Task-Specific Knowledge Management in a Process-Centred SEE

Harald Holz[1], Arne Könnecker[2], Frank Maurer[3]

[1] University of Kaiserslautern, Department of Computer Science,
D-67653 Kaiserslautern, Germany
holz@informatik.uni-kl.de

[2] Vignette Deutschland GmbH, Heinrich-Heine-Allee 53,
D-40213 Düsseldorf, Germany
arne.koennecker@vignette.com

[3] University of Calgary, Department of Computer Science,
Calgary, Alberta, Canada, T2N 1N4
maurer@cpsc.ucalgary.ca

Abstract. This paper discusses how a process-centered knowledge management and coordination support approach can be used to create learning software organizations. We describe our extensions to the software engineering environment MILOS that allow us to model and interpret information needs that occur during project planning and enactment; this enables MILOS to automatically provide users with task-specific information. In order to capture actual information needs, we propose an extended feedback loop to update the process model stored in an experience base. The result is a knowledge management approach that is process-oriented and supports continuous process improvement.

Keywords: process-oriented knowledge management, process support environments

1 Introduction

Creating effective knowledge management structures is one of the key factors in software process improvement initiatives (like the Capability Maturity Model, Spice, Trillium, etc.). Most often, the knowledge management needs are only mentioned implicitly. Specific organizational structures (e. g. a software process group) are developed for the purpose of managing and distributing knowledge about software development. These structures are costly to maintain, and improving the efficiency of their members by a dedicated tool infrastructure would be very useful.

In general, two mainstreams concerning tool infrastructure can be distinguished: first, process-centred software engineering environments (PSEE) [10] are acknowledged tools to help in planning, managing and executing today's software

projects. Their support is mainly focused on the coordination of the different activities within a project following a defined development process, i.e. focused on project coordination. That is why the support for the individual participating agent in performing tasks is mainly restricted to provide access to input products for a task and to tools to create defined output products.

Secondly, tools that aim at supporting the Experience Factory proposed by Basili et.al. [4] are being developed (see e.g. [1]). They encompass mechanisms which help to capture and make information and knowledge accessible and (re)usable. The main concept here is to document and store any kind of information (e.g., knowledge, experience) and to access it via provided interfaces in a structured way. The common information retrieval based on statistic document analysis techniques mainly used in document management systems (DMS) is extended by, e.g., similarity-based or ontology-based retrieval, to better find relevant information items and to extract new information facts by the combination of two or more existing items.

The MILOS[1] project of the University of Calgary and the University of Kaiserslautern aims at providing an infrastructure that integrates these two mainstreams. Its primary goals are

- to provide access to relevant task-related knowledge when a task is planned or executed
- to integrate knowledge management with project planning and process coordination
- to create a feedback loop from process execution to knowledge management resulting in support for learning software organizations
- to develop tool support (the MILOS system) for this closed-loop approach

Section 2 describes our process-centered knowledge management approach. In Section 3, we sketch the feedback loop for updating the process models. An example for our approach is given in Section 4. The last two sections discuss related work and compare it to our approach.

2 Process-Centred Knowledge Management

As a PSEE, MILOS provides means to define generic process models and to set up concrete project plans based on these models. Furthermore, MILOS' web-based workflow engine supports the enactment of project plans, providing team members with individual to-do lists and relevant documents. During plan enactment, MILOS allows on-the-fly plan changes, notifying team members affected by those changes and updating the workflow engine accordingly [13].

Whereas MILOS supports project planning, coordination and enactment on the project level, so far there has been little or no support for the individual project member when confronted with a specific task. Rather, it is assumed that he already is equipped with all the information relevant to perform the task that has been assigned

[1] Minimally Invasive Long-term Organizational Support

to him, or that he at least knows how to retrieve that information. However, studies show that people often are not aware of information that might be relevant to them, even though this information has been explicitly stored in the company's organizational memory [14].

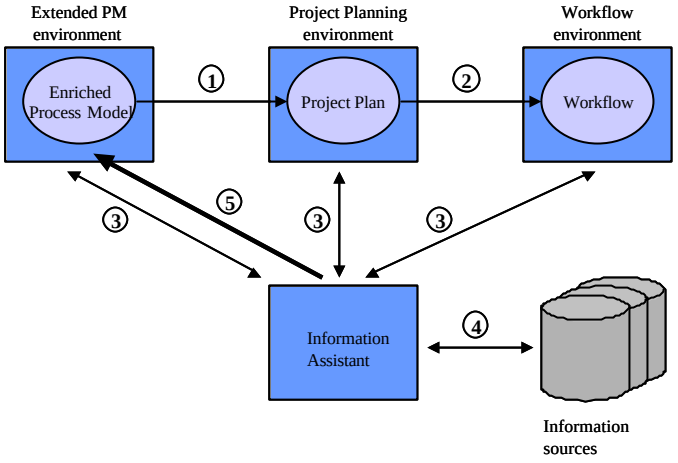


Fig. 1. An abstract view of the knowledge delivery concept within PSEEs: process models (PMs) with associated information need objects (INOs) can be created in an extended PM environment. These enriched PMs can be used as templates for a project plan (1), which is interpreted (2) by a workflow engine using the added scheduling information. Users carrying out activities in the planning or workflow environment are supported by the Information Assistant with information (3), based on available INOs from the PM. The IA queries (4) information sources on demand to retrieve information and deliver it (3) to a user. From within the IA, a user can post questions that are added to the appropriate process model element (5) as part of the feedback loop (see Section 3).

Therefore, we extended MILOS by an "Information Assistant" (IA), an explicit information delivery concept. The concept is based on an explicit modeling of information needs that might arise during specific tasks, together with an automatic query to the appropriate information source from which the answers can be retrieved; Fig. 1 gives an overview on this extension.

Generic Process Models are reusable workflow descriptions of software development processes. Primarily, these models describe different tasks[2], ways how to solve tasks (called *methods*), conditions describing when a task can be started, criteria its results must fulfill, required qualifications for tasks, and the document flow between tasks. For a more detailed description see [17].

In order to provide task-specific knowledge delivery, we have extended our representation of process models by *information need objects* (INOs; see Fig. 2). INOs represent potential information needs that might arise during planning or enactment of specific processes; they are characterized by the following attributes:

[2] In the following, task and process are used synonymously.

– **representation:** a textual representation that describes the information need, e.g. “What bugs have been reported for component <componentName> ?“. The represention can contain variables (delimited by ‘<’ and ‘>’) that correspond to attributes defined in the process model. During planning or enactment, they will be substituted by their current values.

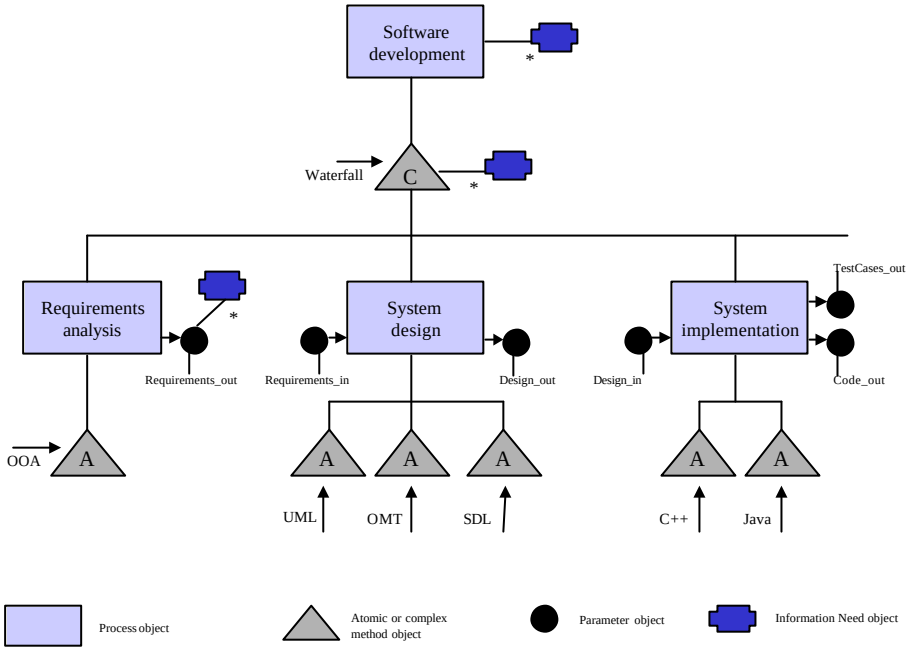


Fig. 2. A part of an example process model, where INOs have been attached to different model elements, e.g. to processes, complex methods (i.e. process decomposition), atomic methods i.e. process specialization) and parameters. The “*” denotes that several INOs can be associated with a model element.

– **information source:** specifies which resource can be used to retrieve information that might be helpful to satisfy the defined information need. Sources can be e.g. bug tracking systems, information agents, databases, or document management systems. A number of typical interfaces (eg. SQL, CGI, HTTP) have been predefined within MILOS to allow access to different types of information sources.
– **Query representation**: specifies a query in the syntax required by the given information source, `e.g.: “?bug_status=NEW&product=<component Name>”.` The query will be executed automatically to retrieve potentially helpful information items from the information source.
– **Supported roles:** specifies one or more roles to which the retrieved information will be relevant. For example, that way we can distinguish between information needs typical for planners or enactors.

– **Information category:** specifies one or more categories under which the INO will be organized; examples of predefined categories for the role `planner` are: "`agent assignment`", "`project scheduling`" etc.
– **Precondition:** a Boolean expression that specifies when the information need normally occurs and the query can be launched, e.g.: "`componentName.hasValue()`"

Thus, a process model not only serves as a means to store knowledge about best practices, but also maintains generic links to knowledge items that have been found useful in the past when planning or enacting specific processes.

Project Plans are set up by the planner by tailoring generic process models to the needs of a concrete project. This includes:

- selecting processes and methods from the generic process model that should become part of the project plan. For example, the generic process model contains a process "System Design" with methods "UML", "OMT" and "SDL". The project manager can add an instance of the process " System Design" to the project plan and then select "UML" from the set of alternatives as his method of choice.
- standard project planning: Assigning tasks to responsible agents, scheduling tasks, cost estimation etc. These activities are commonly supported by COTS tools like MS Project.

In order to support planners in these knowledge intensive activities, we want to enable our PSEE to provide them with situation-specific information that helps them in their decision-making.

Since project plans are tailored process models, plan elements inherit the generic INOs associated with the corresponding process model elements. In the context of the current project, variables referenced in the INO definitions can now be substituted by their values (e.g. the variable <assignedAgent> is substituted by the team member to which the task under consideration is currently assigned to). As a consequence, INOs that are intended to reflect a planner's information needs can now be presented to him by the Information Assistant, in order to signal available access to potentially useful information.

Using a flexible **workflow engine**, project plans are interpreted in order to actively guide human users in their work. It manages the state of the processes contained in the plan, to-do lists for individual users, the products created during process enactment, and traceability relationships between process entities [19]. In addition, the INOs associated with plan entities are presented to those team members on whose to-do lists the entities appear; thus, they are given situation-specific access to potentially relevant information.

Before we illustrate the functionality with an example scenario in Section 4, we briefly sketch the proposed feedback loop in the following section.

3 Creating a Feedback Loop for Continuous Learning

To support an organizational learning process, our approach links process-centered knowledge management with project planning, enactment support and experience packaging. We follow a four-step process (see Fig 3).

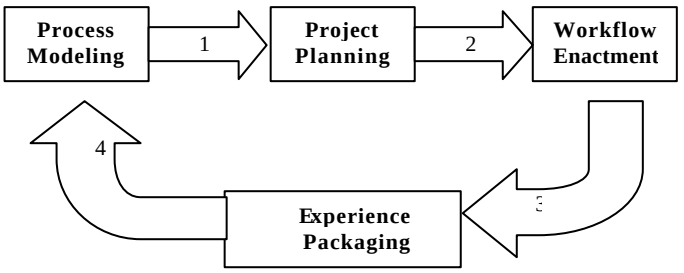


Fig. 3. Knowledge Utilization Cycle

The process model library contains descriptions of best practices in software development for a given company. In step 1, the project manager selects processes and methods from the library to create an initial project plan. This plan includes references to background knowledge (generic queries, modeled as INOs) that were stored in the process model. However, the planning process is incremental: we can change the plan at any time during process execution. The current project plan is the basis for enactment supported by our workflow engine (2). Using a standard Web browser, a user is able to connect to the workflow engine, access her to-do list, accept/reject/restart a task, access task inputs, and edit task outputs. After completing a task, the outputs are accessible as input for the successor tasks.

The next step is to extract reusable process knowledge from the current project plan (3). A user can select parts of the project plan (e.g. a method for a specific task or the newly added information resources) and upload/integrate them into the generic process model library (4).

During planning and enactment, the Information Assistant allows users to formulate new information needs in the form of questions. These information needs are associated with the task in the context of which they occurred. Thus, members of the software process group are alerted to existing information needs during experience packaging, and can try to complete the partially specified INO. That way, next time the respective process occurs in a project, users can be provided directly with the requested information.

4 Examples

In the following, we illustrate how we set up the Information Assistant within MILOS for use in our environment. First, we give examples of information needs that we intend to be able to satisfy, and for which we set up (or are currently in the process of setting up) appropriate information sources. Second, we illustrate our system's

functionality with the help of an example scenario motivated by our own project's software development process.

Table 1. Typical information needs of planners

Category	Information need
Anticipating difficulties	• What problems occurred in similar tasks? How were they solved? • What relevant problems/shortcomings/bugs with the tools used in the current task are known?
Agent assignment	• Which agents match the skills required for the task? • Which agents have performed a similar task before? • Which agents are available at the time period in question?
Effort estimation	• What quality models exist for the task? • What was the time effort of similar task?
Task refinement	• What standard refinements exist for the task? • How were similar tasks performed in the past?

4.1 Modeling Information Needs

Both planners and plan enactors are confronted with a set of standard problems that occur for every task, e.g. anticipating potential difficulties, finding a skilled team member with free capacities, estimating the time effort, and, depending on the task's granularity and the team member's expertise, developing an outline of how to perform the task. Table 1 lists examples of typical planning information needs related to these problems.

In order to enable the Information Assistant to satisfy these information needs, we require the availability of appropriate information sources from which the desired information can be obtained, e.g.: (i) the organization's resource pool, including skill information for each agent, (ii) a calendar database providing schedule dates for agents, (iii) bug and issue tracking systems, project and task case-bases, (iv) an experience base storing process and quality models, (v) a DMS holding general documents, etc.

Table 2. Example of a generic INO that is associated with every task

INO attributes	Value
Representation	"Who has performed similar tasks before?"
Query	retrieve_similar_tasks(<currentTask>)
Information source	Task case-base
Precondition	Task case-base available
Information category	agent assignment, task execution
Supported roles	planner, enactor

As mentioned above, information needs are represented as INOs that reference an information source; Tab. 2 shows an example INO that references a task case-base. In order to launch the query execution, the IA must be able to access the task case-base via a predefined interface. In accordance to this interface, it must also replace the variable <currentTask> with the appropriate representation of the task for which the INO's query execution is requested. In the example of our task case-base, this means replacing the variable with the current task's string representation; this representation becomes part of a parameter string for a CGI-script that performs the similarity-based search.

In addition to information needs that typically occur for all tasks, we capture information needs that only occur for certain tasks. Processes in the process model are enriched with INOs as depicted in Fig. 4, where we define an INO that reflects a coder's information need to view the list of bugs that have been reported for the component he is currently implementing.

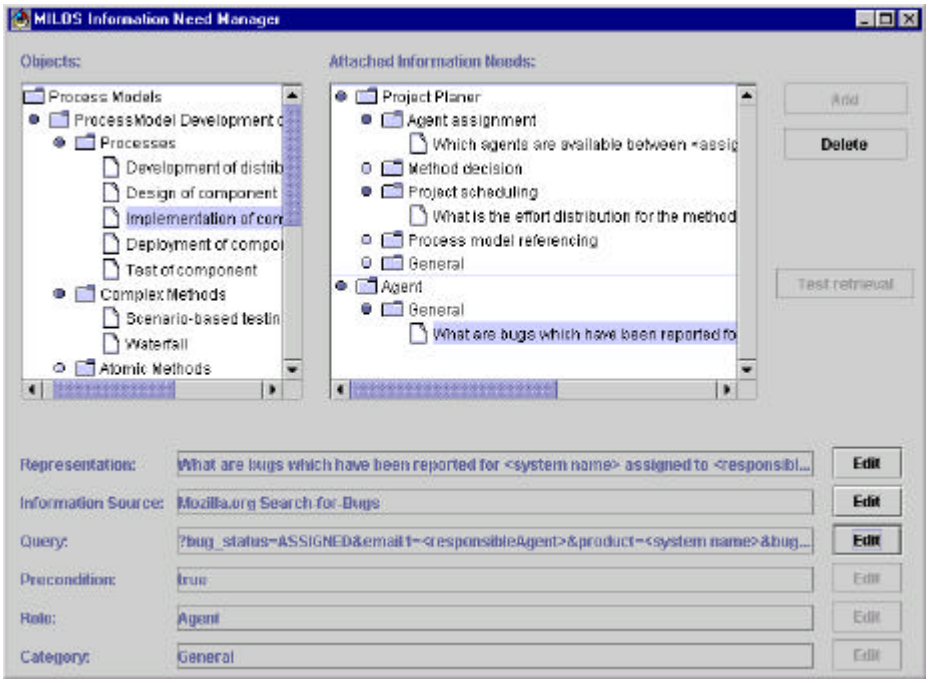
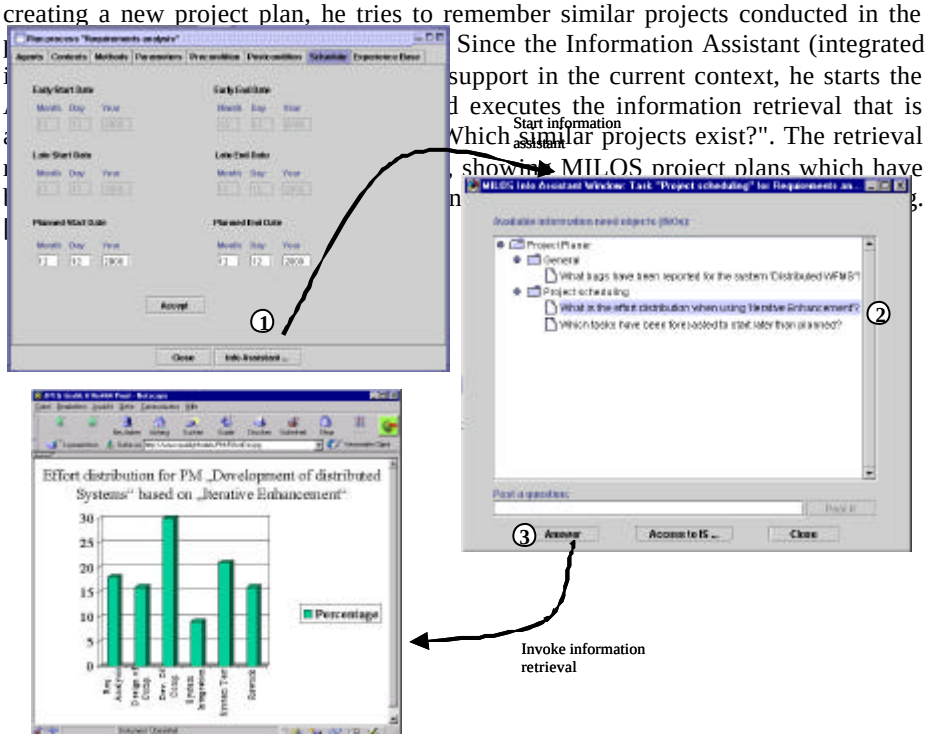


Fig. 4. The MILOS INO editor: from the tree in the upper-left part of the window, the user has selected the process object 'Implementation of component' from a process model. The tree in the upper-right part displays the associated INOs, using the role and information category attributes for structuring. The attribute values of the selected INO are shown in the lower part of the window

4.2 Example Scenario

In our example, the project planer starts the MILOS PSEE to set up a new project, i.e. to define a project plan. For his current project he provides a characterization by specifying the project name ('Distributed workflow management system'), the system architecture type ('Distributed'), and the estimated duration ('2 years'). Before creating a new project plan, he tries to remember similar projects conducted in the



Since the Information Assistant (integrated support in the current context, he starts the executes the information retrieval that is Which similar projects exist?". The retrieval shown MILOS project plans which have

Fig. 5. The project planer starts the Information Assistant from the scheduling tab within the MILOS project planer via the enabled button (1) to get support in his scheduling task. The IA presents him the current relevant and available INOs and he selects the question "What is the effort distribution for the process model 'Development of distributed systems'?" (2). After he pressed the "Answer" button (3) in the IA, the browser in the lower left picture opens and present him an effort distribution diagram.

Inspecting the most similar project plan, he can see that this plan used the process model 'Development of distributed systems' from the MILOS PM library. Hence, he browses the process model library and selects this process model as a basis for his project plan. As a result, the processes from the process model now define tasks within the project plan. Besides the specification of these processes, their parameters, and the possible methods, the corresponding instances in the project plan include the INOs that are attached to the process model objects.

The chosen process model maintains knowledge about the development method 'Iterative enhancement' which the project planer thinks is appropriate especially because of the long project duration and the component-oriented architecture. Next, the planner wants to make a rough estimation of the time effort required for the task 'Requirements analysis'. This can be done by consulting a quality model that describes the effort distribution with respect to process steps in the chosen process model. Using the Information Assistant (see Fig. 5), he accesses the information associated with the question "What is the effort distribution when using Iterative Enhancement?" which is listed in the information category 'Project scheduling'. The questions that relate to this information category are available within the Information Assistant because the current work context of the project planer is 'project scheduling'. A corresponding diagram retrieved from the experience base for quality models is displayed in a window as shown in Fig. 5. According to the effort distribution and the estimated total time of 2 years for his project, he can now provide rough estimates for all tasks.

Furthermore the planer has to assign responsible agents to each of the tasks specified by the process steps in the project plan. He wants to do this for a design task for which he has not yet chosen a method. Existing alternatives are 'UML' and 'SDL'. He accesses the agent assignment UI within the MILOS project planer component for the design task. As the project planer is fairly new in his job and the department, he requires information whether any agents are working in the department which have experience in one of these methods. The Information Assistant provides an INO within the information category 'agent assignment' which models this information need represented as "Which agents have experience with 'UML' or 'SDL'?". The question is coupled with the retrieval on a skill database for agents, which yields a list of agents with the required skills. He can see that for both methods agents are employed in the department. He decides to use 'UML' as he personally has used this method before and is convinced of its quality. Due to this decision, his current work context changes as the method decision for the design task gets the value 'UML'. This is propagated to the Information Assistant which updates

his list of INOs. The general question about experienced agents for all two methods is not longer required as a decision has been made.

Since he has already provided start and end dates for the design task, he is now interested in agents which have experience with 'UML' and are available in the given time frame. Thus he uses an according question from the Information Assistant to query the schedule database for available agents, checking only those agents that have been retrieved as agents with UML skills.

The project planer knows from formerly planned projects that another method for design is 'OMT'. Besides the planning for the current project he is interested if any agents in the new department have experience with this method. As 'OMT' is not a defined method in the process model he has received from the PM experience base, the Information Assistant does not offer support for this question. But as the retrieval for 'UML' and 'SDL' has been done on a skill database, he uses the Assistant to access the query engine interface of this information source. Now he defines his own query to search for agents with 'OMT' experience and launches it. As he thinks this information need might occur again in the future (maybe even the 'OMT' method can be modeled as additional method in the process model) he starts the INO Manager within MILOS and adds a corresponding INO to the INO list of the 'Design' process into the information category 'Agent assignment'. From now on, this INO appears whenever someone uses the support concept while doing agent assignment for the 'Design' task.

Likewise, as with the agent assignment for the 'Design' task, the project planner proceeds with the remaining tasks in the project plan.

Agents that participate in the project can access individual workspaces provided by the MILOS workflow engine. The project planer has assigned a task 'Implement WFE component' during project planning to the agent 'Alice' and, accordingly, this task appears in her to-do list (see Fig. 6 for an illustration of the to-do list UI).

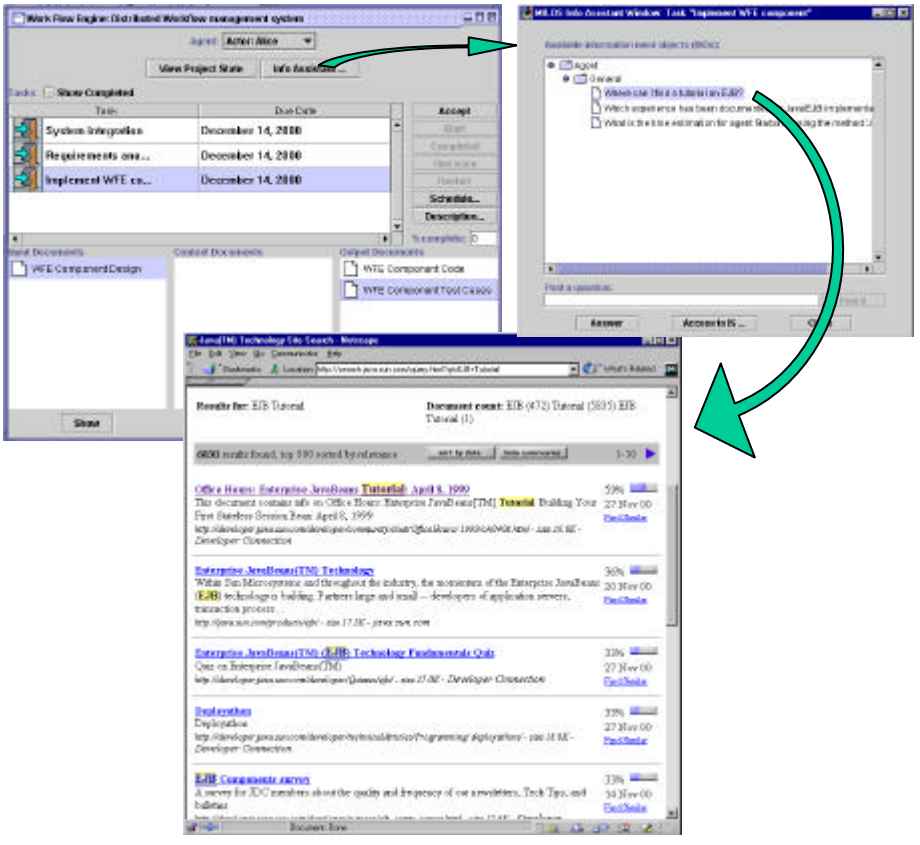


Fig. 6. INO execution to find an EJB tutorial: agent "Alice" has selected the question "Where can I find a tutorial on EJB?" in the Information Assistant (left picture). After she has pressed the 'Answer' button at the bottom, a browser opens and presents her a list of links which have been retrieved from the Javasoft homepage to the topic "EJB Tutorial". She can now refer to the hyperlinks to access the information items

The workspace allows her to browse the information associated with each task, e.g., scheduling information or the input and output products that have been specified in the project plan and are listed in the lower part of the window when she selects a task.

When she starts working on the task 'Implement WFE component' she runs into a problem while trying to implement an EJB. She remembers that this once has been explained to her in a tutorial. She launches the Information Assistant again and identifies her question "Where can I find a tutorial on EJB?" in the presented list, which is exactly what she is looking for. She selects the INO and executes the retrieval (as shown in Fig. 6) to receive a list of items found on the Javasoft homepage which has been known as a good source by the one who modeled the INO. She follows the hyperlink to the EJB tutorial, in order to refresh her knowledge while following the tutorial steps.

A further look in the Information Assistant window guides her to the question "Which experience has been documented for Java/EJB implementation?". She does not even know that any experience is documented related to that, but seemingly these documents exist as otherwise this information need would not appear in the Information Assistant. Since there are documented experiences, there must also be agents who have made these experiences. As she can not find a corresponding information need in the presented list of questions (i.e. INOs), she uses the posting functionality of the Information Assistant to state her question and post it. The Information Assistant attaches the question to the process object in the project plan that defines the task 'Implement WFE component' and notifies the project planer (i.e. manager) that an information need has occurred for this object that has not been satisfied with the given INO specifications. The manager can now initiate further actions, either to provide the required information for agent 'Alice' and/or to model this information need as a persistent INO for future support.

In summary, besides having used the provided information need support via the execution of modeled INOs within the Information Assistant, the project planer has modeled at least one new INO about the 'OMT' method and Alice has posted a question which has been attached to project plan object. When a project is finished the process model can be written back into the MILOS library. Using this technique the newly modeled INOs and the posted questions become persistent in the PM library, here in the process model 'Development of distributed systems'. They are available in the future every time a project planer reuses this model for a project plan. The posted questions need to be modeled into a completely specified INO (e.g. coupled to an information source) to be usable in the future. Before that, they serve as a marker for an INO modeler that an information need occurred for the specific process model object to which it is attached.

Using this feedback loop, process models that are frequently used and maintained provide better support each time they are tailored for a project. The approach thus creates a learning software organisation as it helps to package required information

and to provide it at the right time (i.e. while planning or enacting a particular task) to the right agent.

5 Related Work

Related work comes mainly from two areas: Software Process Improvement and Knowledge Management. We first discuss software process improvement and we analyze some KM approaches.

Most process improvement approaches, e.g. capability maturity model, require describing the development processes more or less formally. Within the framework of software process modeling, several languages were developed that allow for describing software development activities formally ([15], [6], [2])

Software process models represent knowledge about software development. They describe activities to be carried out in software development as well as the products to be created and the resources & tools used. These models can be a basis for continuous organizational learning as well as the actual basis for the coordination and the management of the software engineering activities.

Software process modeling and enactment is one of the main areas in software engineering research. Several frameworks have been developed (e.g. procedural [15], rule-based ([11],[16]), Petri net based [3], object-oriented [5].

Managing software process knowledge is also the goal of the experience factory approach [4]. They distinguish between the organizational structure to manage the software knowledge (the experience factory department) and the activities that have to be carried out to build an experience factory.

Knowledge management and organizational memory is currently a hot topic in research (for example, see ([7], [18]). The use of case-based reasoning technology for experience management in software engineering is discussed in [1]. Their approach is not directly linked to project execution support and therefore fails in providing an operational feedback loop between project execution and organizational learning. Their approach could be easily integrated with MILOS by pointing our CBR queries to their system.

The expert system group at the German DFKI is also following a process-centered approach to knowledge management. Their tool is based on a business process modeling approach and is, compared to MILOS, fairly inflexible at enactment time: changing the process by simply replanning the project with a COTS tool is not supported. On the other hand, their approach is based on domain ontologies [12] that allow for a semantic-oriented information retrieval.

Ontology-based retrieval is also investigated by [8]. Ontology-based KM approaches are not providing a process-centered approach and therefore require the user to specify queries for a given task (whereas MILOS can provide users with predefined queries for tasks). A similar argument holds for hypertext-based approaches (e.g. [9]): they do not provide task-oriented access to knowledge but force the user to navigate to it.

6 Conclusion

In this paper we described our approach for supporting learning software organizations. The MILOS system is an Internet-based process-centered knowledge management environment that structures knowledge around development processes.

Linked to a process, the user can find methods (describing ways how to perform the process to reach its goals), products (input and outputs to the process), knowledge about the qualifications needed to perform the process and knowledge about typical information needs, together with ways to satisfy them.

The process-centered structure of the system has the following advantages:

Processes are "natural" entities for managers and team members: they are well used to thinking in processes (e.g. for project planning).

For their daily work, people don't need knowledge per-se but knowledge for performing specific tasks. A process-centered knowledge management system associates explicitly the task with the knowledge needed for it.

By linking relevant information sources and generic queries to task, the "lost in hyperspace" problem is reduced because the user is actively guided to available knowledge needed instead of being forced to somehow find relevant information on his own.

The feedback cycle creates a learning software organization: our approach allows to package "good" elements of successful project plans into reusable process models and, hence, supports the implementation of a continuous process improvement strategy for software companies.

In Section 1, we mentioned that we are aiming at an improved efficiency of an organization's process group. Whereas undoubtedly the introduction of new tools at first results in an increased workload, we argue that, in the long run, the proposed Information Assistant will relief the process group from answering standard questions in the same way as help-desk applications already have proven to do in other domains (e.g. trouble-shooting for technical devices). We expect he Information Assistant to provide answers to the users' standard questions (especially to those of new employees), so that the human experts in the process group need only be consulted for new, more difficult problems.

In order to collect first experiences with the approach presented here, we intend to apply it within our own software development projects (e.g. master theses). Whereas the coordination aspect often can be neglected in these personal processes, a lot of knowledge about the MILOS implementation, its design and concepts (both past and future) is typically required, making it very difficult for new students to get started with their work.

Acknowledgements

The work on MILOS was supported by the DFG (as part of SFB 501: "Development of large systems with generic methods"), NSERC, The University Of Calgary, and Nortel with several research grants. We would like to thank Sigrid Goldmann and

Martin Schaaf for their valuable input, and tec:inno - empolis knowledge management division[3] for providing us with their CBR middleware 'Orenge'.

References

1. Althoff, K.-D. & Bomarius, F. & Tautz, C. (1998). Using Case-Based Reasoning Technology to Build Learning Software Organizations. In Proceedings of the 1st Interdisciplinary Workshop on Building, Maintaining, and Using Organizational Memories (OM-98), 13th European Conference on AI (ECAI'98), Brighton, http:// SunSITE.Informatik.RWTH-Aachen.DE/Publications/CEUR-WS/Vol-14/
2. Armitage, J., and Kellner, M. (1994). A conceptual schema for process definitions and models. In D. E. Perry, editor, Proc. of the Third Int. Conf. on the Software Process, IEEE Computer Society Press.
3. Bandinelli, S., Fuggetta, A., and Grigolli, S. (1993). Process Modeling-in-the-large with SLANG. In IEEE Proceedings of the 2nd Int. Conf. on the Software Process, Berlin (Germany).
4. Basili, V. R., Caldiera, G., and Rombach, H. D. (1994). Experience Factory. In Encyclopedia of Software Engineering (J. J. Marciniak, ed.), vol. 1, John Wiley Sons.
5. Conradi, R., Hagaseth, M., Larsen, J. O., Nguyen, M., Munch, G., Westby, P., and Zhu, W. (1994). EPOS: Object-Oriented and Cooperative Process Modeling. In PROMOTER book: Anthony Finkelstein, Jeff Kramer and Bashar A. Nuseibeh (Eds.): Software Process Modeling and Technology, 1994. Advanced Software Development Series, Research Studies Press Ltd. (John Wiley).
6. Curtis, B., Kellner, M., and Over, J. (1992). Process modeling. Comm. of the ACM, 35(9): 75–90.
7. Davenport, T.H. & Jarvenpaa, S.L. & Beers, M.C. (1997). Sloan Management Review, 37 (4):53-65, Improving Knowledge Work Processes,1997.
8. Decker, S., Erdmann, M., Fensel, D., and Studer, R.(1999). Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In R. Meersman et al. (eds.), Semantic Issues in Multimedia Systems, Kluwer Academic Publisher, Boston.
9. Euzenat, J. (1996). Corporate Memory through Cooperative Creation of Knowledge Bases and Hyper-documents. In Proceedings of the 10th Knowledge Acquisition, Modeling and Management for Knowledge-based Systems Workshop (KAW'96), Banff.
10. P.K. Garg, M. Jazayeri: *"Process-centered Software Engineering Environments"*. IEEE Cumputer Society Press, 1996.
11. Kaiser, G. E., Feiler, P. H., and Popovich, S. S. (1988). Intelligent Assistance for Software Development and Maintenance, IEEE Software.
12. Kühn, O. & Abecker, A. (1997). Corporate Memories for Knowledge Management in Industrial Practice: Prospects and Challenges. In Journal of Universal Computer Science 3, 8, Special Issue on Information Technology for Knowledge Management, Springer Science Online. URL: http://www.iicm.edu/jucs_3_8/ corporate_memories_for_knowledge.
13. Maurer, F., Dellen, B, Bendeck, F., Goldmann, S., Holz, H., Kötting, B., Schaaf, M.: *"Merging Project Planning and Web-Enabled Dynamic Workflow Technologies"*. IEEE Internet Computing May/June 2000, pp. 65-74.
14. Mahe, S. and Rieu, C.; Towards a Pull-Approach of KM for Improving Enterprise Flexibility Responsiveness: A Necessary First Step for Introducing Knowledge Management in Small and Medium Enterprises. In: Proceedings of the International Symposium on Management of Industrial and Corporate Knowledge (ISMICK '97), Compiegne, 1997.

[3] www.tecinno.de

15. Osterweil, L. (1987). Software Processes are Software Too. In: Proc. of the Ninth Int. Conf. of Software Engineering, Monterey CA, pp. 2-13.
16. Peuschel, P., Schäfer, W., and Wolf, S. (1992). A Knowledge-based Software Development Environment Supporting Cooperative Work. In: Int. Journal on Software Engineering and Knowledge Engineering, 2(1).
17. Verlage, M., Dellen, B., Maurer, F., and Münch, J. (1996). A synthesis of two software process support approaches. In Proceedings 8th Software & Engineering and Knowledge Engineering (SEKE-96), USA.
18. Wielinga ,B.J. & Sandberg, J. & Schreiber, G. (1997). Methods and Techniques for Knowledge Management: What has Knowledge Engineering to Offer, Expert Systems with Applications 13, 1, 73-84.
19. Dellen, B., Kohler, K., and Maurer, F. (1996). Integrating Software Process Models and Design Rationales. In: Proc. of Knowledge-Based Software Engineering Conference (KBSE-96), IEEE press.