A Competence Knowledge Base System as Part of the Organizational Memory

Minghong Liao, Knut Hinkelmann, Andreas Abecker, and Michael Sintek

German Research Center for Artificial Intelligence (DFKI) GmbH
P.O.Box 2080, D-67608 Kaiserslautern, Germany
Phone:+49 631 205 3474, Fax:+49 631 205 3210
{liao,hinkelma,aabecker,sintek}@dfki.uni-kl.de

Abstract. Personal competences of experienced employees are the most important knowledge assets of knowledge-work oriented enterprises. Thus, it makes perfect sense to start IT support for enterprise knowledge management with a system that facilitates finding of appropriate contact persons for business tasks which require specific knowledge, experiences, or skills. We propose such a competence knowledge base system (CKBS) which builds upon an ontology-based model of competence fields, the use of which allows (i) comprehensive multi-criteria organization and queries for personal competences, (ii) complex heuristic inferences for finding knowledgeable persons in spite of vaguely specified information needs, and (iii) easy integration of the CKBS into an overall organizational memory information system.

1 Introduction

Enterprises are realizing how important it is to "know what they know" and be able to make maximum use of their knowledge. Especially in industrialized countries with highly educated but expensive employees, products and services must be outstanding in terms of innovation, flexibility, and creativity. A prerequisite for being able to face current and future challenges is the systematic management of the knowledge assets of the enterprise (see, e.g., [20]).

Despite the use of knowledge-based systems for decision support and expert task assistance, of groupware and workflow tools for improving communication and collaboration, and of advanced document management and artifact repositories for better reuse and experience documentation, the tacit knowledge, personal competences, and skills of experienced employees are still the most important resources for solving knowledge-intensive tasks like decision-making, strategic planning, or creative design.

Consequently, one of the first steps to support enterprise knowledge management is often to establish an electronically represented and accessible overview of people's special capabilities, experiences, and key knowledge areas (see, e.g., [1, 16], or Davenport's case study for Teltech [10]). Such a *competence knowledge base system* (CKBS, or: "smart" *company yellow pages*, *skill database*, etc.) can be used for project team formation, troubleshooting in help-desk situations, coaching and advice transfer from experienced employees to newcomers, and also for strategic analyses in order to support long-term skill development in the company.

Practical approaches for building such a system usually rely on textual descriptions of the employees' skills and capabilities, or, in advanced *knowledge mapping* approaches, rely on two-dimensional visualizations of the underlying information space [11]. In our approach, we essentially regard an employee as a "knowledge container" (in a similar way as a book or a multimedia document) such that a personal competence can be described just like the other (tangible) contents of an Organizational Memory Information System (OMIS, shortly OM) [17]. This makes possible sophisticated retrieval mechanisms for searching competent employees and a deeper integration of the competence part into the overall OM system (see also [5]).

This paper is organized as follows: After a more detailed description of the envisioned CKBS usage scenario (Section 2), we describe our ontology-based modeling approach which associates the company's employees with formal concepts of a domain ontology describing their respective competences (Section 3). This section also illustrates ontology-based retrieval heuristics, the more technical formulation of which is described in Section 4. A short discussion of implementation issues (Section 5) and some concluding remarks (Section 6) end up the paper.

2 Application Scenarios for a Competence Knowledge Base System (CKBS)

Looking for people with specific expertise is a common problem in nearly every company. A prototypical example is a help desk where customers call in the case of trouble with a company's products. Such help desks often support a wide range of products or multiple domains. Individuals gain experience and thus develop expertise in particular domains over time. Computer systems try to capture this expertise to make it independent of the availability of a particular human expert (see [12] and [14]). For hard problems, however, an expert—for instance one of the product developers—must still be consulted. In this case, the particular expert must be identified very quickly—a typical application for a CKBS.

Another typical application for a CKBS is forming a project team that must be composed of people with the right expertise for the different tasks of the project. Here, the retrieving problem in a CKBS is further complicated by the fact that in some cases it might be preferable to find people with as many expertises as possible (even if they are not so deep) in order to keep the project team small. For solving a very difficult task, however, it might be advantageous to find experts with high competences—even if the project becomes large.

A related problem is to react on inquiries coming from customers or from other departments in the company. If a prospective customer asks for a solution to a specific problem the company has to direct the inquiry to an expert who can discuss possible solutions with the customer and in the end can make an offer.

Identifying people with a specific expertise in a CKBS is in some aspects different from retrieving information, e.g., in a document management system. There, it is possible to search in the information directly, e.g., by full-text search. In a CKBS, not the competence itself but a description of it is represented. But simple classification or a keyword approach are not sufficient for a CKBS. The specific competences of different people often differ in fine but significant details. Therefore a modeling approach is required that allows to describe people's competences accurately and to implement a retrieval strategy that allows to find a "nearly fitting" expert or multiple experts that together have the desired competence when a specific expert for a particular problem is not available. In the rest of the paper we will sketch an ontology-driven approach for describing and retrieving the competences of people in a CKBS. The presented ideas are prototypically implemented as a DFKI intranet component (Figure 1 shows a screenshot of its user interface).

3 Ontology-Based Competence Modeling

A naive approach to build up a competence knowledge base would be to associate with each employee a number of index keywords describing his or her fields of knowledge; to search for employees competent in a given area would mean to simply

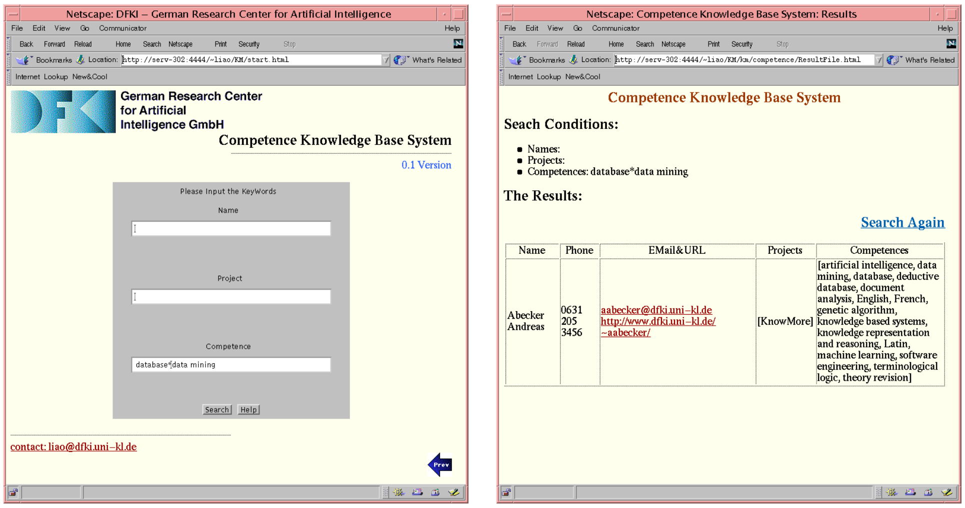

Fig. 1. Input and Output Window of the DFKI Competence Knowledge Base System

look up this table. This would be analogous to keyword-based information retrieval (IR) approaches, thus inheriting all problems of such an approach: keywords are not unequivocal; in a query situation the user often does not know exactly the keyword vocabulary; information needs are often vague or somehow orthogonal to the indexing vocabulary; keyword association is quite imprecise, etc.

In IR research, these problems led to a number of approaches aiming at, e.g., easier query formulation, automatic indexing, probabilistic matching between query terms and index keywords, incorporation of background knowledge into the retrieval process, and so on. One of the most powerful scenarios is defined by the *logic-based approach to IR* [18]: both queries and documents are represented in a formal, logic-based way, and finding a document which is likely to answer the query is understood as a process of logical inference. This view allows to have well-founded and theoretically understood retrieval mechanisms which can be supported by background knowledge represented as logical theories. On the other hand, we have the *conceptual IR approach* [21] which allows as index expressions only elements taken from a formal model of the domain of discourse structuring the concepts of this domain in an is-a/part-of hierarchy or something similar. If conceptual indexing and logic-based retrieval are combined, we can formulate generic as well as domain specific retrieval heuristics over the given domain model in order to support the retrieval inference [8].

In the KnowMore project, we take such a logic-based, conceptual IR approach for indexing and retrieval of the diverse sources of knowledge and information available in an enterprise. Heterogeneous sources are homogeneously described by formal information models. Information models are formulated with a vocabulary defined in some underlying ontologies on (i) logical structure and meta properties, (ii) creation and potential usage context, and (iii) semantic context [5, 6].

Following this idea, also personal (or, group) competences can be formalized as special kinds of "documents" mainly characterized by their owner and their content (i.e., the actual knowledge area). The advantage of this approach is that we can reuse algorithms and structuring ontologies from the overall KM/OM system and search/present competences and explicit (tangible) knowledge sources in an integrated manner. This comprehensive OM approach is still in its early stages and its description goes beyond the scope of this paper. Here, we will focus on some examples for competence retrieval with the help of ontology-based retrieval heuristics.

Example 1: Consider a part of a domain ontology describing the competences of the members of our DFKI research group shown in Figure 2. Suppose Tino to be our only employee, known as competent in the field of object-oriented databases. If we are now searching for a person competent in the field of databases, a simple keyword-based search would fail. However, if we can use the subsumption relationships between competence fields shown in Figure 2, together with a general search heuristics stating that people knowledgeable in a specific area should also be competent in the more general topics, we can expand our scope of search, thus finding Tino who will certainly have some knowledge in general database questions.

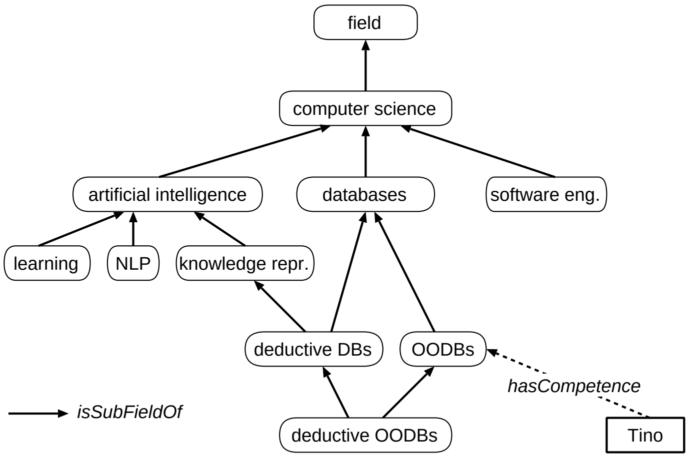

Fig. 2. Simple Ontology on Computer Science Competences

Example 2: In order to exemplify the use of a slightly more complicated search heuristics consider the following scenario. One source of knowledge which is readily available in every company describes which person works (or worked) in which project(s). If we make just the small extension of additionally representing which technologies were relevant in which projects, we can easily infer that people working in a project that deals with some technology are of course competent in this technology.

In Figure 3, we modified the above example accordingly by adding projects and the *worksIn* as well as the *usesTechnology* links in our model. If we are now looking for an expert in deductive, object-oriented databases (DOODB), we find the ESB project[1], and, via the *worksIn* link, we find Mike who is supposed to be competent in DOODB. If we also had to our disposal another general heuristic (similar to the one used in the prior example) which would state that someone competent in a more general area could also be knowledgeable in its specializations, we could again find Tino. However, this conclusion apparently is much more unsafe than the two other heuristics. Thus, it should be possible to formulate sort of a "cascading strategy" pursuing a sequence of search paths with decreasing certainty.

[1] A DFKI application project which employs DOODB technology for contextually-enriched recording of maintenance experiences for a complex coal mining machine [9].

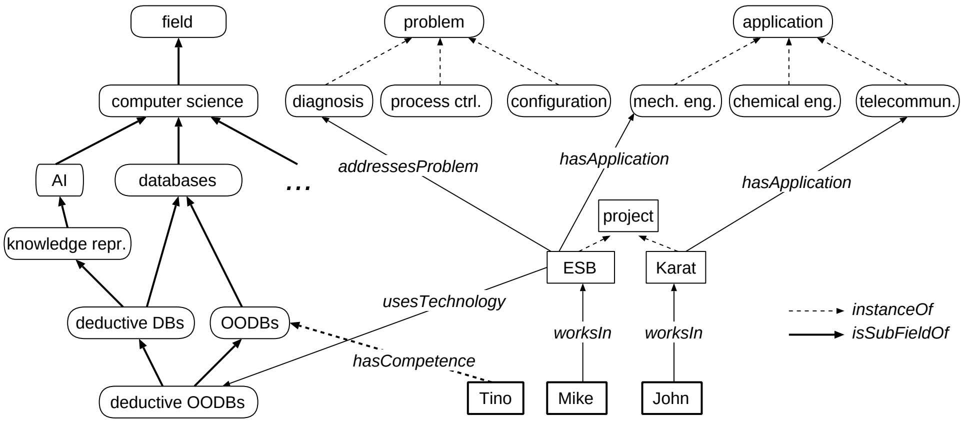

Fig. 3. Extended Ontology Plus Project Structure

In this example, we exploited "sparse" indices with the help of intelligent traversal heuristics. Of course, the same effect could have been achieved with simple retrieval methods and a "complete indexing", putting all knowledge into the indices that is now heuristically inferred at runtime. At first hand, the reasons for choosing our approach were solely pragmatic: in a real-world enterprise, we will always have to live with incomplete information, maintenance problems, and partly out-dated data. Under such circumstances, relying on a minimal extension—which must be put in and maintained—together with powerful completion methods seems to be the preferable approach. But, there is also a more fundamental reason: at least retrieval heuristics referring to the actual query context parameters can by no means reasonably be put into the index database. If, for instance, I want to retrieve people competent in a given area or any more general area which is still more specific than my own knowledge, this information can hardly be put into the indices in advance (for each possible querying employee).

Example 3: In the above two examples, one simple generalization/specialization hierarchy over topics accompanied by the project structure was sufficient for sophisticated competence retrieval. However, a natural, not biased approach to structuring our work did not lead to such a "single-hierarchy" view, but merely imposed a multiple-criteria structure on our work; the different (more or less) orthogonal dimensions were mainly glued together by projects which could be classified wrt. these different dimensions. The basic classification criteria identified were: (i) the *technological fields and approaches* adopted, (ii) the *generic problem class* tackled in a project, and (iii) the *domain of application* a project was situated in.

For instance, the ESB project mentioned above employed (i) DOODB, model-based information structuring, and case-based retrieval mechanisms for (ii) experience storage and diagnosis support in a (iii) mechanical engineering domain, namely coal mining machines [9].

We do not claim that this set of entity and relation types is a definite, optimal structure for organizing CKBSs. However, we have a strong feeling that such an organization—at least as a starting point—is quite intuitive in many company environments, that it is not necessarily more expensive to acquire than a "flat" keyword list for competence description, and that it offers much more benefits. Technically, a more complex structure requires more expressive knowledge representation means;[2] but it also offers more sophisticated means for heuristics formulation and more natural search interfaces.

Consider, for instance, a first customer contact by phone. Here, the application domain and the generic problem class are usually the first information pieces which can be acquired for further focusing the conversation. Depending on the actual topics, it might be useful to directly take into consultation a colleague experienced with the respective application domain—if this is so highly sophisticated that a further discussion already requires some domain knowledge, like, e.g., special areas in chemical industry. It is also often the case that "surprising links" are given by the system because IT experts usually mention only IT specific skills when describing their competence profile. However, via the project membership, we can again suspect that someone who was working in a project on fault diagnosis in a mechanical engineering domain has also some basic ideas on this mechanical engineering area. Since the links *between* independent classification dimensions (IT fields, projects, application domains, generic problem classes) use to be different—like *usesTechnology, addressesProblem*, etc.—from those *within* these dimensions—like *isSubFieldOf, isPartOf, isSubProjectOf, worksIn*, etc.—we can formulate much more exact search heuristics. For instance, if we want to group a brainstorming team for discussing the preferred approach for a project on configuration in electrical engineering, we could pick one member of each project that earlier tackled such a problem in a related domain, or we could select one expert for each technology used in such projects.

In the next section, we will give some more technical hints about how to formalize such retrieval heuristics as described above.

4 Ontology-Based Retrieval Heuristics

If we would do competence retrieval by hand, an intuitive way would be: take as input a graphical representation of the knowledge structures considered (illustrated as directed graphs in Figures 2 and 3), start with the given search items (i.e., marked nodes in the graph) and traverse this graph until a person is found which can be reached from the query items through a "reasonable" sequence of steps. In the examples of the previous section, the meaning of "reasonable sequence of steps" was illustrated by search heuristics describing graph traversal strategies which promise to lead from a competence field to a person which is supposed to know something about this field. Since we believe that such search heuristics are very much too domain and application specific to be formulated once forever and built into the system in a hard-wired manner, we propose a declarative heuristics specification formalism to be interpreted by the CKBS. A heuristics expression is a sequence of formulae of the following form:

$$f_1 \circ f_2 \circ \ldots \circ f_n$$

(denoting the functional composition of the f_i) with

$$f_i \equiv (\lambda)^\gamma$$

where λ is a link or an inverse link (written as $link^{-1}$) and γ is a "partial closure specification", i.e., one of the following path length specifications: n, $n..m$, $\geq n$, * (as abbreviation for ≥ 0), or + (as abbreviation for ≥ 1).

[2] In fact, the question how to optimally represent and exploit ontological information for IR and competence retrieval seems still an open question to us. A more detailed discussion goes beyond the scope of this paper, but is under work. Related problems are investigated, e.g., in [19].

Such a formula takes as input a set of nodes of the directed graph under consideration and, for each node, follows the links specified in the formula in right-to-left order, in each step delivering an intermediary set of nodes as starting point for the next step. "Partial closure" means repeatedly following the same link type (in the case of $\gamma \equiv *$ generating the reflexive and transitive closure of the relation denoted by that link in the ontology). A heuristics formula makes sense if it delivers only person nodes as result set. A sequence of formulae is evaluated in its sequential order with the semantics in mind that less trustworthy heuristics should be denoted last.

Example 1: The first example of the previous section can then be specified as follows:

1. $(hasCompetence^{-1})^1$
   "First search for people directly linked to a search concept."
2. $(hasCompetence^{-1})^1 \circ (isSubFieldOf^{-1})^+$
   "Then look for people competent in some subfield."

For the sake of clarity, we have denoted two formulae here. An alternative formulation would have been: $(hasCompetence^{-1})^1 \circ (isSubFieldOf^{-1})^*$

Example 2: The second example can be denoted as follows:

1. $(hasCompetence^{-1})^1$
   "First search for people directly linked to a search concept."
2. $(worksIn^{-1})^1 \circ (usesTechnology^{-1})^1$
   "Then look for people working in a project applying the technology in quest."
3. $(hasCompetence^{-1})^1 \circ (isSubFieldOf)^1$
   "Finally look for people experienced in the direct superconcept of the topic in quest."

These examples show how heuristics expressions can provide a declarative means for tailoring and tuning the CKBS retrieval engine. The heuristics described should not be understood as prescriptive and valid in all environments. They shall just illustrate how ontology-based search heuristics could look like and how they can be written down in an intuitive, declarative, yet expressive way. Exactly the fact that heuristics may differ significantly from domain to domain, from company to company, or application situation to application situation makes such a declarative and easily adaptable formalism interesting and appropriate.

We illustrated the language constructs needed for formulating our sample heuristics. Of course, there are also boolean connectives useful; but only further work and experiments in a fielded application will show what expressiveness is really necessary in application examples. For a finalized, full-fledged search heuristics language, one could imagine, e.g., some kind of quantification, qualitative path length restrictions, or expressions over query context parameters.

Such a heuristics language is certainly more intuitive and flexible than directly coding search heuristics into the implementation of the retrieval machinery. It is necessary because in our opinion, practical applications cannot be sufficiently solved by few general search heuristics like the ones proposed in most papers on ontology-based IR. Of course, it is still not very easy to use such a language for an end-user, but (i) in practice it will be used by employees at the application programmer level, and (ii) it would be an interesting idea to provide both a graphical browser interface and an automatic retrieval engine and try to automatically derive (e.g., by explanation-based learning) search heuristics from user's manual interaction. It should also be noted that explicitly encoded factual knowledge about people's competences is superior—*if available*. What we propose is to enhance—in a cascading search stratgey—the retrieval facilities of a system which must also be robust in a dynamic and incompletely modeled world.

5 Implementation

The CKBS is implemented with JAVA [2] which allows it to be used on all JAVA-enabled platforms like UNIX, Macintosh, and Windows without porting efforts. The CKBS is designed as a client-server model, its architecture is shown in Figure 4.

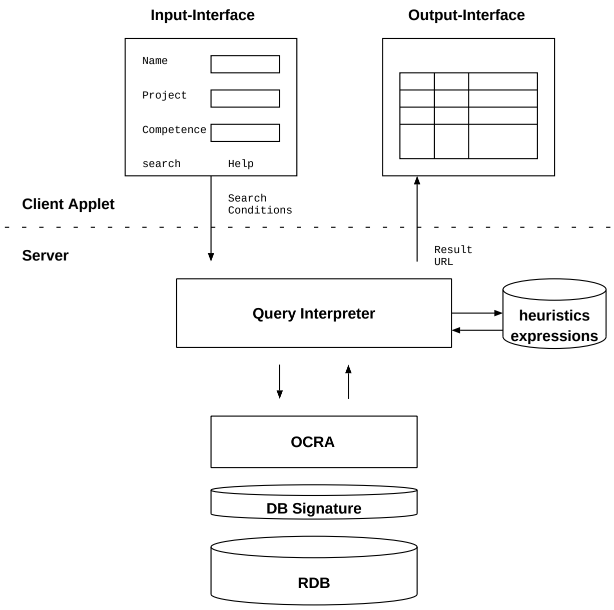

Fig. 4. The Architecture of CKBS

In the input mask (cf. Figures 1 and 4), the tool allows to formulate queries over competence fields, project memberships, or (directly) employee names. Complex queries can be composed using "AND", "OR", and "NOT".

The actual knowledge base with persons and their competence indices, as well as the ontological structure of competence fields, project membership, etc., are stored in a conventional relational database (RDB) which is coupled to the JAVA system code via JDBC [3]. Details about how to efficiently store and access these object-oriented knowledge structures within the relational paradigm can be found in [4]. The relational storage approach together with some additional schema information (denoted in the picture as DB-signature) allows to implement an object-centered relational algebra (OCRA, see [4]) which provides an object-oriented view and access methods with special (weak) deductive capabilities for the underlying data. In detail, the OCRA directly implements the above introduced "partial closure" operator, an essential part of heuristics expressions, which allows to efficiently follow a predefined number of links between objects.

Now, the heart of the CKBS retrieval machinery is a query interpreter which takes the user's complex query and maps it onto OCRA expressions, if needed, with the help of additional search heuristics. It goes beyond the scope of this application-oriented paper to sketch how complex queries (with boolean expressions) interact with retrieval heuristics to be translated into OCRA expressions.[3]

[3] In the current implementation, some parts of the system are much more "hard-wired" than described in the idealized view of Figure 4. However, in the near future we will

6 Conclusions

The study of organizational memories gains more and more interest although IT support is still in its premature stages. A main problem is the dichotomy of the general OM concept which can be understood either people oriented focusing on the intangible parts of the organizational knowledge assets, or document oriented focusing on explicit knowledge representations in documents and artifacts. This dichotomy is reflected, e.g., by the distinction between the process oriented and the product oriented view on Knowledge Management in [13]. It can also be found in many older works on the foundations of OM, see [7] for an extensive discussion of such categorial problems.

In this paper, we proposed a balanced approach which sees personal competences of employees as a "first-order knowledge source" in the OM which should be modeled and retrieved like other knowledge sources in the OM [5]. This requires, however, a sophisticated indexing and retrieval approach. While the semantical issues of well-understood ontology-based information modeling and retrieval are still an open question (see [19]), we were nevertheless able to implement a competence knowledge-base system fully operational in the DFKI Intranet, up to now containing the personal competences of the members of the Knowledge Management research group. Competences are described with respect to a domain ontology and accompanied by enterprise ontology information (project membership); complex queries are evaluated with the help of retrieval heuristics formulated over the structuring ontologies. We presented a first approach for declaratively specifying simple search heuristics and a generic architecture for processing them. Both will be further developed and specified in more detail if we have more practical experiences using the system to support our group internal customer care tasks.

Of course, there are many open questions which can only be clarified with practical experiences. The question how our approach scales up (with growing ontologies as well as large competence databases) seems not to be crucial. Technically, object-oriented databases and knowledge representation systems make rapid progress wrt. handling large ontological structures. Moreover, the number of personal competence profiles will always be neglectable compared with the document, data, and knowledge bases which must be managed in an OM anyway. A similar argument holds for the knowledge acquisition costs spent for ontology building. This is really a difficult problem if one wants to enable intelligent document and knowledge management in a company. However, if one wants to have this, one definitely has to build ontologies. So, these can also be used for competence management. From a user point of view, large ontologies can cognitively be better handled if there is a clear overall structure like in our approach.

Our system seems to be unique in that it treats documents and people in a similar way; other CKBS approaches use to be simple skill databases with full-text search facilities, or are designed for manual search and browsing. Within the ontology-based IR community (see, e.g., [15]) many approaches are designed for manual browsing, too, or propose only very simple ontological structures together with few generic search or query expansion heuristics. This seems too restrictive to us for real-world problems. So, we are working on the generic heuristics expression language for individual tailoring and tuning of ontology-based retrieval systems. Older work by Baudin *et al.* [8] already employed much more complicated retrieval heuristics than usual today, but only in a very specific domain and with more restricted ontological structuring mechanisms (the ontology contains only is-a and part-of links) than in our approach.

---

deliver a full specification of the heuristics expression language and its mapping into OCRA queries.

Minor extensions to our system in the near future will probably comprise an additional graphical presentation of the domain ontology in order to ease query term identification and a "fuzzy matching" facility for tolerating, e.g., small typos when searching for specific project or person names.

References

1. An interview with Tom Davenport and Larry Prusak, about their new book Working Knowledge: How Organizations Manage What They Know. Electronic Publication, http://www.brint.com/km/davenport/working.htm, 1998.
2. JAVA Hompage at SUN. http://java.sun.com/, 1998.
3. JDBC Homepage at SUN. http://java.sun.com/products/jdbc/, 1998.
4. A. Abecker, A. Bernardi, K. Hinkelmann, O. Kühn, and M. Sintek. Techniques for organizational memory information systems. DFKI Document D-98-02, February 1998.
5. A. Abecker, A. Bernardi, K. Hinkelmann, O. Kühn, and M. Sintek. Toward a technology for organizational memories. *IEEE Intelligent Systems*, May/June 1998.
6. A. Abecker, M. Sintek, and H. Wirtz. From hypermedia information retrieval to knowledge management in enterprises. In *IFMIP-98: First Int. Forum on Multimedia & Image Processing*, Anchorage, Alaska, May 1998. TSI Press.
7. L.J. Bannon and K. Kuutti. Shifting perspectives on organizational memory: From storage to active remembering. In *Proc. of the 29th IEEE HICSS, vol. III, Information Systems - Collaboration Systems and Technology*, pages 156–167. IEEE Computer Society Press, Washington, 1996.
8. C. Baudin, S. Kedar, and B. Pell. Increasing levels of assistance in refinement of knowledge-based retrieval systems. In G. Tecuci and Y. Kodratoff, editors, *Machine Learning and Knowledge Acquisition – Integrated Approaches*. Academic Press, 1995.
9. A. Bernardi, M. Sintek, and A. Abecker. Combining artificial intelligence, database technology, and hypermedia for intelligent fault recording. In *ISOMA-98: Sixth Int. Symp. on Manufacturing with Applications*, Anchorage, Alaska, May 1998. TSI Press.
10. Th.H. Davenport. Teltech: The business of knowledge management case study. Electronic Publication by Graduate School of Business, University of Texas at Austin, http://www.bus.utexas.edu/kman/telcase.htm, 1998.
11. M. Eppler. Knowledge Mapping. Eine Einführung in die Wissensvisualisierung. Presentation Slides, http://www.cck.uni-kl.de/wmk/, 1997. In German.
12. G. Kamp. Integrating semantic structure and technical documentation in case-based service support systems. In *Topics in Case-Based Reasoning – First European Workshop, EWCBR-93, Selected Papers*, number 837 in LNCS, pages 392–403. Springer-Verlag, June 1994.
13. O. Kühn and A. Abecker. Corporate memories for knowledge management in industrial practice: Prospects and challenges. In U.M. Borghoff and R. Pareschi, editors, *Information Technology for Knowledge Management*, pages 183–206. Springer-Verlag Berlin, Heidelberg, New York, 1998.
14. D. Logan and J. Kenyon. HELPDESK: Using AI to improve customer support. In A.C. Scott and Ph. Klahr, editors, *Innovative Applications of AI, IAAI-92*, pages 37–53, Menlo Park, Cambridge, London, 1992. AAAI Press / The MIT Press.
15. D.L. McGuiness. Ontological issues for knowledge-enhanced search. IOS Press, 1998. Int'l Conference on Formal Ontology in Information Systems (FOIS'98).
16. D. O'Leary. Knowledge management systems: Converting and connecting. *IEEE Intelligent Systems*, May/June 1998.
17. E.W. Stein and V. Zwass. Actualizing organizational memory with information technology. *Information Systems Research*, 6(2), 1995.
18. C.J. van Rijsbergen. Towards an information logic. In *Proc. of the 12th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 1989.
19. C. Welty. The ontological nature of subject taxonomies. IOS Press, 1998. Int'l Conference on Formal Ontology in Information Systems (FOIS'98).
20. K.M. Wiig. *Knowledge Management: Foundations.* Schema Press, Arlington, 1993.
21. W.A. Woods. Conceptual indexing: A better way to organize knowledge. Technical Report TR-97-61, SUN Microsystems Laboratories, Palo Alto, CA, USA, April 1997.