
適時情報配信のための簡易コンセプトモデリングとコンセプトを用いたタスクのタグ付け

Lightweight Conceptual Modeling and Concept-based Tagging for Proactive Information Delivery

オーレグ ロスタニン* ハイコ マウス* インヤン チャン* 鈴木 剛** 前田 薫***
Oleg ROSTANIN Heiko MAUS Yingyan ZHANG Takeshi SUZUKI Kaoru MAEDA

要 旨

近年、知識集約的な業務へのサポートに関して様々な研究がおこなわれてきた。これらの中には、ユーザ観察に基づく適時情報配信 (Proactive Information Delivery: PID) から、ユーザの作業プロセスや組織などのモデリングを必要とするワークフローに基づいたサポートまで、様々なアプローチがある。一般的にモデリングを伴わない簡便なアプローチは導入コストが低いが、ユーザが必要とする情報を提示する精度が低く、また不必要な情報を提示してしまうことで、情報過多を助長することもあり得る。逆に洗練されたモデリング過程を持つアプローチは高い情報提示精度を持ちうるが、モデルの生成及び更新のコストに関して大きな問題がある。本稿で提案するアプローチは、タスクに関連した知識とプロセスノウハウの漸増型モデリングを利用することで、これらの問題を解決する。本アプローチでは、コンセプトマップ及びコンセプトを用いたタスクのタグ付けを利用し、PIDの精度向上を図る。ケーススタディの結果により、タスクのタグ付けによってPIDの精度が向上できることが示される。

ABSTRACT

During the last decade, a plenty of approaches for intelligent user assistance in knowledge intensive working environments were developed. These solutions vary from a lightweight proactive information delivery (PID) based on a non-intrusive user observation to workflow-based assistance that requires formal modeling of processes, organizations, knowledge domains and task specific information needs. Whereas lightweight solutions have low precision and sometimes yet increase the user's information overflow, approaches based on sophisticated modeling have severe problems with bootstrapping and maintenance. The work presented in the current paper aims to find an optimal integrated solution for user assistance in agile knowledge working environments that exploits a lightweight incremental modeling of task relevant knowledge and process know-how using concept maps and concept-based task tagging to improve the quality of PID results. The results from our case study show that the task tagging is feasible and facilitates PID quality improvement.

* ドイツ人工知能研究所

German Research Center for Artificial Intelligence GmbH (DFKI)

** 研究開発本部 基盤技術開発センター

Core Technology R&D Center, Research and Development Group

*** グループ技術開発本部 オフィスソリューション技術開発センター

Office Solution Technology Development Center, Corporate Technology Development Group

1. Introduction

During the last decade, a plenty of approaches for intelligent user assistance in knowledge intensive working environments were developed. These approaches can be classified using following two dimensions ¹⁾; i) type of supported processes: from weakly structured to strictly structured and ii) type of information modeling: from lightweight to heavyweight. Knowledge intensive work consists of both strictly structured processes that can be formally modeled and enacted using workflow management systems (WFMS) and agile processes (agile knowledge work, AKW) that are highly dynamic that makes them difficult to be formalized, modeled in advance and later reused ²⁾.

Our current work concentrates on the support for AKW, e.g., developing a software or supervising a scientific thesis. Although AKW is dynamic, it is required to manage it to be successfully completed in time. A task list management (TLM) is powerful tool for flexible time management and planning in AKW environments which allows keeping track of tasks being performed in the environments. The popularity of TLM systems is intensified by effective time management e.g., GTD ³⁾. Despite of agility of TLM, one can find certain regularities implied by tasks and dependencies between them. This fact opens numerous opportunities for process know-how reuse in TLM tools ⁴⁾.

Furthermore, a TLM tool is an ideal place for intelligent information assistance required by a knowledge worker coping with tasks. Advocated in the previous paper ⁵⁾, the method of the lightweight proactive information delivery (PID) realized by connecting a TLM system FRODO Taskman ⁶⁾ and an information retrieval system BrainFiler ⁷⁾. Generally, PID has two main purposes, i) minimize users information overload by providing information precisely adapted to the current task's needs, ii) diminish users' risk of overlooking important documents relevant to their tasks.

One can distinguish between light- and heavyweight PID methods based on the required amount of upfront modeling effort.

There are many works describing heavyweight approaches of learning-on-the-job ^{8),9),10)} aiming to improve users' competency level by providing information according to users' information needs and competency level. These approaches claim to ensure a high precision information delivery. Their major bottleneck is a necessity of relatively large effort on process, user and information modeling that makes them difficult to introduce in an enterprise.

We developed TaskNavigator ^{2),11)} in the joint project Virtual Office of the Future (VOF) of Ricoh and DFKI, and applied lightweight PID based on a TLM system to cope with requirements from knowledge intensive works. The second half of the VOF project was dedicated to find and evaluate means that would allow combining advantages of light- and heavy- weight PID, i.e., low modeling effort and high delivery precision.

In this paper we first analyze problems in the lightweight PID in section 2. Then, task tagging as easy way of task annotation is introduced in section 3. Using concept maps as a means for knowledge extraction and modeling as well as its integration with TaskNavigator for the PID improvement is discussed in section 4. The results of a feasibility test conducted in DFKI that showed positive user feedback and acceptance are depicted in section 5. Conclusion with an outlook is given in section 6.

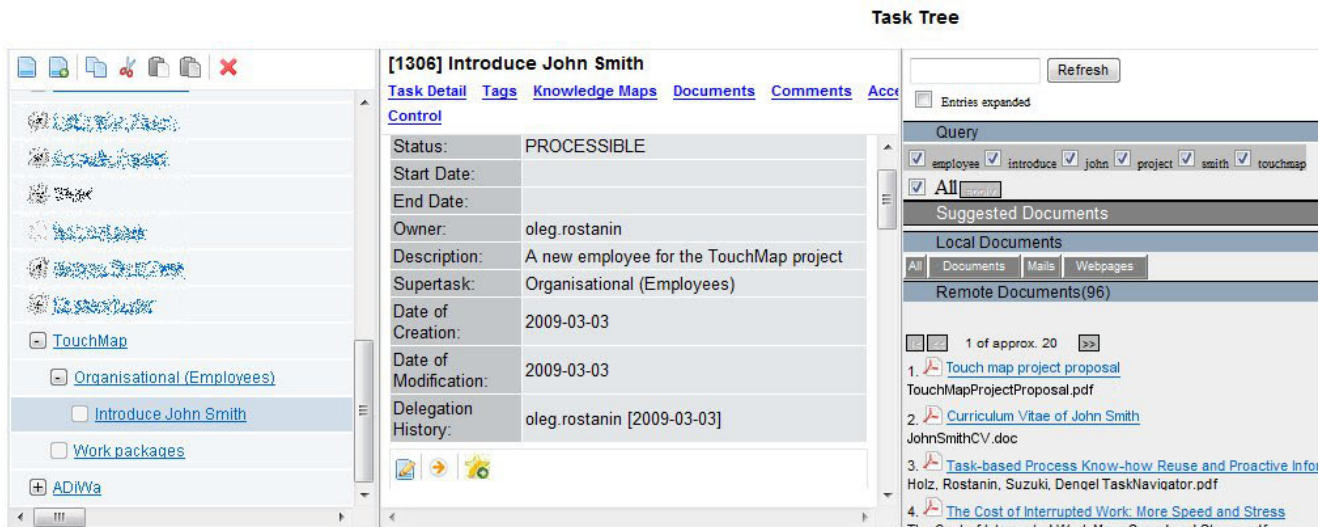


Fig.1 TaskNavigator screenshot.

2. Proactive Information Delivery in TaskNavigator

This section gives a short introduction into the TaskNavigator system and its PID feature. After that, problems in lightweight PID are identified.

2-1 TaskNavigator

TaskNavigator is a novel TLM system providing support for knowledge intensive business processes ²⁾. By the mechanism of task delegation, task comments and notification as well as flexible task structure management implied by work breakdown structure (WBS), TaskNavigator becomes a powerful tool for work coordination and collaboration in small distributed teams. (Fig.1)

2-2 Lightweight PID in TaskNavigator

The unique selling proposition of TaskNavigator is its PID feature. The main idea of PID is to proactively deliver task-relevant information e.g., documents or e-mails related to the task without explicit user request and information modeling in advance. The principle of

lightweight PID is based on the assumption that a context of a task can be described sufficiently by the task information (title, description, etc.) and the information attached to the task. The PID module of TaskNavigator generates a keyword-based query from the current task context represented by a task and its attached documents and sends request to external information retrieval (IR) systems automatically to get task-relevant information. Results from the IR systems are sorted by their relevance score relative to the query and presented to the user in the TaskNavigator GUI (Fig.1, rightmost). The user can navigate in the result list, open documents and attach them to the current task. The current implementation of the system uses the TRMeister database ¹²⁾ for document retrieval from the central TaskNavigator repository in network drives and Google Desktop Search API ¹ to request user's personal documents stored in a local computer.

¹ <http://code.google.com/apis/desktop/docs/queryapi.html>

Example

An example task is given in Table 1

Table 1 example task and generated terms.

Task name	Introduce John Smith
Task description	John Smith is a new employee for the project TouchMap
Comments	Dear Peter, please take this over
Term set	introduce, john, smith, employee, project, touchmap, peter

From the task name, description, and comments, the system automatically generates the term set. The query issued automatically to the search engine on the user's desktop or to the central repository will deliver documents that at most satisfy the given list of keywords, e.g. a description of the project TouchMap or a John Smith's CV (Fig.1, rightmost). These documents can be helpful to successfully and quickly complete the task.

2-3 Problems of Lightweight PID

The first feasibility test of TaskNavigator showed great potential of lightweight PID as well as its bottlenecks^{2),5)}: on one hand, users were often positively surprised for getting useful e-mails or documents provided by PID that would have been overlooked when working on the task. On the other hand, users were sometimes irritated by too many irrelevant documents delivered by PID caused by the query generation from a textual task description. Hereunder we explain the advantage and disadvantage of the lightweight PID.

- *Advantages: The main advantage of lightweight PID is the low level of human effort to make it work: the user just needs to type a new task name in TaskNavigator to get first PID results. A formal modeling of potential information needs of the task is not required (compare to 13)).*
- *Problem (P1): Statistics-based query generation used in lightweight PID can cause unsatisfactory quality of generated queries or search results:*
 - *(P1.1), Weaknesses of the TF/IDF algorithm*
The above example shows that the TF/IDF

algorithm removes the word "new" from the query as it is considered as a stop word. However, in the given context, the word "new [employee]" is essential to express the task semantics.

- *(P1.2), Compound search terms not supported*
Even if the algorithm could identify the importance of the keyword "new" for the current task context, this keyword alone does not have much sense without the combination with the keyword "employee".
- *(P1.3), Verbose descriptions*
Verbose task descriptions sometimes spoils automatically generated query. For the task "Apply a new MySQL-DB¹⁴⁾ for TouchMap weblog" with a verbose description "To install a new WordPress¹⁵⁾ software we need a separate database with the name wordpress on our MySQL server with the user WordPress" would generate the query "apply, mysql, db, touchmap, weblog, wordpress, install, software, separate, database ..." that would result in unsatisfactory PID results: depending on the IR engine, the result set could contain no or too many documents.

Consequently, the quality of results delivered by lightweight PID can be unacceptable in some cases.

Below, we describe an integrated solution of above problems based on concept-based task tagging and lightweight conceptual modeling realized in TaskNavigator.

3. Using Task Tagging to Improve PID Quality

In comparison with the previous approaches e.g.¹⁰⁾, TaskNavigator follows the paradigm of lightweight PID characterized by low level of task context modeling effort. As described above, this could cause problems of a bad

quality of PID results. The objective of the TaskNavigator project was to find an optimal solution that requires a minimally possible modeling effort to achieve acceptable PID results. Our claim here is that:

(C1) introducing implicit bottom-up modeling of the task context by task tagging is feasible and can contribute to the PID quality improvement.

The claim is specified in detail in the following section by introducing tagging technology into PID.

3-1 Using Task Tags for PID Query Refinement

Tagging is a wide-spread technology for lightweight classification and annotation of electronic resources by manually or automatically assigning keywords to them.

Considering tasks in TaskNavigator as resources that are annotated collaboratively by tags, we decompose *CI* into the following sub-claims:

- *(C1.1) Task tags can be used as keywords to refine a search query for task-related PID. Keywords defined by users do not cause problems P1.1 and P1.2 (if multi-word tags are allowed). Moreover, the implicit semantics behind task tags given by users will highlight the most important task aspects suppressing the problem of verbose task descriptions P1.3.*
- *(C1.2) Provided the bag tagging model ¹⁶⁾ is utilized by TaskNavigator, where different users can tag tasks multiple times with the same tag, the popularity/relevance of task-related tags can be used to specify weights of single terms comprising a PID query. A weighted PID query will express the importance of each search term thus yet better defining the task semantics (contributes to solving P1.3).*
- *(C1.3) Provided a list of tags of the parent task is easily available in the current task details, the parent task tags will ease the effort on current task tagging, increasing the system usability.*
- *(C1.4) Task tags can be used to find similarly*

tagged resources (tasks or documents).

In order to implement this new vision on PID, the process of the task-specific information retrieval and delivery will be extended as follows:

- (1) Propose possible tags to the user proactively
- (2) User accepts/rejects tag proposals or tags tasks manually (compound tags are allowed)
- (3) In collaborative task management environment, users can vote for or against the tag assigned by themselves or by colleagues.
- (4) A new PID query is generated by TaskNavigator considering tags and tag votes as (compound) search terms and their weights in the query.

Example

For the task from the previous example, task tagging example is given in Table 2.

Table 2 example of task tagging.

Task name	Introduce John Smith
Term set	introduce, john, smith, employee, project, touchmap, peter
Tag	"new employee", "john smith"

The user accepts the keyword "employee" as a tag and makes a compound tag "new employee" from it. Further, she composes a tag "john smith" from the two auto-generated keywords. These two phrases explicitly defined by the user will be given higher weights than ones for the auto-generated keywords that allow better controlling the behavior of the IR engine and getting more precise PID results.

3-2 Problems of Direct Usage of Task Tags for PID

Although task tagging could solve problems of lightweight PID, there are severe problems going along with tagging such as synonymy (*P2.1*), homonymity (*P2.2*) and polysemy (*P2.3*) ¹⁷⁾. In respect to the information retrieval, the problem of synonymy (includes synonyms, misspelling, different writing styles, and

different languages) is the most critical. The problem of homonymity can emerge, for e.g., if the user tagged a task with "SME" assuming "subject matter expert" but received documents about "small and medium enterprises". The problem of polysemy is sometimes difficult to recognize but it could still spoil IR results: searching for the description of the TouchMap system, the user receives documents about the TouchMap project which are tightly related but still not the same.

3-3 Solving Tagging Problems by Lightweight Conceptual Modeling

A standard way of solving the problem of synonymity and misspelling is to use mechanisms supporting controlled vocabularies during the tagging process (e.g. auto-completion supported by SKOS² or thesauri). To solve homonymy and polysemy problems, more sophisticated ontological modeling of the task-relevant domains can be utilized. A sound modeling of task context is a very complex endeavor that is practically impossible for every task in TaskNavigator. A complete modeling of the task-related knowledge domains that can be used for precise PID⁸⁾ is effort consuming and hardly possible in an open knowledge intensive working environment. As an alternative, we offer a lightweight modeling of tasks and task-relevant knowledge domains that partially includes the functionality of controlled vocabulary. The proposed solution is based on the idea of concept maps and is described below in detail.

2 <http://www.w3.org/TR/2008/WD-skos-primer-20080829/>

4. Lightweight Conceptual Modeling using Concept Maps

Concept map is a methodology for a graphical knowledge presentation created in early 1970s by J. Novak¹⁸⁾. A simple information model behind standard concept maps lacks a control of the vocabulary. Formally defined concept facets described by domains and ranges are also missing, as opposite to ontologies. These simplifications restrict using concept maps in the semantic web applications that require formal inference.

A concept mapping tool LeCoOnt³ developed at DFKI is aimed to combine the graphical expressiveness and intuitiveness of standard concept maps, a simple but well-defined concept model as well as vocabulary control to provide a universal platform for lightweight modeling of knowledge domains and typical user tasks using a concept map paradigm. We claim that introducing such formalisms into the information model of concept maps will suffice to solve problems of tag-based PID described above. Moreover, concept maps can have positive side effects to deeper solution to problems *PI.1-PI.3* as compared to simple tagging.

Fig. 2 shows a screen of the LeCoOnt tool currently editing a concept map "How to introduce a new employee?". The concept map defines a concept "New employee" as well as the task type "Introduce a new employee" consisting of several subtasks. Every concept in LeCoOnt consists of the following fields: i) a unique "uri" that allows to identify the concept; ii) a "stereotype" specifying the concept type; iii) a "label" (textual name, comparable with skos:prefLabel in SKOS notation); iv) "alternative label" consisting of a ";"-separated label list, e.g. abbreviations; v) "translations" consisting of a ";"-separated list of labels in different languages; vi) an informal textual description. The user can freely define relation names between the concepts. To avoid

3 <http://lecoont.opendfki.de/>

redundant relations, the user is supported by an auto-completion feature when creating a new relation between concepts that shows relation types already defined in the LeCoOnt database. Furthermore, LeCoOnt allows associating documents and URLs with any concept (e.g. a wiki page "NeuerMitarbeiter" for the concept "New employee").

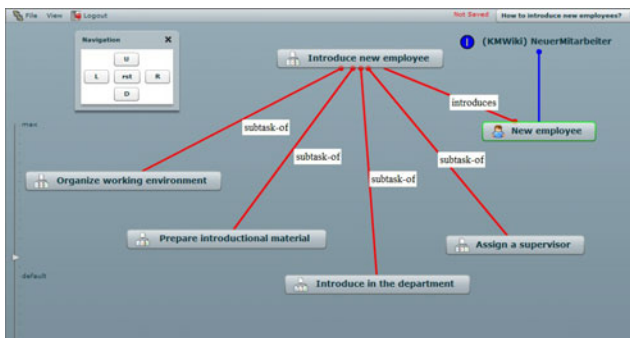


Fig.2 LeCoOnt.

TaskNavigator integrates LeCoOnt as means to control the vocabulary used for task tagging: when the user tries to add a tag to a task that already exists, a LeCoOnt service warns the user and delivers a list of concepts that eventually can be used to tag the task according to the user input (auto-completion function). The user can select a proposed concept as a task tag or create a new tag. This new tag will be stored in the LeCoOnt database as an unbound concept (independent from concept maps) that can be later reused for domain or task modeling.

4-1 Concept-Based PID

Having introduced a controlled vocabulary maintained by LeCoOnt, we are able to use it to identify which concepts from the LeCoOnt database match the current task context and not to rely on unstable results of statistics-based keyword extraction (see *PI.1-PI.3*). Fig. 3 (right bottom) shows a proposal to use the concept "New employee" as a tag for the task "Introduce John Smith". Tag proposals based on the LeCoOnt DB are

generated by an information extraction (IE) engine iDocument¹⁹ that is integrated into LeCoOnt. The user can tag the current task with proposed concepts or attach concept information items to the task. Fig. 3 (left middle) illustrates tags accepted by the user and attached to the task "Introduce John Smith".

Fig.3 Concept-based PID.

Attached concepts (concept-based tags) together with their alternative labels will be used by the PID engine to generate new PID queries. A simple PID query expansion enabled by the underlying concept model will ease the problem of synonymity (*P2.1*) when searching the documents. Relations of a tag to other concepts in the LeCoOnt database can be used to disambiguate meanings of keywords presented by tags and filter the resulting documents set delivered by PID (currently not implemented).

Example

A concept based query generation example is given in Table 3.

Table 3 generating query from concepts.

Task name	Provide introduction material for John Smith
Tag	"Provide introduction material"
Concepts	"Provide introduction material" (not important), "Digital paper"(important),
Automatically generated query	"digital paper" (weight=100), "digitaes papier" (weight=100), "digital" (weight=6), "paper" (weight=4), "digitaes" (weight=6), "papier" (weight=4)

The task "Provide introduction material for John Smith" tagged with concepts "Provide introduction material" (voted as not important), "Digital paper" (voted as important, translation "digitaes Papier") will produce the above query with weights.

In implementing the concept-based PID we used the some heuristics, e.g., tags defined by users as important (just added to the task) get a weight of 100, less important (added but voted negatively) get the weight of 50, and not important (voted more than once negatively) get a weight of 0. In some cases it is necessary to include not only concept labels but also single tokens comprising labels. Such search terms get a weight proportional to the length of term relative to the length of the label. This ratio is divided by 10 to get the final term weight. Heuristics mentioned here were tested during the TaskNavigator feasibility study and were found by users as adequate.

4-2 Using Concept Maps as Lightweight Process Models

Whereas the task tagging represents a bottom-up approach to task modeling, the LeCoOnt tool can be used as means to lightweight top-down task modeling. In Fig. 2 an informal process model "Introduce a new employee" created in LeCoOnt is shown. Having attached the concept "Introduce a new employee" as a task tag, a TaskNavigator user can decompose the task into subtasks according to the task model defined in the concept map. Created subtasks will be automatically

tagged by corresponding concepts from the concept map and inherit information items attached to the concepts.

5. Feasibility Test of Concept-Based PID

In order to show the feasibility of the approach, a case study was conducted at the DFKI knowledge management department. An excerpt of evaluation settings is given in Table 4. The users can be classified into two groups: i) 7 intensive users those who used TaskNavigator for part of their work and initiated 97% of the tasks created in the case study, and ii) the rest with rather short usage period and small number of own created tasks.

Table 4 overview of evaluation.

Subjects	4 students 9 researchers 2 consultants
Duration	3 months
Created tasks	376
Attached documents	624
Attached comments	164
Attached tags	485
Created concepts	70
Notifications	104

The type of tasks conducted with TaskNavigator ranged from personal tasks such as workshop preparation, writing publications, or diploma theses to project tasks such as project organization, proposal preparations, or customer relations. Over the case study period, 458 tags were added to tasks. During task tagging, 70 new concepts were created, thus, the concept base evolved also during task execution both with reusing existing tags and contributing new ones not yet reflected in the concept base.

The breakdown of the given 458 tags is shown in Table 5. Considering both numbers of tasks and given tags, each task got enriched description by 1.2 tags in average. Over 80% of tags were reused by some means, which means a number of tags being used in the system is fairly

maintained to reduce previously mentioned risks. Over half (54%) of the tags are automatically provided by the system. This support by the system facilitates tag use to enrich task context. Finally 24% of the tags were proposed by the concept-based PID and added to tasks by users.

Table 5 breakdown of given tags.

Manually created and added	15%
Manually added (reused) from existing tag clouds	31%
Automatically given by task decomposition	30%
Proposed by concept-based PID and added to users' tasks by users	24%

For the investigated tasks, the subjects compared the query generated from the task's textual context to the query generated from the concepts attached to the tasks. Once tags were available, usually the tag-based query terms were rated better. However, this depends on the preciseness of used task description, e.g., a precise task name "related work on Semantic Desktop" vs. "find related work".

The overall impression of the subjects was, that both, lightweight standard PID and tag-based PID have benefits and drawbacks, e.g., tag-based PID is considered providing better quality of queries than ones from standard PID; however, the former still requires user effort to assign tags to tasks while the latter is fairly automated. Therefore, both approaches should be combined. For instance, standard PID could be used for a task without tags.

6. Conclusion

This paper explains the notion of lightweight PID and its shortcomings and proposes an advanced extension based on both bottom-up and top-down lightweight task modeling that allows solving problems of lightweight PID. As a feasibility test with real users showed, both lightweight and extended PID approaches complement each other and should be used together. Whereas the

advanced PID solves many problems of the lightweight one, a lightweight PID can help to solve the problem of a system cold start specific to tag-based PID: when there are only few concepts available in the knowledge base, standard PID keyword proposals can be used as a basis to create concepts to initialize the knowledge base.

Although the first feasibility test results were positive, a lot of improvement potentials, like better usability and workplace integration were identified by test users. There were also some conceptual aspects that could not be tackled in the project's time frame, e.g., the model of a task context defined in this work considers neither different user skill and knowledge levels nor different user roles⁹⁾. This could be a topic of the following research.

Acknowledgement

Work funded in part by "Stiftung Rheinland-Pfalz für Innovation" (Virtual Office of the Future)

7. Reference

- 1) H. Holz, et. al. (2005) From Lightweight, Proactive Information Delivery to Business Process-Oriented Knowledge Management. *Journal of Universal Knowledge Management*. 2, 2005,
- 2) H. Holz, et. al., (2006) Task-Based Process Know-how Reuse and Proactive Information Delivery in TaskNavigator. In *Proceedings: CIKM '06. ACM Conference on Information and Knowledge Management*, November 6-11, 2006, Arlington, USA.
- 3) D. Allen (2001). *Getting Things Done: The Art of Stress-Free Productivity*. Penguin Books. ISBN 0-14-200028-0.
- 4) U. V. Riss, H. M. Jarodzka and O. Grebner. (2007). *Pattern-based task management & implicit knowledge - How to mobilize implicit knowledge*. 4. *Konferenz Professionelles Wissensmanagement*. Potsdam.

- 5) H. Holz, et. al., (2005) A Lightweight Approach for Proactive, Task-Specific Information Delivery In: Proc. of I-KNOW'05 - Special Track on Business Process Oriented Knowledge Infrastructures (BPOKI)
- 6) L.v. Elst, et. al. (2003) Weakly-structured workflows for knowledge-intensive tasks: An experimental evaluation. IEEE WETICE Workshop on Knowledge Management for Distributed Agile Processes (KMDAP03). IEEE Computer Press.
- 7) BrainFiler is a software product of the brainbot technologies AG, Germany.
(<http://brainbot.com/site3/produkte/brainfiler>)
- 8) C. Christl, et. al. (2008) Deploying semantic web technologies for work integrated learning in industry. A comparison: SME vs. large sized company in: Sheth, A. et al. (eds): Proceedings of the ISWC 2008, 7th Intl. Semantic Web Conference, Karlsruhe, Germany, Oct 26-30, 2008, pp. 709-722, Springer
- 9) O. Rostanin, et. al., (2006) Project TEAL: Add Adaptive e-Learning to your Workflows. In Proc. of I-KNOW'06, 6-8 Sept. Graz.
- 10) A. Schmidt, S. Braun, (2006). Context-Aware Workplace Learning Support: Concept, Experiences, and Remaining Challenges. First European Conference on Technology Enhanced Learning - EC-TEL 2006, Crete, Greece. Nejdl, W., Tochtermann, K. (eds.). LNCS vol. 4227, Springer, 2006, pp. 518-524.
- 11) T. Suzuki, et. al. Agile Workflow Management and Proactive Information Delivery in TaskNavigator, Ricoh Technical Report 32 (2006)
- 12) T. Ikeda, et. al., (2005) TRMeister: a DBMS with High-Performance Full-text Search Functions. 21st Intl. Conference on Data Engineering (ICDE 2005), 2005 IEEE.
- 13) A. Abecker, et. al. (2000) Context-Aware, Proactive Delivery of Task-Specific Information: The Know-More Project. DFKI GmbH. Intl. Journal on Information Systems Frontiers (ISF) 2 (314); Special Issue on Knowledge Management and Organizational Memory. Kluwer 2000. pp. 139-162
- 14) MySQL is a trademark of MySQL AB
- 15) WordPress is a trademark of Automattic Inc.
- 16) Cameron Marlow, et.al., HT06, Tagging Paper, Taxonomy, Flickr, Academic Article ToRead: <http://www.danah.org/papers/Hypertext2006.pdf>
- 17) S.A. Golder, B.A. Huberman, (2006) The structure of collaborative tagging. Journal of Information Science, 32, 198-208, pp. 101-127.
- 18) J.D. Novak, (1998). Learning, creating, and using knowledge: Concept maps as facilitative tools in schools and corporations. Mahwah, NJ: Lawrence Erlbaum Associates.
- 19) B. Adrian, A. Dengel. (2008) Believing Finite-State cascades in Knowledge-based Information Extraction. Dengel, A. et al. (eds.): KI 2008: Advances in Artificial Intelligence. Springer.