A Conceptual Modeling Approach to Semantic Document Retrieval

Terje Brasethvik [1] and Jon Atle Gulla [2]

[1] Norwegian University of Science and Technology, Trondheim
brase@idi.ntnu.no
[2] Elexir Sprach- und Informationstechnologie, Munich*
jag@elexir.de

Abstract. This paper describes an approach to semantic document retrieval geared towards cooperative document management. In our conceptual modeling approach, a semantic modeling language is used to construct a domain model of the subject domain referred to by the document collection. This domain model is actively used for the tasks of document classification and search. Moreover, linguistic techniques are used to facilitate both the construction of the model and its use. This paper presents our prototype model-based classification and search tool and how it is applied on a document collection from a Norwegian company.

1 Introduction

Conceptual modeling languages, techniques and tools serve different purposes with respect to Information Systems. In particular conceptual models serve as a vehicle of communication between various stakeholders, they facilitate information exchange and they provide (a formalized) documentation of several perspectives of the system as well as the Universe of Discourse. In areas like information and knowledge management and ERP systems, models are not only used during system development, but may also be used actively as access points to information during the whole system lifecycle [1]

Several approaches to using conceptual models in the design of web-based information systems have been proposed (see e.g. [2][3]). However, there are relatively few approaches that use conceptual modeling techniques as access points to information on the Web. This is rather surprising, as the ability to describe information precisely is central in much current research and standardization efforts, such as the Semantic Web [4] and the OntoWeb [5] initiatives. Of particular importance here is the construction of ontologies to facilitate information exchange. So far, efforts in these areas have come from Artificial Intelligence and Library Science, rather than from the modeling communities. One possible reason for this is that conceptual modeling techniques can be too labor intensive for the magnitude of the world wide web in general.

* Elexir is a subsidiary of Fast Search & Transfer, http://www.fast.no.

It is our claim that conceptual modeling languages are useful in information retrieval and information management on the web. In smaller and more restricted settings than the whole web, such as Intranets, Special Interest Groups and Project web sites, these techniques may be made to scale. Also, in these settings web tools will have to be more situation-specific and are often designed to support the activities of the users, particularly with respect to document classification and retrieval.

This paper describes an approach to semantic document retrieval geared towards cooperative document management. A semantic modeling language is used to construct a domain model of the subject domain referred to by the document collection. Whereas this domain model is central in document classification and search, linguistic techniques are used to facilitate both the construction of the model and its use. The techniques represent an interface between the textual documents and the formal concepts represented in the model and partly automate the user tasks.

We start in Section 2 by describing the notion of cooperative document management and the problems related to semantic document retrieval. Section 3 describes our approach in detail, while Section 4 presents our prototype applied on a small example from a Norwegian company, while section 5 gives a presentation of related work.

2 Cooperative Document Management

Work in organizations today is document intensive. A substantial amount of information in an organization is resident in documents. The ability to retrieve and extract information from these documents is becoming increasingly important [6][7] and is a natural part of such efforts as knowledge management and organizational memories: *"Insights drawn from the documents are seldom made explicit, although persons who work in teams, form a group or belong to a community would greatly profit from sharing their knowledge." [6].*

In semantic document retrieval, the challenge is to express document semantics in a formalism that enables efficient document retrieval and increases reuse of documents and their enclosed knowledge. This is a distinguishable part of document management and is something we will refer to as *cooperative document management* for the rest of this paper.

As long as documents are in their production phase, they are often subject to *local management*, i.e. they are guarded by their authors and distributed to a limited audience like the project group. The problem of document management arises when these documents are to be made available outside their local production context, i.e. when they are to be used and reused by other actors, possibly in different settings and locations. Web technology in various flavors is used to make documents available, as in Intranets and community web sites. However, there are no explicit mechanisms available to describe the semantics of these documents and to classify and organize them for the benefit of later retrieval and use. For these purposes, one turns to document descriptive meta-data [8][9]. The need for meta-data is situation specific. Various schemes for describing document semantics for example use of selected keywords, written abstracts, text-indices or auto-summaries and even collectively created free-text descriptions. Our approach draws on the use of keywords from controlled vocabularies.

2.1 The Use of Conceptual Modeling

With controlled vocabularies, most of the work is put into the definition of the concepts in the vocabulary. A well-defined vocabulary becomes a tangible representation of the subject domain of the documents that can be applied directly in semantic description and retrieval of documents. Vocabularies are often created and expressed in their own "subject languages" and are useful in their own right: *"Using these to retrieve information provides a value added quality, can transform information into knowledge. When this happens, the subject language becomes an analogy for knowledge itself." [10, p 127]*

Fundamental to our approach is the idea that both document descriptions and query expressions can be created using concepts from the vocabulary. The "language problem" in retrieval systems arises when the descriptions and queries do not match.

With controlled vocabularies, the language problem is approached by the proper definition of the vocabulary and by the way this vocabulary is put to use in the actual semantic description and retrieval system. In our approach, concepts in the vocabulary must be given a domain specific definition and users must be able to interpret these concepts in order to create document descriptions and query expressions.

Models are constructed whenever knowledge should be communicated to others and examined independently of the knowledge originator [11]. Modeling languages provide the necessary mechanisms for converting personal knowledge into publicly available representations. Given appropriate tool support, modeling languages are in their very nature intended to support the shared construction of domain semantics, a process quite similar to the construction of domain vocabularies for document management. Conceptual modeling languages allow us to define concepts, their properties and interrelations, and to communicate this information to the stakeholders.

In an approach to document retrieval, the definition of concepts in the model must be textually grounded, i.e. the definition of a concept must be related to its appearances in the document text. We use linguistic analysis to create a *domain model lexicon* containing text level terms and word forms, thus representing the interface between the conceptual model and the document text.

2.2 The Example of Company N

Throughout this paper, we will illustrate our approach with an example from a large-scale Norwegian company, *Company N*. When projects are carried out, all documentation is stored in separate project databases in Lotus Notes, where documents are organized according to the appropriate project activity (or sub activity and task). Documents in this sense can mean anything from small half-a-page notes, written directly in the Notes text editor, to special documents like requirements specifications, that may be several hundred pages long and are added to the database as attachments. For documents written in Notes, the system offers a free-text search facility. The database naturally structures information according to properties relevant for projects, such as activity names, titles, descriptions, responsible, deadlines and so on. These are the contextual meta-data attributes that facilitate browsing and retrieval from the database.

Even if Lotus Notes provides advanced search facilities, these are hampered by several factors; Notes search is not efficient in cases where the volume of documents increase, there are large variances in document types, and of course when the document text is not available in Notes, but only as a "binary" attachment. Consequences of this is that users feel they have to really know which database to search, and even sometimes be familiar with the structure of the database in order to successfully find documents. Company N also uses a web-based Intranet built on top of Lotus Notes. In order to improve distribution of documents and information, selected documents are extracted from the Lotus Notes database and published on the fixed intranet.

3 The Approach

Our approach to semantic document retrieval is based on two fundamental assumptions: First, we believe that it is possible to create a domain model that defines the semantics of the subject domain and that both semantic document descriptions and query expressions can be created as selections – or fragments - of this domain model. Second, we believe that the users may use this domain model interactively in order to classify and retrieve documents semantically.

3.1 Overall Description

The functionality of the document system includes three fundamental processes:

- The construction of the domain model. This is mainly a manual process, though the model must reflect the subject domain of the documents. To speed up the modeling, we run several linguistic techniques on representative sets of documents from the domain.
- The classification of documents is done by selecting the appropriate domain model fragments and providing some meta data. Classifications may be made more precise by labeling relationships between the concepts. The process involves the automatic matching of document text against domain model concepts and leads to a proposal that shows the users which concepts were found in the model.
- The retrieval of documents is normally done by entering text queries that are matched against the model before they are sent to the retrieval system. However, the domain model may also be used interactively both for creating the search expression and for subsequent query refinement.

Figure 1 illustrates the construction of the domain model. In addition to the manual modeling and definition effort, we include two additional steps. We start out by analyzing a set of representative documents from the domain. The text of the documents is prepared by some linguistic techniques (described in section 3.3) before being compared with a reference document corpus. The result is a list of highfrequented terms from the document domain, as well as a correlation analysis that indicates some possible relations between these terms. This list is then taken as input to the cooperative and manual modeling process.

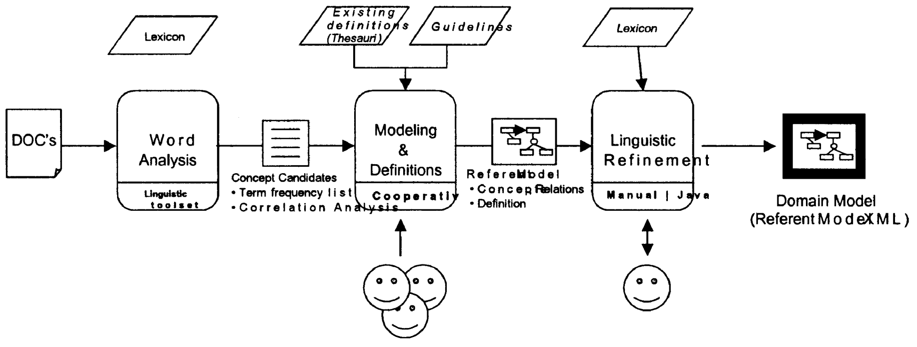


Fig. 1. Constructing the domain model

In our system we use the Referent Model Language [12] and the corresponding "Refedit" [13] modeling editor to define concepts and their properties. In order to prepare the models for automated matching against document text, we perform what we call a linguistic refinement of the model. For each concept in the model we add textual definitions of the concept, as well as what we call a term list, i.e. the list of terms/words that we will look for in a text as designators for concepts. Today this list is created manually, but this could be partly automated with the use of a synonym dictionary or some linguistic domain analyses to be described later. Furthermore, we add all conjugated word forms for the terms in the list. For Norwegian, these word forms are extracted from a full-form dictionary.

The process of classifying documents is described in figure 2. Again, this could be performed as a completely manual process, having the user interact with the model in order to select and refine the model fragment that best describes the semantics of the document at hand and to enter the meta-data attributes appropriate for this domain. However, if the model is prepared with the linguistic enhancements, we may perform an automated matching of the document text to give the users an indication as to which concepts could be relevant for a document. The user may also work simultaneously on a group of documents. For a group of documents, the matching provides an overview of how the text in the documents is reflected by the domain model concepts.

To utilize the relations between concepts, there are several possibilities. The correlation analysis from the initial document analysis step provides a statistical measure that indicates the strength of a relationship. These strengths or weights can be used to suggest expansions to either the document classification or the later queries. Simpler weighting algorithms such as counting the times a relation is used, is found in indexing approaches [14]. Our approach has so far been to go with the traditional modeling way, i.e. to provide meaningful names for a relation. For the task of classifying documents, these names can be entered at classification time for each document. Provided relation names are stored with the model as synonyms for the relation and then provided as "synonyms" for the relations in subsequent classification tasks.

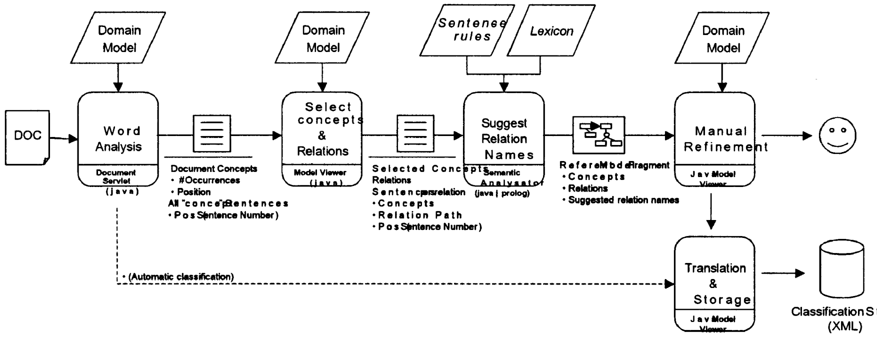


Fig. 2. Using the domain model for semantic classification of documents

In the end, the completed document description is converted to an appropriate storage format and installed in a classification store. The intention is to apply a standard meta-data representation format (such as the Dublin Core, RDF-XML serialization syntax) or to be able to use whatever indexing and retrieval machinery is available in the intranet setting. Today, we create our own DC-like description file in XML format.

Documents are retrieved by selecting model fragments that reflect the information need of the user. Even if the queries have to be created in terms of model concepts, the user may start with a text only query that is matched against the model in a similar way as for classification. However, in order to exploit the fact that the domain model represents "a map of available" information, our system supports a model-based interface to document retrieval that allows the users to interactively create and refine query expressions and visualizes how the result set is distributed in terms of model concepts.

3.2 System Architecture

An overview of the system architecture is presented in. The client, denoted CnS client (classification and search), is implemented as a standalone Java application. The CnS client has two modes, as indicated by the name. For classification, the client supports working simultaneously on a set of documents that are locally managed in a user profile. Also the linguistic enhancement of the model is performed using this client. In the search mode, the client communicates with a standard Web-browser for listing and viewing documents.

The server side components are implemented as a variety of add-ons to a standard Web server; mainly cgi-scripts and servlets. The client communicates with the server by executing http get/post commands and receiving XML encoded replies. As it is our goal to interface our system with existing retrieval machinery or document storage systems, we have not developed any extensive solutions to these functions, they are best considered working prototypes that are applicable in stand-alone settings.

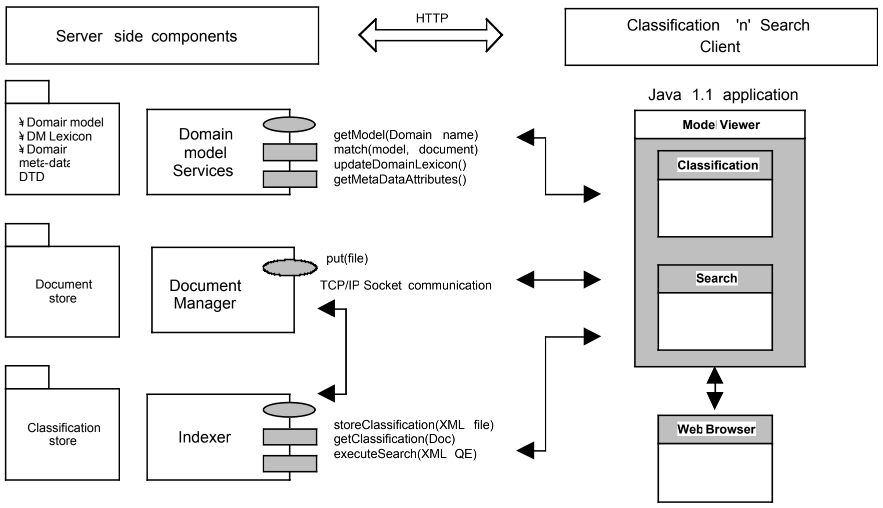


Fig. 3. Overview of system architecture

Currently all server-side storage of information is done using XML files. The main components on the server-side are:

- Domain services: these are cgi-scripts written in Perl that "deliver" the domain model and its additional files upon request. Models are stored as XML files. The linguistic enhancements, which we may denote the *domain model lexicon,* are stored in a separate XML file that refers to the concepts and relations in the model file. This lexicon is updated from the client.
- The domain services also include the document-model matcher that is used to match the text of a document against the concepts in the model by using the corresponding domain model lexicon. The matcher returns match-sets, describing for each matched concept, the match count as well as the list of sentences that included a reference to this concepts. The sentences are used to provide examples of concept occurrences and to provide input to the naming of relations. Currently the matcher only accepts pure text and HTML documents.
- The document manager is a simple Java servlet that accepts documents and stores these in a document directory, providing a unique URL for these. This component is only used when necessary in order to ensure that all documents can be accessed through a URL, the preferred point of access to documents from the other components.
- The indexer is a cgi-script written in Scheme. The name is somewhat misplaced, as no actual indexing takes place in our prototype, the script just accepts the final document description XML files created by the client and stores and manages these. The name indicates however, that in a real setting, this module will be replaced by an interface to existing indexing and retrieval machinery. The indexer also accepts query expressions in XML format, evaluates these against the stored description files and formulates the XML containing the query results returned to the client.

3.3 Linguistic Dependencies

The concepts in our domain model are abstracted from their linguistic realizations in the documents. This means that a particular concept may be associated with a number of word forms. The word forms may be inflections of the same stem, though also synonyms and other semantically related word forms may be grouped under the same concept. Since it is a very tedious process to go through the documents manually and define a set of appropriate concepts, we make use of a whole battery of linguistic scripts.

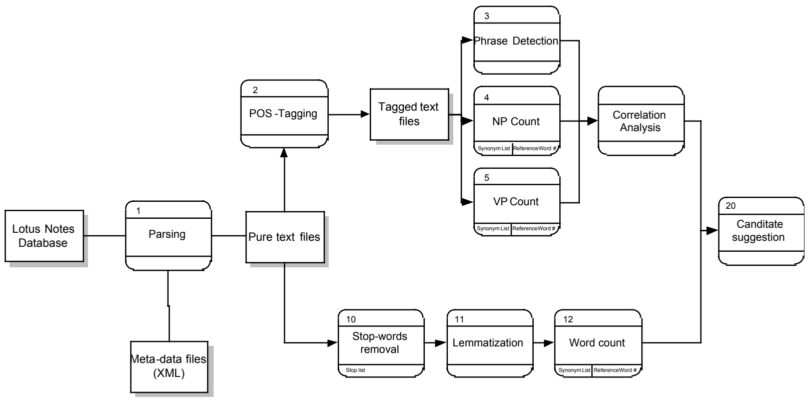


Fig. 4. Performing a linguistic analysis of documents in order to suggest concept candidates

Consider the document flow in figure 4, which shows how candidate concepts are generated from a set of Notes documents in Company N. After stripping off Notes-specific tags and other structural information in the documents, we use the resulting text files for two kinds of analyses:

- Detection of prominent phrases: Noun phrases (NPs) and verb phrases (VPs) may be good indications of concepts to be included in the domain model. To analyze grammatical phrases, we first tag the text with a simple part-of-speech tagger. Highly frequent phrases are detected, counted and inserted into our list of candidate concepts. For the relationships between concepts, we also need to expose the verbs commonly used in connection with these prominent nouns and noun phrases.
- Detection of prominent stems: After removing stop words from the document, we replace the word forms with the corresponding stems (lemmatization). We then count the number of occurrences of each stem and enter the most frequent stems into the list of candidate concepts.

With this approach, we help the domain experts in defining the concepts to be included in the domain model. A list of candidate multi-term concepts and a list of single-term concepts are presented to the user. However, we are only able to generate concept candidates, and a manual verification and refinement of this candidate set is necessary before creating the actual model.

4 System Walkthrough

In this section we go through a case study, in which our system was used to classify project documents for Company N. Our example is selected from an actual project database in Lotus Notes. For the subject domain “collaboration technologies” - a high level terminology document with definitions of central terms from the domain has formed the basis for our domain model. These terminology definitions are part of an initial ”mission statement” for the project. However, the abstract terms found here are rarely found in the text of the actual project documents. Thus, in order to bridge the gap between the texts and the high-level terminology document, we have to add more detailed concepts to the model. In particular, we use the general abstraction mechanisms of our modeling language to specialize the original abstract concepts. Furthermore, we run the texts through the linguistic analysis in order to extract terms that can be associated with the concepts in the model. It is interesting to note that in the Company N example, the linguistic analysis discovers exactly the same terms in both Norwegian and English. Most of the documents are written in both languages. Since classification and retrieval is performed by concepts and we can attach terms in both languages to these concepts, our approach can just as easily classify documents in both languages.

4.1 Classification

Figure 5 shows our CnSClient in classification mode with a fragment of the particular domain model and a small set of corresponding documents. The domain model fragment shows the hierarchy of collaborative processes, which in the terminology definitions from Company N is defined as an aggregation of coordination, production and other activities. The fragment also shows the specialization of the concepts representing *coordination* and *production* activities.

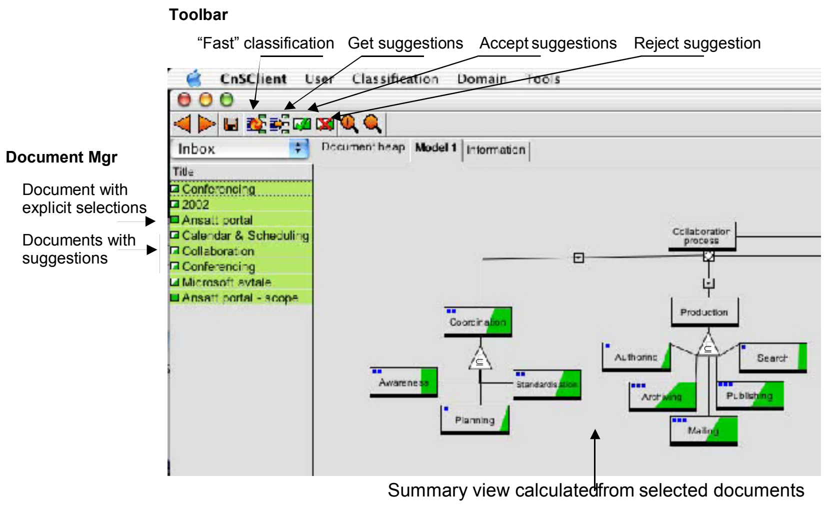
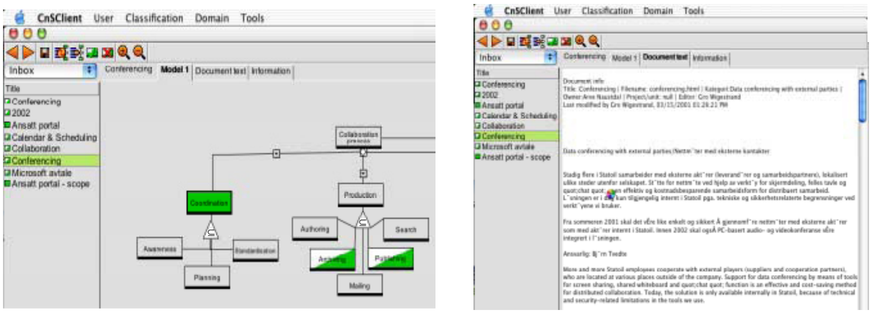


Fig. 5. The CnS tool: Classification modus - a working set of documents classified by way of the domain model

The classification of a document according to the domain model amounts to selecting the model concepts considered relevant for this document. In the classification mode, the toolbar provides the user with the options of getting suggestions from the server-side model-matcher as well as quick-keys for accepting or rejecting these suggestions. The “Fast” classification button lets the user accept whatever suggestions the server provides without further examination – our alternative to automatic classification. The document management function allows the user to manage a local set of documents that are under classification, these may be divided in folders. As is illustrated in the figure, the user may work with both one document at a time or a selection of documents simultaneously. The figure shows a summary view of all the selected documents. In this view the user has first received suggestions from the model-matcher and is now manually refining these suggestions. Suggestions are marked with a green triangle. In the document list, documents with unprocessed suggestions are marked with a small green triangle, while documents where the user has made actual selections (or accepted the suggestions) is marked with a filled rectangle.

The model-view shows a summary of how all the selected documents match the model. Green triangles illustrate concepts that have suggestions, the size of the triangle (along the bottom line) illustrates the percentage of documents in which this concept is suggested. The more suggestions, the more the triangle grows from right to left. Similarly, if a suggestion is accepted or a concept is manually selected, the triangle becomes a rectangle. As with suggestions, in the summary view the top of the rectangle grows from right to left according to the relative amount of documents in which this concept is selected. For example the concept of *archiving* is suggested in a little more than half of the documents, while it is actually selected in roughly one third. The concept of *authoring* however has only a small amount of suggestions and no actual selections. The blue dots in the upper left corner of the concepts is a proportional measure on how many hits this concept had in the last matching, and this number is unaltered by the actual selections. The way of summarizing the matches and selections for a set of documents allows the users to view how the model-concepts reflects the text of the documents and can also be used to illustrate how well a model is able to discriminate documents in a collection.



Document info
Title: Conferencing | Filename: conferencing.html | Kategori:Data conferencing with external parties |
Owner:Arne Naustdal | Project/unit: null | Editor: Gro Wigestrand
Last modified by Gro Wigestrand, 03/15/2001 01:26:21 PM

Data conferencing with external parties (Nettm˜ter med eksterne kontakter

Stadig flere i Statoil samarbeider med eksterne akt˜rer (leverand˜rer og samarbeidspartnere), lokalisert ulike steder utenfor selskapet. St˜tte for nettm˜te ved hjelp av verkt˜y for skjermdeling, felles tavle og quot;chat quot; [illegible] en effektiv og kostnadsbesparende samarbeidsform for distribuert samarbeid. L˜sningen er i dag kun tilgjengelig internt i Statoil pga. tekniske og sikkerhetsrelaterte begrensninger ved verkt˜yene vi bruker.

Fra sommeren 2001 skal det vÊre like enkelt og sikkert Å gjennomf˜re nettm˜ter med eksterne akt˜rer som med akt˜rer internt i Statoil. Innen 2002 skal ogsÅ PC-basert audio- og videokonferanse vÊre integrert i l˜sningen.

Ansvarlig: Bj˜rn Tveite

More and more Statoil employees cooperate with external players (suppliers and cooperation partners), who are located at various places outside of the company. Support for data conferencing by means of tools for screen sharing, shared whiteboard and quot;chat quot; function is an effective and cost-saving method for distributed collaboration. Today, the solution is only available internally in Statoil, because of technical and security-related limitations in the tools we use.

(a) (b)

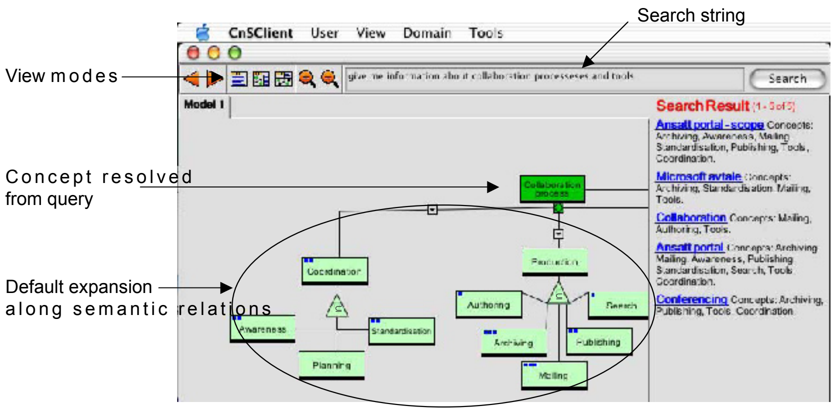
Fig. 6. Classfication of one document (a) and examining one documet text (b)

Figure 6a shows a user working on one single document. The coloring of the concepts follows the same rules as in the summary view, but with a single document, the suggestions are shown as a full triangle (archiving and publishing) and actual selections become a fully colored green rectangle (coordination). Users may accept and reject a single suggestion by clicking either on the green or white part of the concept respectively, or all suggestions at once by using the toolbar buttons. When examining classifications for one document, the user can examine the full text of the document by clicking on the document tab figure 6b.

Before saving the finished classifications, also the selected contextual meta-data attributes for the documents must be entered. Before storing the final classifications, the client performs a check of the selected document's classifications and provides warnings to the user according to some predefined rules. Currently the system will also signal if documents are classified according to only one concept or if a whole set of documents are all classified according to the exact same concepts.

4.2 Retrieval

In the retrieval mode, the user may start with a string representation of the query and then refine the query by interacting with the model. Search results are computed continuously during model interaction. Figure 7 illustrates the mapping of a search string to a model concept (collaboration process). By default, query expressions are expanded along the hierarchical abstraction mechanisms of the model (here aggregation and generalization) illustrated by a lighter shade of green the default expansion may be overridden simply by clicking on the abstraction symbol. Also in search mode, the blue dots are used to indicate the number of hits for a concept. They are shown also for concepts not included in the selection, in order to illustrate which concepts will return more documents in a subsequent refined search.



Search Result (1 - 5 of 5)

Ansatt portal - scope Concepts: Archiving, Awareness, Mailing Standardisation, Publishing, Tools, Coordination.

Microsoft avtale Concepts: Archiving, Standardisation, Mailing, Tools.

Collaboration Concepts: Mailing, Authoring, Tools.

Ansatt portal Concepts: Archiving Mailing, Awareness, Publishing Standardisation, Search, Tools Coordination.

Conferencing Concepts: Archiving, Publishing, Tools, Coordination.

Fig. 7. Retrieval: Interacting with the model to create a search expression

5 Related Work

We define our approach as a cooperative document management system. Shared or common information space systems in the area of CSCW, like BSCW [15], ICE [16] or FirstClass [17] mostly use (small) static contextual meta-data schemes, connecting documents to for example people, tasks or projects. In general these systems rely on free-hand descriptions of the semantics of documents. An interesting approach to semantic classification of text in a cooperative system is found in the ConceptIndex system [6], where concepts are defined by attaching them to phrases – or text fragments – from the documents. Concepts are only defined by their occurrences and the actual domain vocabulary is not visible in the approach. This is contrary to our use of conceptual modeling, which provides mechanisms for explicit definition of the domain model.

The idea that users putting information on the web should also describe the semantics of this information, is at the very heart of the Semantic web initiative [18][4]. Semantic web relies on the Resource Description Framework [19] a semantic network inspired language for constructing meta-data statements. Pure RDF does however not include a facility for specifying the vocabulary used in the meta-data statements. This is however under development through what is denoted RDF-Schemas [20] and there are interesting work on document query-facilities that exploits the information in the RDF-schema when querying for documents [21]. Also, pure RDF does not include mechanisms to ensure shared semantics of information. For this, RDF statements should be connected to Ontologies. Ontologies [22][23] are nowadays being collaboratively created [24] across the Web, and applied to search and classification of documents. Ontobroker [25] and Ontosaurus [26] allow users to search and also annotate HTML documents with "ontological information". Domain specific ontologies or thesauri are used to improve search-expressions. The medical domain calls for precise access to information, which is reflected in several terminological projects, such as [27][28][29][30]. A very interesting project is the “OntoQuery” [31] project which uses domain specific ontologies to define the semantics of natural language phrases and further as a basis for semantic document retrieval.

Most of the focus in ontologies for Semantic web has been to support computers – not humans – looking for information. The idea is to support software agents that can be used to enable intelligent services such as travel planning agents, or eCommerce shopping agents. Therefore, in contrast to conceptual modeling languages, the current advocated ontology languages, such as RDF-S [20] , DAML [32] and OIL [33] do not have a graphical visual representation, do not have graphical editors and case-like tool support and are not intended for direct user interaction. In conceptual modeling, the domain model is intended to support human communication, which requires a human-readable notation. In our approach, this is necessary to make the model directly applicable in the classification and search interface.Web search engines today increasingly use language technology to improve their search. Natural Language Analysis enables the use of phrases or semantic units, rather than “simple” words in retrieval. Phrase extraction from text has led to advances in IR and has been the basis for Linguistically Motivated Indexing - LMI – [34][35][36]. [37] provide a survey of NLP techniques and methods such as stemming, query expansion and word sense disambiguation to deal with morphological, lexical, syntactic and semantic variation when indexing text. An increasing number of search tools offer categorization as part of the search functionality. Posting a query, the user may also select one or more categories for the search. Each category is defined internally by means of a domain dictionary that contains the characteristic words and phrases in the category. The document category is included in the index, making it possible to restrict the search to one or more categories. The scientific search tool from Elsevier Science, Scirus [38] is a good example of this approach. The Northern Lighthouse web search engine applies automated generation of categories to let users refine their search expressions.

Like our model-based search approach, category-based search establishes links between documents and abstract semantic units. A category is an abstraction of related concepts, forms a semantically meaningful unit, and can be both part of a super-category and decomposed into sub-categories. Compared to our model approach, the categories tend to be at a higher level of abstraction. Even though you may specify a model describing a vast area like "scientific publications," you would usually use the model approach on fairly small, but well-defined domains. Category-based search and model-based search are in fact compatible, in the sense that domain models may be employed to describe the actual concepts defining the lowest-level categories. This exposes the concepts underlying the category to the user and allows the user to explore the whole category in a systematic way. Work on the connection between Natural language statements and formal statements of some kind in a CASE tool have been a challenge for many years [39]. Constructing models from a larger set of documents, however, the system needs more sophisticated techniques for handling linguistic variation when proposing model elements. [40][41] give examples of CASE tools that have integrated advanced parsing and understanding [39].

6 Concluding Remarks

This paper presented an information retrieval system that concentrates on concepts rather than on key words. Whereas traditional systems depend on syntactic matching of query terms with document key words, our system abstracts concepts from the documents and matches these against query concepts. Problems linked to word inflections, synonyms, phrasing and spell-checking are easier to handle in this framework. An additional feature of concept-based retrieval is that we do not see the words in isolation, but also take into account how words or concepts are semantically related to each other.

Also, the retrieval system represents a shift from traditional text-based retrieval interfaces to graphical model-based ones. Whereas text-based retrieval is fast for small, independent queries, the model-based approach allows the user to explore the structure of the domain while formulating his query. A more explorative retrieval method is made available, where the user is given feedback even before any documents are retrieved. In this way, the user learns more about the domain as a whole and can take the distribution of documents into account before posting his query.

It should be noted that we do not consider our retrieval system a replacement of traditional search engines The system presented here is not intended as a generalpurpose search engine. Domain models assume that the document collections address structured, well understood domains. For unstructured domains, the models would be too fragmented or unreliable to be of any help. Also, in spite of the abstraction mechanisms in modeling languages, domain models seem more effective in small domains than in large ones. For large, complex domains, the challenge is to come up with domain models that are small and easy to use for the end-users. We must be able to handle several views of a model, or even several models. In addition, it should be possible to focus the models into an area of interest (other than just zooming). This is most notable during search, when users want to specialize their search expressions by narrowing in on the interesting concepts. Our blue-dot symbols for concepts with hits, illustrates the concepts that may be used to narrow a search expression. We are currently experimenting with allowing the users to “dim away” concepts without hits and then to compress the remaining concepts by squeezing them closer together.

Acknowledgements

This work is performed at the Information Systems group at IDI, NTNU, directed by Professor Arne Sølvberg. The CnSClient application is implemented by Audun Steinholm, while most of the server side scripts is written by Arne Dag Fidjestøl.

References

1. Gulla, J. A. and T. Brasethvik (2001). A model driven ERP Environment with search facilities. Applications of Natural Language to Information Systems NLDB, 2001.
2. Poncia, G. and B. Pernici (1997). A methodology for the design of distributed web systems. CAISE*97, Barcelona, Spain, Springer Verlag, 1997.
3. German, D. M. and D. D. Cowan (1999). Formalizing the specification of Web applications. Advances in conceptual modeling (Workshop of ER'99), Paris, Springer Verlag, 1999.
4. W3CSW (2001). Semantic Web Initiavive, http://www.w3c.org/2001/sw/, Accessed:December 2001.
5. Fensel, D. (2001). The Ontoweb Project, http://www.ontoweb.org: http://www.ontoweb.org, Accessed:July 2001
6. Voss, A., K. Nakata, et al. (1999). Collaborative information management using concepts. 2nd International Workshop IIIS-99, Copenhague, DK, Postproceedings published by IGP, 1999.
7. Sulllivan, D. (2001). Document Warehousing and Text Mining, Wiley Computer Publishing.
8. Weibel, S. (1995). "Metadata - The foundations for resource descriptions." D-LIB Magazine(July 1995).
9. Miller (2001). Semantic web and digital libraries. European Conference on Research and Advanced Technology for Digital Libraries, Germany, 2001.

10. Svenonius, E. (2000). The intellectual foundation for organising information. Cambridge, MIT Press.
11. Bunge, M. (1998). The Philosophy of science - from problem to theory, revised edition, Transaction Publishers.
12. Sølvberg, A. (1998). Data and what they refer to. Conceptual modeling: Historical perspectives and future trends, In conjunction with 16th Int. Conf. on Conceptual modeling, Los Angeles, CA, USA, 1998.
13. Refedit (2000), "The referent model editor homepage", http://www.idi.ntnu.no/~ppp/referent/, Accessed: February 2002.
14. Pubgene (2001). The Pubgene database and tools, http://www.pubgene.org, Accessed:December 2001.
15. BSCW (1999). Basic Support for Cooperative Work on the WWW, http://bscw.gmd.de, Accessed:May 1999.
16. Farshchian, B. A. (1998). ICE: An object-oriented toolkit for tailoring collaborative Web-applications. IFIP WG8.1 Conference on Information Systems in the WWW Environment, Beijing, China., 1998.
17. FirstClass (1999). FirstClass Collaborative Classroom, http://www.schools.softarc.com/, Accessed: May, 1999
18. Berners-Lee, T., J. Hendler, et al. (2001). "The Semantic Web." Scientific Amerian(5).
19. W3CRDF (1998). Resource Description Framework - Working Draft, http://www.w3.org/Metadata/RDF/, Accessed:March 2000
20. W3CRDFS (2000). The RDF Schema Specification, http://www.w3.org/TR/2000/CR-rdf-schema-20000327/.
21. Karvounarakis, G., V. Christophides, et al. (2000). Querying semistructured (meta)data and schemas on the web: The case of RDF and RDFS, http://www.ics.forth.gr/proj/isst/RDF/rdfquerying.pdf, September 2000.
22. Gruber, T. (1995). "Towards Priciples for the Design of Ontologies used for Knowledge Sharing." Human and Computer Studies 43(5/6): 907-928.
23. Uschold, M. (1996). Building Ontologies: Towards a unified methodology. The 16th annual conference of the British Computer Society Specialist Group on Expert Systems, Cambridge (UK), 1996.
24. Domingue, J. (1998). Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. 11th Banff Knowledge Aquisition for Knowledge-based systems Workshop, Banff, Canada, 1998.
25. Fensel, D., J. Angele, et al. (1999). On2Broker: Improving access to information sources at the WWW: http://www.aifb.uni-karlsruhe.de/WBS/www-broker/o2/o2.pdf, Accessed: May, 1999.
26. Swartout, B., R. Patil, et al. (1996). Ontosaurus: A tool for browsing and editing ontologies. 9th Banff Knowledge Aquisition for KNowledge-based systems Workshop, Banff, Canada, 1996.
27. Galen (1999). Why Galen - The need for Integrated medical systems, http://www.galen-organisation.com/approach.html, Accessed:March 2000.
28. OMNI (1999). OMNI: Organizing Medical Networked Information, http://www.omni.ac.uk/, Accessed: May, 1999.

29. Soamares de Lima, L., A. H. F. Laender, et al. (1998). A Hierarchical Approach to the Automatic Categorization of Medical Documents. CIKM*98, Bethesda, USA, ACM, 1998.
30. Spriterm (1999). Spriterm - hälso och sjukvårdens gemensamma fakta och termdatabas, http://www.spri.se/i/Spriterm/i-prg2.htm, Accessed:March 2000.
31. Ontoquery (2001). The Ontoquery project - description, http://www.ontoquery.dk, Accessed:December 2001.
32. DAML, P. (2000). The DARPA Agent Markup Language Homepage, http://www.daml.org, Accessed:December 2001.
33. OIL (2001). Ontology Interface Layer, http://www.ontoknowledge.org/oil/, Accessed: December 2001.
34. Schneiderman, B., D. Byrd, et al. (1997). "Clarifying Search: A User-Interface Framework for Text Searches." D-Lib Magazine(January 1997).
35. Strzalkowski, T. (1999). Natural Language Information Retrieval, Kluwer Academic Publishers.
36. Strzalkowski, T., G. Stein, et al. (1998). Natural Language Information Retrieval - TREC-7 report. TREC-7, 1998.
37. Arampatzis, A. T., T. P. van der Weide, et al. (1999). Linguistically Motivated Information Retrieval, University of Nijmegen.
38. Scirus (2001). SCIRUS for scientific information only, http://www.scirus.com, Accessed:December 2001.
39. Métais, E. (1999). The role of knowledge and reasoning i CASE Tools, University of Versailles.
40. Fliedl, G., C. Kop, et al. (1997). NTS-based derivation of KCPM Perspective Determiners. 3rd Int. workshop on Applications of Natural Language to Information Systems (NLDB'97), Vancouver, Ca, 1997.
41. Tjoa, A. M. and L. Berger (1993). Transformation of Requirement Specifications Expressed in Natural Language into EER Model. 12th Int. conceference on Entity-Relation approach, 1993.