OntoNaviERP: Ontology-Supported Navigation in ERP Software Documentation

Martin Hepp[1,2] and Andreas Wechselberger[1]

[1] E-Business and Web Science Research Group, Bundeswehr University Munich
[1,2] Semantics in Business Information Systems Group, STI, University of Innsbruck
mhepp@computer.org, andreas.wechselberger@unibw.de

Abstract. The documentation of Enterprise Research Planning (ERP) systems is usually (1) extremely large and (2) combines various views from the business and the technical implementation perspective. Also, a very specific vocabulary has evolved, in particular in the SAP domain (e.g. SAP Solution Maps or SAP software module names). This vocabulary is not clearly mapped to business management terminology and concepts. It is a well-known problem in practice that searching in SAP ERP documentation is difficult, because it requires in-depth knowledge of a large and proprietary terminology. We propose to use ontologies and automatic annotation of such large HTML software documentation in order to improve the usability and accessibility, namely of ERP help files. In order to achieve that, we have developed an ontology and prototype for SAP ERP 6.0. Our approach integrates concepts and lexical resources from (1) business management terminology, (2) SAP business terminology, (3) SAP system terminology, and (4) Wordnet synsets. We use standard GATE/KIM technology to annotate SAP help documentation with respective references to our ontology. Eventually, our approach consolidates the knowledge contained in the SAP help functionality at a conceptual level. This allows users to express their queries using a terminology they are familiar with, e.g. referring to general management terms. Despite a widely automated ontology construction process and a simplistic annotation strategy with minimal human intervention, we experienced convincing results. For an average query linked to an action and a topic, our technology returns more than 3 relevant resources, while a naïve term-based search returns on average only about 0.2 relevant resources.

1 Navigation in ERP Software Documentation

ERP systems like SAP R/3, myERP, or ERP 6.0 are very complex software packages, which makes new users and experienced staff alike largely dependent on online help and other online documentation. At the same time, it is a software category of utmost commercial relevance. Now, due to the broad scope and amount of detail of ERP software, the associated documentation is mostly huge; and the online help and other parts of the documentation combine terminology from business management (e.g. „depreciation“), the various application domains (e.g. „ECR“ in the retail sector), and SAP-specific language. With regard to the latter, also, a very specific vocabulary has

A. Sheth et al. (Eds.): ISWC 2008, LNCS 5318, pp. 764–776, 2008.

evolved, in particular in the SAP domain (e.g. SAP Solution Maps or SAP module names). This vocabulary is not clearly mapped to business management terminology, as taught at schools and colleges.

In effect, search and navigation in ERP documentation is unsatisfying for many users, since they are unable to express a query using the terminology from their current context or professional background. Instead, they need to be familiar with the particular SAP terminology in order to describe what they are looking for; a skill that imposes a lot of friction on new employees using ERP software. Even for the vendors of respective software, it is extremely difficult to produce and maintain a consistent documentation, in particular due to synonyms and homonyms.

Semantic technology is obviously a promising technology for helping out. However, the enormous size of respective documentation, the ongoing evolution, and pressing business constraints render the creation of perfect ontologies and annotations unfeasible. The particular challenge lies in developing an approach that brings a substantial improvement at little cost, i.e., that minimizes the amount of human labor in the process.

For our evaluation, we have taken the data from a subset of the SAP Logistics branch of functionality, namely the SAP Level II View ("Business Blueprint") regarding the Material Master branch (called "SAP Library" - Material Master ("LO-MD-MM"[1])). The respective part of the documentation consists of only 144 HTML files with a total file size of 1.12 MB. Still, the total number of different words and word groups in this small part exceeds 20,000!

The remainder of the paper is structured as follows: In section 2 we explain the OntoNaviERP approach. In section 3 we summarize the conceptual model for our representation. In section 4 we describe the implementation work carried out. In section 5 we evaluate the technical contribution and discuss the approach in the light of related work. Section 6 summarizes the main points and concludes the paper.

2 OntoNaviERP Approach

Our overall idea is to (1) construct a consolidated set of ontologies covering the general business management domain, the SAP software and solutions domain, and particular industry branch or application domains; (2) integrating those ontologies at a conceptual level, (3) augmenting them with synonyms from Wordnet synsets and other resources, (4) developing a highly automated annotation strategy and infrastructure based on off-the-shelf GATE/KIM technology (see e.g. [1]), and (5) designing a suitable user interface. Figure 1 illustrates our approach.

The main competency question the system should support can be defined as follows:

CQ: *Which [document | part of a document] is relevant as [instruction | term definition | reference] for a software user who wants to [create | modify | retrieve | delete | carry out a certain business function on] a certain business object?*

[1] http://help.sap.com/saphelp_erp2005vp/helpdata/en/ff/516a6749d811d182b80000e829fbfe/frameset.htm

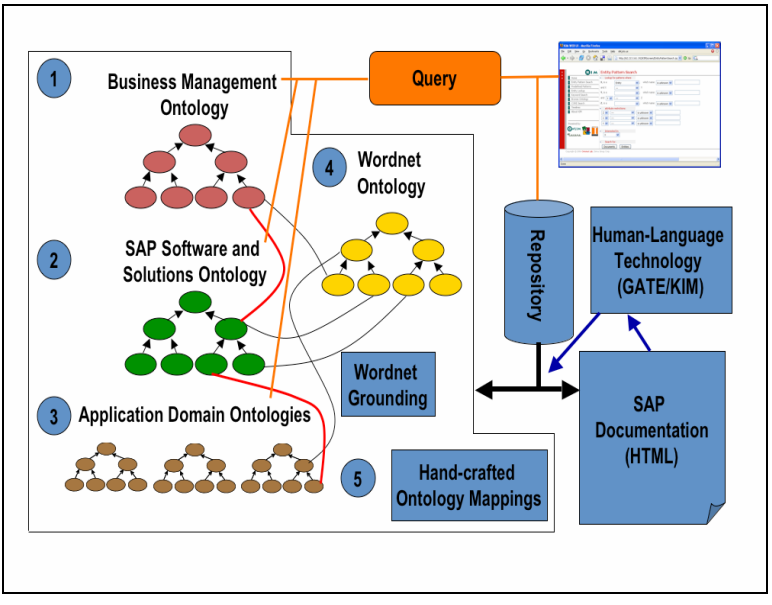

Fig. 1. OntoNaviERP Approach

Of course, the set of competency questions can be extended. However, we would like to stress that we are aiming at a cost-efficient solution that brings a substantial improvement over the state of the art. Given the huge size of both the corpus of text and the vocabulary, a more sophisticated approach is not per se more appropriate. In the long run, we also want to consider the individual usage context and the user's professional background and skills. However, a major problem in using the potential of such extensions is being able to capture respective data without imposing too much additional effort on users.

3 Conceptual Model

Our core conceptual model for supporting search in the SAP software documentation is as follows: First, we assume that a document or part of a document is characterized by (1) whether it offers instructions, explains terminology, or points to further references; (2) which type of action it describes on which type of business object (e.g. tangible or intangible resource or data set). For the type of documents, we use just four classes, a top-level class *TypeOfContent* and three subclasses *Instruction*, *Term Definition,* and *Reference*.

The topic covered by a document is for us always defined by a pair of an *Action* and a *Business Object*. This could for example be "create new client data set", "change ordering quantity", or "find supplier". Again, this may sound like a rather simple conceptual model, but we will see later that it is sufficient to bring substantial improvement.

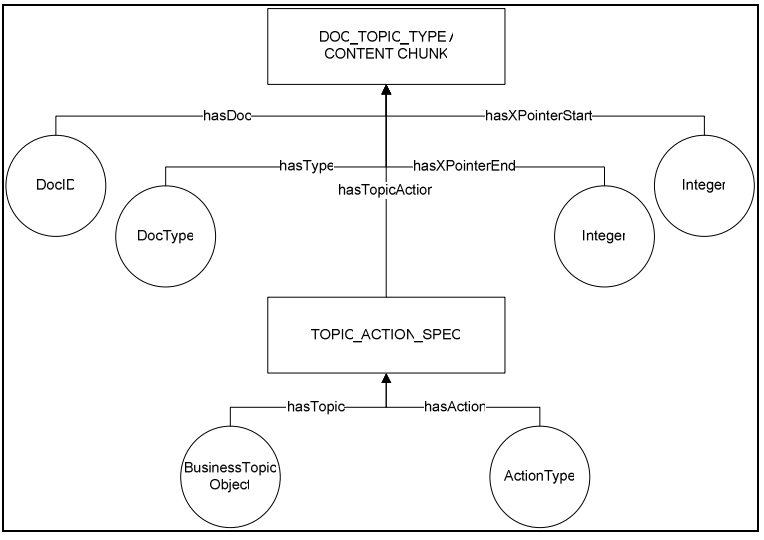

Fig. 2. Representing the type of actions and the business object

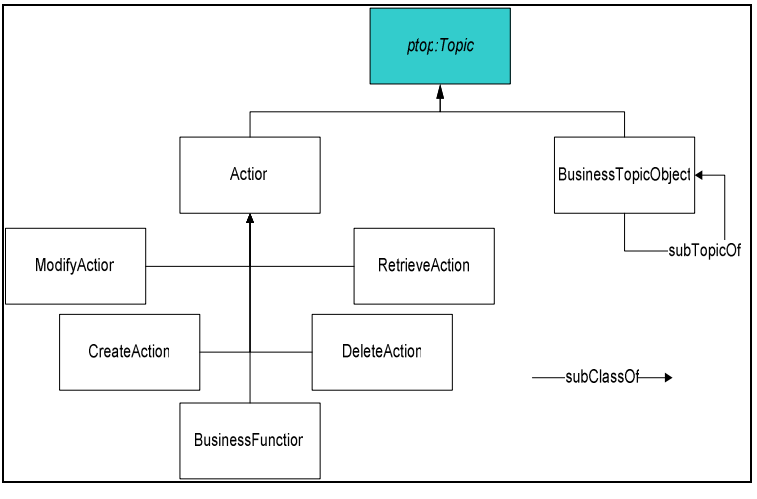

Fig. 3. User Actions and Business Topics as subclasses of Proton `ptop:Topic`

Also, we are dependent on a highly automated annotation process, for which lightweight structures are more promising. One key advantage of this conceptual model is that it reduces the natural-language analysis to spotting the occurrence of named entities representing actions or business objects, which works well with standard GATE/KIM technology without complex linguistic analysis. Figures 2 and 3 illustrate our approach.

4 Implementation: The OntoNaviERP Application

In this section, we describe the implementation of the OntoNaviERP prototype and summarize our experiences.

4.1 System Architecture

For the OntoNaviERP application, we use a very straightforward system architecture, based on mature, mainstream Semantic Web components. In detail, we use Sesame as a repository for the ontologies and the knowledge base, and GATE, KIM, and Lucene for the named entity recognition and other annotation tasks. For controlling the annotation we use KIM directly. For querying the knowledge base, we employ a dedicated GUI implemented as Java Server Pages which access the KIM API.

The KIM platform uses special ".nt" files as input for the named entity recognition. Among other details, they explicitly list the lexical variants of each named entity defined in the ontology. For creating these files from a given OWL ontology augmented by synonyms and other lexical variants, we developed a special converter application. Figure 4 shows the respective architecture.

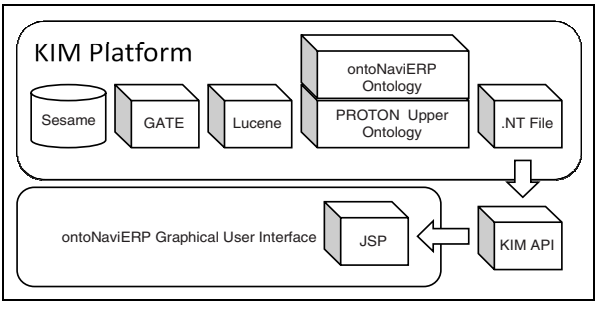

Fig. 4. OntoNaviERP Architecture

4.2 Ontology Engineering

For developing the respective ontologies, we had to meet the following requirements. First, the KIM/GATE infrastructure requires that the PROTON System Module must be present and imported in our own ontology, The `owl:Class` *Entity* must be the superclass of any proprietary ontology class that shall be considered by GATE/KIM for annotating resources. As for the exact location of a domain ontology in the KIM/GATE environment, there are three options: (1) it can be used instead of the PROTON Top Module, (2) instead of the PROTON Upper Module, or (3) in combination with the PROTON Upper Module. The KIM documentation[2] recommends using the domain-independent PROTON Top Module as the basis for any particular domain ontology, and we followed that advice. One positive side-effect of that choice is that the ontologically clean top-level branches "Abstract'', "Happening'', and

[2] http://www.ontotext.com/kim/doc/sys-doc/index.html

"Object'' force us to make good conceptual choices for all of our more specific elements.

For building the OntoNaviERP ontology, we used the following approach: For the *Action* and *Document Type* classes, we simply created respective elements using Protégé, and enriched them by suitable lexical variants of popular synonyms. For all synonyms, we defined a dedicated `owl:annotationProperty` that can later be used to derive the .nt files for KIM/GATE automatically using our tool.

For the *Business Objects* branch, we followed a straightforward approach:

(1) We used the SEO Studio Lite tool (Free Edition, version 2.0.4, build 3452), which is originally a tool for search-engine optimization for Web masters. For us, it returns tables with frequencies of occurrence for all single words or word combinations out of 2 or 3 words. This can be used to get a quick understanding of the active domain vocabulary. Since it was clear that we could not manually engineer an ontology that completely reflects the 20k+ words domain vocabulary, we ordered the resulting lists by descending frequency and considered all single words and 2- or 3-word groups that are used at least ten times in the total text corpus. That cut-of point was mainly determined by practical reasons, i.e., how much time we had available for building the ontology.

(2) We created a term cloud from the concurrency data in order to get a visual aid on the relative importance of certain terms. This step was not really needed, but was perceived a helpful cognitive aid during the ontology engineering process.

(3) Then, we generated a skeleton ontology based on all terms semi-automatically. We mainly applied a script to generate candidate concepts in OWL, consolidated similar concepts, and then manually made them specializations of the PROTON Top Module.

(4) As a last step, we used the Wordnet plug-in for Protégé to augment the concepts by synonyms and lexical variants. We store all synonyms in the ontology using a proprietary `owl:AnnotationProperty` *hasSynonyms*.

In a couple of days, we were able to produce a medium-size ontology for the SAP logistics domain that contains a large amount of synonyms and lexical variants for all entries. One must note that the lexical variants are only necessary because including a stemming engine in the current KIM/GATE package proved burdensome to us, which is why we discarded that option for the moment.

Table 1 summarizes the metrics of the resulting ontology. We can see from the ratio of all concept pairs (action + business objects) vs. the number of concept pairs occurring in at least one document that there is a good fit between the ontology and the document corpus - roughly 60 % of all possible conceptual combinations appear at least once.

We can also see that the 415 pairs of action and business objects on the conceptual level multiply to 27,500 term pairs at the language level, indicating the strength of the consolidation achieved by the ontology.

Note that the ontology needs only a few properties, because we just use very basic recognition of named entities.

Table 1. Metrics of the OntoNaviERP SAP Logistics Ontology

Classes		**132**
Conceptual level		
	Action classes	5
	Topic classes	127
	Concept pairs: All	635
	Concept pairs: Subset of pairs that appear in at least one document	415
	Subconcept pairs: Subset of pairs that appear in at least one document	268
Lexical level	Synonyms	383
	Synonyms for actions	126
	Synonyms for topics	257
	Term pairs: All	32382
	Term pairs: Subset for which the respective concept pairs appear in at least one document	27500
Properties	Total	3
	Object	2
	Datatype	0
	Annotation	1

4.3 Annotation

As for the annotation of the corpus itself, we employed the standard KIM/GATE package with existing JAPE rules; we did not modify the named entity recognition nor carried out a linguistic analysis. For putting it to work, we first had to derive an .nt Gazetteer List for the annotation and for the later search. For that, we used a small online tool based on the Jena Semantic Web Framework Java API. It takes as input any OWL ontology and creates from that an .nt file which includes instances of/for the concepts in the OWL ontology, and uses our *hasSynonyms* annotation property to build the *hasMainAlias, hasAlias* and the *subTopicOf* transitive property for the *subTopicOf* relation. We will make that tool available for other KIM/GATE users shortly, for we found it quite useful.

Then, we applied a two-stage annotation strategy: First, we used KIM/GATE to store links to all occurrences of known named entities in the repository. Second, we manually decided for each of the 144 HTML documents on the main content type (instruction, term definition, or reference). That took only minimal effort. As a future extension, one could make that distinction individually for action and business object pairs inside the documents.

For populating the KIM/GATE annotation, the following steps are necessary:

(1) Copy the *.owl and the *.nt files into the correct KIM directory. The correct directories are for the OWL files,

C:\…\kim-platform-1.7.12.15\context\default\kb\owl

and for the .nt files

C:\...\kim-platform-1.7.12.15\context\default\kb

(2) Edit the file "sesame.inmem.conf" so that our *.owl and *.nt files are included as imports. The file is at

C:\...\kim-platform-1.7.12.15\config

(3) Start KIM, Sesame and Tomcat, and populate the knowledge base. We used a one-time batch annotation run, since the HTML files are static. On-the-fly annotation would work, too, except for the manual step of classifying the type of document. Figure 5 shows the annotation step, and Figure 6 how the recognized entities are highlighted in the generic KIM interface.

Now, with our "brute-force" annotation strategy, we annotated all documents that contain a pair of action and business object anywhere in the text. Since we first thought that was too simple an approach, we added a filtering algorithm that considers a document relevant only if the two words representing the action and the business object respectively are within a range of +/- 25 words, as has been done in traditional information retrieval. However, this extension shows useful only for pure keyword search. As soon as we search at the conceptual level, the impact of that filter becomes limited. Instead, we use a ranking algorithm based on the distance and frequency of occurrence.

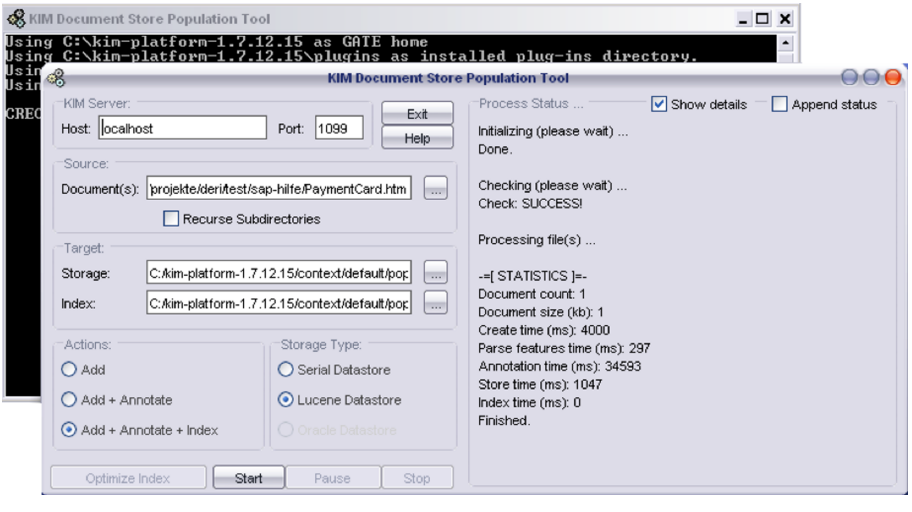

Fig. 5. KIM/GATE Annotation

Fig. 6. Recognized entities in the generic KIM interface

4.4 User Interface

We also developed a user interface that hides the ontology-based search behind user-friendly controls.

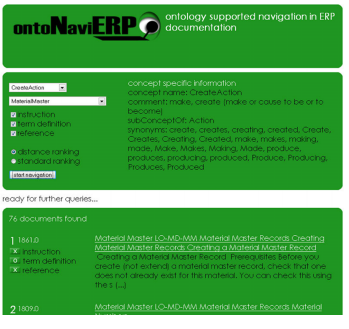

Fig. 7. OntoNaviERP User Interface

Users can check the types of documents they are interested in and specify the action and business objects. For each chosen conceptual element, all stored synonyms are displayed. Figure 7 shows the interface.

5 Evaluation and Discussion

In the following we summarize our evaluation of the technical contribution of our approach and compare it with the effort for ontology modeling and knowledge base population.

5.1 Contribution of Semantic Technology

From the subset of all term pairs for which the respective conceptual pairs occur at least in one single document, we drew a representative random sample of n=50 (with the random integer generator at http://www.random.org). Then, we determined the number of retrieved documents and the share of truly relevant documents from those documents for the following four techniques:

Technique 1: Number of documents containing both terms in its exact lexical form (we of course ignore capitalization, since that has been standard in keyword-based retrieval for decades).

Technique 2: Same as T1, but only those containing both terms within a 50-words range (25 words left and right)

Technique 3: Number of documents including either combination of a) the given topic term, its synonyms, its subconcepts, or the synonyms of the subconcepts and b) the given action term or its synonyms.

Technique 4: Same as T3, but only those documents containing the relevant named entities reflecting actions and business objects within a 50-words range.

So in short, techniques 1 and 2 represent the state of the art in simple keyword-based search in ERP documentation, and techniques 3 and 4 are the OntoNaviERP approach. Tables 2 and 3 summarize the results of our evaluation. Note that the precision for techniques 1 and 2 are based on very small return sets, since many search patterns do not appear in this exact lexical form.

Table 2. Impact of OntoNaviERP on retrieved documents and precision

	Technique 1: Term-based			**Technique 2: Term-based with 50 words range**			**Technique 3: OntonaviERP**			**Technique 4: OntoNaviERP with 50 words range**		
	Retrieved	Relevant(*)	Precision(*)	Retrieved	Relevant(*)	Precision(*)	Retrieved	Relevant(*)	Precision(*)	Retrieved	Relevant(*)	Precision(*)
Avg	**0.38**	**0.16**	*0.44*	**0.02**	**0.02**	*1.00*	**11.46**	**3.46**	**0.63**	**5.96**	**2.90**	**0.65**
Min	0.00	0.00	*0.00*	0.00	0.00	*1.00*	1.00	0.00	0.00	0.00	0.00	0.00
Max	6.00	3.00	*1.00*	1.00	1.00	*1.00*	72.00	10.00	1.00	50.00	10.00	1.00
Median	**0.00**	**0.00**	*0.42*	**0.00**	**0.00**	*1.00*	**5.50**	**2.00**	**0.55**	**3.00**	**1.00**	**0.80**
STD	1.10	0.55	*0.43*	0.14	0.14	*n/a*	14.93	3.04	0.33	8.62	3.03	0.37

* Of the first ten results retrieved

The results are very encouraging: Where the mean of retrieved documents in keyword-based search is only 0.38 documents per pair (Technique 1), the ontology-based search (Technique 3) returns, on average, more than 11 documents, and thus almost 30 times as many. Now, one would expect that the simple expansion of a search to synonyms and lexical variants, plus a small subsumption hierarchy would lead to a sharp decrease in precision. However, surprisingly, this is not the case. While the OntoNaviERP approach returns almost 30 times as many documents, more than 60% of the returned documents are relevant, as long as we only look at the top ten documents in our ordered result set. This is the more encouraging as we did not employ any tuning with regard to named entity disambiguation. In other words, the same synonym can be assigned to multiple concepts, and our simplistic annotation counts them for both if found. It seems that the homonyms among the terms are rarely used. There is for sure room for further improvement of the named entity recognition.

Table 3. Statistics on the number of additional, relevant documents

	Effectivity: Number of additional, relevant documents found by OntoNaviERP			
	Additional, relevant documents by technique 3	**Comparison: Relevant retrieved documents with technique 1**	**Additional, relevant documents by technique 4**	**Comparison: Relevant retrieved documents with technique 2**
	T3-T1	T1	T4-T2	T2
Avg	3.30	0.16	2.88	0.02
Min	0.00	0.00	0.00	0.00
Max	10.00	3.00	10.00	1.00
Median	2.00	0.00	1.00	0.00
STD	2.90	0.55	2.99	0.14

While the filtering based on the word distance increases precision in keyword-based search, it has minimal impact on the ontology-based search.

5.2 Discussion: Cost and Benefit

While the technical improvement alone is already very encouraging, it should also be judged in the light of the minimal, straightforward ontology engineering and annotation approach we use. As said, the annotation effort was limited to classifying 144 Web pages according to three branches (instruction, term definition, or reference), and running the out-of-the box named entity recognition of the KIM/GATE platform. Creating the ontology was basically extracting roughly 140 classes, assigning them to PROTON abstractions, and adding a lot of relevant synonyms and lexical variants. By using a dedicated `owl:AnnotationProperty`, such terms could be productively added and maintained directly in standard OWL editors like Protégé. The Gazeteer file was quickly generated from the OWL file using our lightweight conversion tool.

5.3 Related Work

While there is a mature body of literature on the core techniques, like named entity recognition, ontology-supported information retrieval, and ontology learning from text, we found no previous works that apply ontologies for ERP software documentation. This surprised us, because the blend of terminology from multiple spheres, e.g. college textbook general management terminology, vendor-specific business terminology, vendor specific systems terminology, and industry-branch terminology coexists wildly, both in the authoring processes and among the software users.

There is some work on deriving ontologies and populating knowledge bases from software documentation in general, e.g. annotations from APIs etc. Such particular work on ontology learning from software artifacts is described in [2]. The closest works in our direction from the Semantic Web community are [3] and [4], but while both address software documentation, they do not target large Common-Off-The-Shelf ERP packages like SAP solutions. For a general overview on ontology learning and population, see e.g. [5].

The KIM/GATE environment is described in [1]. A conceptual framework for the business process space, which is shaped and reflected by an ERP landscape, is presented in [6] and [7].

In another context, Holger Bast and colleagues [8-10] have worked on completing queries for facilitating search with their CompleteSearch approach; but again, this is not yet applied to ERP documentation. We are considering to using respective techniques for a more intelligent UI, though.

In information systems literature, the problem of modeling activity options for users has been discussed in [11], and the alignment of ERP software documentation and the system configuration has been addressed in [12]

6 Conclusion

We have shown how the navigation in ERP software documentation can be improved substantially by using standard KIM/GATE technology plus rather lightweight ontologies that are massively augmented by synonyms derived from frequent terms in the corpus and a standard Wordnet plug-in for Protégé. We obtained the relevant terminology using readily available search-engine optimization tools.

Despite a mostly automated ontology construction process and a simplistic annotation strategy with minimal human intervention, we experienced convincing results.

It comes as no surprise that ontologies can help improve precision and recall in a large body of text, in particular as long as the effort for creating the ontology and annotating the corpus are not considered and the corpus is stable. Both, however, is not given in ERP software documentation. The sheer size of the documentation and the used terminology makes manual ontology engineering and manual supervision of the annotation unattractive. Thus, we wanted to develop a pragmatic and cheap approach that relies on current semantic technology to tackle a real business problem. Eventually, we were surprised about the substantial improvement our solution shows. As next steps, we will work on more intelligent user interfaces and on trying to consider user skills and backgrounds, the context of a search task, and the customization status of the software for further improving our approach.

Acknowledgements. The work presented in this paper has been supported by a Young Researcher's Grant (Nachwuchsförderung 2005-2006) from the Leopold-Franzens-Universität Innsbruck, and by the European Commission under the project SUPER (FP6-026850). The authors would also like to thank Ontotext for great support with the KIM package.

References

[1] Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D., Goranov, M.: KIM - Semantic Annotation Platform. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870. Springer, Heidelberg (2003)

[2] Sabou, M.: Extracting Ontologies from Software Documentation: a Semi-Automatic Method and its Evaluation. In: Proceedings of the Workshop on Ontology Learning and Population, ECAI 2004, Valencia, Spain (2004)

[3] Ambrósio, A.P., Santos, D.C.d., Lucena, F.N.d., Silva, J.C.d.: Software Engineering Documentation: an Ontology-based Approach. In: Proceedings of the WebMedia & LA-Web 2004 Joint Conference, Ribeirão Preto-SP, Brazil (2004)
[4] Witte, R., Zhang, Y., Rilling, J.: Empowering Software Maintainers with Semantic Web Technologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519. Springer, Heidelberg (2007)
[5] Buitelaar, P., Cimiano, P., Magnini, B.: Ontology Learning from Text: Methods, Evaluation and Applications, vol. 123. IOS Press, Amsterdam, The Netherlands (2005)
[6] Hepp, M., Leymann, F., Domingue, J., Wahler, A., Fensel, D.: Semantic Business Process Management: A Vision Towards Using Semantic Web Services for Business Process Management. In: Proceedings of the IEEE International Conference on e-Business Engineering (ICEBE 2005), Beijing, China (2005)
[7] Hepp, M., Roman, D.: An Ontology Framework for Semantic Business Process Management. In: Proceedings of the 8th International Conference Wirtschaftsinformatik 2007, Karlsruhe (2007)
[8] Bast, H., Chitea, A., Suchanek, F., Weber, I.: ESTER: Efficient Search on Text, Entities, and Relations. In: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Amsterdam, The Netherlands (2007)
[9] Bast, H., Weber, I.: The CompleteSearch Engine: Interactive, Efficient, and Towards IR&DB Integration. In: Proceedings of CIDR 2007, Asilomar, CA, USA (2007)
[10] Bast, H., Majumdar, D., Weber, I.: Efficient Interactive Query Expansion with Complete Search. In: Proceedings of CIKM 2007, Lisboa, Portugal (2007)
[11] Soffer, P., Golany, B., Dori, D.: ERP modeling: a comprehensive approach. Information Systems 28, 673–690 (2003)
[12] Knackstedt, R., Winkelmann, A., Becker, J.: Dynamic Alignment of ERP Systems and their Documentations. An Approach for Documentation Quality Improvement. In: Proceedings of the Americas Conference on Information Systems (AMCIS 2007), Keystone, CO, USA (2007)