Knowledge Search within a Company-WIKI

Stephanie Müller[1], Nils Kritzler[1], Alexander Tartakovski[1], Ralph Bergmann[1], and Ralph Traphöner[2]

[1]University of Trier,
Department of Business Information Systems II,
54286 Trier, Germany
{muel4102|krit4101}@uni-trier.de
{Alexander.Tartakovski|bergmann}@wi2.uni-trier.de

[2]Ralph Traphöner
empolis GmbH
Europaallee 10, 67657 Kaiserslautern, Germany
ralph.traphoener@empolis.com

Abstract

Usage of Wikis for the purpose of knowledge management within a business company is only of value if the stored information can be found easily. But foundational characteristics of a Wiki, which are its easy and informal usage, result in a big amount of steadily changing, unstructured documents. The widely-used full-text search provides often search results only of insufficient accuracy. In this paper, we present an approach which improves the search quality enormously. It makes use of index-based search which is supported by a meaningful knowledge base. The structure of the necessary background knowledge is offered by the concept of ontology and created based on the content of the Wiki. Search results are more precise and complete as knowledge-base search supports bilingualism (multilingualism), synonyms as well as the origin of term.

1 Introduction

The concept of Wiki [Baeza-Yates, 1999] provides a simple and efficient possibility to create knowledge and makes it accessible. It is especially suited for the purpose of knowledge management within a company because of the great acceptance of employees.

However, the authoring simplicity results in a while in a big amount of steadily changing, unstructured documents. Consequently, the users often lose the overview about the available content. Full-text search, which is usually implemented within Wiki-systems, doesn't help sufficiently to overcome that problem [Cesarano *et al.*, 2003]. It doesn't take the relationships between concepts and objects, synonyms, and bilingualism among other things into account and provides therefore often only insufficient search-results [Cesarano *et al.*, 2003; Money and Turner, 2004]. In this situation the user acceptance decreases, since the wanted information could be often found only after several attempts of using full-text search.

The same development could be observed at empolis GmbH[1] after the introduction of Wiki for the purpose of knowledge management. Since its implementation in February 2004, the amount of pages increased up to 5,500 in November 2004. After this great expansion the user acceptance began to decrease.
empolis GmbH and the Department of Business Information Systems II, University of Trier founded a project with the objective of overcoming the explained difficulties by developing a knowledge-based search function to enable improved access to the information filed in the Wiki.

In this paper, we present the concept and the realisation of the search function using a combination of following technologies: Semantic Web, Text mining, and Case Based Reasoning (CBR) [Davenport and Prusak, 1998; Davenport and Grover, 2001; Leuf and Cunningham, 2001]. A domain specific ontology provides a vocabulary for the semantically annotation of the content. The annotation is constructed automatically with the help of text mining technology. Similarity-based search on the semantically observed content is done using CBR-retrieval technology.

The second chapter describes the application of Wiki within a company in terms of knowledge management. While the third chapter introduces the concept of knowledge-based search, the forth chapter demonstrates its realisation. The last chapter concludes the paper with summary and discussion.

2 Wiki to emphasise knowledge-sharing

"Increasingly, knowledge is recognized as an organization's most valuable resource and the best foundation of sustained competitive advantage" [Maedche, 2002]. Knowledge management is rapidly becoming an integrated business function as companies realise that effective management of intellectual resources is connected to competitiveness [Abecker and Elst, 2004].

[1] empolis is an arvato AG subsidiary, an international media service company and part of Bertelsmann AG. It is supplier of enterprise content and knowledge management solutions.

The difficulty consists in the gathering of knowledge as well as its creation, allocation, storage and relocation.

One instrument to organise and cross-link knowledge is a Wiki [Baeza-Yates, 1999]. This concept offers a forum for its users to share knowledge and look up information. It simplifies and encourages knowledge sharing as its usage is held very easy, simple and quick. The handling of Wiki does not conform to a lot of rules and there is also no need to setup specific software

These characteristics lead to a lack of a formal structure as well as a dynamic changing landscape of a Wiki which makes it very difficult to keep an overview of the content. Especially the constant growth, which is anarchical and uncontrolled, makes this task more and more complicated. Additionally, many inner-company Wikis are kept in several languages which aggravate the task.

The existence of a Wiki is only of interest if not just the storage of knowledge is realised in an easy and uncomplicated way but also the relocation of the stored information is quick and simple. The main aim of a Wiki is the reusability of knowledge, but this is only achieved if the needed information can be found easily. To reach this aim the improvement of the search functionality is needed to grand better finding of relevant information and is presented in the following.

3 Knowledge-based search supported by the concept of ontology

The approach mostly used within a Wiki is full-text search; but it is already widely known that results are not satisfactory. Using full-text search, a result is only a 100% hit, if the title of an article corresponds exactly to the query. The problem of this method is the total ignorance of similarities between words like singular and plural or different words used for the same thing; multilingualism is not cared for as well. The listing of results contains many irrelevant articles, misses out several relevant documents and the ordering of relevance does not reflect the real order of importance of the articles. This insufficient search functionality leads to the decrease of usage of Wiki for knowledge sharing.

To improve the insufficient search functionality, the approach of a knowledge based search function is presented in this paper. Following this approach, an ontology provides the necessary background knowledge. Outgoing of this knowledge base, case base reasoning is used to represent the content of the several articles and get better findings when executing the search functionality.

3.1 From index-based to knowledge-based search

The first step while developing knowledge-based search functionality is the creation of an index which represents the content of an Wiki article. Search is then performed by comparison of the query with the index of the several documents. Index-based search provides a faster access to the content of a Wiki. Regarding its usually large number of documents which is also constantly increasing, this methodology is most appropriate in that context. If provided by a good knowledge base, it is also able to offer a better finding of relevant information.

The main problem which has to be solved is the ambiguity of natural language; it manifests itself in the synonym and polysemy phenomenon [Money and Turner, 2004]. The synonym phenomenon refers to the problem that the same concept can be represented in many different ways. The fact that words can have different meanings in different contexts is defined by polysemy [Cesarano *et al.*, 2003]. To provide a fast and reliable knowledge-based search, the knowledge of the language use within a Wiki is essential. Especially, the problem of imprecise interpretation of the search-query and the consequential need of "processes to ‚interpret' the query, to retrieve the expanded query condition according to the interpretation, and to evaluate the closeness of the result to the original query" [Liu, 2001] require the knowledge of the language use within the Wiki. Without this interpretation based on the knowledge, satisfying results of the user query are not possible.

As the Wiki landscape is changing constantly, a manually indexing of the various articles it not appropriate in this context. For automatically indexing documents, its content has to be classified without human intervention. That means there is a need to provide knowledge, realised through several categories where the documents can be mapped to. This scheme has to represent the content of the Wiki.

3.2 Ontologies to provide the necessary background knowledge

The knowledge base is realised by the usage of ontologies. It provides a common understanding of things of the world and for that reason are means to bridge the 'semantic gap' existing between the actual syntactic representation of information and its conceptualisation [Davenport and Grover, 2001].

Ontologies are the key means to annotate unstructured documents with semantic information, to integrate information and to generate specific views that make knowledge access easier [Davenport and Grover, 2001]. They provide the domain knowledge for the realisation of knowledge-based search.

Before mapping Wiki articles according to the domain ontology, the last one demonstrating the conceptual model of the Wiki has to be created. This scheme represents the set of classes, instances as well as their relationships, which map the content of the Wiki. A thesaurus completes that model; synonyms, pseudo-synonyms as well as acronyms are included to enhance semantic understanding [Cesarano *et al.*, 2003].

After the ontology is created, the metadata of various articles has to be produced which means indexing the documents. Its metadata has to represent the content of an article further on. The extracted words of an article are mapped on the concepts of the ontology. The metadata resulting of this process consists of ontology-concepts which present the content of each article. It is stored together with the corresponding article and is available for the searching function from then on. To support the search functionality it is appropriate to classify each article after creation or editing

Figure 1 Process of the creation of metadata

Afterwards, the retrieval is done while mapping the search query on the ontology and result the concepts which represent the query context. These are then compared with the metadata of the articles instead of mapping the document text itself. This process results in relevant findings for the user.

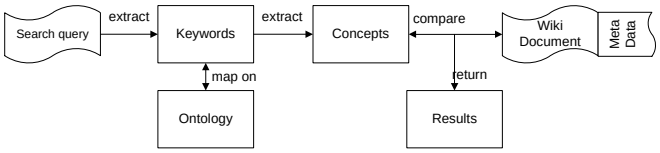

Figure 2 Query processing for retrieval

4 Realisation

The realisation of the above explained theoretical approach can be divided into two parts: The creation of the ontology and the development of the search functionality itself.

4.1 Identification of the specific domain knowledge of the Wiki

As the ontology needs to illustrate the content of the Wiki further on, the domain knowledge of the Wiki has to be detected. Outgoing of this data, the ontology can be created.

The starting point for this procedure it the collection of all articles which are contained in the Wiki; this set is named corpus in the following. To find out which words of the corpus reflect the content of the Wiki, word frequency lists are used. The word frequency list is build with the help of concordance programs. A concordance can be described as "an alphabetical index of all the words in a text or corpus of texts, showing every contextual occurrence of a word" [Aamodt and Plaza, 1994].

The first step is sorting out useless words which do not describe the content of the articles. These words are called 'stop words' and are so common that they are worthless in giving any information about the essence of an article. Resulting is the concordance of the remaining words. These are ordered according to their frequency. Next, there is a need of a decision, how often a word has to be present to be important enough for the assimilation into the ontology, words with a lower frequency are deleted accordingly. That choice is dependent on the total number of words on the list and for that we cannot give any recommendation for a reasonable figure.

After this proceeding, the word frequency list still contains several words which do not imply any relevance for the ontology. As language is ambiguous, it is not possible to sort out every useless word during the concordance process. This has to be done manually. There is also a need to remove words which might be significant but are used in several contexts and for that their meaning is not definite. After the removal of these not significant words, the content of the resulting list reflects the environment of the Wiki and for that is the initial point in creating the ontology.

4.2 Ontology creation process

Outgoing of the modified word frequency list, the manual creation process of the ontology can take place. To keep the overview, a visualisation tool is used.

"There is no 'correct' way or methodology for developing ontologies" [Maletic and Marcus, 2001], there exist several approaches which depend on the application that one has in mind. One possible way is to start with a rough first pass which is then refined in an iterative process [Maletic and Marcus, 2001]. As this methodology fits into the given context we decided for it.

This is done by allocation of a class to every word of the list. So, an overview of the content comes into being. These classes represent a starting point for filling the ontology with subclasses and upper classes. The filling process is executed finding the right place inside the ontology for every word of the word frequency list. It has to be decided where to put it into the hierarchical scheme. Following this procedure, additional facts have to be taken into account: sometimes there exist more than one place where the class belongs to or there is the necessity to create additional classes to merge several classes. The last step within this process is the consideration if there is a need to create additional classes which fit into the given context and therefore enrich the ontology.

Apart from is-a and kind-of relations which are already contained in the hierarchical scheme, other sorts of relations between several classes are created as well. For the performance of the CBR-based search functionality it is sufficient to define solely the strength of connection between classes, instead of complete and explicit definition of all relations. The strength of connection is a measurement which can be evaluated to find relevant search results. This is realised by similarity weights which range from 0 for 'no relation' to 1, which means 'equals'. This provides a view of the world that is understandable for the search engine and enables it to retrieve documents which contain similar keywords to those formulated in the query.

The last step is the creation of a thesaurus for every class. This feature supplies a base of keywords which stand for the respective class and serves for the semantic annotation executed by the text mining functionality. The figure below shows an extract of an example ontology.

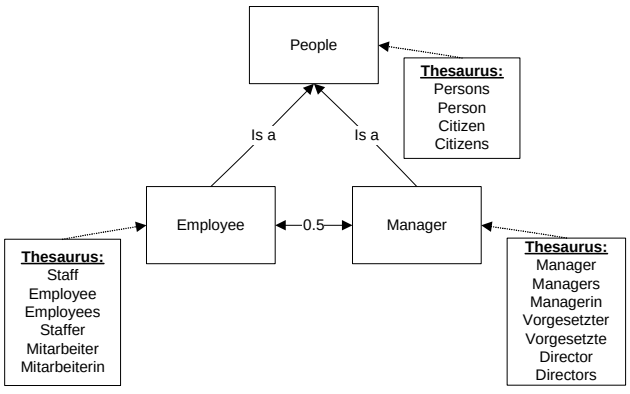

Figure 3 Extract of an example ontology

4.3 Embedding the ontology into CBR-Suit

To execute the search it is necessary to embed the ontology into a CBR suit. The case model has to be

created according to ontology. To start with the transferring process, all classes of the ontology are taken as attributes into the CBR suit. These attributes characterise the several cases in the case base. The thesaurus which has been created for every ontology class describes the valid type definition for every attribute respectively. According to the extract of the ontology displayed above, the corresponding case model includes the attributes 'people', 'employee' and 'manager'. The valid type definition for 'employee' contains 'staff', 'employee', 'employees', 'staffer', 'Mitarbeiter', 'Mitarbeiterin'. To enhance a structure within the unstructured collection of attributes, the hierarchical class scheme is adopted to enable inheritance. Attributes which have been created out of an upper class contain all attributes which are transferred from its lower classes. Following the example above, the type definition of the attribute 'people' contains the elements 'person', 'persons', 'citizen', and citizens'. Additionally, according to the hierarchical class scheme, it contains the attributes 'employee' and 'manager', and their type definitions respectively.

All other relations which have been created between several ontology classes are transferred into the CBR model as well. They are included as similarity measures between the respective attributes. They are means to calculate similarities and for that reason support the retrieval of similar cases when executing the CBR cycle. Outgoing of these similarity weights related cases can be found. The usage of this measurement is illustrated in more detail in the following when describing the text mining functionality.

To evaluate the content of every Wiki article, a text mining functionality is necessary. Its scans the articles and selects all attributes contained in the case model which describe and represent the content of the respective article. An array of relevant attributes represents the article further on. It can be described as a case in the case base. This proceeding has to be executed for every article to create the respective case base. To select relevant attributes, the text miner scans each article for those words which are contained in the type definition of every attribute. Next, those attributes are stored together with a reference of the original article and represent a case in the case base. The following figure shows some example documents with corresponding cases.

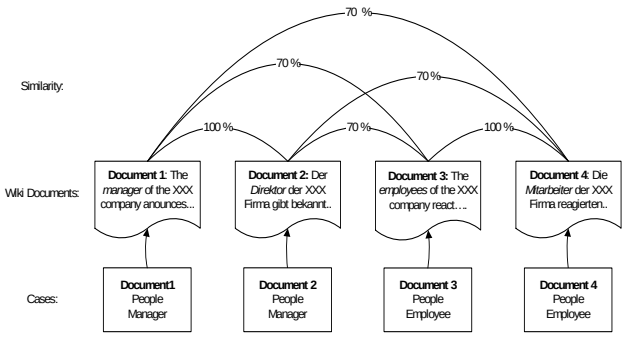

Figure 4 Creation of the case base

The attributes which are mapped to Document 1 and Document 2 are equal. 'Manager' is included in the same type definition as 'director' which belongs to the attribute 'manager'. As this attributes is included in the type definition of 'people', this attribute is included as well. As the cases of both documents contain exactly the same attributes, they have a similarity of 100 %. To compare document 2 and document 3, their case shows one similar attribute. Additionally, the attributes 'manager' and 'employee' are rated a similarity of 50 % in the case model. As a result these cases are rated similar 70 %. Similarities between cases are calculated by a similarity function which considers the amount of equal attributes as well as the amount and extend of similar attributes between two cases.

The CBR cycle starts with the input of the search query. With the usage of the text miner and the case model, a new case is created. Next, the similarity function calculates the similarity of this new case to the ones in the case base and retrieves the most similar ones. They are represented to the user in order to relevance.

To realise the search functionality we used the open retrieval engine orenge which has been developed by the project partner empolis GmbH. It has been created especially to execute intelligent case-based and knowledge-based searches and for that reason is also useful for our purposes.

orenge offers the possibility to store a data model to provide a knowledge base which supports the search functionality; here the case model can be entered. The so called aggregate class contains all attributes of the case model. To keep the taxonomic relations, a class structure is offered. That means, for every attribute which contains other attributes, a class is created which contains the attribute itself as well as all containing attributes.

Inheritance is supported. Additionally, the classes 'filename' and 'content' are included; they contain the original data of the Wiki files. The following figure illustrates the aggregate class:

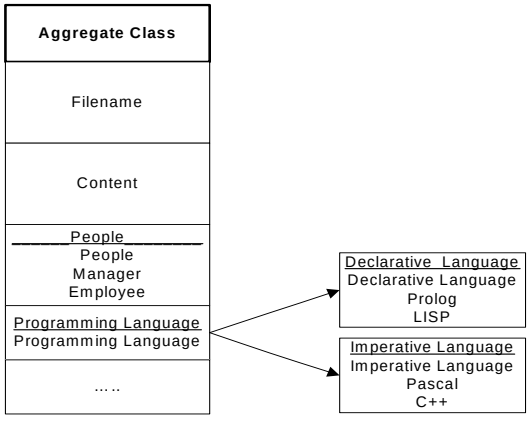

Figure 5 Aggregation class

The class 'People' contains the three attributes as already described in this chapter. The class 'Programming Language' contains only the attribute 'programming language', but two subclasses which contain attributes respectively. To describe this domain more specialised, these classes could contain subclasses to distinguish between object orientated and structured programming languages. As more precise a domain is modelled as more precise are the search results.

For every attribute in the data model, a thesaurus is entered which conforms with the type definition respectively. Furthermore, it is possible to enter similarity

weights between attributes of the same class as well as attributes between different classes.

After the data model is entered in orenge, the orenge Textminer parses the Wiki articles and created the several cases as described above. Each case consists of the name of the Wiki file as well as its content. Additionally every relevant attribute is included; that means every word of the article is scanned by the Textminer and for those words which are included in the thesaurus of an attribute, this attribute is stored in the case base to represent the content of the respective article.

Subsequently of the creation of the case base, orenge is able to execute the search functionality. The entered query is scanned by the Textminer following the same proceeding as the indexing of the articles. As a result, an array of attributes describes the query further on. The next step is the calculation of similarity to every article. Those articles which posses exactly the same attributes reflect a similarity of 100 %. According to similarity the results are listed and the user can evaluate the offered articles.

. It has to be considered that after every change of the data model the indexing process has to be done again. Additionally, after every change of a Wiki article, its index has to be updated to represent the actual content of the article.

5 Outlook and Conclusion

This paper addresses the problem of decreasing acceptance of inner-company Wikis, which occurs when the content becomes large and chaotic. The acceptance of employees decreases on the one hand because of loss of an overview over the available information and on the other hand because of lack of the search functionality. Full-text search, which is usually implemented within Wiki-systems, does not support the users sufficiently since the quality of the search-results is low.

The intention of the project described in this paper is to make the application of Wiki for the purpose of knowledge management within companies more attractive by development of an improved search functionality.

We introduced a realisation approach to knowledge-based search functionality which outperforms full-text search.

According to this approach, the domain knowledge of the Wiki is represented via ontology, which is created in the semiautomatic manner. For this purpose, the whole document corpus is analysed using concordance programs and, after manual validation, the remaining data can be taken over into the ontology as classes and relations. Following, after manual validation and extension, the ontology is embedded in the CBR-suit orenge.

Both, every document from the corpus and every query, are semantically annotated with text mining software contained in the CBR-suit which has access to the constructed domain ontology. Every semantic annotation of any document or any query is regarded as a single CBR-case within orenge. The search for the relevant documents is then carried out as a CBR retrieval process.

Based on the meaningful domain model, the quality of the search results is increased to high extend. The provided knowledge guarantees that the annotation is of a high quality and matches the content of the articles. Knowledge-based search copes easily with the weaknesses of full-text search such as “the gap between the user’s information need and the actual query strings they specify” [Cesarano *et al.*, 2003].It finds relevant articles regardless of which synonym is used to formulate the query. Another advantage is the support of multilingualism and different word forms. Results are represented according to relevance; that means not only 100 % hits are displayed, but also articles with related content.

However, it has to be considered that good search results depend exclusively on a good data model. The richer it is the better are the results. As it is of such importance, it has to be paid regard that the model is extensive and correct. Furthermore, the search does only operate sufficiently, if the data model spans the whole context of the provided data base. This makes a continuous improvement of the model absolutely important. But it has to be considered that this process is extremely time-consuming as well as costly. The maintenance of the data model is highly significant to achieve good results. If done continuously, this guarantees a good search functionality that works within unstructured documents and outranges full-text search to a great extend.

References

[Aamodt and Plaza, 1994] A. Aamodt and E. Plaza. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications, Vol 7, Nr. 1, 1994.

[Abecker and Elst, 2004] A. Abecker and van L. Elst van. Ontologies for Knowledge Management. In: Staab, S., Studer, R. (Editors), Handbook on Ontologies, Berlin 2004, Pages 435-454.

[Baeza-Yates, 1999] R. Baeza-Yates. Modern information retrieval. Addison-Wesley, New York 1999.

[Cesarano *et al*., 2003] C. Cesarano, A. Acierno d’, A. Picariello. An Intelligent search Agent System for Semantic Information Retrieval on the Internet: In: Proceedings of the 5th ACM international workshop on Web information and data management. New Orleans, Louisiana, USA, 2003, Pages 111-117.

[Davenport and Grover, 2001] T. Davenport and V. Grover. General Perspectives on Knowledge Management: Fostering a Research Agenda. In: Journal of Management Information Systems, Volume 18, Issue 1, 2001.

[Davenport and Prusak, 1998] T. H. Davenport and L. Prusak, Working Knowledge: How Organisations manage what they know, Harvard Business School Press, Boston, Mass. 1998.

[Leuf and Cunningham, 2001] B. Leuf and W. Cunningham. The Wiki Way Quick Collaboration on the Web, 1. Print, Addison-Wesley, Boston, Mass. [and others] 2001.

[Liu, 2001] H. Liu. Intelligent Search techniques for large software systems. Thesis. Ottawa-Carleton Institute for Computer Science, School of Information Technology and Engineering, University of Ottawa 2001.

[Maedche, 2002] A. Maedche. Ontology Learning for the Semantic Web. 1. Edition, Kluwer Academic, Dordrecht 2002.

[Maletic and Marcus, 2001] J.I. Maletic and A. Marcus. Supporting Program Comprehension Using Semantic and

Structural Information. In: Proceedings of 23rd ICSE, Toronto 2001, Pages 103-112.

[Money and Turner, 2004] W. Money and A. Turner. Application of the Technology Acceptance Model to a Knowledge Management System. In: Proceedings of the 37th Hawaii International Conference on System Sciences, 2004.