Information and Knowledge Management www.iiste.org
ISSN 2224-5758 (Paper) ISSN 2224-896X (Online)
Vol.6, No.3, 2016

Workflow Management of Sprints and Software Tests: Coordination, Consensus, and Cooperation in the Enterprise Wiki of the German National Library of Science and Technology

Sven Strobel
Competence Center for Non-Textual Materials
German National Library of Science and Technology (TIB), Welfengarten 1 B, 30167 Hanover, Germany

Abstract
This paper discusses the workflow management of sprints and software tests at the German National Library of Science and Technology (TIB) and pays special attention to how the enterprise wiki can be used to support forms of group interaction such as coordination, consensus (finding), and cooperation. The enterprise wiki helps to organize *who* has to do *what* by *when* (coordination). At TIB, it essentially supports the project managers in planning, organizing, and controlling the workflows of the project. Moreover, it facilitates and accelerates the decision making within the team (consensus) and helps the team members to share knowledge and develop ideas by editing common documents (cooperation). The workflows for sprints and software tests as well as their management are illustrated by the example of the TIB project *relaunching the TIB|AV-Portal*. The paper argues that establishing workflows in the wiki makes collaborative working more efficient and easier to handle. This is shown with respect to the team's coordination, consensus finding, and cooperation in the project. The paper also specifies the TIB concepts for sprints – a method to develop concepts collaboratively – and software tests. The topic discussed here relates to subject areas such as knowledge management and project management and, more specifically, to collaborative working and computer-supported cooperative work.
Keywords: workflow management, collaborative working, computer-supported cooperative work, project management, enterprise wiki, software engineering, sprint, software test

1. Introduction
Software can greatly support the interaction of individuals in groups such as the members of a project team. The technological support of group interaction is the core topic of the multidisciplinary research field computer-supported cooperative work (CSCW) (cf. e.g. Greif 1988; Bowers & Benford 1991; Ellis *et al.* 1991). CSCW deals with understanding social interaction, and configuring, implementing, and evaluating technical systems supporting social interaction (cf. Gross & Koch 2007: 10).

The CSCW literature commonly mentions the following forms of group interaction: coexistence, communication, consensus (finding), cooperation, and coordination (cf. e.g. Gross & Koch 2007: 8). *Coexistence* means the simultaneous presence and mutual information of users. *Communication* is the exchange of information between two or more communication partners. *Consensus (finding)* stands for decision making within the group. *Cooperation* is used in a narrower sense, referring to the common retention and editing of data (cf. Gross & Koch 2007: 8; 195f.). *Coordination* is 'the act of managing interdependencies between activities performed to achieve a goal' (Malone & Crowston 1990: 361).

Enterprise Wikis offer a simple and easy-to-use platform to support these forms of group interaction and have become very popular in all kinds of institutions. They provide functionalities that are needed in the context of companies or organizations: this includes helping the staff share and edit common documents as well as externalize its tacit knowledge as part of the organizational knowledge base (cf. Tapscott & Williams 2011: 1-4; Probst *et al.* 2012: 23f.).

Enterprise wikis are modern *knowledge management systems*, insofar as they support the collaboration between several actors to produce knowledge. Wikis are not only suitable for identifying and externalizing tacit knowledge as well as developing and creating added value due to cross-linking. They also help to acquire, distribute, make use of, preserve, and continuously evaluate and improve knowledge (cf. Komus & Wauch 2008: 167).

Furthermore, enterprise wikis are often used as *project management systems* (cf. Ebersbach *et al.* 2005). Project management involves planning, executing, controlling, and closing a project (cf. Nokes & Kelly 2003). A wiki can substantially support the realization of the various project phases. Using the enterprise wiki as a knowledge management and project management system offers several benefits:

- *High transparency*: All the relevant information of the project is displayed in the wiki and may be retrieved by the project members at any time.
- *Central collaboration platform*: Documents can be centrally collected, linked, and commonly edited and developed by multiple users. They are always up-to-date and develop organically. Older versions can easily be restored, which ensures better protection of the content. Users are notified via email about

changes in the document. External project partners may also receive access to certain wiki documents.

- *Documentation & reusability*: All the relevant information of the project is documented in the wiki, which allows the team members to reconstruct at a later point in time why the team has decided in favor of one or the other option. Moreover, the approach, findings, or ideas of the project can be re-used in other projects.

This paper discusses the workflow management of sprints and software tests at the German National Library of Science and Technology (TIB). Special attention is paid to how the enterprise wiki of TIB can be used to support forms of group interaction, namely coordination, consensus, and cooperation (Note 1).

A *workflow* is a defined series of activities in a work system (cf. Borghoff & Schlichter 2000: 332). It defines how tasks are structured, who carries them out, the sequence in which they are to be carried out, how they are synchronized etc. (cf. Gross & Koch 2007: 92). *Workflow management* encompasses the planning, execution, and controlling of workflows (cf. Aalst & van Hee 2002).

Sprints are a central component of the agile product management framework Scrum (cf. e.g. Deemer *et al.* 2012; Schwaber & Sutherland 2013). The sprint concept we use at TIB defines a sprint as a short working cycle (2 - 4 weeks), in which a sub-concept is collaboratively and mainly virtually developed and approved (cf. Strobel 2015: 713). *Software tests*, on the other hand, check the correctness of software items in relation to previously specified requirements (cf. Kaner *et al.* 1999). The main objective of software tests is debugging, i.e. localizing and resolving defects.

Sprints and software tests are tasks that are collaboratively performed by the team. The single activities of the team members have to be coordinated in order to achieve the overall goal. In other words, sprints and software tests require a workflow management, which specifies, executes, and controls the interdependent activities of the team members.

The paper is structured as follows: Section 2 introduces the TIB project, in which the sprints and software tests were performed. Section 3 and 4 discuss the workflow management of the sprints and software tests. Section 5 provides a conclusion.

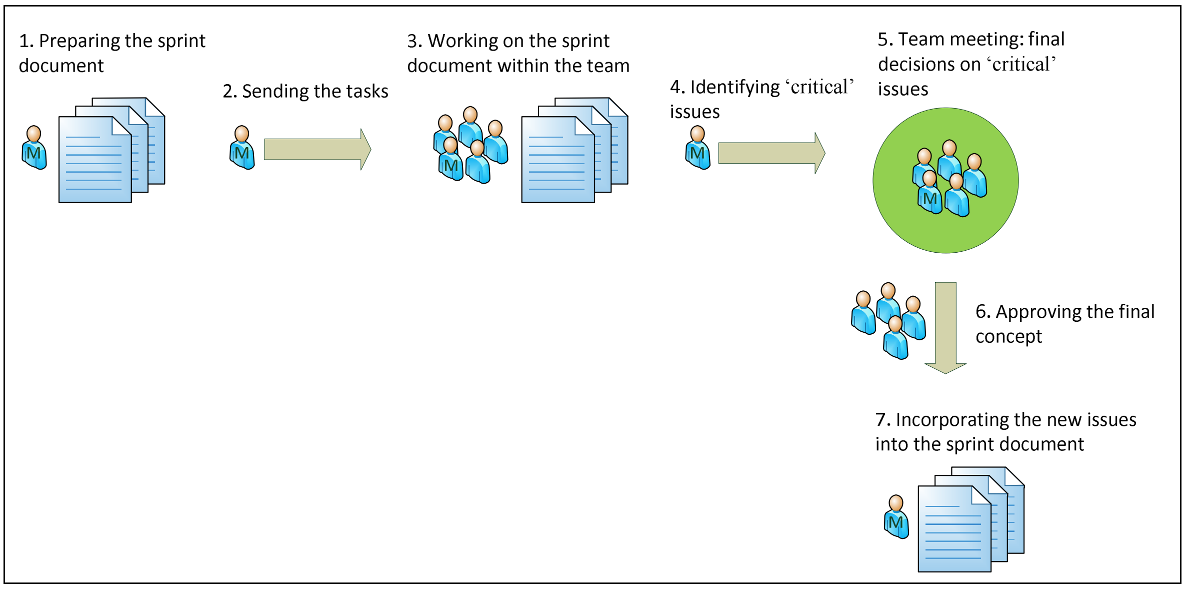
2. The Project *Relaunching the TIB|AV-Portal*

The workflow management of the sprints and software tests illustrated here corresponds to two working packages of the TIB project *relaunching the TIB|AV-Portal* (hereafter referred to as *relaunch project*). The aim of this project was to adapt the screen design and interaction concept of the video portal of TIB (TIB|AV-Portal, Note 2) to the TIB main portal (Note 3). It was decided that the TIB|AV-Portal and TIB main portal would receive a new look and feel as a consequence of the newly formed foundation of the TIB and University Library at Hanover. The project was carried out by a small cross-functional team of 5 persons and ran for one year. The TIB|AV-Portal was relaunched together with the main portal on 4 January 2016.

The course of the project and working packages (WPs) can be summarized as follows: (WP-1) gathering the requirements for the new screen design and interaction concept; (WP-2) creating a user interface prototype based on the collected requirements; (WP-3) writing requirement specifications; (WP-4) evaluating the prototype and specifications collaboratively using sprints in order to optimize the prototype and specifications; (WP-5) commissioning external screen designers to create a new and responsive screen design based on the optimized prototype; (WP-6) testing the new screen design; (WP-7) commissioning external software developers to implement the new screen design plus interaction concept defined by the requirement specifications; (WP-8) testing the implementations (software tests); (WP-9) debugging and deploying the implementations (cf. the detailed description of the project in Strobel (to appear)).

3. Workflow Management of the Sprints

At TIB, we developed a specific sprint concept with a workflow and roles that are not defined in this way by Scrum. Strictly speaking, we rather work with the sprint idea than in the Scrum framework. The TIB concept defines a sprint as a short working cycle (2 - 4 weeks), in which a sub-concept is collaboratively and mainly virtually developed and approved. The overall concept is incrementally developed by the repeated execution of sprints. In the *relaunch project*, the concepts were requirement specifications for the new screen design and interaction concept of the TIB|AV-Portal. The conceptual developer of the project team took on the role of the sprint master, while the other four team members acted as sprint participants. The sprint master's task was to plan, steer, and control the sprint workflow (Note 4). The sprint workflow of TIB is briefly summarized below (cf. figure 1).


1. Preparing the sprint document
2. Sending the tasks
3. Working on the sprint document within the team
4. Identifying 'critical' issues
5. Team meeting: final decisions on 'critical' issues
6. Approving the final concept
7. Incorporating the new issues into the sprint document
M
Figure 1. Sprint Workflow

1. The sprint master prepared the sprint document in the wiki: he determined start and end of the processing time for the sprint document, defined the sprint goal, wrote a detailed description of the concept, and assigned scheduled tasks to the sprint participants. In the *relaunch project*, the concept description consisted of requirement specifications, which had been organized according to functionalities, e.g. functionalities of the start page or hit list. These specifications had been written by the conceptual developer in (WP-3) and were successively included in the sprints in order to be collaboratively evaluated and developed (WP-4). The concept description involved screen shots of the prototype to visualize the requirements.
2. The sprint master sent the scheduled tasks to the sprint participants using a Wiki function. This marked the actual beginning of the sprint. The participants' task was to examine, complement, and eventually approve the concept description.
3. The sprint participants completed their tasks, commented on the concept, and proposed modifications and additions in the sprint document. The sprint master in turn commented on the participants' issues.
4. The sprint master identified 'critical' issues in the sprint document about which decisions were still needed and planned a team meeting to clarify them. (If there were no 'critical' issues, step 4 and 5 could be skipped.)
5. At that meeting, the team members made final decisions on the 'critical' issues.
6. The sprint participants approved the final concept.
7. The sprint master incorporated the new issues into the sprint document.

The sprint yielded requirement specifications which had been collaboratively evaluated, developed, and approved by the team. The total set of requirement specifications was incrementally produced by applying the sprint cycle several times. The sprints helped the team to facilitate and accelerate complex decision making.

3.1 Sprints – Coordination

The wiki supported the sprint master in coordinating the sprint workflow. The sprint master sent tasks to the sprint participants and controlled their completion using the wiki. He scheduled the tasks in the wiki and assigned them to individual participants. The participants checked off their tasks via check boxes, as soon as they had accomplished them. This information of the wiki helped the sprint master to recognize who had already finished his or her tasks and whom he had to remind at the end of the sprint to do them. He also recognized 'critical' issues by means of the sprint participants' supplements and processing status of the tasks. If, for example, the task 'examine and complement the concept description' had been checked off by all sprint participants but there were missing checks for the task 'approve the concept description', the sprint master knew that there was still need for discussion and he had to organize a team meeting.

Furthermore, the sprint master created an overview page to control the workflow. The overview page indicated the status as well as start and end of the processing time of the single sprint documents. It showed the sprint master (1) which sprint documents he still had to work on in order to send them to the participants with the next sprint (status*: in progress*); (2) which sprints waited for approval in which he had to supervise the completion of tasks (status: *waiting for approval*); (3) and which sprints had been completed in which he had to incorporate the new issues into the concept description (status: *completed*).

3.2 Sprints – Consensus

The sprint was not completed until all participants had approved the concept description. When approving the concept, the following conditions applied: (1) issues which had not been commented on by the participants or about which there had been universal agreement were considered as accepted; (2) 'critical' issues were those issues about which decisions still needed to be made, as was the case with divergent opinions, proposed modifications, or supplement suggestions.

The sprint master organized a team meeting, in which the 'critical' issues were exclusively discussed. He started the meeting by briefly summarizing the participants' pros and cons about every 'critical' issue he had found in the sprint document. After that, the team discussed the alternatives. Team members often could be convinced in these discussions and changed their mind. In cases where general consent was not possible, decisions were reached by majority voting. The goal of the team meeting was to come to a decision and not postpone it. In the *relaunch project*, we were often able to make final decisions on 10 to 15 'critical' issues in a two-hour meeting.

Having clarified the 'critical' issues, the team members approved the final concept. This step ended the sprint. The wiki had helped to speed up the consensus finding within the team. The sprint master incorporated the final decisions into the sprint document, i.e. he worked out the concept description and wrote into comment fields why the team had decided for or against specific options.

3.3 Sprints – Cooperation

The sprint master wrote the concept description, providing the sprint participants with a first draft. The sprint participants were supposed to examine and complement it. The concept description was structured by specific points. To every point in the concept description, there was a comment field, into which the participants added their suggestions for improvement or corrections, comments or responses. The sprint master incorporated these issues into the concept description, specifying the concept. In cases where one team member had added very detailed technical comments, the sprint master asked that person to incorporate these issues into the concept description.

The sprint master had created sprint documents, which were edited by all team members. A classical problem with collaborative working in the wiki is that it does not allow several users to simultaneously edit documents. When several users work on the same document at the same time, content normally gets lost whenever one of the users saves his or her changes. However, with the sprints there was no great risk that the team members would permanently block each other. This was because only five persons – primarily the four sprint participants – worked on the document and also they had one week to do that. In fact, there were hardly any mutual blockades in the wiki when we performed the sprints. However, we could not pursue this strategy when testing. The testing workflow required a different solution to make cooperation in the wiki still possible (cf. section 4).

4. Workflow Management of the Software Tests

Tests are an important instrument of quality management for software engineering. After the external software developers had implemented the TIB requirements for the new screen design and interaction concept (WP-7), the team had to extensively test the implementations in the test system (WP-8). The testing period was limited to one week. The external software developers were supposed to receive a test report as the basis for debugging.

The conceptual developer and IT specialist of the project team acted as test managers, who planned, steered, and controlled the testing workflow. They divided the test cases into two scenarios. In the first scenario, the requirement specifications were tested: Had all requirements been implemented and did they work as defined? The second scenario focused on free testing, i.e. the system was supposed to be tested without specific benchmarks or goals. The two scenarios comprised a total of 58 test cases. For every test case, the test managers created a separate wiki document, which included the following properties: *title of test case, tester, browser version, status, result, test goal, test steps,* and *test documentation. Title, tester, browser, status,* and *result* of the single test documents were passed by wiki function to the testing main page so that the test managers obtained an overview of the whole testing process and could control it (cf. figure 2).

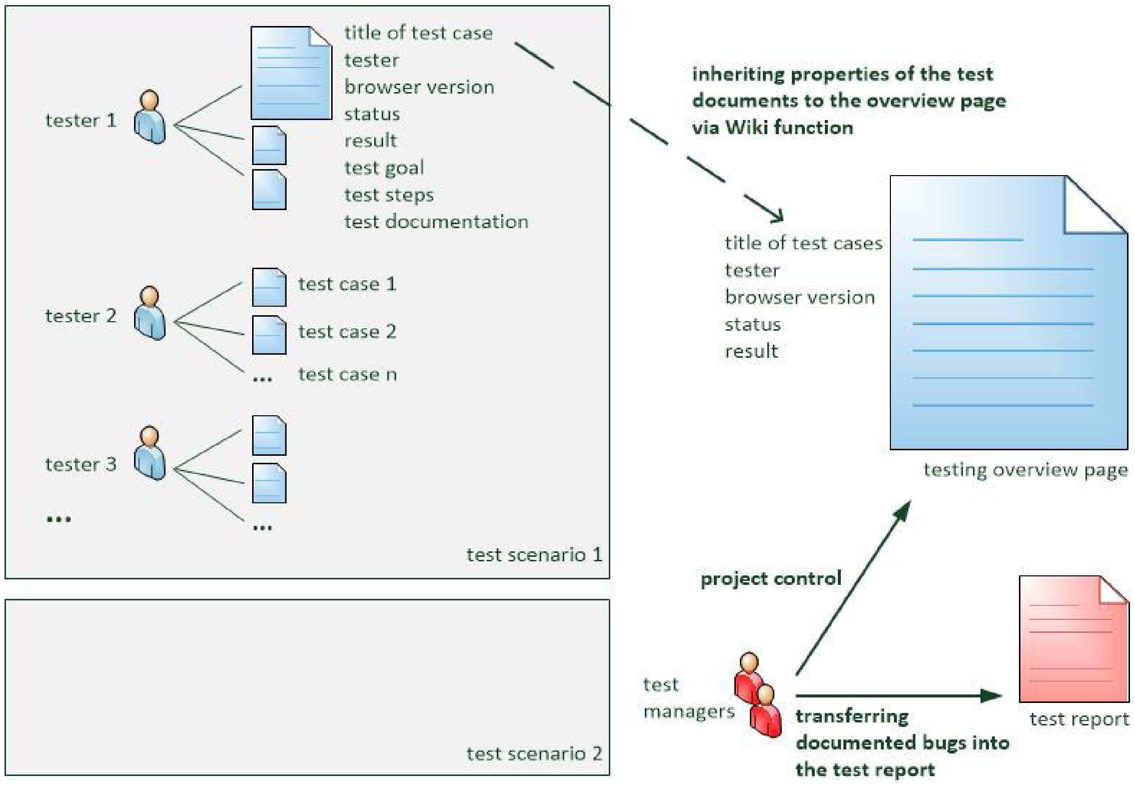


Figure 2. Testing Workflow

The testing workflow included 2 test managers and 10 testers, with the test managers also being testers. The testers were supposed to complete their test cases, which had been assigned to them by the test managers. They had to read and capture the test goal, perform the test steps as defined, and document bugs they had found with detailed descriptions and screen shots. Having done this, they specified the status (*done*) and result (*passed* vs. *failed*) of the test case and indicated the browser version they had used.

The test managers, on the other hand, briefed the testers, monitored the testing process, tried to reproduce bugs which had been found by the testers, summarized all reproducible bugs in a test report, and planned the debugging with the external software developers. The test report, which was finally sent to the software developers, contained about 30 pages with numerous screen shots.

The software developers fixed the defects in several iterations. The test managers controlled the debugging by going through the test report. After all points of the test report had been resolved, the implementations were deployed in the production system (WP-9).

4.1 Software Tests – Coordination

The wiki helped the test managers to coordinate the testing workflow. The test managers assigned the 58 test cases to the individual testers using wiki macros. Furthermore, they summarized all the relevant information for the testers on the overview page, including testing period, test instructions, access data for the test system, a list of the browsers to be used, etc.

The testing overview page was the essential instrument for the test managers to control the workflow. The *status* of the test cases specified on that page enabled the test managers to recognize which tests had not yet been accomplished. This allowed them to remind certain testers to finish outstanding test cases. Moreover, they could detect which tests were positive (*passed* – no bugs found) or negative (*failed* – bugs found). For preparing the test report, only the negative test cases counted – the positive test cases could simply be ignored. The test managers were thus able to concentrate on those test cases which were relevant for the test report. This saved them a lot of time.

In cases in which the situation had changed significantly during the testing period, the test managers were able to respond to this quickly by modifying the test goal and test steps in the wiki and asking the tester to do that test again.

4.2 Software Tests – Consensus

The testing workflow necessitated no real consensus finding within the group. The testers independently documented the bugs they had found, and the test managers then decided whether to incorporate these bugs in the test report.

4.3 Software Tests – Cooperation

The test managers were faced with the challenge to organize the testing workflow in such a way that 10 users could work in the wiki on a common task at the same time. Editing a single wiki document was out of the question because the testers were supposed to intensely test the system in just one week and to document bugs they had found immediately. The testers would otherwise have blocked each other constantly. Therefore, the test managers created for each of the 58 test cases a separate wiki document, some properties of which were passed to the overview page by a wiki function. As a result, the testers could work completely independently and did not block each other. The test managers, on the other hand, obtained an overview of the whole testing process, helping them to monitor it. Due to the inheritance function, the testers collaboratively compiled the information on the overview page, without having to directly edit this document.

Given the fact that the testers had very different skills and perspectives, a wide range of bugs was found, but they would have never been found by only one person. In other words, the testers' diversity produced testing results with a much greater quality. This mechanism is often referred to as *wisdom of crowds* (cf. e.g. Surowiecki 2005). Raymond (2000: 9) puts it in a nutshell: 'Given enough eyeballs, all bugs are shallow'.

5. Conclusion

The workflow management for sprints and software tests at TIB shows how the enterprise wiki can significantly support group interaction. The wiki helps to achieve the overall goal of collaboratively performed tasks of the project. It especially helps the managers, e.g. the sprint master, test manager, or project manager, to plan, steer, and control the workflows. This paper concentrated on the three interaction forms coordination, consensus, and cooperation. The CSCW literature also mentions the interaction forms coexistence and communication (see above), which have been disregarded here. These two interaction forms can be supported by the wiki as well. For example, the wiki informs the other users via email when one of the users has changed the document or checked off a task. The change history of the document provides permanent information about which user has changed which parts of the content (coexistence). We also used the wiki commentary function in the *relaunch project*. This function allowed the team members to comment on the whole document or individual passages. These comments were normally responded to by other team members. Comments and responses were archived in the wiki documents as part of the project communication.

We intend to represent different kinds of workflows in wiki templates in order to achieve a higher standardization, efficiency, and availability of information in our department. These workflow templates will receive descriptions which define the field of application, sequence of the single activities, roles of the participants, virtual and non-virtual components, problems to be solved, etc. Teams will be able to search the wiki and find workflows they can use or adapt in their own projects.

References

Aalst, Wil van der & van Hee, Kees (2002). *Workflow Management: Models, Methods, and Systems*. MIT Press: Cambridge, Massachusetts.

Borghoff, U. & Schlichter, J. (2000). *Computer-Supported Cooperative Work*. Springer: Berlin.

Bowers, J. & Benford, S. (1991). *Studies in Computer-Supported Cooperative Work: Theory, Practice, and Design*. Elsevier: Amsterdam & New York.

Deemer, P. et al. (2012). *The Scrum Primer. A Lightweight Guide to the Theory and Practice of Scrum*. Version 2.0, InfoQ.com.

Ebersbach, A. et al. (2008). *Wiki: Web Collaboration*. Springer: Berlin.

Ellis, C., Gibbs, S. & Rein, G. (1991). Groupware – Some Issues and Experiences. *Communications of the ACM,* 34(1), 38-58.

Greif, I. (1988). *Computer-Supported Cooperative Work: A Book of Readings.* Morgan Kaufmann: San Mateo.

Gross, T. & Koch, M. (2007). *Computer-Supported Cooperative Work*. Oldenbourg: München.

Kaner, C., Falk, J. & Nguyen, H. Q. (1999). *Testing Computer Software*. John Wiley & Sons: New York.

Komus, A. & Wauch, F. (2008). *Wikimanagement. Was Unternehmen von Social Software und Web 2.0 lernen können*. Oldenbourg: München.

Malone, T. & Crowston, K. (1990). What is Coordination Theory and How Can It Help Design Cooperative Work Systems? *Computer-Supported Cooperative Work (CSCW '90) Conference Proceedings,* 357-370.

Nokes, S. & Kelly, S. (2003). *The Definitive Guide to Project Management. The Fast Track to Getting the Job Done on Time and on Budget*. FT Press Club: Harlow.

Probst, G., Raub, S. & Romhardt, K. (2012). *Wissen managen: Wie Unternehmen ihre wertvollste Ressource optimal nutzen*. Springer: Wiesbaden.

Raymond, E. (2000). *The Cathedral and the Bazaar*. Version 3.0. http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/index.html. Retrieved 25.01.2016.

Schwaber, K. & Sutherland, J. (2013). *The Scrum Guide^{TM}, the Definitive Guide to Scrum: The Rules of the Game*, scrum.org.

Strobel, S. (2015). Einsatz von Sprints in der Produktentwicklung der Technischen Informationsbibliothek. *Bub*, 67, 713-715.

Strobel, S. (to appear). Developing the Web Portals of the German National Library of Science and Technology: Tools and Workflows Used. *Qualitative and Quantitative Methods in Libraries*.

Surowiecki, J. (2005). *The Wisdom of Crowds: Why the Many Are Smarter Than the Few and How Collective Wisdom Shapes Business, Economies, Societies, and Nations*. Random House: New York.

Tapscott, D. & Williams, A. (2011). *Wikinomics: How Mass Collaboration Changes Everything*. Atlantic Books: London.

Notes

Note 1. At TIB, we use the enterprise wiki Confluence by Atlassian.
Note 2. av.tib.eu
Note 3. tib.eu
Note 4. I based the expression ‘sprint master’ on ‘scrum master’ from the Scrum framework. However, note that the roles are totally different.

Acknowledgment

The publication of this article was funded by the Open Access fund of the Leibniz Association.