Knowledge and Business Processes: Approaching an Integration

Steffen Staab & Hans-Peter Schnurr
AIFB, Univ. Karlsruhe, D-76128 Karlsruhe, Germany
http://www.aifb.uni-karlsruhe.de/
{staab,schnurr}@aifb.uni-karlsruhe.de

This paper describes a principled approach towards creating an IT-support environment for knowledge workers. Starting in the analysis phase our approach paves the way for putting an intelligent assistant to work within a typical business process environment. Thereby, this support intelligence is centred around the interdependencies between documents providing the background knowledge and business processes.

1 Introduction

Support for the knowledge worker in her daily work may take many different guises: For instance, *business process management* and workflow technology cover the more rigid forms of work, such as processing standard customer contracts. *Groupware systems* further cooperation and communication, giving the knowledge worker the capabilities to access and disseminate knowledge. *Information access and retrieval* tools facilitate the user's interaction with different knowledge sources. However, what is largely missing so far is an environment that integrates the business process aspects of *weakly structured* knowledge work with an environment that *actively supports* the worker in using and adding to knowledge resources.

In addition, we must cope with three major technology problems apply to practical any knowledge management setting. First, distributed knowledge should be collected and implicit knowledge should be made explicit by inferencing. For instance, if one document states that person A was in project B and another document tells that project B was done for customer C then it may be interesting to know *explicitly* that person A knows something about customer C. Second, the knowledge worker should be able to concentrate on her core work and should not be distracted by learning new tools. Third, the knowledge worker should not be forced to work through extensive lists of information provided by an overly eager personal assistant. This means that the support provided must be concise.

The metaphor we have in mind is one about an intelligent assistant who looks over one's shoulder and answers questions one might have at a particular point of work. In order to avoid disconcerting dialogues, the assistant should actively propose reasonable questions and pre-fetch corresponding answers for the task at hand. Thereby, we assume that the knowledge repository is not one that is very structured, rather we presume that documents in common text processing formats are intermingled with graphical presentations, tabular information and some knowledge in databases. Hence, we claim plausibility of our approach as a real-world scenario for the information technology environment of common knowledge workers.

In this paper we want to show the design of a general methodology that paves the way for going from a common intranet and archive solution to the kind of IT support just outlined**.** Our approach is centred around office activities of typical knowledge workers, such as project planners, requirements

engineers or sales representative. Typically, this kind of work involves a large amount of background knowledge, the origins of which are distributed across the whole company as has also been sketched above. In the following we first draft a *project planning scenario* that will then serve as our running example in the rest of the paper. Subsequently, we will give an outline of major business structures that we take into account, *viz*. *intranet environments*, *business documents* and *business processes*. These structures are brought together in the section on our methodological approach. Finally, we give a description of an architecture of our approach and a review of related work.

2 Project Planning Scenario

Let us here consider a common project planning scenario. The setting is given as follows: a project manager in an IT-consulting firm has to compile a team - mostly from current employees of his company. Thereby, he has to meet the following planning requirements:

- Participants must be available for the project,
- participants should have particular technical knowledge that is needed for the project, and
- they should have some knowledge about the client in this project or at least about a client with a similar industry background.

Currently, there are several approaches to handle this problem. First possibility, the team is compiled from a set of people the manager knows by chance. This, however, means that the project starts with handicaps that could be avoided with a more appropriate team: *(i)*, effective techniques may not be applied in the most efficient way - or they may not applied at all; *(ii)*, the goal of the client is easily misunderstood, because a deeper understanding of his background is missing, etc. Second possibility, project listings circle in the company. This approach suffers from its incompletion, *i.e.* many of the most appropriate persons may miss to read the listing, as well as from inacceptable time lags between their initiation and their completion. Third possibility, all the information may be maintained by a human resource department, e.g. in a central database that may be used to retrieve the necessary informations. However, this approach requires significant overhead. Though project documentations and human readable employee descriptions are already available in forms of common documents, this information must also be kept up to date in the database.

For our scenario we have the following - realistic - assumptions: First assumption, the project manager compiles a project plan that she uses to estimate the man power and expertise she needs for the project, *e.g.* with a project planning software that supports the creation of *network plans*[1], common spreadsheet or text processing software. What is important at this point is that persons who execute this task on a regular basis usually hold on to a particular tool and a particular way of executing this task with this tool. For instance, it is quite common that a project manager creates a *template* (or uses a template that is provided by her company) in order to execute and document the planning task.

Second assumption, the information that she relies on is drawn from her personal knowledge, from the knowledge of people she asks, and from the knowledge available in other *project documents* and in the *intranet*. Naturally, the IT environment has no insight into the project manager’s mind and communications between people may only be exploited when it happens electronically. However,

1. Network plans are common means for describing and detecting dependencies between work tasks and availability of personnel.

the third type of knowledge that is available in the documents of the enterprise is the one that can always be accessed electronically and, thus, its the one we want to exploit for our knowledge support mechanisms. A common document describing a former project is depicted in Figure 1. Though the example is completely fictitious - like the client company - the structure is quite realistic.

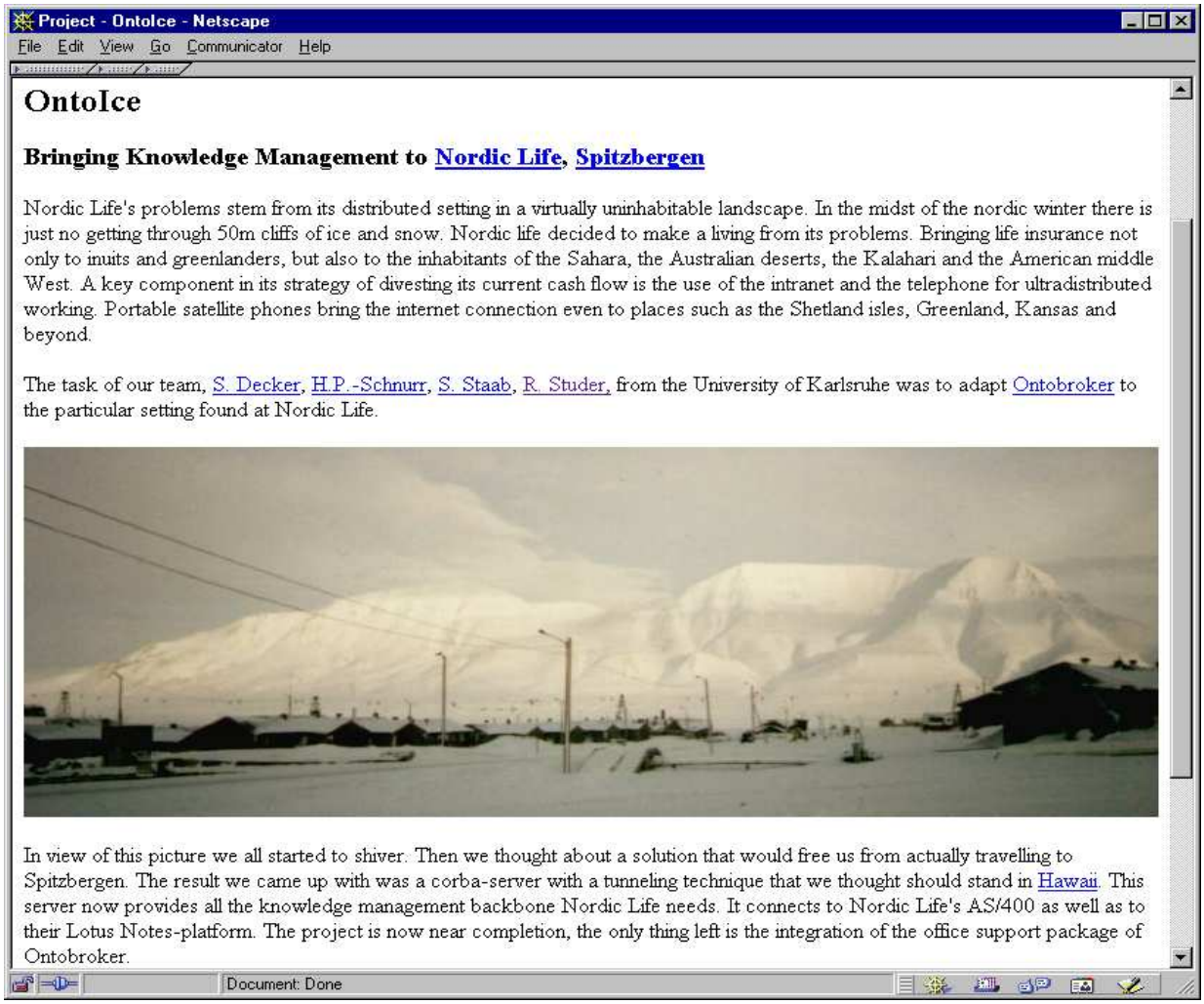

Figure 1: A fictitious example project page.

Let us now assume that a groupware platform exists that handles scheduling tasks, such as Lotus Notes™ or Netscape Professional™. The knowledge that is necessary in order to fulfill the three requirements mentioned at the beginning of this section can be found as follows:

- Availabilities may be retrieved from the scheduling database,
- technical knowledge may be found on employees' web pages
- industry specific knowledge may be inferred from employees' participations in projects at project web pages.

However, it is very tedious for the project manager to gather the information she needs from these different sources. Indeed, it may even be difficult to find these different information sources at all. The remainder of this paper will show how the knowledge for the project planning task can be

provided automatically, once the project planning task has been analysed and an appropriate methodology and IT support has been introduced to the enterprise.

3 Ontobroker as Intranet Environment

Typical intranet environments comprise at least two technical services. First, they offer means to store documents. In particular, current technology tends to make the boundary between file servers and intra-/internet servers, on the one hand, and web pages and files, on the other hand, disappear.[2] Hence, we may assume that all documents are available in the intranet. Second, intranet environments offer technology for navigating the intranet environment and finding information. While navigation is realized by common web clients, such as Netscape™, additional retrieval facilities are given by gathering and search tools like Verity™ or Shoe [Heflin et al., 1998].

Our KM methodology builds on the framework given through the *Ontobroker* approach as described in [Benjamins et al., 1998] and [Decker et al., 1999]. In order to provide a concise picture of our approach we give an overview of the main modules and capabilities of Ontobroker, before we will integrate these different building blocks into our KM support approach. The main components of Ontobroker from a functional view point are the underlying ontology, the annotated document sources and the inference and query engine. These are outlined in the following.

3.1 Ontology

The ontology is a specification of a shared conceptualization. As such it enables the communication of concepts between different participants of a discourse - or between a knowledge provider and a knowledge recipient. In our application the ontology provides the semantic basis on which we build for inferences as well as for accessing the facts. It comprises three different types of assertions (cf. Table 1).

Table 1: Partial ontology for human resource management.

Concepts	Attributes	Rules

Object[].
Person :: Object.
Employee :: Person.
Manager :: Employee.
Consultant :: Employee.
Project :: Object.
Company :: Object.
Manufacturer :: Company.
FinanceCompany :: Company.
Insurer :: FinanceCompany.
LifeInsurer :: Insurer.
Bank :: FinanceCompany.
	Person [
	firstName =>> String;
	lastName =>> String;
	email =>> String;
	phone =>> String;
	participantOf =>> Project;
	hasCompExperience
	=>>Company]
	
	Project[
	projectName =>> String;
	projectGoal =>> String;
	client =>> Company;
	member =>> Person;
	leader =>> Person]
		FORALL Proj1, Pers1
		Proj1 : Project
		[member ->> Pers1]
		<->
		Pers1 : Person
		[participantOf ->> Proj1]
		
		FORALL Pers1, Proj1, Comp1
		Proj1 : Project
		[member ->> Pers1,
		client ->> Comp1]
		->
		Pers1 : Person
		[hasCompExperience->> Comp1]

2. We only mention here solutions like Notes Domino™, Hyperwave™, and their likes.

First, a set of concepts that one talks about. Second, a set of attributes that link different concepts and also associate concepts with properties. Third, a set of rules that define the semantics of concepts and attributes. These rules are used to enforce a particular semantics, e.g. the symmetry of relations or the inheritance of concept attributes from super to subconcepts.

3.2 Annotated Document Sources

Ontobroker uses the ontology as a conceptual basis for concepts and relations it can handle and reason about. However, only the annotated document sources establish the linkage between abstract concepts and real facts. Hence, it may be described in the ontology what a project is in general, but the particular project ‘‘OntoIce’’ must be declared to be a project on the corresponding project page. Ontobroker allows for three major types of declarations. First, HTML^A is an extension of HTML that is compatible with all major HTML browsers. It allows for the formulation of knowledge facts. An HTML^A-enhanced web page editor has been developed that supports the creation of annotated documents. Second, RDF definitions provide a schema for meta-annotations that can also be digested by the Ontobroker gathering engine. Third and most important here, XML allows for the annotation of text with meta-tags. We consider that a particular set of XML-tags link text instances, *e.g.* projectNames, to concept and attribute definitions in the corresponding ontology. For instance, ‘‘OntoIce’’ is defined as the name of a project (cf. Figure 1 and Table 2).

Table 2: The XML project page.

```
<project>
        <projectName>OntoIce</projectname>
        <projectGoal>Bringing Knowledge Management to
                <client type=„LifeInsurer“>Nordic Life</client>, Spitzbergen
        </projectGoal>

Nordic Life's problems ...
...
The task of our team,
        <member>S. Decker</member>, <member> H.P.-Schnurr </member>,
        <member> S. Staab </member>, <leader> R. Studer </leader>, from the
        <university>University of Karlsruhe</university> was
...
</project>
```

3.3 Query and Inference Engine

A gatherer searches through the intranet documents, extracts all the facts stated in these document, *e.g.* that a project named ‘‘OntoIce’’ exists and that it has the members ‘‘S. Decker’’, and stores these facts in the knowledge base.

The query and inference engine then allows the retrieval of these facts from the knowledge base. In addition, it allows to infer implicit knowledge, *e.g.*, knowledge that cannot be found on a single web page. Table 3 compares the types of queries that may be used with different information access methods as well as the type of answers that may be derived from these queries. The semantics of the table needs to be understood as follows: Methods that are on the same height or below of an example query can be process this query. As becomes evident, the Ontobroker system allows for more

powerful queries than either of the other three methods, while it remains downward compatible. In addition, Ontobroker produces answers that are focused enough such that they may be of immediate help in the knowledge management scenario sketched above - unlike the keyword search techniques. The database approach already allows for semantic queries, indeed, deductive databases may even give the same range of answers as ontobroker[3]. However, the database approach would require that all the information were given in a rigidly schematized format and this requirement can hardly be fulfilled in any realistic knowledge management setting.

Table 3: Comparison of different retrieval methods.

Method	Example Query	Answer
Keyword Search	Find documents with keywords ‘‘OntoIce’’ and ‘‘Nordic Life’’.	Document with some highlighting
Thesaurus-supported Keyword Search	Find documents the text of which contains a Project and an Insurer.	Document with some highlighting
Database Queries	In which projects did Joe Doe participate?	Like ‘‘Joe Doe participated in P,Q, and R.’’
Ontobroker	Is there a consultant who has working experience with insurers?	Like ‘‘Joe Doe has working experience with the life insurers X, Y, and Z.’’

4 Business Documents as Knowledge Repository

Current business documents come in many different forms: letters, faxes, order forms, notifications, receipts, memos, private home pages, project home pages, etc. Nevertheless, it is quite common that these different forms are not chosen arbitrarily, but rather these forms are often standardized up to a certain point, indeed they often come with a particular semantics, such as the short notes that allow the reader to determine sender, reader, urgency, and further actions that need to be taken with a very short glimpse (e.g. check boxes for ‘‘please answer’’). Similarly, letters are usually not allowed to come in a completely free form, but they are usually composed with a particular corporate identity in mind. This corporate identity defines fonts, but it goes even further and pervades the way a company presents itself to the outside world.[4]

With our approach we go even one step further, since we also link these documents with corporate identity styles to the enterprise’s ontology (or ontologies). This leads us to annotated document templates, the and, thus, make them available as explicit knowledge repository. Beforehand, however, we want to introduce the technical platform that we rely on.

4.1 SGML/XML Documents as Technical Platform

SGML (Standard Generalized Markup Language, [ISO, 1986]) and a subset of it, XML (eXtensible Markup Language, [W3C, 1998a]), are standardization efforts that aim at a general scheme for exchanging documents. Given the widespread support among major computer software providers that XML has found recently[5], it is reasonable to assume that the structure of any business document

3. Note that F-Logic [Kifer et al., 1995] that Ontobroker builds on was originally conceived as a theoretical background for deductive databases.

4.

5. This also includes graphics formats, such as vector graphics (cf. [W3C, 1999] on SVG, scalable vector graphics).

will be accessible by way of XML annotation and query tools in the very near future. In our scenario, this is also of particular interest, because SGML/XML gives us the power to reason about document structures and contents as will be described further down in the paper.

4.2 Annotated Document Templates

Based on the considerations from the preceding sections, we have decided to build our approach on annotated document templates. Annotated document templates outline the general structure of a business document. For instance (cf. the left column of Table 4), a project plan may consist of text sections that define the author of the plan, the project participants, a network plan table that describes the dependencies betweeen participants, the single tasks, etc.

Table 4: The XML structure for a project plan.

<project> <author> </author> <plandate> </plandate> <participants> <member> </member> </participants> <networkplantable> </networkplantable> <tasks> <task> </task> </tasks> </project>	<project> <author> Joe Doe </author> <plandate> March, 15th, 1999 </plandate> <participants> <member> Joe Doe </member> <member> Hans-Peter Schnurr </member> <member> Steffen Staab </member> </participants> <networkplantable>here goes the table</networkplantable> <tasks> <task>Analysis of Nordic Life Business Processes</task> <task>Analysis of Nordic Life Organigram</task> </tasks> </project>

The (XML) annotations describe the semantics (and, possibly, some layout) of the document structures. When the user fills in parts of the document (cf. the right column of Table 4) in order to complete his business task, then she connects the information she provides with corresponding metainformation.

This approach is not only viable for completely new plans. One can also build on successful previous plans that just need some restructuring in order to make them viable for the new tasks. Hence, reuse is supported just like the completion of new plans - which is important in the project planning scenario, since, previous experiences are often used to estimate costs for new projects. In spite of this structuring, the templates leave a lot of leeway for user adaptations that may become necessary to describe the plan any further, *e.g.* with text, tables or even figures.

In order to provide a thorough use of document templates in an enterprise, similar measures have to be taken as for introducing a corporate identity, though at a greater, and therefore more laborious, level of detail.

In the next section we will discuss how these document templates can be connected to the business process analysis and the support function for the knowledge worker.

5 Business Processes

In this section, we consider some related work on business processes with a (comparatively) rigid structure that we build upon. This work adopts a formal *high-level Petri net* point of view.

Particularly interesting for our approach is the use of document contents for controling the business process in a special version of Petri nets, *viz*. so-called SGML nets.

5.1 Rigidly Structured Business Processes and SGML nets

Petri nets are a well-known, well-founded formal notation to model the behavior of a dynamic system. Due to the possibility of visualizing the dynamic aspects of business processes as well as changing the level of abstraction, they constitute one of the major formal methods for modeling business processes[6] (*cf.* [van der Aalst, 1998], [Weitz, 1998]). The basic Petri net concept is represented by a triple *N = (P,T,F)*, where *P* is the set of places (passive elements) and *T* the set of transitions (active elements). A place (represented by circles in Figure 2) may be interpreted as an object store and a transition (rectangles in Figure 2) specifies a class of operations on its adjacent places. The *flow relation F* is a set of directed arcs leading either from a place to a transition, constituting an input arc, or vice versa, constituting an output arc.

A *marking* of a net *N* assigns a set of tokens to every place in N. It can be interpreted as a global system state. When a transition occurs, tokens are removed from the places connected to this transition via the input arcs, input places, and inserted into the places connected to this transition via the output arcs, output places. Thus, for each transition pre- and postconditions can be formulated and, hence, sequences, alternatives, concurrency and iteration can be modeled.

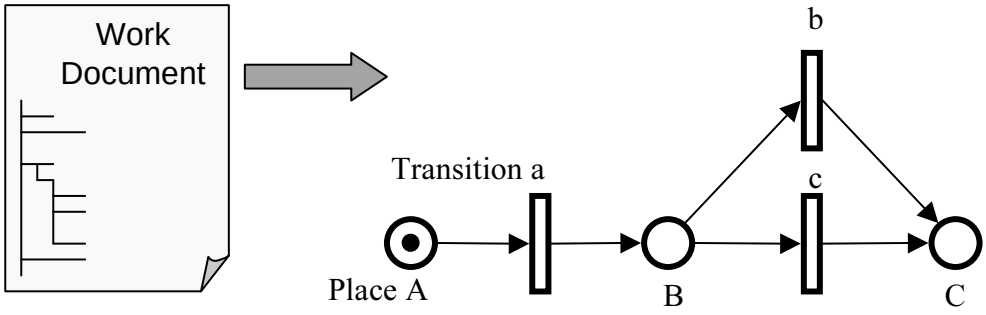

Figure 2: Workflow Defined by a Petri Net Model.

An SGML net (cf. [Weitz, 1998]) comprises the basic concepts of Petri nets, but instead of just a flow of atomic markers it also allows the modeling of a flow of SGML documents. In addition to simple transition rules that delete markers from input places and insert them into the output places of a transition, the transition rules perform operations on the SGML documents, such as checking off a text passage or iterating over a list. These kind of transition rules are defined by way of match operators that make use of the structure provided by SGML.

Still, the way SGML nets are defined only captures the (comparatively) rigid parts of the workflow. From the system's point of view the less rigid parts are equivalent to transitions with non-deterministic, since exogeneously triggered, operations. The selection of an appropriate transition operation is delegated to the user. In order to support her decision making, we want to present

6. Informal methods for workflow models include, *e.g.*, data flow diagrams (*cf.* [Georgakopoulos et al., 1995] for a survey on methods and tools).

appropriate knowledge from the document repository of the enterprise. This purpose is realized by *context-based views* that use pattern-matching on XML-structures.

5.2 Context-based Views

The observation that we build on is that knowledge work, though often unstructured, still involves a large amount of subtasks and document parts the structure of which is stable over time. We analyse these structures with the goal of finding the interdependencies between the *process*-oriented subtasks and the *document part*-based representation. The underlying idea is that the documents themself describe the progress that has been made towards achieving the goal and, thus, indicate the knowledge that may be of help for the worker - similarly as SGML structures determine the transitions that may occur at a given state of the system.

For example, a project plan may be considered complete when the initially stated subgoals that must be achieved are checked off. During the task of compiling a team the worker wants to get some support with finding the appropriate team members. Similarly, a product specification may be complete when there are no documents left with ‘‘dangling links’’. A listing of documents that one could reasonably refer to, *e.g.*, due to type constraints could be of utmost help.

An example is illustrated in Figure 3. The current work document(s) represent the task that must be performed, *e.g.* a project plan needs to be compiled. This work involves communicating with prospective project participants. The problem often lies in identifying appropriate participants and in establishing the communication link. As for the first, our methodology allows the establishment of blueprint questions like ‘‘Which person in the company knows about X and has capacity for projects as of Y?’’ (cf. Table 3). As for the second, given that a person is identified, the fax number(s) or e-mail adresses may be retrieved simply by given the name information. Thereby, it is not required that the fax number is stated in a corresponding database entry. Rather it may be given in the signature of a mail in the general accessible mail archives, or on a home page or only indirectly: via the group that this person belongs to.

5.3 An Integration of Context-Based Views and SGML nets

In our model, the state of the system is given by the combination of a state variable (corresponding to places in the SGML net) and implicitly through the way SGML queries match certain semantic predicates[7]. Depending on this state, the system may either initiate a new task (a transition in the SGML net) or it may display context information, like the fax number of an addressee.

The kind of help that could be useful is defined by the state that the document is in and by the task which is executed on it. Thereby, tasks and states are interrelated. The states determine whether a task may be executed and the executed tasks (together with the intitial states) determine the state of the document. It is common to annotate transitions with logic predicates that define when a transition may be executed. In addition, we define logic predicates at transitions and at places that define context-based views.

This process and knowledge management integration will be implemented in F-Logic [Kifer et al., 1995]. F-Logic rules describe what help to offer depending on the facts that are stated in the document and the state that is currently reached by the document. By this way a flexible, declarative

7. We define these pattern matches in analogy to the XSL specification draft [W3C, 1998b]. However, for better readability we here use a pseudo code style in our example of Figure 4.

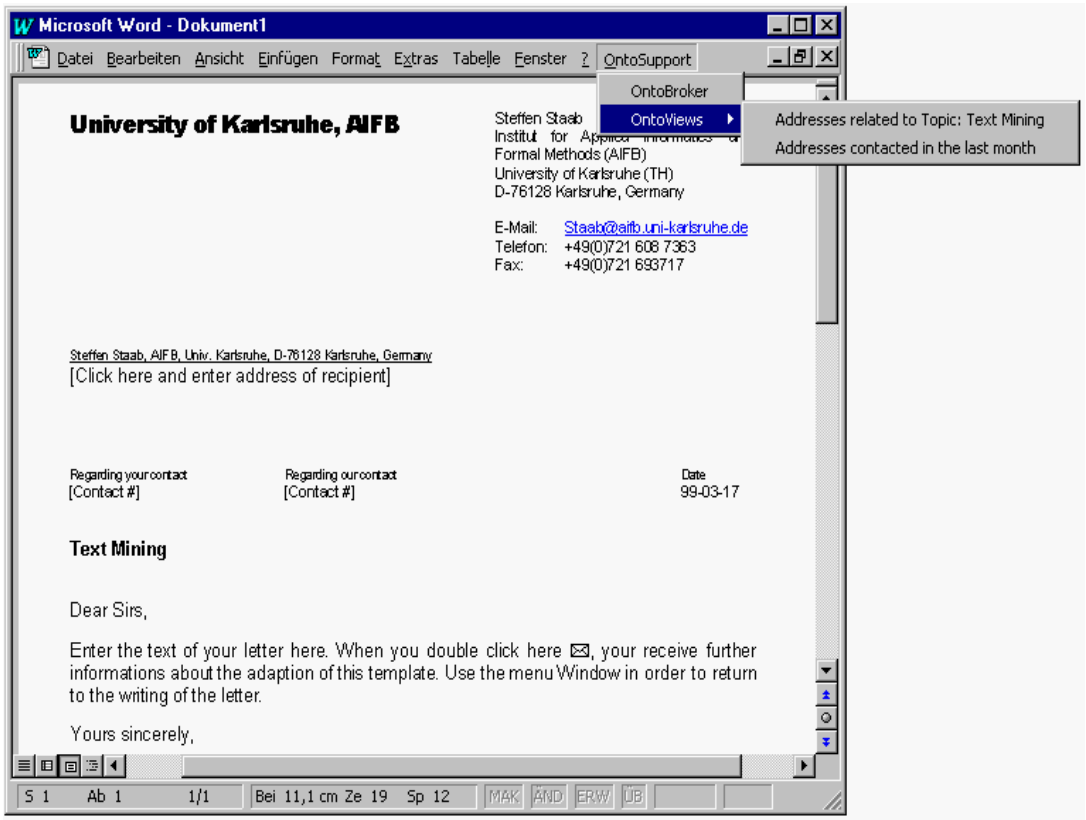

Figure 3: A knowledge worker's daily IT-environment.

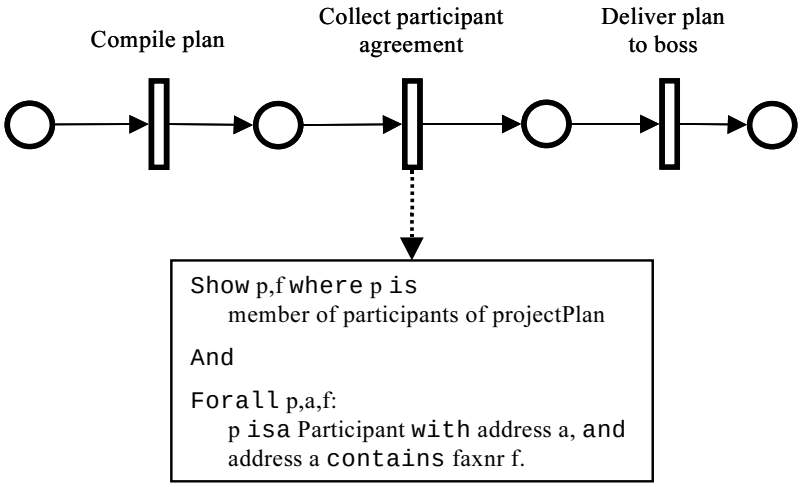

Figure 4: Integrating workflow and context-based views.

mechanism is exploited that might be easier to maintain than hand-crafted imperative code. In addition, we rely on a methodology already used for the inference engine and, thus, avoid an abundance of technologies.

For the future, we think of four major extensions: First, plan recognition techniques (e.g., Bayesian Networks [Charniak & Goldman, 1993]) could try to determine the overall task that is currently executed in order to help the user in completing this task. Second, planning techniques could

determine whether the current state in combination with the tasks executed so far still allows for a viable prosecution of the business process [McAllester & Rosenblitt, 1991]. Finally, learning techniques could observe persons at work and learn automatically how to help them (Comparably to [Barnekow & Staab, 1999]).

6 Methodology

With our methodology we approach an integration of the knowledge management and business process aspects described in the preceding sections. As is widespread practice, we distinguish the phases of building and using the IT support environment (cf. Figure 5). Furthermore, we divide the first phase - as is also common - into analysis and design.

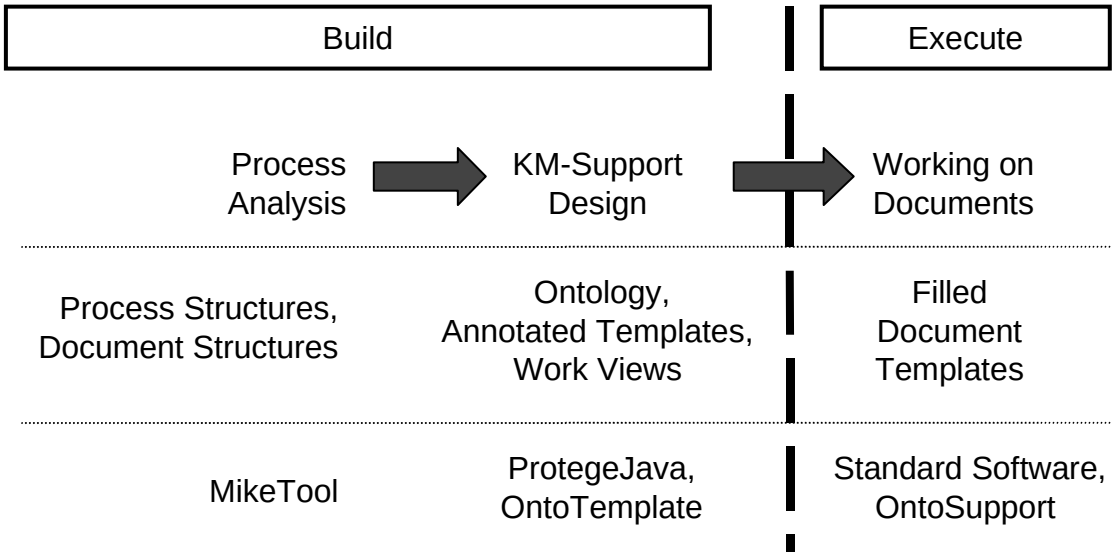

Figure 5: The methodological dimensions of our KM approach.

During the knowledge management preparation phase (build), existing business structures, i.e. processes and documents, are analysed. The results are used to design the knowledge management support structures. During the execution phase these structures are given life and, by the way of everyday work, the knowledge repository fills up. These three phases are now elaborated in this section.

6.1 Process and Document Analysis

In this first step of the build phase, we focus the analysis on processes which should be supported by the Knowledge Management system and on documents, playing a key role in the business process and the knowledge process. Keeping in mind that weakly structured processes are in our scope, a modeling approach such as applied in common workflow management systems is too rigid and therefore not applicable to a knowledge management setting. Nevertheless, our assumption is, that knowledge work, though often unstructured, still involves a large amount of subtasks parts of which are stable over time.

Our approach employs a division of processes into tasks and subtasks. The context is analysed and typical questions one might have at a particular point of work are compiled. Thus, one may find the

interdependencies between the subtasks and the document-part-based representation. The analysis of these subtask and document structures is supported with the MIKE tool [Steinberg, 1999]. MIKE (Model-based and Incremental Knowledge Engineering) [Angele et al., 1998] is an approach for developing knowledge-based systems. MIKE integrates semiformal and formal specification techniques together with prototyping into a coherent framework. All activities in the building process of a knowledge-based system are embedded in a cyclic process model. For the semiformal representation we use a hypermedia-like formalism which serves as a communication basis between knowledge worker and knowledge engineer during knowledge acquisition. The semiformal knowledge representation is also the basis for formalization, resulting in a formal and executable model, specified in the Knowledge Acquisition and Representation Language (KARL). Since KARL is executable, the model of expertise can be developed and validated by prototyping. A smooth transition from a semiformal to a formal specification and further on to design is achieved because all the description techniques rely on the same conceptual model to describe the functional and non-functional aspects of the system.

6.2 KM-Support Design

In the second step of the build phase, we design the Knowledge Management support system. This support system contains a domain ontology describing the content of the documents, a context-model with an overview of context-based views and samples of annotated templates (cf. the basic dimensions of ontological modeling: enterprise, information and domain ontology in [Abecker et al., 1998]. These knowledge descriptions are generated according to the task and document structures in the process analysis phase. The transition of the formal specification defined with the MIKE tool in the first step of the build phase into annotated templates, context-based views and domain ontologies in the second build phase is supported with ProtegeJava [Eriksson et al., 1995] and OntoTemplate. ProtegeJava is a suite of tools to model a domain. It contains a graphical editor to model ontologies. The formal specification, *viz.* the process and the document structures, acquired with the MIKE tool in the process analysis phase, constitutes the input to ProtegeJava. By this way one may smoothly integrate existing ontologies with the new domain model or integrate the new domain model into upper ontologies. For the future we want to extend ProtegeJava such that the creation of rule descriptions and rule hierarchies is also supported. In a final step, we want to add facilities for the definition of work views.

6.3 Execution Phase

As already indicated people use annotated templates that are then filled during the course of regular work. The user may add further annotations if she wants to, but is not obliged to do so. The environment is embedded in common standard software that may become even more accessible for the programmer of such environments, since XML has gained widespread popularity among major suppliers of standard office software. The knowledge management environment we provide is geared towards natural and convient use by the knowledge worker. In particular, this environment realizes one of the major ideas of knowledge management, *viz.* that an IT tool may only act as a *facilitator* for sharing, creating or retrieving knowledge, but never as a key player in creating, evaluating or contributing knowledge. Hence, the use of our support environment should never be made obligatory. Rather, the intention is the provision of active help for the user by push technologies - *without annoying the user.*

7 IT Architecture

The IT architecture of our approach is summarized in Figure 6[8]. On the right hand side one finds the On2broker parts[9] that form the backbone of our knowledge management system. We here only describe the coarse modules, detailed descriptions may be found in [Fensel et al., 1999] and [Decker et al., 1999].

- *Archive*: The annotations in the documents assert facts that constitute the available background knowledge. A typical archive is an intranet environment as introduced in Section 3.
- *Crawler*: The crawler gathers these facts from the archive and deposits them in the database.
- *Database*: The database serves for efficient retrieval and storage of facts.
- *Inference engine*: The inference engine draws conclusion based on the facts contained in the database. Its inferences are initiated through queries. Depending on the queries, rules in the ontology are used to extend the set of facts - which are then also reflected by an enlarged fact database.

The extension we show in this paper is depicted mainly on the left side of Figure 6:

- *Annotated Templates*: A set of annotated templates facilitates the creation of facts by the knowledge worker. Most annotations need not be given explicitly, but happen as a ‘‘side effect’’ of regular work. At this point it becomes crucial to support common tools, such as standard text processing, spreadsheet or presentation software. This support function will probably become easier to implement, when software companies adapt XML as a strategic data interchange platform, which seems to become the case right now.
- *Process Control*: The process control narrows down the context of the current tasks. It helps to automate rigid process parts of a knowledge management scenario[10]and enhances the choice of the appropriate knowledge management support mechanism.
- *Context-Based Views*: In particular, the process control allows a more selective choice of context-based views that give the knowledge worker additional insights into the facts contained in the archived documents.

Our whole approach is based on an ontology as its underlying skeleton. The ontology establishes a common vocabulary and enforces semantic relationships between concepts and, thus, provides the means for communication between the different modules.

---

8. Note that direct interaction between work document and database as depicted in Figure 6 may happen, e.g. if the work document is a database, but this will not happen very often.
9. On2broker is an improved version of Ontobroker (cf. [Fensel et al., 1999]).
10. This automation may even include common workflow mechanisms.

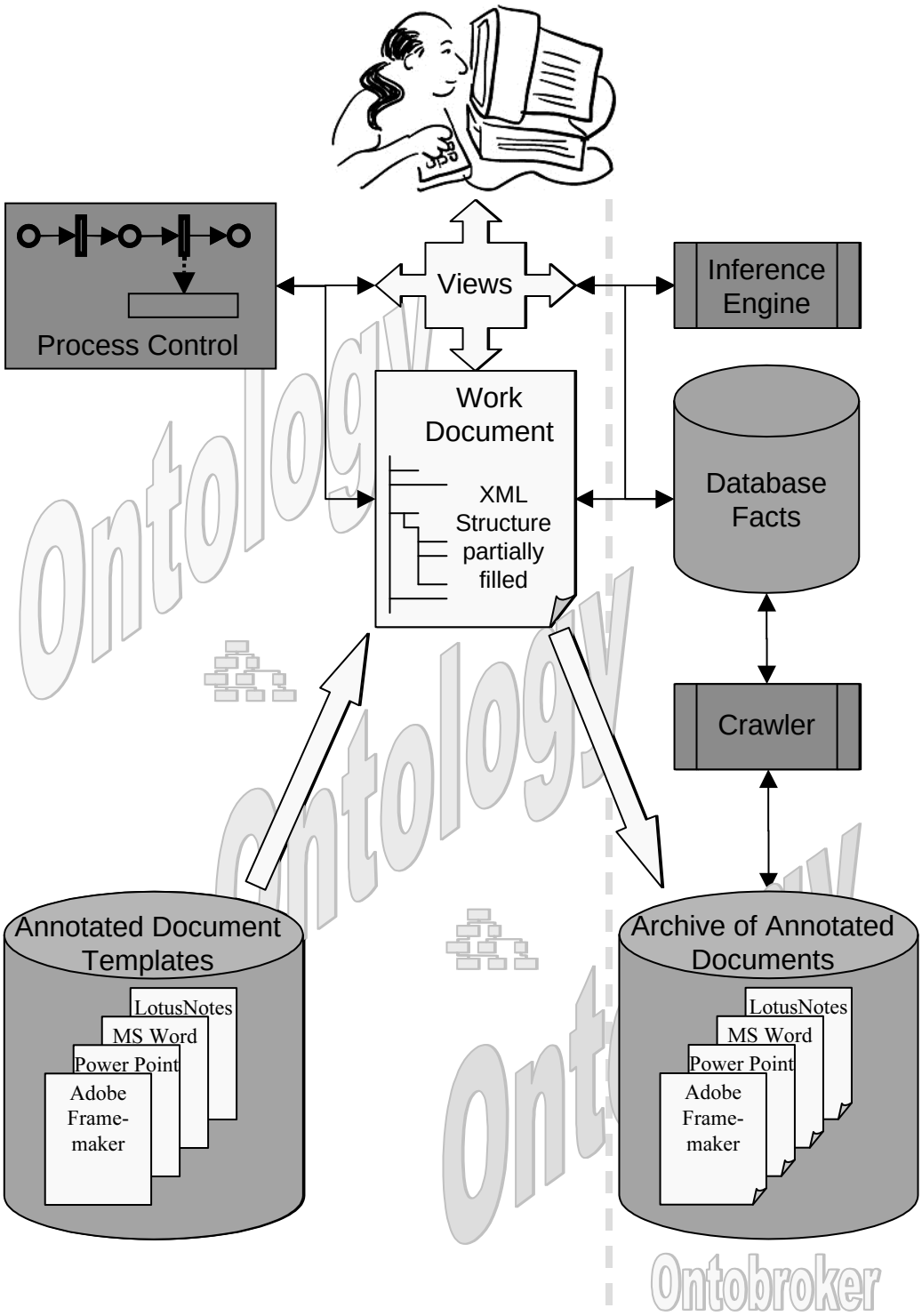

Figure 6: The principal IT architecture.

8 Related Works

Our starting point has been a common intranet environment, in which all documents are put such that they are widely available for reuse and general information. The requires a rather open minded environment, but this policy of open doors is not too seldom nowadays [Sellens & Wilson, 1998].

The next step has been an integration of distributed factual knowledge with an ontology as its conceptual backbone. Thereby, we relied on the system ontobroker, but similar systems like, *e.g.*, Shoe [Heflin et al., 1998] could be used in a very similar manner. Indeed, [Benjamins et al., 1998] already outlined how Ontobroker could be used for knowledge management, however in their approach the user had to bear all the burden of doing the right things at the right time, while our approach goes in the direction of telling the user what might be useful for him in his very next task.

A central point in our approach is reasoning about document structure and contents. Here we take over W. Weitz's view of SGML documents in a workflow process [Weitz, 1998], but extended his approach to include the knowledge management side and the weakly structured parts of processes. Our considerations also fit well with ones by [Abecker et al., 1998] about the nature of processes and documents in the processes. However, with our approach we breathed life into these formerly rather abstract notions.

Nearest to our integration of workflow and knowledge management aspects are works by [Huber, 1998], [Reimer et al., 1998], and [Wargitsch, 1998]. [Huber, 1998] builds on a Lotus Notes intranet environment that lets the user define a simple ontology and small workflows. However, his approach is less principled and does not lend itself easily for modeling and process planing goals. In particular, he cannot query facts, not to speak of implicit knowledge, but only documents.

[Wargitsch, 1998] describes an approach that puts elements of a process management system into an organizational memory. However, his purpose lies not so much in supporting the knowledge worker in his processes, but rather in supporting the process manager in adapting and optimizing the workflow. Hence, his knowledge management goal has a very particular scope. Ultimately, one should be able to integrate his approach and ours in order to provide adaptation possibilities based on workflow structures and based on underlying background knowledge.

[Reimer et al., 1998] supports the user with particular tasks. For this purpose, they use rather rigid process structures that are build from declarative business rules. We, in contrast, leave all the decision with the user and try to provide him with information that might facilitate his problem solving.

Based on the same principle idea, *viz*. the use of document oriented work, a comparable tool, which is however less sophisticated in terms of generality and outreach to the intranet, has successfully been used for re-engineering specifications at SD&M (the

corresponding methodology for document-based software engineering is described in [Denert, 1991]).

Our approach builds heavily on considerations by [Abecker et al., 1998] who establish a common grounding for documents, organization and knowledge (reflected by their information, enterprise and domain ontologies, respectively), but they do not go as far in drawing inferences and using this knowledge for a push technology and a tight integration of workflow and knowledge management, such as we do.

9 Conclusion

We presented an approach towards supporting the knowledge workers in an enterprise that is based on the tigth interconnection of documents, knowledge, organization and processes. In order to make the system practicable, we circumvent the knowledge annotation bottleneck by providing templates that are filled anyway in the course of the work process. In order to succeed, however, we plan to support the user in his own setting (i.e., framemaker™, lotus notes™, word™, etc) and though this will take some pain to achieve it will be a must for successful KM technology.

Our approach still shows several loose ends that need to be tied up by further research: First, we will have to take care of mappings between multiple ontologies or mappings between XML structures (DTDs, document type definitions) and ontologies, because different departments in an organization will not be willing to commit themselves to a standard vocabulary. This problem worsens with a tightly knit value chain that includes several companies or with virtual enterprises.

Second, we did not consider the boundary between the single workspace and the overall business process. There one has to find a way to provide flexibility beyond the single knowledge worker. Sharing complete documents is certainly a first step, but not the last. We imagine a technology that gives BP and KM managers full control over the rigidity of the work process. A sliding scale should allow the choice between common workflow techniques, models such as presented in [Reimer et al., 1998] and - at the other end of the spectrum - our approach. On this scale, this we think is a trueism, the process should be made as flexible as possible and as rigid as necessary.

Third, we envision a solution for the security problem that arises when all documents are accessible *a priori*. This may be solved by distributing the inference engine. Different inference servers allow for different types of access. Data that are private may not be used for inferences otherwise privacy is not secured.

Finally, we still must elaborate on the formal basis of the integration of workflow aspects with knowledge management tasks. The non-deterministic operations we model in SGML-nets still lack any formal grounding.

Nevertheless, our approach provides a new, we believe promising, way towards an integration of knowledge and business processes. A research prototype is currently under development and scheduled for the end of July. We hope we can give a live presentation at the IJCAI workshop.

10 Acknowledgments

The research presented in this paper has been partially funded by the German Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF) under grant number 01IN802 (project ``GETESS´´).

11 Bibliography

[Abecker et al., 1998] Andreas Abecker, Ansgar Bernardi, Knut Hinkelmann, Otto Kühn, & Michael Sintek. Toward a Technology for Organizational Memories. In: *IEEE Intelligent Systems*, 13(3), 40-48, 1998.

[Ackerman & Mandel, 1995] Mark Ackerman and Eric Mandel. Memory in the Small: Combining Collective Memory and Task Support for a Scientific Community. Journal of Organizational Computing and Electronic Commerce, accepted for publication in 1999.

[Angele et al., 1998] Jürgen Angele, Dieter Fensel, Dieter Landes, & Rudi Studer. Developing Knowledge-Based Systems with MIKE. In: *Journal of Automated Software Engineering*, 5(4), October 1998, 389-418.

[Barnekow & Staab, 1999] Thomas Barnekow & Steffen Staab. An Architecture for Recovering Business Operations Bottom-Up. In: *HCI '99 - Proceedings of the 8th International Conference on Human-Computer Interaction*, Munich, Germany, August, 22-27, 1999 (to appear).

[Basili et al., 1994] Victor R. Basili, Gianluigi Caldiera & H. Dieter Rombach. The Experience Factory. In: *Encyclopedia of Software Engineering*, Wiley, 1994.

[Benjamins et al., 1998] V. Richard Benjamins, Dieter Fensel, & Asunción Gómez Pérez. Knowledge management through ontologies. In: *[PAKM, 1998]*, pp. 5.1-5.12, 1998.

[Charniak & Goldman, 1993] E. Charniak & R. Goldman. A Bayesian model of plan recognition. *Artificial Intelligence Journal*, 64(1), 53-80, 1993.

[Decker et al., 1999] Stefan Decker, Michael Erdmann, Dieter Fensel, & Rudi Studer, Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information, In: R. Meersman et al. (eds.), *Database Semantics, Semantic Issues in Multimedia Systems*, , pp. 351-369, Boston, MA: Kluwer, 1999.

[Denert, 1991] Ernst Denert. *Software engineering: methodische Projektabwicklung.* Berlin; Heidelberg: Springer, 1991 (in German).

[Eriksson et al., 1995] H. Eriksson, Y. Shahar, S. W. Tu, A. R. Puerta, and Mark A. Musen. Task Modeling with Reusable Problem-Solving Methods, *Artificial Intelligence*, 1995, 79(2):293---326.

[Fensel et al., 1999] Dieter Fensel, Andreas Witt, Jürgen Angele, Stefan Decker, Michael Erdmann, Hans-Peter Schnurr, Steffen Staab, and Rudi Studer: On2broker in a Nutshell. In: *WWW8 - Proceedings of the 8th World Wide Web Conference*, Toronto, May 11-14, 1999.

[Georgakopoulos et al., 1995] D. Georgakopoulos, M. Nornick, & A.P. Sheth. An overview of workflow management: From process modeling to workflow automation infrastructure. *Distributed and Parallel Databases*, 3(2), 119-153, 1995.

[Heflin et al., 1998] Jeff Heflin, Jim Hendler, & Sean Luke. Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web. In: AI and Information Integration. Proceedings of the AAAI-98 Workshop. Madison, Wisconsin, USA.

[Huber, 1998] Harald Huber. Document research based on collaborative provided structure knowledge. In: *[PAKM, 1998]*, pp. 11.1-11.9, 1998.

[ISO, 1986] International Organization for Standardization. ISO 8879. Information processing - text and office systems - standard generalized markup language (SGML), October 1986.

[Kifer et al., 1995] Michael Kifer, Georg Lausen, & J. Wu. Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the ACM*, 42, 1995.

[Landes et al., 1998] Dieter Landes, Kurt Schneider & Frank Houdek. Organizational Learning and Experience Documentation in Industrial Software Projects. In: A. Abecker, S. Decker, N. Matta, F. Maurer, U. Reimer (eds.), *Building, Maintaining and Using Organizational Memories.* Proceedings of the Workshop at ECAI '98 (European Conference on Artificial Intelligence. Brighton, UK, August 23-28, 1998), pp. 47-63.

[McAllester & Rosenblitt, 1991] David McAllester & David Rosenblitt. Systematic Nonlinear Planning. In: *AAAI '91 - Proceedings of the Ninth National Conference on Artificial Intelligence*. Menlo Park, CA/Cambridge, MA: AAAI Press/MIT Press, 1991, pp. 634 - 639.

[PAKM, 1998] Ulrich Reimer (ed.), *PAKM-98 - Proceedings of the 2nd International Conference on Practical Aspects of Knowledge Management*, Basel, Switzerland, October 29-30, 1998.

[Reimer et al., 1998] Ulrich Reimer, Andreas Margelisch, Bernd Novotny, Thomas Vetterli. EULE2: A knowledge-based system for supporting office work. *ACM SIGGROUP Bulletin*, 19(1), pp. 56-61, 1998.

[Sellens & Wilson, 1998] Corinne Sellens & Owen L.F. Wilson. The CMG knowledge intranet. In: *[PAKM, 1998]*, pp. 22.1 - 22.5, 1998.

[Steinberg, 1999] Ulf Steinberg. *MikeTool.* Master's thesis. AIFB, Univ. Karlsruhe, 1999 (in German).

[van der Aalst, 1998] Wil M. P. van der Aalst. The application of petri nets to workflow management. *Journal of Circuits, Systems and Computers*, 8(1):21--66, 1998.

[Wargitsch, 1998] Christoph Wargitsch. An organizational-memory-based approach for an evolutionary workflow management system - concepts and implementation. In: Nunamaker, J. (ed.), *Proceedings of the 31st Annual Hawaii International Conference on System Sciences*, Vol. I, Los Alamitos: IEEE Computer Society Press, 1998, pp. 174-183.

[Weitz, 1998] Wolfgang Weitz. Combining structured documents with high-level petri-nets for workflow modeling in internet-based commerce. *Journal of Cooperative Information System*s, 7(4), 275-296, 1998.

[W3C, 1998a] World Wide Web Consortium. Extensible Markup Language (XML) 1.0. W3C Recommendation, February 10th, 1998. http://www.w3.org/TR/1998/REC-xml-19980210.

[W3C, 1998b] World Wide Web Consortium. Extensible Stylesheet Language (XSL). Version 1.0, working draft, December 16th, 1998. http://www.w3.org/TR/1998/WD-xsl-19981216.

[W3C, 1999] World Wide Web Consortium. Scalable Vector Graphics (SVG) Specification. W3C working draft, February 11th, 1999. http://www.w3.org/TR/1999/WD-SVG-19990211.