Task-based Process Know-how Reuse and Proactive Information Delivery in TaskNavigator

Harald Holz, Oleg Rostanin, Andreas Dengel
DFKI, Erwin-Schroedinger-Str. 57,
67663 Kaiserslautern, Germany
{firstname.lastname@dfki.de}

Takeshi Suzuki, Kaoru Maeda, Katsumi Kanasaki,
Ricoh Co., Ltd. 1-1-17 Koishikawa, Bunkyoku
112-0002 Tokyo, Japan
{firstname.lastname@nts.ricoh.co.jp}

ABSTRACT
Knowledge management approaches for weakly-structured, adhoc knowledge work processes need to be lightweight, i.e., they cannot rely on high upfront modeling efforts. This paper presents TaskNavigator, a novel prototype to support weakly-structured processes by integrating a standard task list application with a state-of-the-art document classification system. The resulting system allows for a task-oriented view on office workers' personal knowledge spaces in order to realize a proactive and context-sensitive information support during daily, knowledge-intensive tasks. Moreover, TaskNavigator supports process know-how reuse by proactively suggesting similar tasks or relevant process models, based on textual similarities. Finally, we report on a feasibility test and a case study that have been conducted in order to evaluate the system in the context of daily research task management and software requirements analysis.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval] H.4.1 [Office Automation]: Groupware; Workflow management, H.5.3 [Group and Organization Interface]: Computer-supported cooperative work; Web-based interaction, H.2.4 [Systems]: Distributed databases.

General Terms
Management, Performance, Human Factors

Keywords
Agile workflows, Process-oriented knowledge management, proactive information delivery

1. INTRODUCTION
The recent emergence and popularity of new desktop search engines such as Google Desktop Search[1], x-friend[2], MSN Desktop Search[3], etc. has clearly shown the need for tools that help users manage their personal knowledge space (PKS).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'06, November 5–11, 2006, Arlington, Virginia, USA.
Copyright 2006 ACM 1-59593-433-2/06/0011...$5.00.

Typically, the documents needed by a knowledge worker for the task at hand are spread over various places such as e-mail folders, file system folders, or paper stacks on the desk. While the concept of a desktop-wide search certainly relieves the user from the burden of querying several different information sources (e-mail, local and network drives, etc.), current desktop search engines still follow the standard, passive query/retrieve model: the user has to explicitly 'pull' for information that might be relevant for a task he is currently trying to accomplish. Besides being inefficient, empirical studies have shown that such pull approaches typically lead to suboptimal reuse rates of available documents [11]. To address this issue, several business process-oriented knowledge management approaches have been developed for proactively providing process participants with information that is relevant with regard to their current tasks [2]. However, as most of these approaches rely on static workflow/process specifications, they are typically inadequate for weakly-structured processes such as knowledge-intensive office work processes.

Currently, state-of-the-art workflow and document management systems offer valuable support only for routine activities in office work. Despite this support, it has been claimed that knowledge-intensive office work has not reached satisfying increases in productivity in recent years (cf. [15]). The reason for this perceived lack of productivity increase in such office work is seen in the insufficient understanding of the nature of knowledge-intensive work and the lack of adequate integration of information support and work activities. From our experience, knowledge work consists of both agile and strictly-structured processes that often are highly interleaved. Whereas recent project support systems aim at uniformly supporting users in both kinds of processes [14], an integrated approach for information support in the form of proactive information delivery still seems to be missing.

In this paper, we present TaskNavigator, a novel workflow management system capable of supporting agile workflows. The system aims at improving productivity of users and user groups during knowledge-intensive work, by enhancing process information reuse individually and among the group members.

[1] http://desktop.google.com/

[2] http://www.x-friend.de/

[3] http://toolbar.msn.com/

This paper is organized as follows: in Section 2, we provide an overview on the approach for lightweight process-oriented knowledge management that underlies the TaskNavigator prototype. Sections 3 and 4 describe TaskNavigator's functionalities in detail and illustrate the usage of the system with the help of an example scenario. Evaluation results are presented in Section 5, followed by a discussion of related work (Section 6) and a conclusion (Section 7).

2. THE PERL CYCLE

Our work aims at developing a light-weight approach for business process-oriented knowledge management (BPOKM) that can be applied for flexible knowledge work processes such as R&D or consulting processes. Since such processes cannot be defined and modeled sufficiently in advance to allow for workflow-like enactment support, standard approaches to BPOKM (see e.g. [3]) that are based on formal process models and formally specified information needs cannot be applied. Moreover, companies are often not willing to make high upfront investments into process and knowledge modeling activities when it is unclear if and when these investments are going to pay off.

Therefore, we propose the following bottom-up approach, called PERL cycle, for introducing BPOKM into a company in order to support knowledge workers in their daily activities, without requiring upfront process- or ontology-modeling. Figure 1 depicts the PERL cycle graphically: the approach is based upon an integrated tool support for flexible task management (i.e. a to-do list application) and proactive (i.e. push-like) information delivery. The cycle builds upon the following two assumptions:

1. Knowledge workers are willing to use a task list application provided by their company in order to keep track of their current tasks.
2. Information that is relevant for the knowledge workers' tasks is accessible from their desktops in the form of electronic documents, e.g. files, emails, Wiki pages, etc.

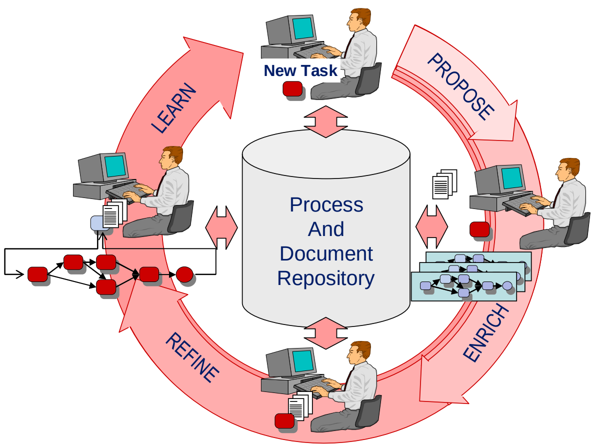

Figure 1: PERL cycle

Consequently, Figure 1 depicts a (logical) central repository where the knowledge workers' tasks and documents are stored within the company. Four phases, called "Propose", "Enrich", "Refine", and "Learn" are grouped around this repository, and need to be supported by an appropriate tool environment.

Propose: For a newly created (or newly selected) task, the system should proactively provide the knowledge worker with both available documents and process know-how that might be relevant in order to successfully complete the task.

Enrich: In order to receive more relevant document suggestions, the system should support the knowledge worker in easily enriching a task's description informally, e.g. by associating documents created or handled in the context of the task.

Reuse: The system should support the knowledge worker in reusing process know-how by both proactively suggesting similar former tasks (together with their decomposition into subtasks) from the repository and process guidelines, either textual or formal process definitions (if available) for the current task. Reuse of similar tasks should be supported by enabling the knowledge worker to create corresponding copies of the subtasks of similar tasks; reuse of process models should be supported by enabling the knowledge worker to instantiate subtasks according to the decomposition specified in the process model. In both cases, new (sub-) tasks are being created, for which the cycle will start again with phase "Propose".

Learn: The system should continuously learn by storing the actual decomposition of each task into subtasks, store new documents (e.g. emails, downloaded papers, work results etc) in the repository, support generalization of tasks into formal process models, learn how to categorize tasks to corresponding process models, and improve its proactive document suggestions over time.

In order to analyze the feasibility of such an approach, we developed a prototype, called TaskNavigator, implementing most parts of the PERL cycle. In the following, we will describe TaskNavigator's functionalities in more detail.

3. TASKNAVIGATOR

In accordance to the PERL approach, TaskNavigator provides tool support for:

- Agile task management
- Proactive information delivery
- Process know-how reuse

We will present these functionalities in the following sections in more detail.

3.1 Agile Task Management

TaskNavigator provides users with the standard functionalities of common task management tools known from MS Outlook. Users can create and edit tasks, specifying a task's due date, priority, current state, etc. In addition, TaskNavigator allows users to maintain task-specific, hierarchical bookmark lists by attaching documents or URLs to a task and organizing them in (task-specific) folder structures.

Moreover, users can define simple ad hoc workflows by specifying predecessor relationships tasks, expressing that a task should remain in state "waiting" until its predecessors are in state "completed". However, unlike rigid workflow management systems, the order of working with tasks in TaskNavigator is not strictly predefined. Task states have a recommending character, and users

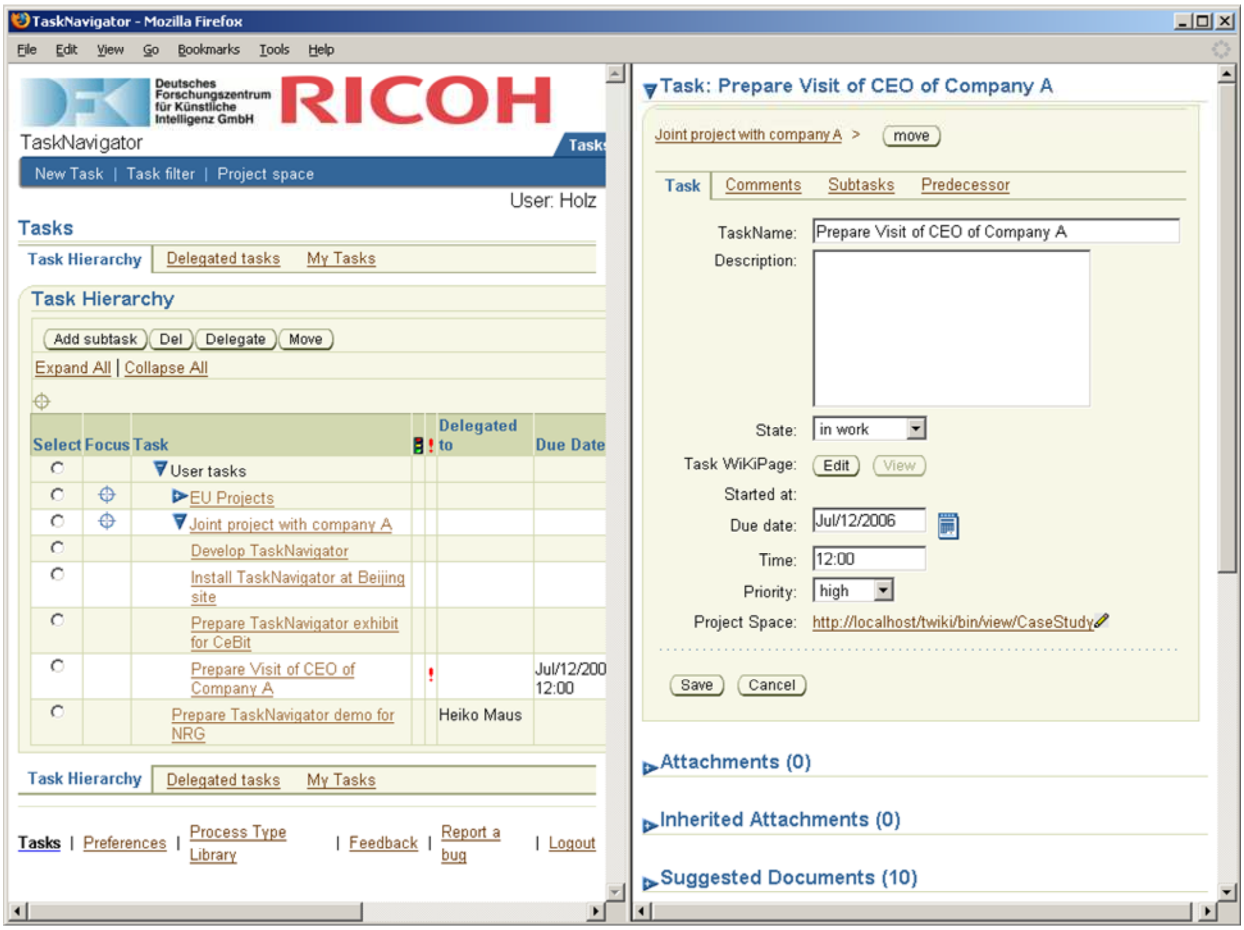

Figure 2: Hierarchical view on the user's tasks

may start working on waiting tasks before preceding tasks are completed.

Tasks in TaskNavigator can be delegated to other users of the system (including subtasks and attached bookmarks). Automatic email notifications are sent to delegates, and the delegated tasks are added to their task lists. TaskNavigator allows delegatees to either accept or reject a delegated task; in either case, the delegator is informed via email and can track the task's current state on his list of delegated tasks.

In order to help users organizing their tasks, TaskNavigator allows decomposing a task into subtasks, resulting in hierarchical work breakdown structures. Figure 2 depicts an example screenshot from TaskNavigator: the left-hand frame shows an expanded task/subtask hierarchy, while the right-hand side shows details for the currently selected task "Prepare Visit of CEO of Company A".

3.2 Building Personal Knowledge Spaces

In order to build a TaskNavigator document repository that contains relevant information, we start with the contents of the company's shared network drives and users' local file systems. In order to exploit the users' native structures, their desktops as well as the structures from the company's shared network drives, we make use of BrainFiler [4], a commercial system which realizes a personalized document management environment, allowing multi-criterial classification of documents, search functionalities such as Boolean search and document similarity evaluation, as well as incorporation of remote (peer-to-peer) BrainFiler instances. BrainFiler enables a user to build a personal information model by allowing to import (and synchronize) native structures such as e-mail folders, bookmarks, and file directories together with contained e-mails respectively documents (see Figure 3). The imported structures are shown as trees (usually interpreted as *is-a* hierarchies) and can be arranged in different views. The meaning of the nodes (interpreted as concepts) is determined statistically by a document term-similarity vector on the basis of the assigned documents

A user is now able to elaborate the personal information model by creating new or rearranging existing structures, building relations between concepts (a concept can have multiple parents), and assigning documents to several concepts (i.e., annotating/tagging a document with concepts).

These structures then can be used for a conceptual search (e.g., all documents annotated with concepts *X* and *Y*) as well as a combination with the keyword-based search (e.g., all documents annotated with concept *X* and containing term *T*).

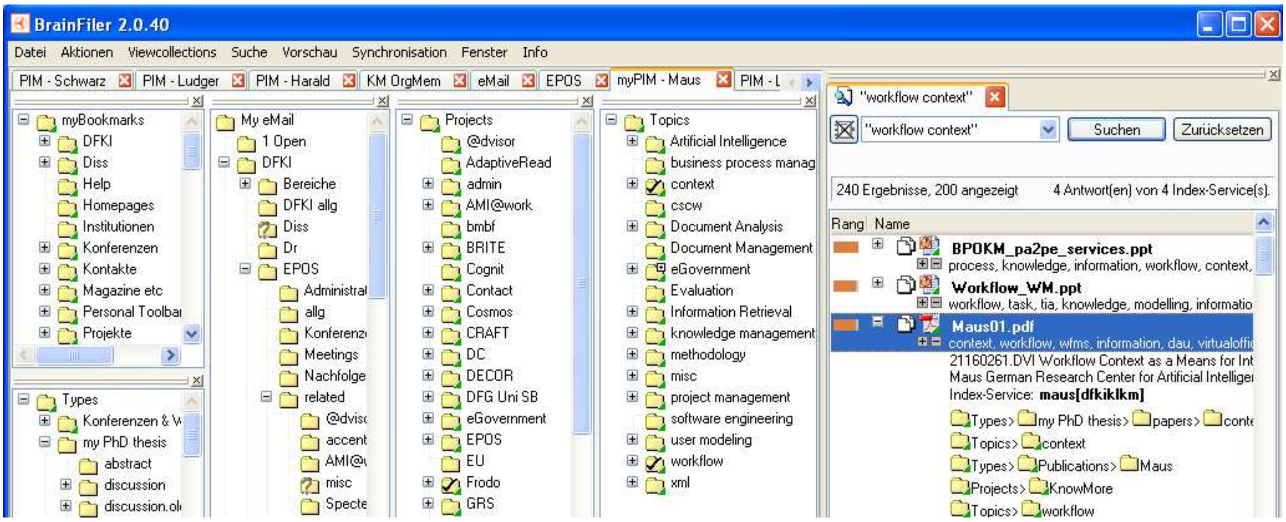

Figure 3: Multi-criterial indexing of documents with BrainFiler

With BrainFiler, the knowledge worker has a personal desktop search spanning nearly all information sources, allowing multi-criterial classification and different views on his personal document collections as well as on those of his workgroups.

The created BrainFiler indexes for personal and company-wide documents provide the basis for task-specific, proactive information delivery.

3.3 Proactive Information Delivery

In addition to task-centered information structuring, TaskNavigator realizes task-oriented proactive information delivery (PID), i.e. the system automatically retrieves potentially relevant documents from various different information sources and suggests these documents to the user (see e.g. Figure 2, pane "Suggested Documents"). For each suggested document, a short excerpt is shown in form of the most relevant terms extracted by BrainFiler below its hyperlink which allows direct access to the document. For document retrieval, TaskNavigator transparently triggers a query to BrainFiler (see Figure 4). The query for a task is determined by the task's name, the task's description, and the attached bookmarked documents. Technically, this is realized by creating a (virtual) concept (see Section 3.2) within BrainFiler for each task, that contains the task's name and description as files, as well as all bookmarked documents. Using BrainFiler's document classification suggestion functionalities, all documents that BrainFiler suggests to be classified under the task-specific concept node (and exceeding a user-defined relevance threshold value) are listed by TaskNavigator as suggested documents for that task.

TaskNavigator also uses the PID component to proactively suggest process know-how in the form of similar tasks or available process types (see Section 3.4).

Figure 5 shows the task detail frame with the expanded Suggested Documents section. Users can modify the query manually by editing keywords or selecting concepts, and can also provide positive or negative relevance feedback on selected documents by pushing "+" or "-" button respectively. As a consequence, TaskNavigator will update the list of suggested documents, taking into account the relevance feedback. If a user considers a suggested document relevant for the current task, he/she has two alternatives to associate the task with the document:

- Making the user's own copy of the document
- Making a link to the document

User and system interaction related to the proactive information delivery is logged by the system in order to evaluate the effectiveness of the PID (see Section 5).

3.4 Process Know-How Reuse

Nowadays, the success of any enterprise heavily depends on the competence and productivity of each employee. The productivity can be drastically increased by reusing process know-how created and gathered within the enterprise over the years [19].

During knowledge-intensive workflows, rigid and agile processes are often interleaved [16]. While modeling rigid processes is supported by state-of-the-art process modeling tools, agile processes are a subject of discussion in current scientific and technological research. Despite the flexible and unpredictable nature of agile processes, it is highly desirable to be able to reuse them, especially for knowledge-intensive work.

The easiest case of process know-how reuse without requiring process modeling efforts is the so called instance-based task reuse. In case of instance task reuse, users are provided with the functionality for retrieving similar tasks (see Figure 8).

In TaskNavigator, similar task search is based on the BrainFiler functionality for finding similar document categories. Every task in TaskNavigator corresponds to a certain BrainFiler category: the similarity between two tasks is derived from the similarity between the corresponding document categories computed by BrainFiler.

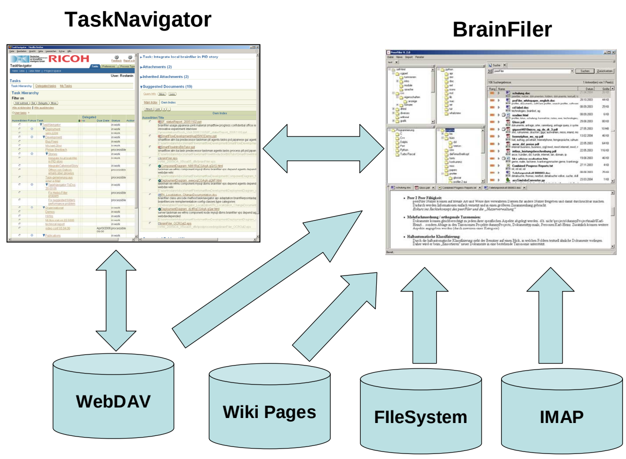

Figure 4: PID with integrated DMS BrainFiler

The information about retrieved similar tasks (i.e. task descriptions, decomposition into subtasks, information items attached to similar tasks) can be adapted and reused during the enactment of the current process.

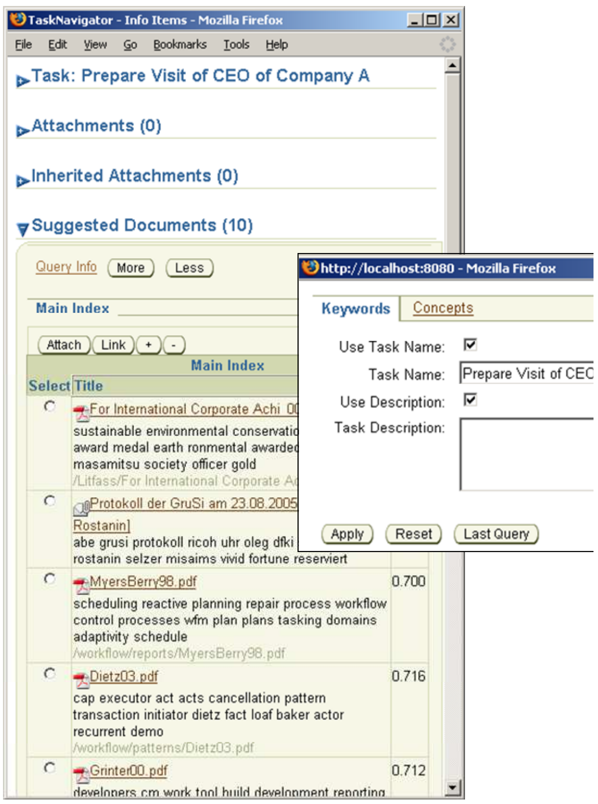

Figure 5: Suggested documents and query editor

Newly created task instances are retained in the respective repository (i.e. the task case base) and can be reused by the user himself or others. By analyzing the rate of different reuse activities, the most frequently reused tasks (typical tasks) can be identified. If these frequently recurring tasks are relevant for the enterprise, and the enterprise is willing to make an investment into process modeling, such typical tasks can be modeled more formally using process types.

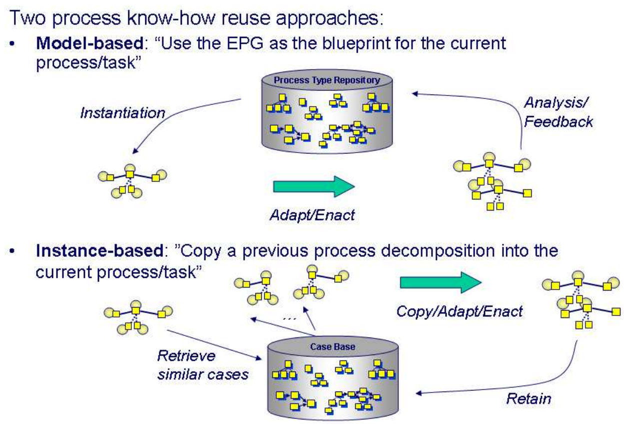

Figure 6: Model-based and instance-based activity reuse approaches.

A process type is a semi-formal abstract description of a process activity containing a number of attributes:

- a textual description of the activity,
- pre- and post- conditions for the activity,
- possible approaches for executing the activity (decompositions)
- activity-related documents.

Every process type decomposition contains zero or more sub-activities. Tasks created as instances of a certain process type inherit the properties of that process type like related documents or possible decompositions into sub-activities. During process type instantiation, users can choose the necessary decomposition of the process type and tailor the task structure according to their situation. Process types are managed within the *Process Type Library* (PTL), a central repository where the available process types are stored and maintained. While the PTL is regarded the center of process know-how, the PTL should be available for all project members with proper access control.

Our suggestion is that by combining agile process modeling, task instance reuse, and proactive information delivery, we provide a basis for effective sharing of process know-how among participants of agile knowledge intensive processes.

3.5 Collaborative Documenting Space

The concept of a Wiki has become a widely used collaboration platform in both small and large institutions. Its flexibility and lightweight characteristics match very well with TaskNavigator concepts.

In TaskNavigator, Wikis enable the following aspects to help users collaborate on tasks and their related information:

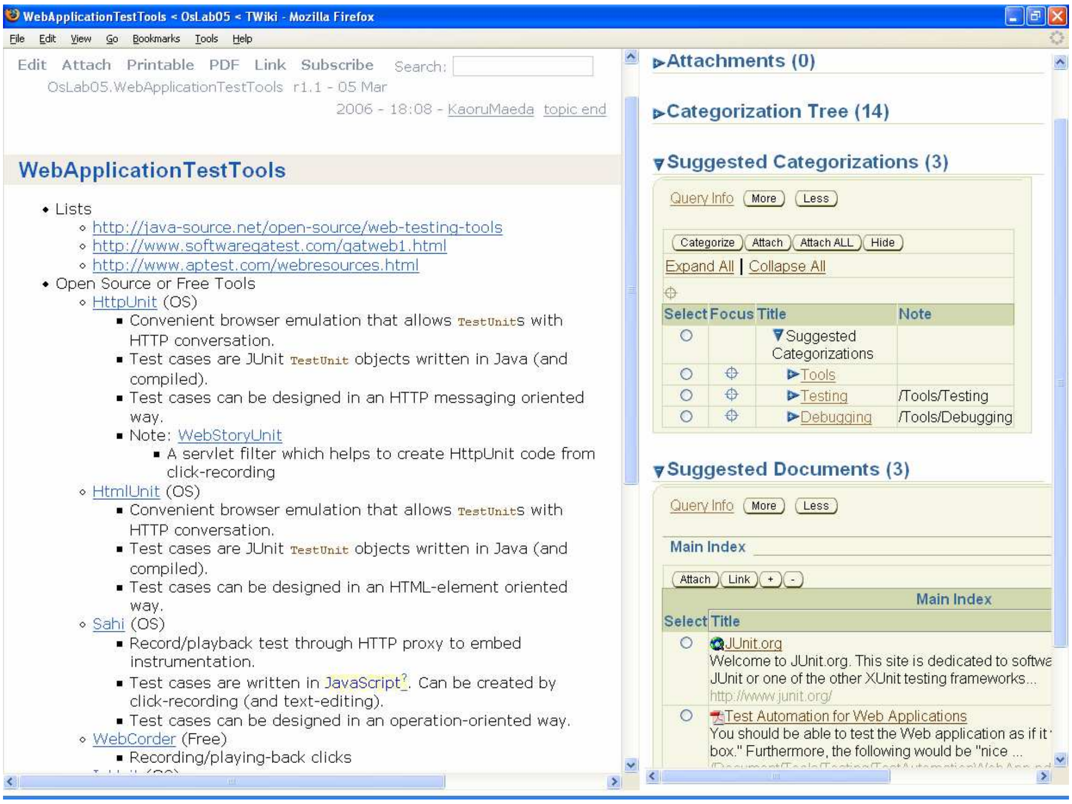

Figure 7: Collaborative authoring platform built upon Wiki and PID functionality

- An additional document repository for PID: TaskNavigator already integrates WebDAV repositories and repositories of uploaded task attachments. Wikis adds another type of document repository that enables embedded cross-reference links and collaborative authoring. Task-subtask relationships can be easily mapped to document-subdocument relationships by representing each section in a document as a Wiki page. Wiki pages are indexed by the DMS BrainFiler (Figure 4) and become candidates for PID suggestions just as documents in other repositories.
- Means for informal process descriptions: Corresponding to each process type, users can associate a Wiki page to have informal process descriptions. Users who are about to instantiate a process type can find its descriptions and how-to's. They are also free to add comments, new suggestions, and additional references. This encourages building and sharing best practices in PTL.
- A platform for task-related discussions: When a task is delegated to a user, discussions between the delegator and the delegatee are facilitated with a Wiki. TaskNavigator provides an easy way to create a Wiki page that corresponds to a task. This "task-specific Wiki page" can be used as a communication platform for all users involved (or as a scratch pad for a single user). When the task is finished, the page remains as a record how the user performed that task.

First experiences with TaskNavigator revealed some shortcoming of our approach:

- While working on a task, typical users spend a considerable amount of time working on documents rather than on the task list. PID is also useful when creating documents. In addition to task-oriented PID a document-oriented PID is necessary.
- Wiki pages should contain functions to categorize and organize pages according to different criteria. This prevents the common "lost in Wiki space" problem. Potential categorizations may come from file/email folders already existing in the personal or shared information space. Means to reuse existing categorizations for Wiki pages are needed.

We extended the PID functionality to solve these problems. PID now suggests related documents as well as document folders when users are viewing/editing Wiki pages. TaskNavigator performs an information retrieval using the page name and its contents as query string. This is our first approach to provide document-oriented PID. Suggested related documents/folders can be attached to the Wiki page like tasks. These new attachments are immediately visible to other users who view the page.

Wiki page categorization is also extended with PID. Wiki categories are now unified with document classification categories in the DMS BrainFiler. PID recommends possible categorizations for the current Wiki page based on the similarity of its contents to Wiki pages that are already categorized. Users can accept the recommendation by clicking a button. This Wiki page categorization is also visible from task-oriented PID. User groups can share and evolve information categorization hierarchies, uniformly from both task- and document-oriented views.

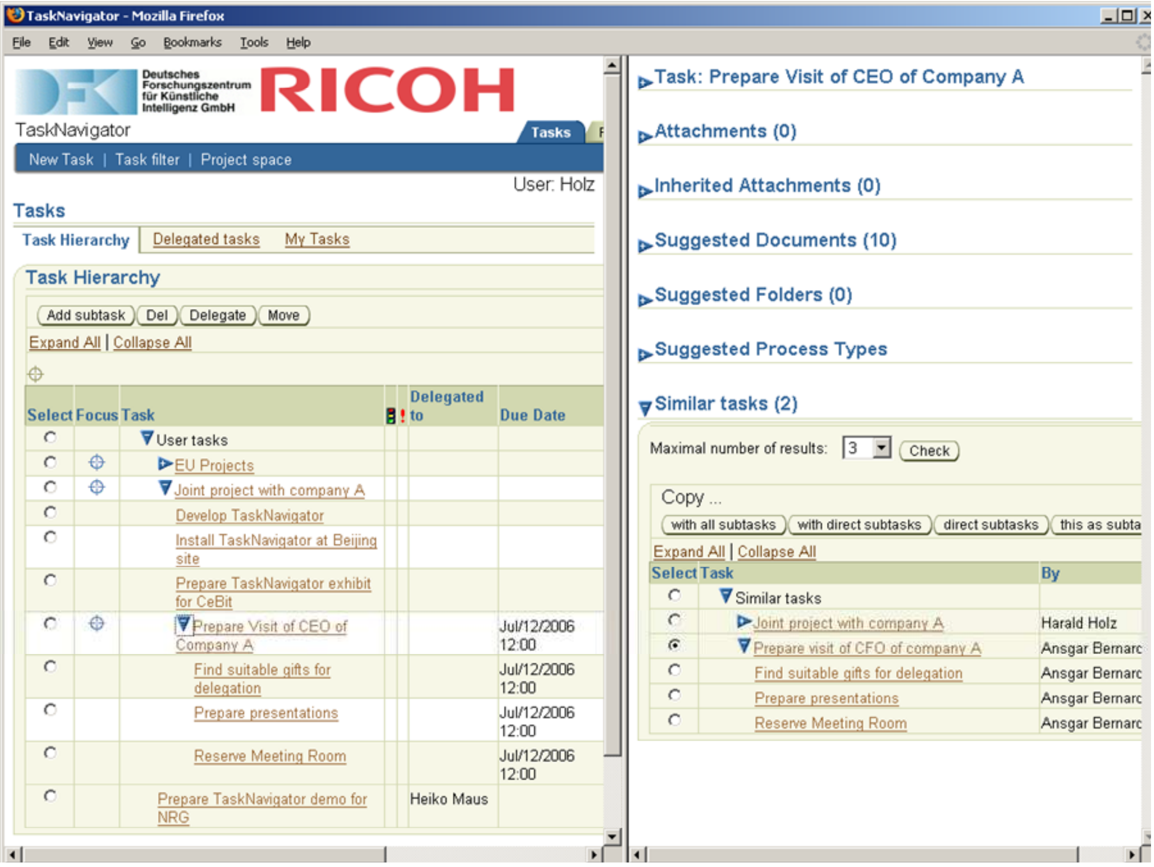

Figure 8 : Similar task retrieval

Figure 7 illustrates how Wiki based PID may help the user. In this example, the user is working on a Wiki page that describes different software testing tools. TaskNavigator searches its repositories to find documents and document folders related to testing. Also, Wiki categories related to testing and tools are suggested as a possible categorization.

4. Usage Example: Preparing a Visit of a Foreign Delegation

Let us assume the following situation: a TaskNavigator user in company A is preparing the task "visit of CEO of company A" that is a Japanese company. The user creates a new subtask "Prepare visit of CEO of company A" in the task "Joint project with company A" (see task hierarchy on Figure 2), and clicks on it to see task details in the right frame. The TaskNavigator system performs a query through the integrated DMS and retrieves documents from the document repository that might be relevant for preparing the visit. For the current task the system has found 10 documents (see "Suggested documents" pane on Figure 5).

Listed on top is an e-mail containing details about the CEO's visit and a scanned article about the CEO. The user decides that these 2 documents are interesting. He can download them or attach them to the task (see attach and link buttons on Figure 5). Since the user has not prepared such a visit before, he wonders if colleagues might have. After clicking the "check" button in the "Similar tasks" pane (see Figure 8) he receives a list of tasks that are likely to be similar to the current one. The task "Prepare visit of CFO of company A" once performed by a colleague looks very similar. The colleague decomposed this task into three subtasks:

- Find suitable gifts for delegation
- Prepare presentations
- Reserve meeting room.

Because all subtasks are relevant for the current visit preparation as well, the user copies this subtasks to his own task list via TaskNavigator. Moreover, for the task "Find suitable present", the user notices that his colleague attached some documents on Japanese business culture that he found useful. Seemingly, he had searched the Internet for related information. The two documents he found and attached are now also available to the current user.

5. Evaluation

In this section, we describe the experiments we conducted to evaluate TaskNavigator's performance. While pilot tests were conducted with students, the feasibility test and the case study were done with office workers.

5.1 Feasibility test

In this section, we explain the experimental settings of the feasibility test and its results. The feasibility test was conducted with computer science researchers at the Knowledge Management department at the DFKI. Around 20 researchers used the TaskNavigator system for their daily task management. They were experienced scientists working with large volumes of information and they understood the main concepts of TaskNavigator - PID and Agile Task Management (ATM).

Since some researchers at the same time intensely used BrainFiler, which is the backend system of the TaskNavigator, we provided the local PID functionality that allows searching documents on the local desktop realized by local BrainFiler installation. The document repository contains over 20000 documents, including papers, specifications, and other shared documents of the group. The duration of the test was two months; the first half was used to get accustomed with the system, during the latter half, we evaluated the users' activities.

We collected several metrics to evaluate the usage of the system functionalities. This enabled us to validate the qualitative feedback. Here we show an excerpt:

- Number of attached information items.
- Number of access to the attached information items.
- Number of suggested information items.
- Number of access to the suggested information items.
- Number of created tasks and delegated tasks.

In combination with basic metrics, e.g. the number of tasks and the total time spent on the system, we can derive useful measures to evaluate the system, e.g. the number of suggested information items per unit time or per task.

A summary of the results of the DFKI feasibility test is shown in Table 1, containing the usage data of around 20 users for their daily tasks. Since the contexts of the experiments are completely different, we cannot directly compare the results from the SRCB case study with the feasibility test at the DFKI.

Table 1 – Excerpt of metrics from DFKI feasibility test

Created tasks	518
Delegated tasks	10
Attached information items	151
Access to the attached information items	163
Suggested information items	30198
Access to the suggested information items	271

The amount of created tasks and the numbers of suggested and accessed documents indicate that TaskNavigator was used intensively. It is remarkable that not all users were equally engaged with TaskNavigator (7 users created 6 or less tasks). On the average, every user created 25.6 tasks. Even after the test was concluded, about 50% of the users (10 people) continued using TaskNavigator for their daily work.

Considering the amount of attached (151) and accessed (163) information items, some information items were used more frequently than others. In addition, the number of created tasks (518) and the accumulative number of suggested information items (30198) indicate that on the average 58 information items were suggested for a single task.

The number of accesses to the suggested information items (271) is considerably smaller than the number of suggestions. This is due to the fact that TaskNavigator suggests a newly calculated set of information items every time a user accesses to a task. Consequently, the accumulative number of suggested information items became large. However, comparing the number of accesses to attached information items (163) with the number of accesses to suggested items (271), the information items suggested by TaskNavigator seems to have been regarded as useful as the information items that were manually attached to tasks by the user.

The large difference between the numbers of created tasks (518) and delegated tasks (10) reveals that the system was mainly used for organizing one's own work. Since the researchers are working in a similar context and have been in touch with each other every day, they might not have needed the task delegation function during the short test period.

5.2 Case study

The TaskNavigator case study was conducted at the Ricoh Software R&D Center in Beijing (SRCB) with experienced researchers as test persons. Here, we tested the requirements analysis activity, which is an example of knowledge-intensive work. Two researchers at SRCB took the role of requirements engineers analyzing requirements of a software product. Three other researchers at SRCB were domain experts to help the two researchers by providing product information. They communicated with researchers in the Ricoh Software Research and Development department (SRDG) in Tokyo, who were customers. In the case study, they analyzed a product with regard to the replacement of an existing component. In this analysis, they could utilize design documents of the existing component. The duration of the case study was three months, the first month was used to get accustomed to the system, and during the latter two months we tested the system. The two requirements engineers provided output documents about use cases and domain models for the analysis. The summary shown in Table 2 shows the activities of the requirement engineers during the last month derived from a questionnaire. We analyzed the results of the metrics and the questionnaire integrally.

Table 2 – Excerpt of metrics from SRCB case study

Created tasks	95
Delegated tasks	20
Attached information items	56
Access to the attached information items	174
Suggested information items	3420
Access to the suggested information items	53

In comparison with the results of the feasibility test, the number of task delegations per user is high. Most of the delegations were for reviewing requirements. The number of the attached information items per task and the number of accesses to the attached information items are also high. The ratio of the numbers of items and accesses shows that the researchers accessed the attached information items 3.1 times on the average. Almost the same number of accesses was realized regarding the suggested information items. As we explained in the feasibility test, the ratio of suggested items and accesses is lower than the one of the attached items. On the average 36 information items were suggested for a task by PID. The higher number of the access to the attached items could mean that the requirements analysts attached very valuable information items to their tasks. Compared to that, there is still room for improving the accuracy of PID. In the questionnaire, one of the two analysts mentioned that the PID functionality was useful during the requirements analysis activities, especially for understanding the problem domain, while the other one complained that information suggested by PID (predefined process information of requirements analysis) was not adequate. This shows that the usefulness of the PID functionality depends on the information repository, and on the expertise of the user. We assume the PID functionality is useful especially for new employees or individuals who are newly assigned to a project.

In the questionnaire, the subjects indicated that the ATM enables them to organize tasks faster and to improve the efficiency and productivity in planning and organizing tasks. In addition, they also stated that the ATM functionality is easy to learn and intuitive. They also mentioned that the TaskNavigator system is easy to use in general, including learning the operations. However, they slightly disagreed to both efficiency improvement and productivity increase by applying TaskNavigator. This could be caused by the quality of the stored information in the repository. For the case study, we prepared requirements engineering process and domain information by conducting pilot tests on requirements engineering with students, and by conducting a survey on the internet. However, it is still very difficult to cover all the relevant information for a certain requirements analysis. Inadequateness of the information in the repository was also noted by one of the requirements engineers.

Aside from the difficulties with the information repository preparation the case study results show that the subjects gave a positive feedback on the system functionalities, the ATM and the PID. This supports our idea that, reusing tasks or task models effective in situations where the integrity of prior information is doubtful.

6. Related Work

The issues addressed by the approach presented here stem mainly from the areas of process-oriented knowledge management and desktop search engines. In the following, we briefly compare existing works with the approach described in this paper. Most work on integrating knowledge management and process support has been done in the field of business processes (see [2] for a recent overview of Business Process-Oriented Knowledge Management). Prominent approaches such as EULE [12], OntoBroker [17], WorkBrain [18], PreBIS [6], or DECOR [1] focus mainly on fairly static (in contrast to weakly-structured) processes with regard to proactive information delivery; hence, they rely on structured task representation and ontologies. Caramba [7] realizes an activity-based knowledge management approach for ad-hoc processes by enabling knowledge workers to link knowledge artifacts to tasks. However, only artifacts that have already been linked to a task are made accessible for the task's enactors; a proactive distribution of potentially relevant artifacts based on the content of artifacts already linked to the task is not provided.

The CALVIN project [10] investigates lessons learned systems supporting the process of finding information relevant to a particular research task. CALVIN learns about information sources by automatically recording cases that represent the consulted information sources. As the user browses for information, the system maintains the user's current research context (e.g., a set of keywords describing the main topics) and compares it with former contexts. If the similarity between the current and a former context exceeds a certain threshold, the resources associated with the former context are presented to the user as relevant in his current context.

Other approaches to provide light-weight, proactive information delivery are based on collaborative filtering (CF) technology, e.g., GroupLens [13] or Entree [5].

Current desktop search engines (e.g., Google Desktop Search, x-friend, MSN Desktop Search) do not yet have a notion of a user's task or some other retrieval context. An exception is blinkx[4] that provides on-the-fly recommendation links to available documents that are relevant to the user's active window (e.g., an open document or e-mail editor).

7. Conclusion

In this paper we presented TaskNavigator, a prototype that realizes a light-weight approach to task-specific, proactive document delivery. The term vector similarity-based approach used here by relying on BrainFiler's functionalities is intended to complement our earlier work on more heavy-weight approaches based on formal process models and ontologies [8][9], which require considerably more modelling effort on behalf of the users. Although we used a flexible workflow management system as a basis for the prototype, the presented approach is also applicable to simpler to-do list applications as found in the personal information management tools (e.g., PDAs) of today's office workers. It should be noted that the approach allows starting formal modeling of processes and information needs at any time, should the company be willing to make such investments.

The current TaskNavigator version has been evaluated in form of several feasibility studies, and one case study in the context of a distributed software development project. Due to lack of a quantitative evaluation, we cannot show statistical significance of the results. However, based on the positive qualitative results collected so far concerning TaskNavigator usage, and taking into account the positive evaluation results already obtained for our process-embedded information support [8], we believe that an efficiency gain can also be achieved in an everyday office setting with the approach presented here, by making documents more easily available during the office worker's tasks, and helping to prevent that relevant documents might be overlooked.

Acknowledgements

Work funded in part by "Stiftung Rheinland-Pfalz für Innovation" (InnoWiss, TEAL). The authors thank Christian Denger and Michael Eisenbarth (Fraunhofer Institute of Experimental Software Engineering) for preparing the questionnaire used in the case study.

[4] http://www.blinkx.com

8. REFERENCES

[1] Abecker, A., Bernardi, A., Ntioudis, S., Mentzas, G., Herterich, R., Houy, C., Müller, S., and Legal, M. (2001). The decor toolbox for workflow embedded organizational memory access. In ICEIS 2001, 3rd International Conference on Enterprise Information Systems, Setbal, Portugal.

[2] Abecker, A., Hinkelmann, K., Maus, H., and Müller, H.-J., editors (2002). Geschäftsprozessorientiertes Wissensmanagement. xpert.press Springer.

[3] Abecker, A., Papavassiliou, G., Ntioudis, S., Mentzas, G., Müller, S., Methods and Tools for Business-Process Oriented Knowledge Management - Experiences from Three Case Studies. In: F. Weber, K.S. Pawar and K.-D. Thoben (eds.): ICE 2003 - 9th International Conference of Concurrent Enterprising, Espoo, Finland, 16-18 June 2003. 245-254.

[4] BrainFiler home page. http://brainbot.com/site3/produkte/brainfiler.

[5] Burke, R. (1999). Integrating knowledge-based and collaborative-filtering recommender systems. In Proc. Workshop on AI and Electronic Commerce.

[6] Delp, M., Bhm, K., and Engelbach, W. (2004). Pre-built information space: Some observations on the challenges of process-oriented knowledge management. In Proceedings of IKNOW'04, Graz, Austria.

[7] Dustdar, S. (2004). Reconciling knowledge management and workflow management systems: The activity-based knowledge management approach. In proceedings of IKNOW'04, Graz, Austria.

[8] Elst, L. v., Aschoff, F.-R., Bernardi, A., Maus, H., and Schwarz, S. (2003). Weakly-structured workflows for knowledge-intensive tasks: An experimental evaluation. In IEEE WETICE Workshop on Knowledge Management for Distributed Agile Processes (KMDAP03). IEEE Computer Press.

[9] Holz, H. (2003b). Process-Based Knowledge Management Support for Software Engineering. dissertation.de Verlag.

[10] Leake, D., Bauer, T., Maguitman, A., and Wilson, D. (2000). Capture, storage and reuse of lessons about information resources: Supporting task-based information search. In AAAI-2000 Workshop on Intelligent Lessons Learned Systems.

[11] Mahe, S., Rieu, C., "Toward a Pull-Approach of KM For Improving Enterprise Flexibility", 1st Int. Symp. on Management of Industrial and Corporate Knowledge (ISMICK'97), Compiègne, France, October 1997.

[12] Reimer, U., Margelisch, A., Novotny, B., and Vetterli, T. (1998). Eule2: A knowledge-based system for supporting office work. ACM SIGGROUP Bulletin, 19(1), 56–61.

[13] Resnick, P., Iacovou, N., Suchak, M., Bergstorm, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In Proc. ACM 1994 Conference on Computer Supported Cooperative Work.

[14] Riss, U., Rickayzen, A., Maus, H., and Aalst, Wv (2005). Challenges for Business Process and Task Management Journal of Universal Knowledge Management. Special Issue on Knowledge Infrastructures for the Support of Knowledge Intensive Business Processes 0(2), 77-100.

[15] Schuett, P., "Wissensmanagement im E-Learning", J.UCS, the Journal of Universal Computer Science, www.jucs.org, as part of a special issue on Knowledge Management in 2003

[16] Schwarz, S., Schwach strukturierte Workflows für das Wissensmanagement in Unternehmen. Master's thesis, Fachbereich Informatik, Universität Kaiserslautern, August, 2000.

[17] Staab, S. and Schnurr, H.-P. (2000). Smart task support through proactive access to organizational memory. Knowledge-Based Systems.

[18] Wargitsch, C., Wewers, T., and Theisinger, F. (1998). An organizational memory-based approach for an evolutionary workflow management system - concepts and implementation. In 31st Hawaiian Int. Conf. on System Sciences, HICSS.

[19] Wiig, K.M. Perspectives on Introducing Enterprise Knowledge Management. In proc. Of the 2nd Int. Conf. on Practical Aspects of Knowledge Management (PAKM98)