Exploiting semantic aspects to evolve a text-based search on a legacy document management system

Johann Grabner, Andreas Mauczka, Mario Bernhart and Thomas Grechenig
Research Group for Industrial Software
Vienna University of Technology, 1040 Vienna, Austria
{johann.grabner, andreas.mauczka, mario.bernhart, thomas.grechenig}@inso.tuwien.ac.at

Abstract

Semantic technologies provide the means to create a more efficient human-computer interaction. In this paper we showcase how to use semantic technologies in the design of a Document Management System (DMS) to re-engineer a legacy DMS using a text-based search. Due to the historical growth of the database, a text-based search without using domain knowledge delivers bloated result-sets that prevent an efficient use of the DMS. The requirement of an intelligent search can be suitably addressed by using semantic technologies, thus we propose such a new design of a DMS. We employ concepts used in ontologies to provide search results based on hierarchies that are based on concepts of natural language. We construct a metamodel for the application domain to deliver fitting suggestions along with the search results. Finally, we realise a prototype as a proof-of-concept based on the introduced design to highlight the evident advantages of a semantic approach.

1. Introduction

In recent times social interaction and collaboration entered the spotlight of websites, flaunting the title Web 2.0 [10]. After the wide acceptance and success of applications in Web 2.0, the semantic web as an evolutionary follow up of the classical web, which extends the classical web by using the context of information to process information, is rapidly gaining importance[9]. Folksonomies of Web 2.0 are the product of the collaboration of large amounts of individuals with unique perspectives. To be able to gain a foothold in daily life, semantic web products will have to extend Web 2.0 products like Folksonomies by semantic aspects.

A text-based search retrieves information containing the exact or terms similar to the search string. If you search just for a certain text, a text-based search is capable of delivering satisfying results. A text-based search is very likely to fail to find the context behind a search string, however. In the present study we introduce a search capable of getting both the information and the context of a given text.

Legacy systems tend to evolve into behemoths that might get the required task done, but rather slowly. This is especially tedious when the legacy system is a stand-alone system of an organisation - and every user is forced to work in this system. The legacy system analysed in this work is a document management system, called internally Office Information System, used for filing and searching of documents. This document management system (DMS) has, due to natural growth of the document base, outlived the usefulness of its own text-based search as search results tend to deliver unmanageable amounts of data.

Therefore the capabilities of text-based search are seemingly met and a new approach was suggested, which is presented in this paper. We propose a concept for a semantic search for this DMS and we implement a prototype for the users of the legacy system. We will employ tagging-mechanics and constructs of natural language to support search, file and even navigation through the database. We realise a customised triplestore by using open source technologies. Furthermore we use a Folksonomy with semantic extensions to provide qualitative suggestions for categorisation, navigation and search. Finally, we give a short introduction of the metamodel used to model semantic relations between hypernyms and instances.

2. The Legacy Document Management System

The office (Organisational Unit E010 - “Organisation and Coordination”) of the Vienna University of Technology is responsible for all incoming documents by mail, the distribution of these documents to the according organisational units and finally for the archiving of these documents for a period of ten years. The existing workflows are supported by a legacy software system named “OIS” (Office Informa-
tion System). The existing system is in its final software lifecycle stage, since it is not capable of meeting today's requirements and problems in the processing of the existing amount of data arise. The system is in use since approximately 20 years and more than 200,000 files have been entered into the system, averaging at 10,000 files per year.

Examples of problems with the legacy DMS are:

- Lack of support when filing documents - currently the order of the system is maintained by the experience and established conventions of the users.
- No filing across topics - the legacy system can only attach documents to one topic. Content covering multiple topics cannot be realised at the moment.
- Traceability of content - the description of content of a document is currently subject to undocumented conventions. These conventions determine which attributes are used to describe the content (e.g. contract name, paragraph, ...).
- Limited number of search results - the maximum number of results is currently limited. Due to the nature of the DMS, not all relevant search results are listed by the system.
- Step-wise refining of search criteria - based on a result set, it is currently not possible to step-wise refine the search criteria to get better results.

The existing problems are faced by proposing a design and a prototype realisation as a proof of concept of an innovative and semantic document management system (DMS). By using a semantic approach we hope to improve the search beyond the capabilities of a classical, text-based search engine. To improve acceptance of the new DMS, current system users were heavily involved during design and implementation of the prototype. We integrate domain knowledge of the OIS users to find possible innovations in the area of DMS as the user group is considered as very experienced DMS users.

Finally a prototype of the DMS is implemented that integrates the semantics of the domain into the current workflows. By taking these semantic relationships into consideration, this prototype is capable of assisting during search and filing of documents by giving content and context related suggestions. To be able to give these suggestions, the system needs to derive the knowledge from the domain and from the current usage of the DMS by the users.

3. Design and Implementation of a Semantic Solution

The scenario described earlier is augmented by a number of factors that have to be taken into consideration for the final solution. These are current user-problems, additional requirements beyond the scope of the legacy DMS gathered in stakeholder-interviews and project management input. The boundaries of the new semantic solution are given by current workflows and processes. The search-function is considered a primary factor for the quality of the DMS in the given scenario as it is of central importance for the workflow in the enterprise. Another important factor are standardised descriptions and how to handle them in a user-friendly manner. The integration of semantic aspects into a folksonomy is required for a working tagging system in the context of this work. As it is one of the major problems of the current system and the system continuously grows every year, suggestions for navigation through the data sets, but also for tagging have become a key requirement that needs to be addressed.

In the following chapters we show how we addressed these concerns in our concept and we give a short insight into the technical realisation in form of a prototype.

3.1. Improving the Search Beyond the Scope of a String Comparison

A text-based search is capable of returning exact or similar matches. This scope however is too narrow for the requirements of the stakeholders because it does not provide accurate search results in the existing database. A semantic factor is missing. Due to this deficiency, semantic relationships between terms won’t be recognised and therefore excluded from the search results. To address these issues the system is able to ask the desired meaning of a term, if the user just entered a homonym. Furthermore the system is capable of mapping alternate naming to the same term. By supporting homonyms, synonyms and abbreviations in the DMS a new quality of search results for the user can be achieved.

3.2. Standardised Descriptions

Natural language with its numerous possibilities of expression and different points of view of similar terms represent obstacles in the standardisation of textual descriptions. This lack of standardisation prevents an automated processing of descriptions. To avoid these pitfalls, we employ the use of tags. This concept is being used in the Web 2.0 with big promise. However, using tags does not guarantee standardised description texts. But the simplification of language lowers the risk of complex descriptions and improves the automated processing of these descriptions.

3.3. Folksonomy with Semantic Extensions

A certain degree of freedom for creating tags in the daily usage of the system is a required trade off to increase user-
acceptance of the application. Similar to [9] and the success of Web 2.0, we believe a collective intelligence of the users is more productive than restrictions on the system. This holds especially true considering the collective need to have a usable tool. The focus of the concept therefore lies in the support of the workflows of the users and to create a more efficient work-experience. By using semantics in the processing of the data, the system is capable of supporting the user by proposing meaningful suggestions that are similar in content (and not just textually). Furthermore suggestions build a fundamental design element for navigating the existing datasets. To provide qualitative suggestions for categorisation, the language concept of hypernyms [4] is used in addition to homonyms and synonyms. By using the natural boundaries of the application domain, it was possible to build a metamodel based on the relationships between the generic terms (similar to the ontologies presented in [6]). We use only one metamodel in the application instance for the prototype.

Connections between hypernyms are modelled using only one kind of semantic relation. This relationship is a hierarchical relationship and shows which hypernym is the subordinate of the other (a meronym). Therefore, the focus is on the existence of a relation between two hypernyms and not on a detailed description of the relation itself. The user has to define the kind of relation of the suggested tags with the entered term himself, the system just shows connected tags. The system relies on the associations of the user to the shown combinations of different tags.

The simplified ontology of the user-domain reflected by the metamodel is the key element for semantic extensions. When performing a search with a combination of tags, the result set not only will contain matching tags, but also combinations of subordinate tags of the search tags (meronyms of the search tags). A tag used as a keyword in a search therefore represents a semantic tag space (all meronyms of the tag are included), containing the existing semantically relevant tags in the system.

### 3.4. Suggestions for navigation and tagging

Each result set of a user search comes with suggestions for further navigation[2]. These suggestions include recommendations derived from the domain knowledge to further constrain the result set of the follow up search and the tags created together with the tags of the search criteria Figure 1. To create the domain knowledge driven recommendations, meronyms of the entered tags are used. To refine the initial search, the user can add meronyms step by step to semantically tune the result set until it fits his expectations. The system also delivers suggestions for creating a tag or tagging documents (see [5]). During tag creation the user gets a suggested list of meronyms related to the created tag after choosing a hypernym. The metamodel is used to find the tags qualified for a relation Figure 2. For tagging documents, the system suggests combinations of tags sorted by frequency of appearance. Thus the user is informed which sets of tags were used for prior tagging of documents. By showing possible matchings, the system guides the user to place his documents in the proper (best-fitting) filing area. The system supports the user by using the domain knowledge to provide suggestions in every aspect of the application.

The structure of the hypernyms is built using three semantic axes (based on concepts presented in[12]). These axes are time, place and content. The suggested tags for navigation are grouped according to hypernyms on the semantic axes Figure 1. A strategy for visualisation of the search results is selected depending on the numbers of tags related to the search criterion. If there are less than six entries, the tags are listed consecutively. For more than six and less than 100 tags a combo-box is used. If more than hundred related tags exist, a text field with auto-completion is used together with a list of the top five tags used together with the search criterion.

### 3.5. Technical Implementation

The concept presented in the earlier sections is implemented in a prototype. The database technology was con-
strained due to a given software environment of the legacy application. Therefore a relational database system had to be used to create a customised triplestore. Due to the constraints earlier using a stand-alone triplestore or an existing triplestore based on a relational database management system (RDBMS) had to be discarded. The underlying RDBMS of the legacy application is MySQL\(^1\), thus MySQL is the RDBMS on which the customised triplestore is build on. The triplestore is realised by using two tables in the database. The first table holds unique identifiers of the resources (realised as URIs) including the used literals. The second table holds ternary relationships between the resources of the first table and is representing therefore the triplestore. This option for realising the triplestore was not chosen deliberately but to conform to organisational constraints on the prototype.

As there were no such constraints on the choice of the web application framework, we chose “Ruby on Rails”\(^2\). Development started out swiftly due to numerous Code- Generators and Plugins available. RESTful Routes \(^3\) are used to give access to the existing resources in the system. Every tag has a URL, which is used as unique identifier for the resources in the database. Auto-completion for entering tags is implemented using functionality provided by the “Yahoo! User Interface Library”\(^4\).

---

\(^1\)http://www.mysql.com/
\(^2\)http://rubyonrails.org/
\(^3\)http://guides.rubyonrails.org/routing.html#restful-routes
\(^4\)http://developer.yahoo.com/yui/
\(^5\)http://dublincore.org/
The type of the resource is specified by its given hypernym. Synonyms and denominators or abbreviations are optional for the descriptions. The predicates “dcterms:title”, “dcterms:identifier” and “dcterms:alternative” are included from the standard to put tags and literals into the relations according to the DCMI Standard5. We use “dcterms:hasPart” to express meronym-like relationships between two tags. For reasons of performance additional attributes of resources are stored in conventional database tables. These attribute-tables hold the application-specific, DMS relevant data about the corresponding resources in the triplestore. To improve data queries, statistics about tags and documents are recorded in caching-tables. These tables are constantly updated. Furthermore, to improve performance consecutive queries on the same table are congregate into a single query.

4. Related Work

Folksonomies and the extensions of Folksonomy by using semantic aspects is currently a hot topic due to high activity in this area in the semantic research. Marchetti et al. identify in [8] weaknesses of existing, collaborative tagging-systems. Based on services of Wordnet6 and Wikipedia 7 a semantic, collaborative tagging-system is implemented. Entries in Wikipedia and Wordnet are used to allow the user to learn more on a concept behind a tag. To connect a resource and a concept the user is able to choose from different relations. The main difference between the system in [8] and our system is that the system in [8] has external dependencies (e.g. Wordnet, Wikipedia, ..). The specification of a semantic statement is the argument for the semantic search, while our system accepts tags as search input. The main focus in [8] is semantic tagging - no semantic navigation for result sets is implemented.

Hope et al. [4] present a semantic tagging-system for blogs. They use hypernyms for their semantic. Compared to our work, Hope et al. chose a simplified approach as they do not rely on meronyms nor do they have a navigation. Similar to [8] they created a prototype using Wordnet to create an ontology of tags.

An interesting approach to generate semantic tag-hierarchies for navigation in an existing Folksonomy is presented in Laniado et al. [7]. In [11], Specia et al. build a semantic tag-hierarchie by using co-occurring tags in folksonomies. Both Specia et al. and Laniado et al. deliver the foundations for approaches to extend Folksonomies with semantic aspects. Our work relied on their research during migration of the data from the legacy system.

5. Conclusion

The proposed design of a semantic search based on a semantic tagging-system is not only restricted to the use in the DMS area. Any resources can actually be described using the tagging system. By supporting aspects of natural language, the way in which users and machines interact with each other can be improved. Applications that incorporate domain knowledge and aspects of natural language are capable of providing a level of service superior to conventional text-based searches. Concrete suggestions to refine the search result set can be directly derived from domain knowledge. The system is capable of asking the user to correctly specify ambiguous entries and actively support the user with semantic suggestions. We only used hypernyms and meronyms out of the number of possible semantic relations to provide the users with tangible, in natural language common relations. Simplicity is a key factor for acceptance of a system, especially in the domain of document management as users working with a DMS tend to be no computer experts. By passing into the semantic web, applications are required to show the user semantic aspects in the processing of information in a simplified fashion [1].

The prototype of the DMS has an improved search functionality over the legacy system and was widely accepted by the current DMS users. Descriptions and categorisations into filing areas were migrated from the legacy system as tags into the prototype to allow a smooth integration for the users.

5.1. Future Work

There are several possibilities to extend the presented method for realising a semantic tagging-system. The realisation of distributed queries across several application instances in different domains might be a possible extension. Using ontologies to transfer domain knowledge seems an obvious, possible next step. The modelled domain knowledge may not only be used for semantic suggestions but also for answering questions a user might pose. New co-workers might be integrated easier by finding proper visualisation techniques of the domain knowledge. A problem that has not been addressed yet is the tracking of changes in domain knowledge. Filing structures in the DMS, terms and semantics all change over time. There is a strong need for mechanisms to reconstruct changes in the domain knowledge.

References


---

5http://dublincore.org/
6http://wordnet.princeton.edu/
7http://www.wikipedia.org/


