Using Wikis to Tackle Reuse in Software Projects

Jörg Rech, *Fraunhofer Institute for Experimental Software Engineering*

Christian Höcht, *University of Kaiserslautern*

Volker Haas, *brainbot technologies AG*

Software projects in Small and Medium-sized Enterprises (SMEs) are faced with more or less similar types of products (e.g., requirements) when building interactive software systems. The work products from these projects, the experience with these products, as well as the associated products (e.g., test cases) constitute pre-existing knowledge that can be reused in new variants of a software system. We report on parts of the results of the research project RISE (Reuse in Software Engineering) with two German SMEs over the course of two years. Wiki-based technology was used to improve the reuse of this pre-existing knowledge with limited resources with respect to available staff members, time, and money.

In particular, the project was concerned with the question of how to deal with problems in eliciting, documenting, and applying requirements without specialized staff. Wikis were used as they easily allow creating, changing, and deleting content over the Inter- or Intranet. They are often used in software engineering to store project-related information [9] or other content [8].

The result of this project showed that various challenges exist regarding the reuse of requirements knowledge in these small and medium-sized software enterprises.

Software Development in Germany

A fairly recent study in the German software industry [6] showed that a large amount of software companies (98%) in Germany are SMEs, with 77% being micro-enterprises (i.e., with less than 10 employees) and 16% that are called small enterprises – based on the EU classification for companies with 10 or more, but less than 50 employees.

Another recent study [13] in the German software industry indicates that most of the SME companies use agile or lightweight development methods to develop, maintain, and evolve their software. Software engineering

solutions and best practices such as large "heavyweight" or process-driven methods are typically not used due to high costs and a lack of knowledge.

Furthermore, the study showed that many companies (72%) are using an Office Suite (e.g., Microsoft Word or Sun/Open Office) to write their requirement documents, 28% are using no tools for requirements, and only 22% are using specialized tools such as Requisite Pro™ (IBM), Caliber™ (Borland), or DOORS™ (Telelogic) (multiple answers were allowed).

This lack of systematic and specialized tools results in informal, incomplete, and inconsistent requirement documents that are hard to reuse in new projects. Nevertheless, 83% of the participants of the study clearly stated that they often or very often search for reusable documents for their work. Especially in requirement phases, this can be a tricky endeavor, given the fact that 77% rate the completeness of the information in the company as not good, 66% rate the understandability as not good, and 40% rate the correctness as not good. This lack of support and quality in reusing information, esp. in requirements engineering, is also supported by the statement that this phase is the second most requested phase, where 43% of the people have a need for assistance – only programming was requested more often with 61%.

Additionally, in several of our projects with industrial partners, we observed that although the importance of usability is broadly and increasingly accepted, user-centered design processes are hardly used by SMEs in the software industry. Small and medium-sized enterprises often fail to apply user-centered design methods due to additional cost, missing guidelines, and poorly established role models. Even if user requirements are specified, the problem that remains is to guarantee that they are strictly followed during production.

Reuse Challenges

The RISE project (see sidebar "The RISE Project") was started to support software engineers through the reuse of didactically enriched software artifacts from all software development phases in SMEs (Small and Medium-sized Enterprises). Generally, reuse is directly related to extent of existing software parts being used in other systems [15]. However, reuse in this project implies more aspects and does not only apply to software artfiacts:

- What kind of artifacts already exist at the company and how useful are they for different purposes?
- How accessible and searchable are existing artifacts by employees?
- How easily may new artifacts be created and exisiting artifacts modified?

The project research was initiated by the observation that most people in the organizations involved were not pleased with the structure of the knowledge base. In order to elicit the status quo of the industrial partners in RISE, we conducted an in-depth context evaluation. For the execution of the context

Context Of The RISE Project

The project "RISE" (Reuse In Software Engineering) was part of the research program "Software Engineering 2006" funded by the German Federal Ministry for Education and Research (BMBF) that started in June 2004 and ended in December 2005.

The goal of this research project was to improve the reuse of artifacts in software engineering (with a focus on the requirement phase) and brought together researchers from social science (the Department of Educational Sciences and Professional Development at the Technical University of Kaiserslautern) and computer science (Fraunhofer Institute for Experimental Software Engineering (IESE) and the German Research Center for Artificial Intelligence (DFKI)) with industrial partners (empolis GmbH – a small enterprise with a development team of about 40 developers and brainbot technologies AG – a micro-enterprise with approx. 4 developers). In the RISE project, we developed a system for the reuse of software engineering products such as requirement documents. The methodology and technology developed make it possible to share knowledge in the form of software artifacts, experiences, or best practices based on pedagogic approaches.

During the project's lifetime we developed several systems for reuse-oriented organization of knowledge in software organizations on the basis of WIKI and Web 2.0 technology [1]. These systems were targeted towards ease of use and improved the acceptance by the users [2].

References

1. Rech J., Ras E., and Decker B., "*RIKI: A System for Knowledge Transfer and Reuse in Software Engineering Projects,*" in Open Source for Knowledge and Learning Management: Strategies beyond Tools, M. D. Lytras and A. Naeve, Eds.: IDEA Group Publishing, 2006.
2. Höcht C., and Rech J., "*Human-centered Design of a Semantically Enabled Knowledge Management System for Agile Software Engineering,*" in Open Source for Knowledge and Learning Management: Strategies beyond Tools, M. D. Lytras and A. Naeve, Eds.: IDEA Group Publishing, 2006.

evaluation, a period of approx. three months was required – taken into account that not all employees are continuously available.

This evaluation revealed that the partner organizations used multiple repositories to store information about their systems, projects, and customers. Internal information sources (i.e., repositories) with project-relevant information are task cards on a physical blackboard, change tracking system, project folders, universal and private network drives, e-mails, or chat tools (e.g., ICQ) with information in files such as plain text or Microsoft Word documents. For some employees, it was not clear where to store information, as it could be spread across or duplicated in multiple repositories, for example, in the versioning system, change tracking system, or the code itself. Some people even used a Wiki of their own that contained partially documented concepts, ideas, comments, and larger documentations about planned as well as about already developed software systems. Nevertheless, it was not widely used, as its existence was not known by many people and since personal bias existed towards new technologies.

Based on the evaluation of the partners in RISE as well as based on previous projects with industrial partners, we collected the following list of challenges for SMEs in the software sector. These apply to the knowledge transfer and management processes as they were being lived in the organization prior to the introduction of the RISE platform; they were determined by the context evaluation as follows:

- **Recording of information** is limited to few people in the organization and the documented information is only partially complete, consistent, or valid.
- **Reuse of knowledge** is minimal, as the content of the documents has inconsistent structures, incomplete descriptions, or is plain outdated.
- **Locating information** is problematic as it is distributed across several sources with different search interfaces and techniques. For example, important information about the software system itself is documented in text documents, an intranet system, and the source code itself (in this case: Python and ePyDoc). To extract this information, one has to check out the whole software system from the versioning system and search for relevant information on the file level.
- **Sharing knowledge** is cumbersome; there are no templates, guidelines, or checklists to validate if the recorded information has some quality and might be easily reused by one's colleagues.
- **Confidence in the knowledge transfer system** and motivation to share is low as only few people create shareable documents, and if they do, these documents are mostly not accurate or up-to-date. Neither were the authors (resp. other observers) informed about changes made to the content of the system nor were they informed about changes to the navigational or divisional structures (i.e., chapters and sections) of the content. Nevertheless, people would like to share their knowledge in a more persistent way.
- **Workflow for reuse of content** and getting an overview is slow and typically demotivating, as multiple sources have to be searched manually and documents belonging together are not grouped or linked. The discovery of relevant information is perceived as complicated as it is distributed across multiple repositories that all use different (or no) search mechanisms. Furthermore, the search for information in the intranet and file system is term-based (i.e., no stemming or Boolean operators). This was perceived as insufficient and demotivating by the users.

Furthermore, SMEs have several constraints that distinguish them from larger companies. The most critical challenge in German and probably in all SMEs is the lack of money. From this follows that they cannot (or will not) afford the full set of tools that would support the systematic and formal storage of documents such as requirements, test cases, inspection protocols, design, project status, or customer information. Additionally, small software organizations cannot (or will not) afford to implement "best practice" methods available for systematic software engineering. The employees have to have enough motivation of their own to do what is necessary. No one will

perform a task that is not necessary or that is time-consuming without a (relatively) direct or clear benefit. This is also backed by results from other studies by Lubars et al. [10], Emam et al. [4], Nikula et al. [12], and Hofman & Lehner [7], who found that no or only very general (and already available or cheap) tools such as office suites or web sites are used in requirements engineering. As mentioned by Ian Sommerville [14], the requirements process often consists of brainstorming sessions with short informal descriptions on cards, blackboards, or in text documents.

While the tools and methods have already been researched to support the reuse-oriented documentation of knowledge, they still are not implemented as an easily applicable, usable, and integrated framework.

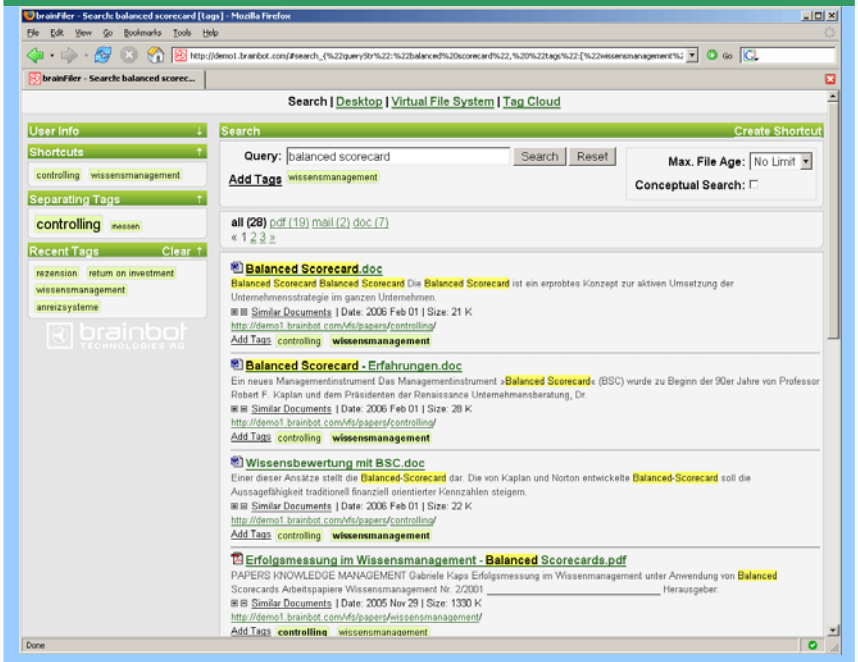
A Wiki-based Solution

We approached these challenges by developing an enhanced Wiki-centered Framework [9] that was targeted to act as a knowledge sharing platform for software development teams with fast and liberal access to deposit, mature, and reuse experiences made in software projects. First, this enhancement could mainly be reached by providing a very flexible and powerful search functionality. Our so-called "Riki" system was extended by search technologies using case-based reasoning technology and ontologies to provide a formal and consistent framework for the description of knowledge and experiences. As a consequence, software developers are now not only able to search the content of Wiki pages, but are also able to find files within other information repositories such as the shared file system.

Second, the RISE platform contains a unified system that allows software developers to annotate any type of content with so-called "tags". Tags have become very popular with the rise of Web 2.0 and various new applications are adopting this principle. Tags can be easily attached to any type of content using the Riki platform. In our project, we decided to use shared tags: once a tag has been attached, it can be seen and modified by all users. This solution was favored in order to support a common vocabulary about artifacts within software development teams. Although this has been considered as being crucial for successful development processes, employees still have the opportunity to use their own "language" by using private tags.

The content in the Riki consists of information on products (requirements, designs, documentations, component interactions, presentations, demo systems), projects (use cases, user stories, test cases, ideas), contacts (employees, customers, partners, suppliers), as well as individual blogs of the employees about

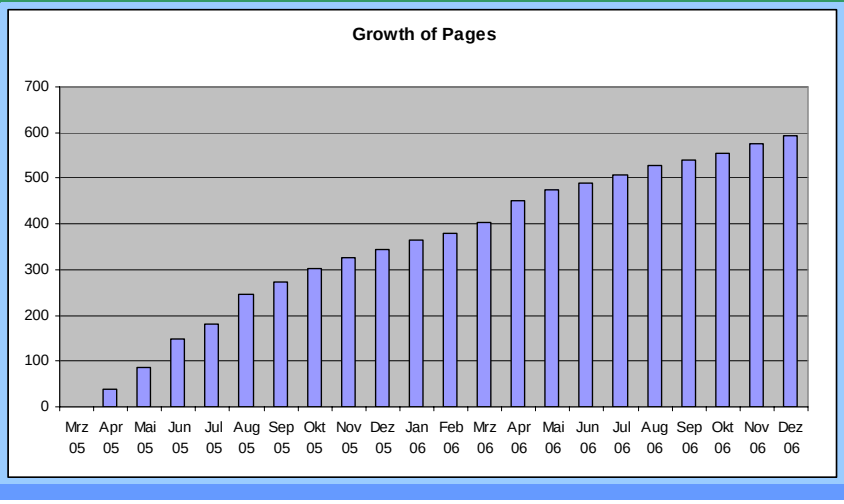
The brainbot brainfiler



Some ideas and concepts developed during the RISE project were used by the project partner brainbot technologies AG within the software ***brainfiler Organizational Memory Server***. The brainfiler Organizational Memory Server is a tool that provides consistent, conceptual organization and intelligent retrieval of information. The unified **access** to distributed data repositories like email, fileservers, issue-tracking systems, versioning systems, and WIKIs is a key feature of the brainfiler software. Documents may be **structured** by tags – this offers a consistent means to collaboratively build up user-centered structures of the information space within a company or a project team. The collaboration of teams – especially in the context of agile software development – is improved: documents and their associated tags can easily be **shared** among authorized users with the web browser. **Finding** relevant documents is facilitated by a keyword search in all accessible documents, a document-based similarity search, and the structural information contained in the tags.

The resulting solution is a system that facilitates accessing, structuring, sharing, and finding information in large, heterogeneous, and distributed document repositories.

Chart 1: Build-up of content

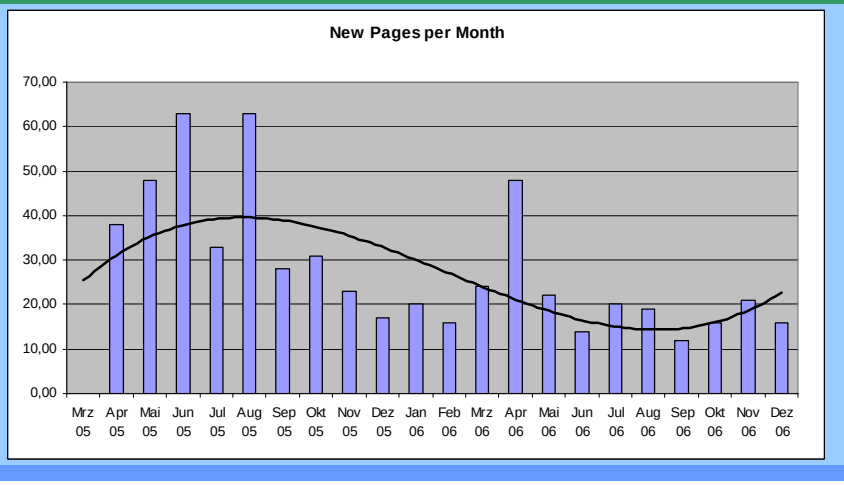


software technologies, software development, etc.

Ontologies and templates enrich the Riki content with semantics that enable us to didactically augment the knowledge within the Riki with additional information and documented experiences. Additionally, the usage of metadata enables the users to build up and use their own individual ontology (in the form of individual tags) that is not bound to compromises or constraints from universal ontologies that might have been constructed in advance. Furthermore, metadata from the universal ontology (i.e., specified beforehand or as defined by other users) was partially used in their own ontology.

Within the Riki platform, the results are clustered by this metadata and the metadata can be used to refine search queries. Annotated pages that were listed in search results remind the users that they have already read them or that they are highly valuable and should be read again.

Chart 2: Growth of content

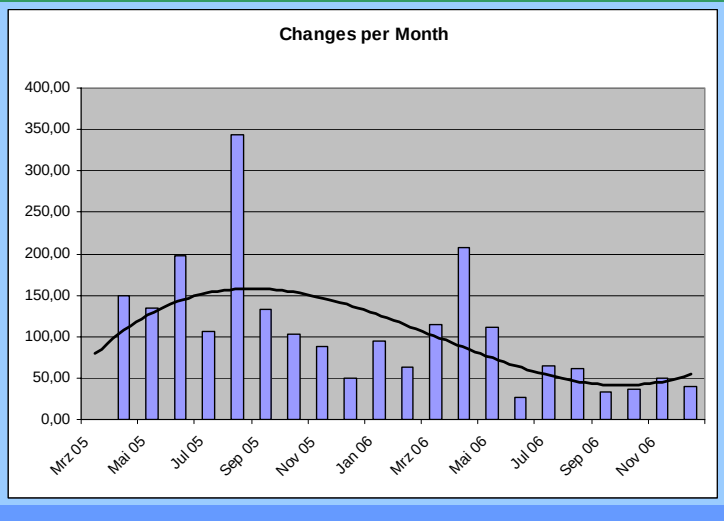


Benefits

In comparison to the status determined at the beginning of the project, the usage of the Riki platform had several effects on the reuse of information in organization. Based on the reuse challenges that were identified at the beginning of the project, we tried to identify the effects of the Riki system on those factors. As elicited with a qualitative group interview, as well as a system log analysis, in the micro-enterprise brainbot (approx. 4 full-time users) we identified the following effects of the Riki platform:

- **More reuse of content** from the RISE platform, as users are more likely to share and search for content. As depicted in Chart 3, the changes made to the content started with 38 pages per month, varied between 27 (min) and 344 (max), and have an average of 105.24 changes per month (i.e., approx. 26 changes per user per month). A peak in August 2005 results from explicit maintenance activities of the content in the system. Two break-ins in July 2005 and June 2006 were caused by holiday season in Germany. Additionally, the trend line (polynominal of 3rd order) depicts the increase of activity at the start that swings into an average value of about 47.33 changes in the last six month of 2006.
- **More data stored** in the RISE platform as technological and social barriers were reduced. Chart 1 shows that the content in the RISE platform grew from zero pages in month 1 to 592 pages after 21 months (450 pages were migrated from the old systems and are not incorporated in Chart 1). The growth as depicted in Chart 2 started with 38 pages per month, varied between 12 (min) and 63 (max), and has an average of 28.19 pages per month (i.e., approx. 7 pages per user per month). Two large projects in August 2005 and April 2006 resulted in a rise of pages and activity. The trend line (polynominal of 3rd order) depicts the in-

Chart 3: Change of content



crease of activity at the start that swings into an average value of about 17.33 new pages in the last six month of 2006.

- **Higher confidence in the system** and increased motivation, as content is easier to find (e.g., by accessing different repositories as well as Wiki pages with unique URLs) and more people are participating in the sharing process.
- **Less barriers for sharing knowledge,** as it is easier and faster to enter information. However, this typically results in low quality of the content. Nevertheless, most users embrace the Wiki idea and record even preliminary information that is revised, commented (e.g., tagged), or removed by themselves or others over time.
- **Faster workflow** for reuse of content and an improved overview due to the integrative view across multiple systems (e.g., file system, e-mail, etc.) in a central, integrated repository.

During the group discussion concerning the usage of the Riki as well as the knowledge culture and transfer, we noted the following characteristics that the users liked very much about the Riki:

- Metadata elements (tags) that can be placed by every user (such as keywords in the form of tags) can be very helpful in indexing the content of a Riki, and the users reacted very positively to being able to index every page with their own metadata.
- The usage of metadata enabled the users to build up and use their own individual ontology (in the form of individual tags) that is not bound to compromises or constraints from universal ontologies that might have been constructed in advance. Furthermore, metadata from the universal ontology (i.e., specified beforehand or as defined by other users) was partially used in their own ontology.
- Searching the information stored in the Riki is widely accepted by the users when the metadata might be integrated into the search process. In the Riki system, all results are clustered by this metadata and the metadata might be used to refine the search query. The search technology exploits co-occurring metadata from multiple users and applies "collaborative filtering" techniques (i.e., "metadata x you search for is also called y").
- Annotated pages that were listed in search results remind the users that they have already read them or that they were highly valuable and should be read again. Similar to the Memex concept by Vannevar Bush (cf. *http://en.wikipedia.org/wiki/Vannevar_Bush*), the metadata might even be used to record a reading sequence for oneself.
- The integrated view a Riki gives on the information in the organization enables the user to send links that are not subject to change, such as mounted devices in the Windows file system.

However, the problem of locating all information in a software organization is currently still a challenge that has its main root in the inhomogeneous tool infrastructures of today's SMEs.

Summary and Outlook

Reuse in general aims at improving productivity and quality by reusing previously developed, standardized, or "off-the-shelf" products (e.g., software components), which have been thoroughly tested or analyzed and possess high quality. Beside the reuse of components, all other software products such as requirements, test cases, or experiences might be reused in software development.

Based on experience from this and several other studies indicate that SMEs typically have a problem with the systematic and formal documentation of knowledge from the software development process. Finally, we observed these challenges and potential solutions for SMEs:

- Information and knowledge about the software projects and products exist during the runtime of a project but get lost soon after its end. Adequate documentation should either be supported automatically or semi-automatically by using a single point of consistent knowledge (SPOCK) to simplify storage and retrieval.
- The Wiki technology has a low usage threshold and enables the simple storage and retrieval of information during a project. As indicated by chart 1 people like to share and reuse information via the wiki - the amount of pages steadily increased over a period of 21 month. However, the informal storage, e.g., of user requirements, opens the door for inconsistent, incomplete, and generally low-quality documents. Nevertheless, most users embraced the Wiki idea and recorded even preliminary information that is revised and improved by themselves or other users over time. Another approach is the use of incentive systems [5] to motivate the sharing of knowledge. In this context, one could also question the overall quality and reliability of that kind of peer-produced content in Wiki-like systems. Paul Duguid examined those "laws of quality" and found that peer-produced projects constantly change: "What is flawed today may be flawless tomorrow" [3]. During the RISE project, we could not address this topic thoroughly enough. But our findings indicate that the overall quality or value of an information space is judged by the users themselves. The users are able to comment about artifacts and evaluate them individually (e.g., by using tags). Incomprehensible, out-of-date, misleading, or unhelpful documents are respectively judged accordingly and are either improved or commented (e.g., using tags).
- Moreover, usability engineering processes and software engineering processes are still not linked to each other. Like Burmester et al. [1], we are strongly convinced that the efficiency of software development processes can be raised significantly by integrating software engineering and usability engineering activities. Therefore, many artifacts are produced, but not actually used, during the production of software systems. We showed how technologies such as the Riki system can support the systematic reuse of

Similar Projects

The two research projects "USEKIT" (http://www.usekit.de) and "ReqMan" (http://www.reqman.de) funded within the same program deal especially with the aspects usability and requirements. The project "USEKIT" [1] aims at an integrated approach for the user-centered design of business critical applications. USEKIT tries to develop and evaluate a user-centered development process that takes into account the special needs and circumstances of small and medium-sized software companies. Another project called "ReqMan" [2] follows aims that are rather similar to the project "USEKIT": the development of an integrated process framework that is suitable for SMEs.

A project called RedSeeds [3], which started in the fall of 2006, has a broader target. The main objective of this project is to create a case-based open framework consisting of a scenario-driven development method (precise specification language and process for the "how-to"), a repository for reuse, and tool support throughout. A reusable case will be a complete set of closely linked (through mappings or transformations) software development technical artifacts (models and code), leading from the initial user's needs to the resulting executable application.

Furthermore, a project with the same name as ours (RiSE [4]) was conducted in Brazil with the goal of eliciting non-technical and technical success factors and best practices related to software reuse [5] and to develop a robust framework for software reuse.

References

1. http://www.usekit.de/
2. http://www.reqman.de/
3. http://www.redseeds.eu/
4. http://www.rise.com.br/
5. E. S. de Almeida, A. Alvaro, D. Lucredio, V. C. Garcia, and S. R. de Lemos Meira, "RiSE project: towards a robust framework for software reuse," in Proceedings of the 2004 IEEE International Con-ference on Information Reuse and Integration, 2004. IRI 2004., 2004, pp. 48-53.

existing artifacts even in small and medium-sized companies.

- If software products get tested too late, these artifacts can only be integrated into the software system with difficulty. Reuse-oriented Wikis enable project members not only to easily produce documents about usage processes or design decisions, but also to integrate them as early as possible during development. This is considered to be very valuable for development teams as far as we observed in our projects: although user-centered design processes can hardly be found in the "real world" of SMEs, these companies are strongly interested in dealing with user requirements as comfortably as possible.

As Carroll states, scenarios are work-oriented design objects: "*They describe systems in terms of the work that users will try to do when they use those systems. A design process in which scenarios are employed as a central representation will ipso facto remain focused on the needs and concerns of users*" [2]. So based on requirements artifacts and scenarios, software development is increasingly becoming a design job that is dominated by creative decisions and has rather well established techniques and processes to transfer requirements into code. Once there are enough artifacts that document requirements, a further step could be to produce usage scenarios. The integration of reuse in traditional process-oriented organizations is typically bound by other success factors [11].

However, we observed that in many companies, hardly any software engineering processes can be found at all. Thus, it should be at least guaranteed that information that is potentially of interest and useful at a certain step during software development can actually be stored and accessed – anywhere, anytime, and by anyone of the team.

References

[1] M. Burmester, J. Machate, and N. Sandweg, "Integrating User-Centered Design Methods with Software Engineering," *I-com*, vol. 3-2005, 31-40, 2005.

[2] J. Carroll, "Five Reasons for Scenario-Based Design," presented at Proceedings of the 32nd Hawaii International Conference on System Sciences, Hawaii, 1999.

[3] P. Duguid, "Limits of self–organization: Peer production and 'laws of quality'," *First Monday*, vol. 11, No. 10, pp., 2006.

[4] K. El Emam and N. H. Madhavji, "A field study of requirements engineering practices in information systems development," presented at Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on, 1995.

[5] M. Feurstein, M. Natter, A. Mild, and A. Taudes, "Incentives to share knowledge," *Proceedings of Hawaii International Conference on System Sciences. HICSS 34, Maui, HI, USA, 3 6 Jan. 2001 * Los Alamitos, CA, USA: IEEE Comput. Soc, 2001, p 8 pp.*, 2001.

[6] M. Friedewald, H. D. Rombach, P. Stahl, M. Broy, S. Hartkopf, S. Kimpeler, K. Kohler, R. Wucher, and P. Zoche, "Status of the software development industry in Germany," *Informatik Spektrum*, vol. 24, No. 2, pp. 81-90, 2001.

[7] H. F. Hofmann and F. Lehner, "Requirements engineering as a success factor in software projects," *Software, IEEE*, vol. 18, No. 4, pp. 58-66, 2001.

[8] B. Leuf and W. Cunningham, *The Wiki Way: Quick Collaboration on the Web*: Addison-Wesley Professional, 2001.

[9] P. Louridas, "Using Wikis in Software Development," *IEEE Software*, vol. 23, No. 2, pp. 88-91, 2006.

[10] M. Lubars, C. Potts, and C. Richter, "A review of the state of the practice in requirements modeling," presented at Requirements Engineering, 1993., Proceedings of IEEE International Symposium on, 1993.

[11] M. Morisio, M. Ezran, and C. Tully, "Success and Failure Factors in Software Reuse," vol. 28: IEEE Press, 2002, pp. 340-357.

[12] U. Nikula, J. Sajaniemi, and H. Kälviäinen, "A State-of-the-Practice Survey on Requirements Engineering in Small- and Medium-Sized Enterprises," Telecom Business Research Center Lappeenranta, Lappeenranta, Finland, Research Report 2000.

[13] J. Rech, B. Decker, and E. Ras, "Intelligente Assistenz in der Softwareentwicklung 2006: Zusammenfassung der Ergebnisse," Fraunhofer IESE, Kaiserslautern IESE Report: IESE-046.06/D, 2006.

[14] I. Sommerville, "Integrated requirements engineering: a tutorial," *IEEE Software*, vol. 22, No. 1, pp. 16-23, 2005.

[15] S. Lauesen, "Software Requirements: Styles and Techniques", Addison-Wesley, 2002