.

KARe explores the use of intelligent agents both as a) a development metaphor, being modeled as a set of artificial agents interacting with human and organizational agents that are part of the enterprise's overall environment and b) technological building blocks, being implemented on the basis of the Java Agent Development Framework (JADE)[1], a middleware containing the basic infrastructure to support agents' to locate and communicate with each other.

2.2. Peer-to-peer Communities

The distributed nature of peer-to-peer networks reflect the naturally distributed character of knowledge and expertise within virtual enterprises (Tiwana, 2003). In virtual enterprises, knowledge and expertise exist sparsely in the members' understanding of each other's knowledge, and in the behavioral and cognitive similarities among individual users. In this respect, KARe aims at uncovering and making salient some of these hidden characteristics so that users might become more aware of the existing organizational knowledge.

Let us consider a virtual enterprise comprised of two or more organizations. The nodes of peer-to-peer network may change or even be completely removed, much in the way partnerships between different organizations may be created and dissolved, having in mind particular objectives and tasks. During the period of partnership, the members of the involved organizations have the opportunity to share knowledge and experience. Consequently, useful pieces of knowledge exchanged before they depart are likely to stay in both organizations, thus peer-to-peer networks function as a self-organizing environment motivating cross-fertilization, so valued in KM settings.

Central to this work is the recognition that social interaction is the driving force behind the creation of knowledge (Nonaka and Takeuchi, 1995). The members of KARe's peer-to-peer community exchange the knowledge artifacts maintained in their personal collection. In other words, KARe aims at imitating the

[1] http://jade.tilab.com/

social processes commonly applied when one has a particular problem to solve during one's daily work. Instead of consulting manuals and documentations, the worker is motivated to get involved in a dialog with workmates, which may lead him/her to grasp more than procedures, the values and tacit strategies adopted in the organization.

2.3. Recommender Systems

Managing information overload has been frequently mentioned as one of the challenges in environments surrounding virtual projects and enterprises (Katzy et al., 2000) (Cole and Gamble, 1997) (Merali and Davies, 2001). Recommender systems support users in selecting items of their interest or need from a big set of items, helping users to overcome the overwhelming feeling when facing a vast information source, such as the web, an organizational repository or the like.

Recommendations may be differentiated by (Montaner et al., 2003): the items they recommend (e.g. movies, web pages, etc.); the nature of the user models they use to guide the recommendations (e.g. history of items accessed by the user, topics indicating user interest, etc.); the recommendation techniques (mainly, how the user model is represented, what kinds of relevance mechanisms are used to update the user model, and which algorithm is used to generate recommendations); and the recommendation trigger, i.e. whether the recommendation is started by the user or by the proactive behavior of the system. Section 3 describes how KARe deals with each of these aspects.

2.4. Taxonomies

Taxonomies and *ontologies* are two types of conceptual models which can be applied to describe a domain of discourse, modeling it as a set of concepts and relations. Although also applied in the past, the Semantic Web has recently increased the interest on using such conceptual models to explicitate the semantics regarding knowledge artifacts (Davies et al., 2003). A taxonomy can be seen as a simplified ontology. On one hand, both conceptual models are identical regarding the use of concepts. However, while taxonomies focuses on hierarchical relations between concepts, ontologies may exhibit more complex relation types, such as part-of and associations. The choice of using taxonomies instead of ontologies in KARe is motivated by the DKM philosophy (Bonifacio et al., 2002), which defends that rather than sharing a unique conceptualization, each organizational member has his own view of his work domain. Thus, each user builds his own conceptual model. As ontologies are generally considered complex and time consuming to be built (Davies et al., 2003), we consider taxonomies as a more realistic model for the common user to create. In a sense, many workers already create directory classifications of this kind, both for physical or digital file systems.

3. KARe: Knowledgeable Agent for Recommendations

KARe is a multi-agent system that recommends artifacts to meet the user needs based on questioning and answering. When we have a real problem at work, we often rely on asking a question to a colleague with whom we share the office, or to someone who is considered an expert in a subject related to our problem. If this is easy in an environment where workmates are collocated, it becomes more challenging when collaboration is virtual. Thus, KARe offers a solution to virtual enterprises, reflecting their intrinsically distributed nature while supporting social interaction among its members.

Asking and answering to questions is an interactive process. The questioner finds a suitable colleague and poses his doubt. Usually, this choice is based on the questioner's assumption that his colleague knows about the targeted subject, besides feelings of trust and comfort towards the responder. The responder, on his turn, is likely to help the questioner, provided that the trust between them is mutual. He will then use his own language and knowledge to provide the answer to the questioner. Besides solving the problem at hand, having the answer gives the questioner the ability to share this new knowledge with other colleagues.

KARe facilitates the questioning and answering process by using a peer-to-peer infrastructure. Each user (a peer) is able to organize his knowledge assets (typically, working documents) according to his own domain conceptualization, using a *taxonomy*. After defining meaningful concepts and their inter-relationships the user distributes the artifacts according to the "matching" concepts in the hierarchy. Figure 1 shows the organization of the personal knowledge assets of three users, connected by a peer-to-peer network.

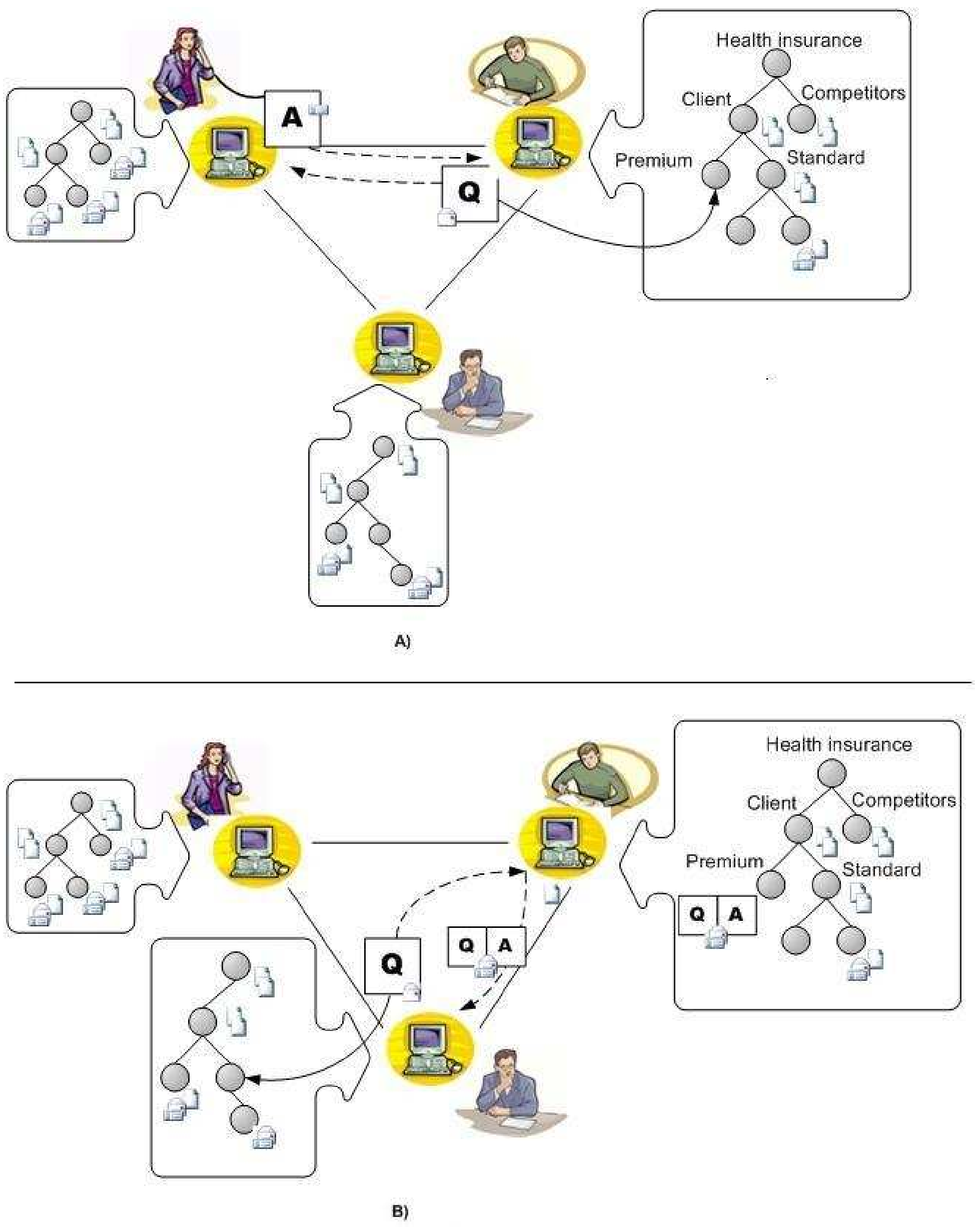


Fig. 1. Using KARe for asking and answering to questions

KARe allows the user to pose natural language questions, searching in other peers' collection for answers among their stored artifacts. The answer can be found among *documents* or *messages* sent by other peers responding previous similar questions. In case no response is found, the system indicates a suitable peer (based on peer's user models) to provide the answer to that specific question. Having received a suitable answer from the indicated peer, the questioner now has this answer classified in his own taxonomy and stored in his system, so that he can be consulted by others regarding the same subject. These processes are illustrated in Fig. 1.

Fig. 1 A) shows how KARe behaves when a user submits a question, manually contextualizing it by assigning it to a concept in his own taxonomy (in this case, the "Premium" concept). The system submits the question to a peer whose user model seems to describe a suitable responder. The user answers to the question submitting it through the system to the questioner. Note that the contextualization of the question may help the responder to understand more about the questioner's doubt. For instance, suppose that this is an insurance company and that Mike's question is "What measures should we take when a client is late with his payment for the acquired services?" Some information is not expressed in Mike's question, for instance: what kind of service is he talking about? However, this information is explicitated by the contextualization of the question, since it is classified under "Health insurance-Clients-Premium" in Mike's taxonomy. Besides clarifying the type of service ("health insurance"), this contextualization also indicates the type of client Mike is referring to (in this case, "premium"), which may have some impact in the responder's answer.

In Fig. 1 B), Mike has already received the answer to his question, which was then classified under the concept he previously indicated. In this way, Mike now stores the answer for his own future reference and for sharing it with peers in need. The figure shows the case in which Joey requests similar information, by posing a question similar to Mike's. In this case, Mike does not need to personally answer to the question, as the system has already found it in his computer, subsequently sending it to Joey.

By storing the same question/answer pair in different peers, we increase the possibility that this knowledge will remain in the organization even when some members are not available anymore. Considering that these peers will be involved in continuous interactions, the knowledge considered "useful" to the community (i.e. information and documents they need for their daily work) is likely to remain in the community, even if the members that originally owned them leave.

If the questioner is not satisfied with the answer automatically retrieved by KARe, the system provides him with a list of possible responders. Responders are chosen based on their characteristics: expertise, reliability, trust, role, collaborative level, and availability. These characteristics are maintained in user models, explained in detail in (Guizzardi, 2006).

4. Recommendation Algorithm description and evaluation

The core of the KARe system consists in mediating the question and answering process. An important part of handling this process comprises the automatic recommendation of existing answers to users' questions. The main focus of this chapter regards the description and assessment of this algorithm, which is based on Information Retrieval (IR) techniques (Baeza-Yates and Ribeiro-Neto, 1999).

In KARe, the knowledge items stored by a peer are not viewed as a flat collection of documents. Instead, the set of documents are structured by a taxonomy which classifies each of these items under a concept of the tree (both leaves and internal nodes). The choice of using taxonomies to classify knowledge artifacts provide the system peers with a contextualized view of knowledge artifacts (as seen in section 3). However, this is just part of the reason behind this choice. Another strong claim we make is that such information may be helpful in aiding our recommendation agents to automatically find knowledge on behalf of the system users.

In a traditional IR system, items are all considered to be part of a unique collection, which should be completely searched when a retrieval request is issued by the user. In KARe, however, taxonomies are used to classify documents. Consequently, the system is able to search for the answer only considering particular nodes of the taxonomy where the answer is most probably located. Besides diminishing computational complexity, this approach allows the system to profit from user knowledge, previously encoded in personal taxonomies, to retrieve related knowledge.

The search process is triggered when a user asks a question, which he/she first assigns to a concept (node) in his/her taxonomy. Hence, a question (or knowledge request) is logically represented not only by the keywords it contains but also by the keywords representing the concept which classifies it. For finding an appropriate answer, KARe must first match two distinguishing taxonomies, analogously to (Avesani et al., 2005; Bouquet et al., 2003). More precisely, when receiving a knowledge request from the questioner, the system must find in the responders' taxonomies which concepts are more likely to contain artifacts that satisfy this request, subsequently retrieving it.

Although our assumption about the gains of applying taxonomies seems reasonable, it can only be proven by testing our algorithm using real datasets. Thus, besides describing the applied techniques, this chapter shall also present empirical data to validate them. The assessment was based on two existing datasets which simulate well the problem at hand, and the evaluation was based on measures of recall and precision of the proposed algorithm in comparison with a standard approach. Notice that a KARe user can also manage the artifacts with a non-structured approach, i.e., having only a flat collection of items. This only influences the quality of results but not the system functioning.

4.1. Information Retrieval Modeling

The approach to retrieve the information is a framework that models the documents and queries into their logical forms, and a ranking function that orders the document set according to queries. Thus, the IR models may be represented as a quadruple $\{D, Q, F, R(q_i, d_j)\}$ where (Baeza-Yates and Ribeiro-Neto, 1999):

- $D = \{d_1, \ldots, d_n\}$ is the set of documents;
- Q is the representation of the user query;
- F is the framework to model documents, queries, and their relationships into a logical form;
- $R(q, d_i)$ is a function to rank documents $d_i \in D$ according to a particular query $q \in Q$.

There are different ways to approach the modeling “framework” and the “ranking function”. This results in different IR models, such as: the boolean model, the probabilistic model and the vector space model. In this work, we use the vector space model, thus describing it in detail.

A collection of text documents is generally represented by a vocabulary, i.e. a small set of index terms determined as representative of such collection. In the vector space model, documents and queries are treated as real algebraic vectors where the dimension of the vectors is determined by the size of a vocabulary. Therefore, once the vocabulary is determined (i.e. the text is pre-processed, determining the index-terms), all documents are represented by vectors. Each dimension of the vectors is calculated based on the frequency of each index term in each document itself. Having this vectorial representation, it is possible to calculate the similarity between couples of documents or between a document and a query.

In Figure 2, the depicted vectors are the abstraction of a query q and a document d_j from a set of documents D, and the angle θ indicates how close these vectors are. There are several measures for vectors similarity, and one of the most used in text documents retrieval, also adopted here, is the cosine of the angle θ formed by the two vectors.

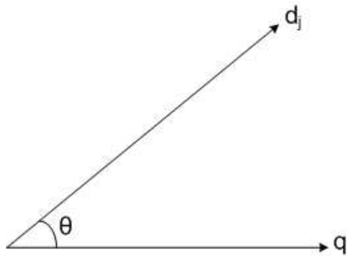


Fig. 2. The cosine function is used to compute the similarity between a query *q* and a document d_j

Each dimension value of the vectors describing the documents is computed using the frequency of the corresponding index term in the document itself. There are several ways to compute these dimensions, and this process is called *index term weighting*. In our algorithm, we adopted the *TF/IDF* representation, which is one of the most used. This method computes the weights in two steps. First, given a document $d_i \in D$, the *term frequency* (TF) f_k is computed for each index term t_k in the vocabulary. Second, the resulting term frequency is weighted multiplying it by the *inverse document frequency* (IDF), which measures the fraction of documents that contain the corresponding index term as follows:

$$d_{i,k} = f_k * \log \frac{N}{n_k} \quad (1)$$

where N is the total amount of documents in the collection, and n_k is the number of documents containing the index term t_k.

The aim of using both *TF* and *IDF* in the weight calculation is on one hand, to increase the weight if a term is very popular in a document and on the other hand, to penalize the weight (i.e. to decrease its value) if the term is present amongst many documents. In this way, the terms of a document which are both representative and discriminant have stronger weight when computing the similarity to a query.

4.2. Algorithm Description

As previously outlined, to find relevant documents in a collection, the standard vector space approach computes the similarity between a query of the user and all the documents in the given collection. Then, the most similar documents are selected as the winners. This approach, however, disregards any knowledge that users may have about the structure of the concepts related to the artifacts being searched. This can lead to noisy or not relevant search results. For instance, trying to search for the word “agents” in a standard search engine (e.g. Google, Yahoo, Looksmart, etc.) results in documents about several different kinds of agents (e.g. chemical agents, software agents, real state agents, travel agents, etc.).

In KARe, the user is allowed to classify his documents according to a personal taxonomy, which represents a personal view of the domain of interest. In this way, similar documents are grouped by the user under the same concept in the taxonomy tree. In addition to that, before submitting the question, the user contextualizes the query, assigning it to a specific concept in the taxonomy. By doing this, the user gives to the system an extra hint on the query's content. Aiming at reducing the noise of the search (not relevant or wrong documents), our algorithm exploits the taxonomic information supplied by the user to determine the region of the search space where the required information is more likely to be found. Besides providing more accurate results, this approach also reduces the computational complexity of the algorithm in comparison with the standard approach. This happens due to the fact that the standard approach needs to search the whole documents collection (complete search space) for an answer. Conversely, following our algorithm, KARe only searchers particular regions of the search space.

To illustrate our approach, we go back to the example earlier presented in section 3. Consider two users Mike and Joey, whose taxonomies are depicted in Figure 3. The taxonomies classify the user's personal documents and also serve to contextualize the user's question. Suppose now that Joey submits the following question as query: “How should we deal with clients' late payment?”, contextualizing it in the ‘Policies’ concept of his taxonomy. Referring back to the previous example, we know that Mike had a similar doubt in the past (i.e. he previously asked “What measures should we take when a client is late with his payment for the acquired services?”) whose answer is now classified under the concept ‘Premium’. But how can KARe know about that?

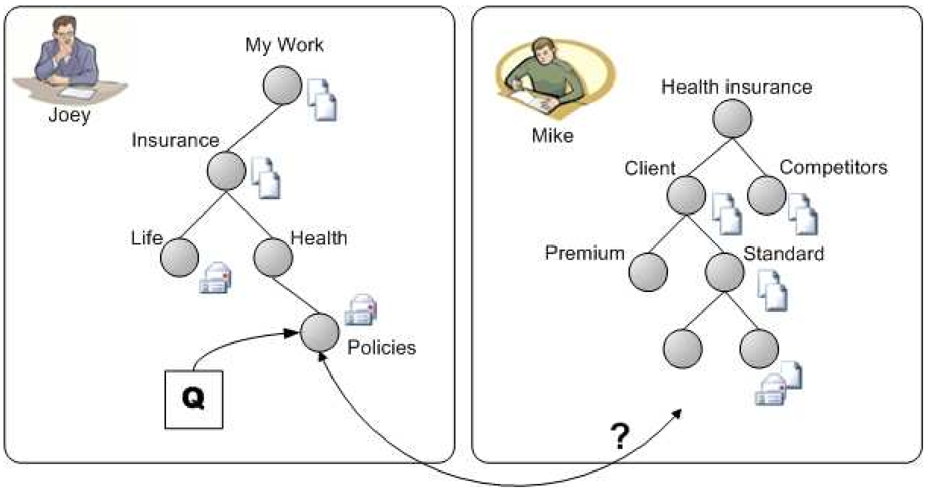
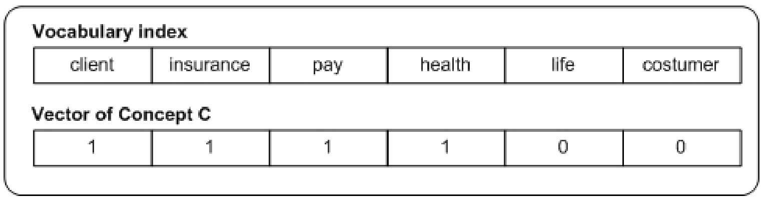


Fig. 3. Taxonomies of Mike and Joey contextualizing documents and questions

Our algorithm should be able to identify which concept in Mike's taxonomy is more similar to the ‘Policies’ concept, where Joey's question is contextualized. Then, the answer can be searched within this concept. Essentially, each of the concepts of the user taxonomy has a vectorial representation, which is an instance on the reference vocabulary containing the weights of the index terms that appear in the documents classified under the concept. Figure 4 illustrates a short vocabulary index and a vector representing a given concept C, which contains the index terms ‘client’, ‘insurance’ ‘pay’, and ‘health’, but does not contain the terms ‘life’ and ‘customer’. In this figure, the used weights are boolean (i.e. ‘1’ indicates the presence of an index term, while ‘0’ indicates its absence). Conversely, as previously outlined, in our approach the weights are given by the *TF/IDF* representation. For finding the most similar concept to a given concept C, the algorithm calculates the similarity between the vector representing C and the vector of each of the concepts in the responder taxonomy.



Vocabulary Index

client	insurance	pay	health	life	costumer

Vector of Concept C

1	1	1	1	0	0

Fig. 4. A short vocabulary index and a vector representation for a concept C in the user taxonomy.

Given a concept C_i, the corresponding reference vector c_i is calculated with basis on the vectors representing the documents classified under that concept. Besides the documents’ keywords, the concept label l_i is also considered in the vector calculation. In fact, not only the label of the concept itself, but in addition also the labels of the ancestors of the given concept are taken into account, as has been earlier proposed in (Adami et al., 2003). More precisely, this is achieved by including the labels of all concepts of the taxonomy in the collection's vocabulary. Consequently, the label of the concept along with the label of the concept's ancestors are considered in the concept's vector calculation. The determination of the concept reference vectors follows the Equation 2.

$$c_{i,j} = \sum_{a \in A_i} l_{a,j} + \frac{\sum_{k \in D_i} d_{k,j}}{|D_i|} \quad (2)$$

where $A_i = \{j \mid C_j \text{ is ancestor of } C_i\}$ indicates the ancestors of concept C_i, $l_{a,j}$ stands for the index term t_j in the label for concept C_a, $D_i = \{k \mid d_k \text{ is classified under concept } C_i\}$ is the set of indexes of documents classified under concept C_i, and $|A_i|$ and $|D_i|$ indicate the dimension of the two sets respectively.

Here, $c_{i,j}$ stands for the weight of the term t_j on the concept C_i. The above equation uses the *TF/IDF* representation already described in section 4.1, both for the document representation and for the concept labels $l_{a,j}$. Equation 2 is basically an average formula, which calculates concept vector c_i based on an average of the weight of the keywords pertaining to all documents classified under concept C_i. This term is then regularized according to the prior knowledge encoded into the concept labels.

We call the process of finding the best matching concept in the responder's taxonomy *query scope reduction*. This is the main novelty of our approach. In summary, the query scope reduction can be seen as a reduction in the *search space* before we retrieve information from it, based on the fact that the required information is more likely to be found in a specific region of this space. Adding this process prior to the execution of the query, we aim to increase the quality of our search, resulting in a less noisy result set, thus recommending mostly pertinent documents to the users. In addition to that, it considerably reduces the computational complexity of the algorithm since it diminishes the set of documents to be searched.

It is important to note that each user taxonomy uses a different vocabulary, i.e. the vectors of the concepts in each taxonomy are created based on different sets of index terms. Consequently, the first step on the query scope reduction is to project the concept vector coming from the questioner in the new space of the responder. This is made by calculating the intersection between the index vector of the questioner and the index vector of the responder. In this way, the concept vector coming from the questioner may be projected into the vocabulary of the responder. This projection is specifically targeted at the problem of coping with different semantic representations of a domain.

After the query scope reduction, the answer to the user's question is searched within the documents classified under the best matching concept(s). For that, all keywords of the user's query are taken into account to select the artifacts of the given concept. In addition to the query's terms, the labels of the concept classifying the query and its ancestors are attached to the query (as extra terms). In this way the query is embedded with enriched contextualized information. The documents are then ranked in a descending order according to the similarity with the query, and the result set is finally sent to the questioner. Our recommendation algorithm is summarized in the pseudo-code shown in Listing 1.

4.3. Case Study

The theory behind our algorithm seems to be consistent, however in order to prove that it actually brings any gains in the efficiency and accuracy of our search, it is advisable to assess it through a case study using real data. The ideal situation would be to experiment our algorithm against two taxonomies classifying real questions and answers. However, as such dataset is not available at the moment, we simulated this dataset using two taxonomies that classify scientific papers in a case study. The question is simulated by the title of a paper and the answer is given by the paper's body. This seems reasonable because a question is usually short, providing us with a few keywords for the search. The answer, on the other hand, tends to be a longer piece of text. For performing the case study, we used two existing taxonomies: the questioner's taxonomy was created by a PhD student to collect papers of her interest, while the one of the responder is taken from the ACM Computing Classification System[2]. Table 1 presents some statistics regarding these two taxonomies.

[2] http://www.acm.org/class/

Listing 1. An excerpt of KARe's recommendation algorithm

```
procedure answer(concVectA, peerQuest, questioner)

//step 1: search the best matching concept for the scope reduction
projConceptVectorA := intersect(concVectA, indexB)

for each (concept on the user B context) {
      s := similarity(currentConceptVectorB, projConceptVectorA)
      if (s > maxSimilarity) {
            bestConcept := currentConceptB
            maxSimilarity := s
      }
}

//step 2: search among the documents in the bestConcept
queryVector := createQueryVector(peerQuest, indexB)
for each (document in bestConcept)
      documentList.add(document, similarity(queryVector,documentVector))
      documentList.sortBySimilarity()

//step 3: send the answer back to the questioner
sendAnswer(documentList, questioner)
```

The case study may be divided into two main phases: preparation of the taxonomies, and execution of the assessment. In the first phase, the following activities were performed:

- a set of papers to be used as queries were selected. These papers should be classified by both taxonomies so that we know which is the contextualizing concept in the questioner's taxonomy and the concept the algorithm should find in the responder's taxonomy;
- all the selected papers were then subtracted from the questioner's taxonomy to avoid bias (i.e. the keywords of the selected papers should not be used to compute the concept vectors in the questioner's taxonomy);
- the titles of all the selected papers were subtracted from the papers classified by both taxonomies to avoid bias (i.e. the title keywords should not be used to compute the concept and document vectors in both taxonomies), as they were to be used as queries;
- all papers and titles were preprocessed removing the stop-words and stemming the resulting terms to common roots, then a vocabulary was created for each taxonomy taking the frequent and discriminative terms, and finally, all papers were encoded using the vocabulary of the corresponding taxonomies.

Table 1. Some statistics regarding the taxonomies

	Questioner Taxonomy	**Responder Taxonomy**
Number of Documents	250	315
Number of Concepts	28	15
Average Documents/Concepts	9	21

The second part of the case study (i.e., the process carried out to execute the assessment) is illustrated in Figure 5. The first step is to manually contextualize the query, by assigning it to a concept in the questioner's taxonomy. In fact, the information regarding which concept should contextualize the query was already known, since the queries were extracted from papers classified under both taxonomies. Next, the query (i.e. a paper title) is preprocessed removing stop-words and stemming, and is then submitted to the responder's taxonomy along with the contextualizing concept's vector. The algorithm then searches for a “similar” concept in the responder's taxonomy (query scope reduction). After the targeted concept is found, the answer to the query is retrieved from the documents within this concept. Finally, we analyze the result set, verifying whether the algorithm is able: a) to find in the responder's taxonomy the *concept* that correctly classifies the paper whose title is the query; and b) to retrieve from the responder's taxonomy, the *specific paper* corresponding to the title used as query.

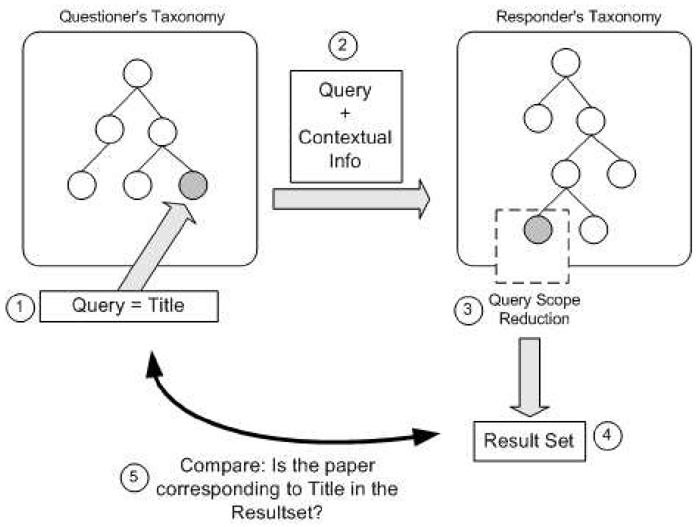


Fig. 5. A schema of the process used to perform the assessment

The case study is, thus, defined as the analysis of the proposed algorithm to evaluate its information retrieval performance measures in comparison with a standard vector model approach. We compared the results of our algorithm with the standard approach based on the vector model (i.e. without the query scope reduction step). Concerning our approach, we considered two options: i) to reduce the search space to the best matching concept only; and ii) to reduce the search space to a small subset of concepts that best matched the query (two and three concepts). We evaluated these approaches in terms of *recall* (i.e. the fraction of relevant documents retrieved) and *precision* (i.e. the fraction of retrieved documents that are relevant). Then, the harmonic mean *F1* of recall and precision was calculated and used to compare our results:

$$F1 = \frac{2 * precision * recall}{precision + recall} \quad (3)$$

Our hypothesis is that our algorithm should, on average, produce better precision and recall results and take less effort than the standard vector model approach. We identified as parameter factors for the case study the number of concepts selected in the responder taxonomy and the question itself. The dependent variable analyzed are the *F1 measure* and the *computational effort* (i.e. number of comparisons between keywords).

4.4. Discussion

We performed 75 queries over our taxonomies, and the results are shown in Table 2. The first column of the table shows the results of the standard approach. The second, third and fourth columns show the results of our approach when returning documents from one, two and three concepts respectively.

Table 2. Results of the case study

	Standard Approach	1 concept	2 concepts	3 concepts
Number of Queries	75			
Documents Found	69	25	37	43
F1-doc	0.920	0.333	0.493	0.573
F1-concept	0.175	0.243	0.238	0.206
Num. of Comparisons	158K	21K	33K	43K

As previously illustrated in Fig. 5, our assessment considers two important results: the ability to retrieve the correct document and the ability to map the query to the correct concept. Hence, *F1* was correspondingly calculated based on two quantities: 1) the number of times the algorithm finds the specific document whose title is being searched (F1-doc); and 2) the number of retrieved documents related to the one being searched, which is the number of documents that are classified under the concept being searched (F1-concept). The second measure is determined counting for each query the number of relevant papers among the best ranked 20 papers.

Observing the number of documents found and the corresponding F1-doc measure it is clear that the standard approach (i.e., search the documents crawling all the papers) is much more effective than our approach (reduced search space). This is particularly true when only the best matching concept is used to limit the search space. Conversely, when searching for related papers, it is evident that our approach gives better results (see the F1-concept measure). In addition to that, our approach considerably reduces the number of comparisons needed to reach a result (see that for 1 concept, it is one order of magnitude faster that the standard approach).

Our approach with varying number of best matching concepts to reduce the search space is worth further discussion. Actually, it is easy to see that, in general, when we increase the number or searched concepts, we increase the chance of finding the specific paper we look for, but we also end up having a result set with more noise. Moreover the number of required comparisons is also increased. Hence there is a trade-off between different elements that must be taken into account when implementing our approach for a specific application.

Our approach has the disadvantage of not finding the right concept many times. This becomes apparent by the comparison of the graphics exhibited by Figures 6. These graphics exhibit comparisons between the standard approach and each variation of our algorithm. Notice, however, that in general there is a correlation between the standard approach and our approach. Actually, we observe that when our approaches have low recall, the standard approach tends to have low recall as well. Conversely, a high recall in the standard approach also coincides with a high recall in our models.

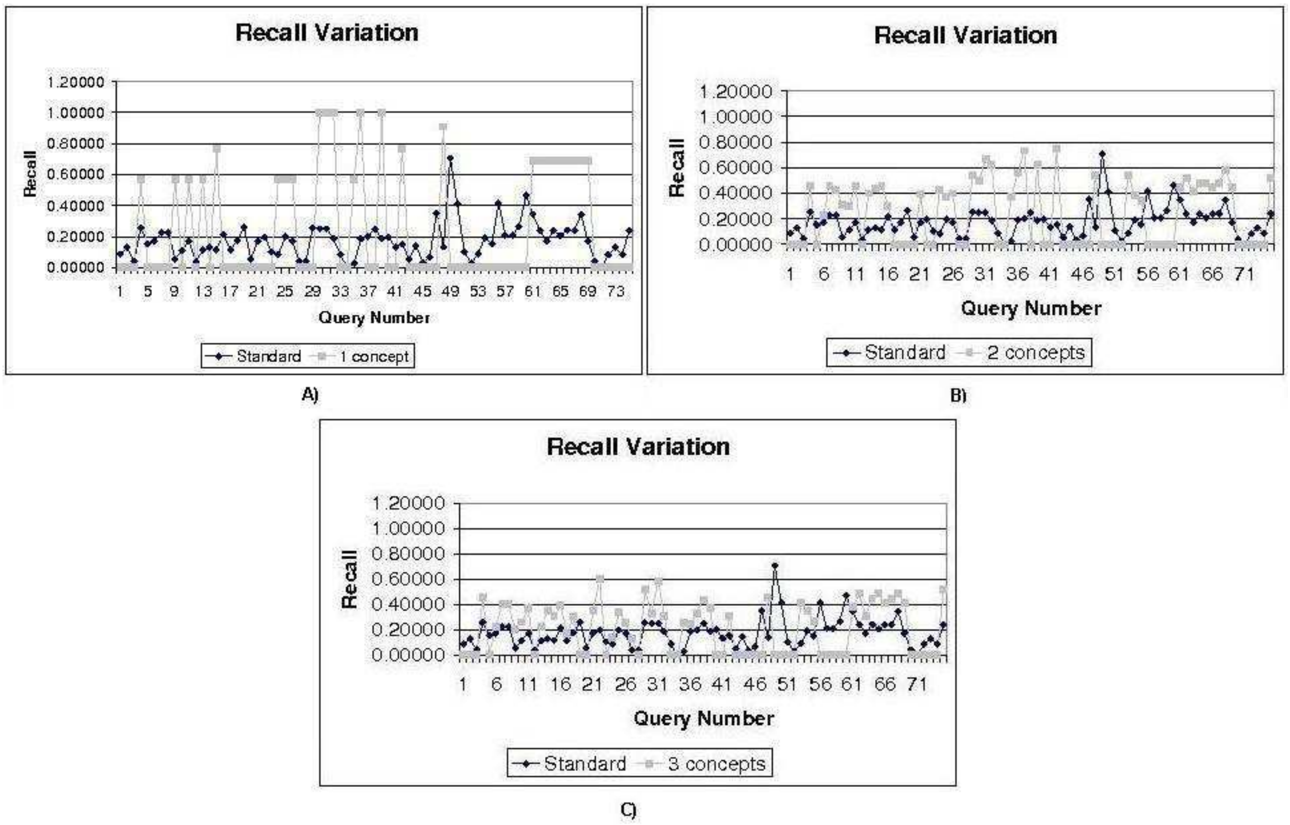


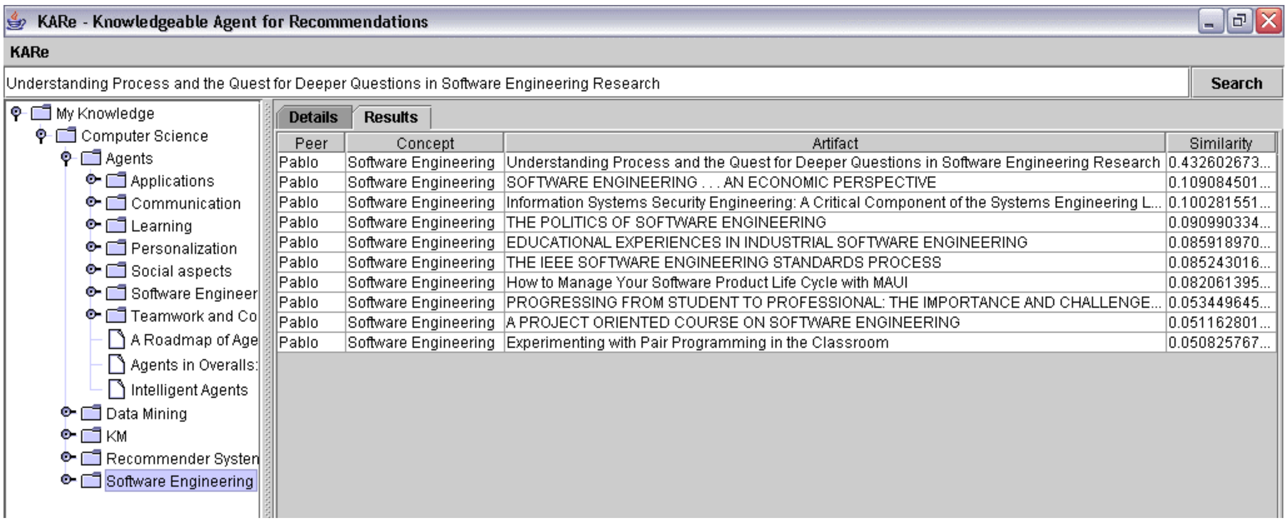
Fig. 6. The graphics depict the recall measure for all 75 queries made with the standard approach and our proposed approach using respectively 1 concept (graphic A), 2 concepts (graphic B), and 3 concepts (graphic C).

In summary, the results show that the best solution requires the right balance between the probability of finding very specific information and the ability of retrieving related information. It is also important to note that increasing the probability of finding specific information also increases the number of comparisons the algorithm should perform, thus turning the algorithm more computational complex. The results also appoint in which direction our work should proceed, i.e., to enhance the ability of the algorithm of finding the right concept. Hence, our future solution will combine both finding specific and related information at once, while also keeping the computational complexity very low.

5. Prototypes

Two prototypes of the KARe system were implemented: a desktop computer version, and a prototype for access in a handheld device.

The main purpose of the **desktop system** is to allow organizational members to exchange knowledge while organizing their personal knowledge items locally (Ludermir et al 2005). Figure 7 shows a screenshot of the desktop prototype. On the left part of the window, the figure depicts a user taxonomy, showing in a tree of concepts, how the user structured his/her knowledge. On the top, there is a text box where the user enters his question, followed by a “Search” button. Having inserted the question, the user may press this button to trigger the searching mechanism. The results of the search are shown in the right side of the screen, classified by peer (the peer from which the artifact was retrieved), and ordered by the similarity of the artifact regarding the question submitted by the user.



Understanding Process and the Quest for Deeper Questions in Software Engineering Research

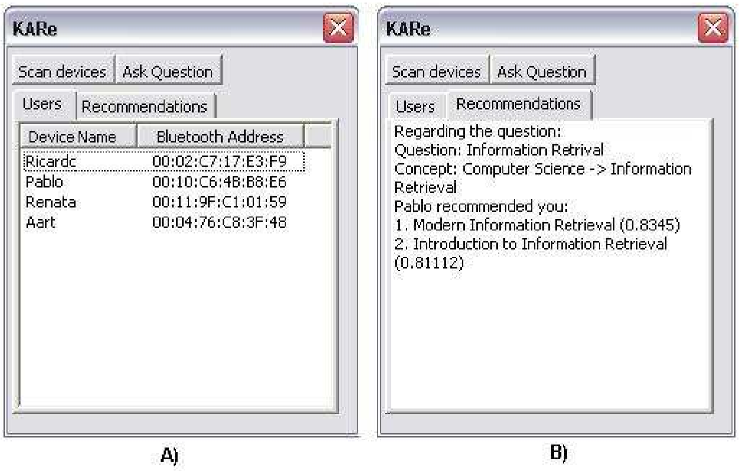
- My Knowledge
  - Computer Science
    - Agents
      - Applications
      - Communication
      - Learning
      - Personalization
      - Social aspects
      - Software Engineer
      - Teamwork and Co
      - A Roadmap of Age
      - Agents in Overalls:
      - Intelligent Agents
    - Data Mining
    - KM
    - Recommender Systen
    - Software Engineering

Peer	Concept	Artifact	Similarity
Pablo	Software Engineering	Understanding Process and the Quest for Deeper Questions in Software Engineering Research	0.432602673...
Pablo	Software Engineering	SOFTWARE ENGINEERING . . . AN ECONOMIC PERSPECTIVE	0.109084501...
Pablo	Software Engineering	Information Systems Security Engineering: A Critical Component of the Systems Engineering L...	0.100281551...
Pablo	Software Engineering	THE POLITICS OF SOFTWARE ENGINEERING	0.090990334...
Pablo	Software Engineering	EDUCATIONAL EXPERIENCES IN INDUSTRIAL SOFTWARE ENGINEERING	0.085918970...
Pablo	Software Engineering	THE IEEE SOFTWARE ENGINEERING STANDARDS PROCESS	0.085243016...
Pablo	Software Engineering	How to Manage Your Software Product Life Cycle with MAUI	0.082061395...
Pablo	Software Engineering	PROGRESSING FROM STUDENT TO PROFESSIONAL: THE IMPORTANCE AND CHALLENGE...	0.053449645...
Pablo	Software Engineering	A PROJECT ORIENTED COURSE ON SOFTWARE ENGINEERING	0.051162801...
Pablo	Software Engineering	Experimenting with Pair Programming in the Classroom	0.050825767...

Fig. 7. A screenshot of the desktop prototype

The KARe **handheld prototype** (Ludermir 2005) is based on the assumption that suitable responders to a specific question can be selected based on their geographical proximity to the questioner. This assumption comes from the realization that people usually share spaces with individuals with whom they share interests, e.g. workmates within an organization, researchers in a conference, and classmates in an educational institution.

By changing user's location, the recommendation is likely to change as well. In contrast, if you try to get a recommendation from the desktop system, the result is the same until the document index is updated. This happens because the KARe's desktop system searches for knowledge artifacts by broadcasting the knowledge request to all peers connected to the network. Thus, such design is slightly modified to accommodate the new search mode. After receiving a request from the user, a search is triggered when the system senses the presence of other peers in the vicinity. Figure 8 A) shows the resulting screen in the handheld after identifying the devices that run KARe. The request is then submitted solely to these peers. In this way, the handheld prototype also avoids the problems of scaling the system to a great number of peers, which still remains to be targeted in the desktop version. Finally, KARe notifies the user in case new recommendations were provided by the contacted peer, as seen in Fig. 8 B).



Device Name	Bluetooth Address
Ricardc	00:02:C7:17:E3:F9
Pablo	00:10:C6:4B:B8:E6
Renata	00:11:9F:C1:01:59
Aart	00:04:76:C8:3F:48

Regarding the question:
Question: Information Retrival
Concept: Computer Science -> Information Retrieval
Pablo recommended you:
1. Modern Information Retrieval (0.8345)
2. Introduction to Information Retrieval (0.81112)

Fig. 8. Screenshots of the handheld prototype

The desktop prototype was developed as two integrated components: Recommender Agents component and Information Retrieval component. The Recommendation Agents component was implemented using the JADE framework. JADE works as a middleware for the agents communication. To enable their communication, an ontology was developed using the Protégé Ontology Editor[3], and implemented in Java classes using the Beangenerator Protégé plug-in[4]. The implementation of the *Information Retrieval* component is based on the use of the Java Lucene library[5]. Lucene is a search engine library that contains implementations of well-known algorithms, such as: the inverted file index, a stopword remover and the stemming algorithm. Persistence of the relevant metadata regarding knowledge artifacts was achieved with the use of XML files. The taxonomy is also represented in an XML file, structured as prescribed in a particularly developed XML schema.

The development of the handheld prototype added an extra component on top of the two components just described. The interface between the new *Peer Discovery* component and the *Recommendation Agent* component is achieved by wrapping up outputs of the former into Agent Communication Language messages that are then sent to the latter. As in the desktop version, the agents are arranged in a peer-to-peer fashion composing a recommender system running on desktop computers. The *Recommendation agents* and *Information Retrieval* components are practically intact. However, a different GUI was developed to run in the iPAQ handheld device. To overcome problems with the limited resources on such devices, the recommendation service was kept in the desktop. This application communicates with the iPAQ through a wireless link to receive the user's inputs and send back recommendations. The GUI was implemented using the Personal Profile API implementation of the Java 2 Micro Edition version (J2ME). Finally, the *Peer Discovery* component was implemented using the Interconnect architecture (Uiterkamp 2005), developed to enable HTTP communication between service hosts and nomadic service components.

6. Related Work

The most distinguishing feature of KARe is given by the consideration of taxonomic information to recommend knowledge artifacts. We have no knowledge of other initiatives that apply taxonomies to aid the process of questioning and answering, adding to the user's query the contextual information provided by the concept to which this query is assigned. Other than this, KARe's distinction is materialized in the query scope reduction stage of the recommendation algorithm, in which a concept of the questioner's taxonomy is matched with concepts from the responder's taxonomy. Matching taxonomies has been targeted before, having gained considerable strength in the last few years, especially boomed by developments in the Semantic Web. In this section, we just cite two initiatives more closely related to ours.

There are mainly two ways of conciliating two different taxonomies A and B. One focuses on mapping labels associated with a concept of taxonomy A into concept labels of taxonomy B. Among the works that adopt such technique, some use only syntactical information of the labels, simply matching keywords, while others go beyond this, considering in addition to syntax, semantic information about the labels, usually

[3] http://protege.stanford.edu/
[4] http://acklin.nl/page.php?id=34/
[5] http://lucene.apache.org/

supported by a dictionary or thesaurus. This is the case of the CtxMatch algorithm (Bouquet et al. 2003), a linguistic-based approach which adopts WordNet[6] lexical reference system[7] to disambiguate and stem labels. This algorithm indicates the relationship between two matched labels, i.e. it informs if the label found in taxonomy B is equal, less specific or more specific than the selected label in taxonomy A. The problem with this kind of technique is that it usually results in low recall. Although the used dictionaries or thesaurus provide valuable additional information about the labels, this is hardly enough and a match in the responding taxonomy is rarely obtained (Avesani et al. 2005).

Our algorithm adopts a different approach of matching taxonomies, by considering not only the labels representing the concepts of the taxonomy but also the keywords of all documents classified under the concept. This adds a great deal of information to the concept representation, usually improving the algorithm's performance at least in terms of recall. A similar approach to ours is adopted by (Avesani et al. 2005). However, this work tries to identify the semantic relationship between the two corresponding nodes, while our approach limits itself to finding one or a few most similar nodes in the responder's taxonomy. In addition to that, another difference may be highlighted. For functioning properly, the approach of (Avesani et al. 2005) requires the two taxonomies to share documents, as the similarity between them is calculated on the basis of this redundancy. Our algorithm does not require such duplication, working well even if there is no redundant information.

7. Conclusions and Future Work

Here we presented KARe, an agent-oriented recommender system that simulates the natural social processes involved in knowledge sharing in virtual enterprises. The suitability of KARe for virtual enterprises can be justified by the fact that the system reflects the natural distribution of this kind of environment. In this way, different organizations can partner and exchange knowledge in a peer-to-peer network. When these organizations depart, the exchanged knowledge remains in all participant organizations, enriching their knowledge base so that it can be reused in the future.

The core of the system is an information retrieval algorithm that was the focus of this chapter. Besides describing such algorithm, this chapter presents empirical results to confirm the algorithm's efficiency, and discusses the system's prototypes. The results of the algorithm assessment showed considerable gains in the recommendation quality are achieved by using the proposed approach. In the future, we aim at confirming this conclusion by evaluating the algorithm against different and larger datasets. Improvements in our evaluation approach can be achieved by experimenting KARe with real users. This would provide us with taxonomies classifying a representative sample of papers, besides unbiased queries. It is however essential to count on several people during a large period of time to have results with statistical confidence

We already envision some possibilities of enhancing the query scope reduction performance. Our research agenda includes the experimentation with the smoothing technique presented in (Sona et al. 2004) to improve the representation of the taxonomic concepts. Smoothing is a technique targeted at situations in which there are many nodes classifying only a few knowledge artifacts. Thus, such technique is suitable for initial stages of system use, when KARe peers are starting to collect their documents and exchange questions and answers.

In addition to this, scalability issues must be targeted before KARe can become a real product. At the moment, we only performed tests using two peers and we can already preview that some problems may arise if more peers are included. This issue particularly regards our desktop prototype and mainly results from the fact that when receiving a knowledge request from the user, the system broadcasts such request to all other peers in the network. We foresee two possibilities to overcome this problem. The first possibility regards the beforehand calculation of the nearest neighbor peer to answer to requests on specific subjects. Thus, the system would know which other peers are more likely to have the answer it seeks, being able to efficiently forward incoming knowledge requests. The other idea we could explore alternatively or in addition to this one is to set up a similarity threshold, limiting the number of documents exchanged between peers to reduce network traffic.

[6] http://wordnet.princeton.edu/w3wn.html

References

Adami, G., Avesani, P., & Sona, D. (2003) Clustering documents in a web directory. In *Proc. of the 5th ACM Int. Workshop on Web Information and Data Management* (pp 66-73). New York: ACM Press.

Avesani, P., Giunchiglia, F., & Yatskevich, M. (2005) A large scale taxonomy mapping evaluation. In Y.Gil et al. (Ed.) *International Semantic Web Conference* (pp 67-81). Berlin: Springer-Verlag.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999) *Modern Information Retrieval*. Addison-Wesley.

Bonifacio, M., & Bouquet, P. (2002). Distributed Knowledge Management: a Systemic Approach. In Minati, G. & Pessa, E. (Eds.) *Emergence in Complex, Cognitive, Social and Biological Systems*. New York: Kluwer Academic/Plenum Publishers.

Bouquet, P., Serafini, L., & Zanobini, S. (2003) Semantic coordination: a new approach and an application. In *Proc. of the Second International Semantic Web Conference* (pp. 130–145). Berlin: Springer-Verlag.

Cole, D. (1997) Application of Knowledge Based Systems to Virtual Organizations. In *Workshop on Using AI in Electronic Commerce, Virtual Organizations and Enterprise Knowledge Management to Reengineer the Corporation*, AAAI workshop.

Davies, J., Fensel, D., & van Harmelen, F. (Eds.) (2003) *Towards the Semantic Web: Ontology-driven Knowledge Management.* West Sussex, UK: Wiley.

Guizzardi, R.S.S. (2006). *Agent-oriented Constructivist Knowledge Management*. PhD thesis, University of Twente, The Netherlands.

Jennings, N. R., Sycara, K. P., & Wooldridge, M. (1998). A Roadmap of Agent Research and Development. *Journal of Autonomous Agents and Multi-Agent Systems*, 1(1), 7-36.

Katzy, B., Evaristo, R., & Zigurs, I. (2000) Knowledge Management in Virtual Projects: A Research Agenda. In *Proc. of IEEE 33rd Hawaii International Conference on System Sciences*.

Liao, S. (2002) Knowledge Management Technologies and Applications: Literature Review from 1995 to 2002. *Expert Systems with Applications*, 25, 155-164.

Ludermir, P. G. *Supporting Knowledge Management using a Nomadic Service for Artifact Recommendation*. Master dissertation, University of Twente, 2005.

Merali, Y., & Davies, J. Knowledge Capture and Utilization in Virtual Communities. In *Proc. of the First International Conference on Knowledge Capture*, Canada.

Montaner, M., Lopez, B., & de la Rosa, J. L. (2003). A Taxonomy of Recommender Agents on the Internet. *Artificial Intelligence Review*, 19, 285-330.

Nonaka, I., & Takeuchi, H. (1995) *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. New York: Oxford University Press.

O'Leary, D.E. (1997) The Internet, Intranets, and the AI Renaissance. *IEEE Computer***,** 30(1), 71-78.

Pumareja, D., Bondarouk, T., & Sikkel, K. (2003) Supporting Knowledge Sharing Isn't Easy - Lessons Learnt from a Case Study. In *Proc. of the Information Resource Management Association International Conference*, Philadelphia, USA.

Sona, D., Veeramachaneni, S., Avesani, P., & Polettini, N. (2004) Clustering with propagation for hierarchical document classification. In *ECML Workshop on Statistical Approaches to Web Mining* (pp. 50-61).

Tiwana, A. (2003) Affinity to Infinity in Peer-to-Peer Knowledge Platforms. *Communications of ACM*, 46(5), 76-80.

Uiterkamp, E. S. (2005) *Nomadic Positioning Services for a Mobile Service Platform*. Master dissertation, University of Twente, The Netherlands.