Tool Support for Scrum

Bachelorarbeit
der Philosophisch-naturwissenschaftlichen Fakultät
der Universität Bern

vorgelegt von

Mascha Kurpicz

3. März 2011

Leiter der Arbeit

Prof. Dr. Oscar Nierstrasz

Erwann Wernli

Institut für Informatik und angewandte Mathematik

Abstract

This work gives insights into how Scrum is used in practice and how tools support it. In order to evaluate the qualitative success of a tool in a project, a better understanding of the human needs they cover is needed. We can then evaluate tools on how they match the needs of the developers. In order to get relevant information about the needs and how they are fulfilled in practice, personal interviews and an online survey were done. Based on these needs, the Scrum Wall and an issue tracker are compared. Weak points and issues that arise with regard to tool support in Scrum were identified and are presented in this thesis. In the end, a prototype of a possible solution approach for the synchronization of information from the Scrum Wall to a computer-related tool is presented: Scrum Detector.

Acknowledgements

A special thank goes to Zuehlke Engineering AG for their collaboration and in particular to Hansjürg Inniger and Ralph Jocham for their time invested in this project. Without them this project would not have been possible.

I would like to express my thanks to my supervisor – Erwann Wernli - for his support, the time invested for discussions and his constructive comments with regard to my work.

Also I want to express my thanks to Professor Oscar Nierstrasz who gave me the possibility to do my Bachelor work at the Software Composition Group and to the other members of the SCG for their constructive comments and support during this project.

Last but not least, I want to thank my family and my friends for supporting me during the work on this thesis and during my whole studies.

Contents

1 Introduction

As John S. Reel writes in "Critical Success Factors in Software Projects" [1], two of the essential factors of managing a successful project are to track progress and make smart decisions. Tools should usually support these two factors. But project management is more complicated than that. Many different aspects must be covered and there are tools that were built to support them. Therefore it is interesting to understand better how different kinds of tools are used in practice and how they support these aspects.

In order to evaluate the qualitative success of a tool in a project, we need a better understanding of the human needs they cover. The developer's needs with regard to tool support must be identified and the different tools can be evaluated on how they match those needs.

Furthermore, it is interesting to understand under which conditions a certain type of tool can be applied better than another kind of tool. To do so, an insight into what tools are used in practice is needed. Scrum is one of several agile development methodologies that are now widely accepted in practice. There seems to be a lack of knowledge of experience reports from Scrum projects in practice on how they were supported by tools.

In this work we will talk about both computer-related tools and lightweight solutions as used often in agile software development. An example for such a lightweight solution is the Scrum Wall. Possible examples for computer-related tools are rather traditional tools such as an issue tracker or a document management system.

In this work various insights into how Scrum is supported by tools are provided. Furthermore, we present the assumptions made based on a set of interviews and a survey about how lightweight solutions as the Scrum Wall can be integrated in a company's IT-development and operation processes. Issues that arise at this regard are also identified.

2 Goal of research

Scrum and other agile methodologies are now widely accepted in practice. Recent studies show that Scrum is even the most popular amongst agile development approaches [2].

2.1 How is Scrum applied in practice?

Many tools exist to support software development (defect trackers, project planning, requirement gathering, reporting, etc.). There are tools that support the software development process in general, and tools that are made for Scrum in particular. Sometimes integrated platforms are offered, such as the Team Foundation Server from Microsoft. Also issue trackers to keep track of items or bugs are used.

In order to understand how Scrum tools are used and whether they are successful to support Scrum, we need to understand first how Scrum itself works in practice, the dynamics of Scrum software development, and whether there is a bias between theory and practice.

In order to get practice-relevant information, a collaboration with Zuehlke Engineering AG was lined up. Personal subjective opinions of different persons in different projects are required in order to see where common insufficiency in the tool support can be discovered.

The idea is to show how in different projects Scrum is used and what tool support was provided in those projects. Furthermore, there is a strong interest in the qualitative evaluation of several experiences we are able to explore. That means a goal is to understand what kind of different Scrum projects there are out there. Also we want to discover if for a certain kind of Scrum project there is a certain type of tool that is particularly advantageous.

2.2 What tools are successful for what?

A tool can be considered successful for Scrum when it meets the needs of the Scrum team. As the satisfaction of the Team Members with regard to a tool is rather subjective, the success of a tool can only be defined in a qualitative way.

In our context, we define success as a measure of the satisfaction people have with the tool w.r.t some high-level needs. The list of high-level needs has been elaborated based on the empirical feedback in the first part. When examining these needs, not only the daily needs but also the more complex occasional needs should be considered.

In addition to that we also want to get insight into specific features offered by tools and plan to investigate how different features are used, mostly in term of the frequency of usage.

In this project, we want to investigate whether existing tools are adequate or not to support Scrum, and how they overlap or complement each other. Tools are adequate to Scrum when they match the developer’s needs and therefore support the developers while working on a Scrum project. One question that we ask is whether we can propose criteria to determine the importance of features with regard to Scrum. That means that on one hand we have the importance of a feature for the developers and on the other hand we have needs and how the tools fulfill those needs. This gives evidence of how needs of different importance are fulfilled.

2.3 What can we improve?

It is a goal of this research to find evidence of the deficiencies of tools. Finding the deficiencies of tools allows us then to propose further approaches on how to handle such unfulfilled needs.

The main point of this thesis is to discuss how lightweight tool solutions - as used for Scrum – are going to get integrated in the complex IT context of companies. We want to know what are the issues that are encountered by different teams in different projects and what can be done in order to improve the current situation.

3 Related work

In this section selected existing researches with regard to Scrum and its tool support are presented.

Several researches about Scrum in distributed teams have been done. For example, Jeff Sutherland et al. write in “The secret sauce for hyperproductive offshored development teams“ that even fully distributed Scrum teams can realize advantages such as productivity, clear communication, high quality and consistency, and transparency and control. In particular, they propose a high-performance Scrum implementation with XP engineering practices inside in order to create a distributed/outsourced Scrum team with the same velocity and quality as a collocated team [3].

With regard to tool support less research has been done. Recently, in May 2010, the Forrester Wave published a paper “agile development management tools” in order to evaluate tool vendors. They found that IBM and MKS led the pack with the best overall current feature set [2].

L.S. Møller et al. propose a tool solution for Scrum in “A Scrum tool for improving Project Management“. Based on personal interviews they derived design requirements, as an intuitive user interface, high accessibility, commitment to Scrum and having a project history, and developed the application Scrummer in order to support the Scrum team [4].

In another study proposed by E.A. Engum et al. a case study reports on an organization’s experience from the transition to a backlog management tool. One of the key points is that a tool is in particular convenient to be used to organize and specify backlog items in transparent way and to handle dependencies. Furthermore they say that the concrete project settings are very important to decide whether a tool helps or not [5].

The Scrum table developed at the Hochschule für Technik Rapperswil provides an approach to avoid synchronization issues between the Team Foundation Server and the Scrum Wall. It represents the data from the TFS on a touchscreen – the Microsoft Surface - and allows interactive Scrum meeting around this table [16].

4 Scrum in a nutshell

To understand this research, a basic knowledge of Scrum and its terminology is necessary. We present here the key points of Scrum and the lifecycle of interaction within Scrum.

The development process with Scrum is divided into several Sprints. After each sprint, a functional increment of product must be produced. This makes it similar to the approach of Feature-Driven Development that is defined as "a process designed and proven to deliver frequent, tangible, working results repeatedly" [6].

In Scrum, in the beginning of each project a Product Owner sets up a list with prioritized items – the so-called Product Backlog.

During the Sprint Planning, the team then chooses a small part of the Product Backlog – the items with the highest priorities – and decides how to implement them. Those pieces are then in the Sprint Backlog.

The team has then a certain amount of time to implement these features (the duration of the Sprint) and does a daily update meeting to assess the progress – the Daily Scrum.

Along the way, a Scrum Master keeps the team focused on its goal.

At the end of each sprint there is a potentially shippable product. The sprint ends with a Retrospective, where the process is analyzed and refined, and a Review, where the working increment of software is shown to stakeholders.

The cycle will be repeated until enough items in the Product Backlog have been completed, the budget is consumed or a deadline arrives [7].

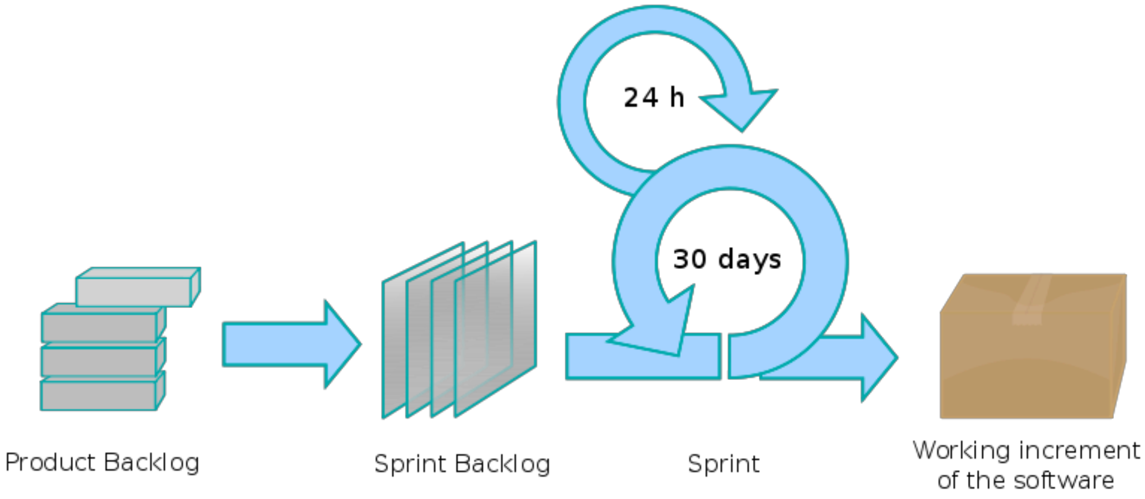


Figure 1 - the Scrum process [8]

Scrum consists of:

- 3 roles: Scrum Master, Team Member, Product Owner
- 5 time boxes: Sprint Planning Meeting, Sprint, Daily Scrum Meeting, Review, Retrospective
- 4 artifacts: Product Backlog, Sprint Backlog, Sprint Burndown Chart, Release Burndown Chart

The Sprint Burndown Chart shows the total task hours remaining for current the sprint. This shows where the team stands with regard to the items in the Product Backlog. The X-axis shows the days in the Sprint, while the Y-axis shows the remaining effort [9].

The Release Burndown Chart is a "big picture" view of a release's progress. The horizontal axis of the Release Burndown Chart shows the sprints; the vertical axis shows the amount of work remaining at the start of each sprint [10].

The Definition of Done defines when an item of the Sprint Backlog can be marked as done. For example, this can include the unit tests or the acceptance of the Product Owner.

Scrum is based on the values of the Manifesto for Agile Development:

> **"We are uncovering better ways of developing**
> **software by doing it and helping others do it.**
> **Through this work we have come to value:**
>
> **Individuals and interactions over processes and tools**
> **Working software over comprehensive documentation**
> **Customer collaboration over contract negotiation**
> **Responding to change over following a plan**
>
> **That is, while there is value in the items on**
> **the right, we value the items on the left more." [11]**

As described in the article "Scrum: stark im Team" [12], another fundamental element of Scrum are the three pillars that characterize Scrum:

- Transparency
- Inspection
- Adaptation

These pillars are realized in Scrum within

- the Product Backlog, the Sprint Burndown and the Release Burndown: Transparency in the process
- the Daily Scrum: Inspection and adaption for the team
- the Sprint Review: Inspection and adaption for the Product Owner, the stakeholder, the team and other involved people
- the Retrospective: Inspection and adaption for the team

Also, in the article mentioned above is affirmed that Scrum is a way of thinking that is based on Leadership and Collaboration instead of being based on Command and Control.

5 Interviews

In a first part of this research, insights into Scrum and its tool support were needed. Therefore, interviews were conducted and interesting topics and lessons learned were documented.

In order to get first data about how Scrum is applied and what tools are used in practice, interviews with 10 different people were conducted. Most of interviewees were from Zuehlke Engineering AG, but from different projects. The duration of each interview was between 1 hour and 1 hour 30 minutes. The interview was done using an interview script, which was evolved and enhanced with the first 3 interviews (see Appendix A). The interviews were then transcribed for further data evaluation.

Before the interview started, data about the professional background of the interviewee were gathered, for example their role in the current project and how many years of experience they have in general and in particular with regard to Scrum. Also we were interested in the number of Scrum projects they had already participated in. However, if the persons had been involved in several projects, we asked them to focus on the most recent one, in order to obtain a more accurate testimonial.

The interview was conducted in two parts. In the first part, the theoretical Scrum process as described in chapter 4 was shown to the interviewee. They were then asked concrete questions about how and how often they are doing the meetings and how often they are using the artifacts proposed by the theory. Also, they were asked about the roles and the use of these roles in their Scrum process. This first part gave us an impression of their project and of their implementation of Scrum. Sometimes first issues that were encountered in their project appeared. Showing them first the complete Scrum process gave them the possibility to compare that process to their own process. The approach of letting the people compare things helped to get them to talk.

In the second part they were asked about their tool landscape, in order to get information about what tools are used in practice to support Scrum. We tried to make a list with all the tools that were used in the project. In this list we considered all kind of tools, also for example testing tools. Also we queried them about what they consider as good and what they consider as negative points regarding the different tools they were using. For those detailed questions we focused in particular on the tools regarding collaboration and the organization of work. This helped to get information about the weak points of the tools and in consequence gives us evidence of possible interview partner's needs.

We mostly stuck to a certain set of questions that were asked in all the interviews, if applicable. Often more specific questions were asked based on the answers given. Also free discussions were enabled what helped us to get further information about the project or the interviewee's opinion.

At the end of the interview, we proposed a rather open question: we asked them about their personal opinion of integrated platforms. We asked them furthermore whether there was anything else they wanted to tell us with regard to Scrum. Often they came up with some interesting points in this free discussion.

5.1 ”We don’t need tools for Scrum!”

A common answer received when interviewing people involved in Scrum about tool support, was that in Scrum there is no need for a computer-related tool. In most of the projects they were using a Scrum Wall on which they pin the user stories. Furthermore they were then standing around this Scrum Wall and moving the cards on the wall during the Daily Scrum or whenever changes were necessary.

Commonly they said that they consider that the big positive point of Scrum, that no digital tool can fulfill, is the human factor. Therefore they often believe that too many tools involved in a Scrum team can even have a negative impact for the team communication. One example for these human factors provided by the Scrum Wall – that appeared quite often in the interviews – was the following:

Imagine you have an open-plan office with the members of a Scrum Team and a big Scrum Wall. If a Team Member now goes to the wall, the others will be notified. Non-verbal communication channels, as for example they might hear his steps through the room, will notify them. Also they might see that this person is hesitating to take a certain task. This hesitation can then lead to communication within the team. The representation in a computer-related tool of this kind of notification is mostly not available.

5.2 But we do use tools for Scrum!

The further we went in the interview, the more we understood that even if the Team Member thinks that there is no need for a tool, they still used one. Almost all of them had some kind of computer-related tool, at least some kind of issue tracker. Different reasons lead them to have a tool anyway. The most popular reasons to have such an issue tracker or similar tool (as for example a wiki or a shared file system) were the following:

- for reporting
- because the Scrum Team is distributed
- because of a non agile context
- for legal reasons / for documentation
- collaboration with other teams (i.e. Scrum of Scrums)

Often the stakeholders expected the team to deliver other reporting materials than just a photo of the Scrum Wall. The Scrum Wall was therefore used additionally to a computer-related tool, even though they mostly confirmed that “when the team is together, it is the best thing to work with cards on the wall”. 5 of 10 interviewees used a tool for reporting reasons.

Three interview partners had experience with distributed Scrum teams. Some of them first tried to use the Scrum Wall, even offshore, projecting the webcam onto a wall during the meeting. To have a consistency of information, they still used a tool. In these cases the computer-related tool was much more important, because it was needed to assure the consistency of the information between the different locations. This could give evidence that the Scrum Wall is not reliable enough.

We realized that sometimes they start using Scrum just in one development team (for example as a pilot), while the rest of the organization is still using a non-agile process (in two interviews this was the case). While they often use just the Scrum Wall within the

Scrum team, they use other tools for the communication with other teams or stakeholders. Also in literature this fact is known, as for example Ken Schwaber writes: “Learning to use Scrum would be pretty simple and straightforward if we didn’t have habits to do things differently. Fitting it into our enterprises, also, would be pretty straightforward if we already weren’t organized and acculturated to do things differently” [13].

For example, during the interviews we received insights into the combination of Scrum and HERMES. If there is a Scrum team in a HERMES context, there is a need for a tool also within the Scrum team in order to organize the collaboration with other teams and other departments and to comply the existing bureaucracy of HERMES.

To fulfill the legal requirements in some cases a special documentation of the development process was needed. Two interviewees mentioned such legal requirements. Therefore they used a tool to track their project data and not only the Scrum Wall. In one interview, a medical software project was mentioned. In another case a rather similar situation appeared for financial software in a banking context. It seems that legal reasons are often involved in software projects.

A Scrum team should be self-organized and communication within the team is discussed often. The interviews gave us evidence that in practice there is often a high coherence between the Scrum team and other teams. 4 of the 10 interviewees mentioned this as a reason for using a tool. No team works in isolation. In one interview there was the architecture office that was not in the Scrum team. In another team a testing department that was not part of the Scrum team did the testing. Also we encountered a project where usability experts were involved. Those usability experts were not part of the Scrum team. Tools can there help to ease the information flow between teams.

5.3 The essence of Scrum

The interviews gave evidence that different people perceive Scrum in different ways. Many people are talking about Scrum as a process, while someone understands Scrum as something more than a process, almost a philosophy.

Also with regard to the main benefit of Scrum different ideas appeared in the interviews. For example someone in the interview mentioned that “not everybody knows everything, but 2-3 people know something together“. This gives evidence that Scrum is perceived as a path towards collective knowledge.

Another view received during the interviews was that many things that are commonly considered a part of Scrum are rather best practices and not part of the process. These are, for example, pair programming, continuous integration, user stories and the planning poker. Other parts are indeed mandatory when using Scrum and can be seen as the core of Scrum. Those core parts of Scrum are the Sprint, the Daily Scrum, the Sprint Planning Meeting, the Definition of Done, the Review and the Retrospective. The interviews gave evidence that most of the teams are doing these core parts of Scrum regularly (8 of 10 interviewees were doing the core parts of Scrum regularly).

In the interviews we often encountered different meanings for the terms task and User Story with regard to Scrum. When talking to a technology expert for Scrum, we found that there is a strict separation between those terms. Strictly speaking we have User Stories in the Product Backlog that are, if necessary, divided into smaller tasks. These tasks are then estimated and implemented by the developers. As soon as all tasks of a User Story have been implemented (as the Definition of Done requires), the User Story will change status to "done" as well. In this thesis, we decided to not strictly separate those terms, as in our data – the interviews – no strict separation was done as for most interviewees this difference seemed to be neither important nor strict.

5.4 Lessons learned

From the impressions received from the personal interviews we can assume that the real dynamics of software development is more complicated than the perceived dynamics. Often on a first sight the software development process seems to be simple. When having a closer look at it, it shows to be more complex. Many different aspects as for example the interactions between different teams must be considered.

Insights on such aspects appeared also in the interviews. The information gained during the interviews was transcribed. Afterwards we grouped the different aspects in themes. That provided a categorization of the feedback received from the interviews. The following themes were mentioned in most of the interviews and can therefore be considered as important:

- ***Low learning curve:*** *How much time can be allocated to learn new tools or processes? Are people willing to invest time and effort on this? What kind of people is willing to invest time and effort on this? Are issues encountered that are caused by an insufficient training or learning?*
- ***Involvement:*** *Does the developer's commitment increase when they feel more involved? Does this involvement influence productivity? Does the input given by the developers really lead to changes in the process?*
- ***Flexibility:*** *What are the aspects of the process or the tools that are typically changed and what are the areas that need to be most flexible? Can the process be adapted when learning from mistakes?*
- ***Accessibility:*** *How can people search and locate meaningful information and how can they navigate through this information? What are the entry points to the written knowledge?*
- ***Awareness:*** *Is interpersonal communication an issue? What does a Team Member need to know about the other Team Members and their activity? Is having a global vision good or bad? In what places people get aware of what is going on? Does this knowledge enable certain management practices (e.g. substitution when someone is sick)?*
- ***Traceability:*** *How is the history feature used? Do changes need to be tracked? Is the "episodic" knowledge important?*
- ***Transparency:*** *Is there "one truth" in the projects? Do different roles/people (e.g. Stakeholders or Team Members) get different views of the information? Are there private bugs, which are bugs only available to the team and not to the stakeholders?*
- ***Reporting:*** *What kinds of reports are used? Are they used to take decisions, to track progress? What kind of report works? Are they trusted? What is the effort that people are willing to spend on reporting?*
- ***Accuracy:*** *Is it an issue to find prevailing and updated information? Are there versioning issues? Do people trust the information? Is there an overhead to keep information up-to-date? What is the right amount of information to keep in written form or to keep in people's heads?*

- ***Integration with other teams:*** *What information needs to be shared and exchanged with other teams? What is the autonomy of each team? What are the practices for a good collaboration?*

In these themes appear again the three pillars of Scrum as described in chapter 4. The transparency appeared directly within the themes, while the inspection and the adaption appeared in an indirect way within the involvement, the flexibility and the awareness.

6 Survey

In the second part of this work, based on the results of the first part (see chapter 5), an online survey was designed in order to get empirical information about the importance of developer's needs and about how often different features are used.

6.1 Scope of the survey

From the input of rather heterogeneous projects we received from the interviews it was impossible to focus on everything in the survey. We decided to focus on the themes described in chapter 5.4.

In the interviews a large choice of different tools appeared. Summarizing them to categories, the following groups could be formed:

- Scrum Wall and similar visual tools
- different types of issue tracker
- document management systems, such as wikis

These are different tools with different intentions, even if at a first sight they seem to have a similar interface. The Scrum Wall helps in particular to get a good visibility for the team. The issue tracker can help to organize work. On the other hand we have a document management system that can help getting accurate and trustable documentation and therefore retains the knowledge.

In order to compare these different tools, questions in the survey applicable to all of them had to be designed. Due to time constraints we decided to drop the document management systems, even though it might be very interesting to do a similar research on them.

We will not consider in particular the problematic of distributed Scrum teams. Furthermore the issues that arise while doing Scrum of Scrums are not considered in this survey.

6.2 Approach

In the interviews only a small set of persons was queried and the projects discussed in the interviews were very different one from another. To objectively compare tools we elaborated a model to compare them and we gathered empirical results – according to the model – via a survey.

The interviews exposed possible real needs of the Team Members. With the real needs we mean the high important daily needs on one hand, and the maybe less important occasional needs that are still of considerable importance on the other hand. We realized that even though those occasional needs are less important, they are still important enough to justify the use of an additional, computer-related IT system.

A hypothesis based on the interviews is that, even if they are using the Scrum Wall for their daily work and do a reporting to a stakeholder once a sprint, this additional tool is still important enough to justify its existence.

In order to understand better those two categories of needs and their fulfillment by the sets of tools currently used by the teams, we decided to put up a short survey among people involved in Scrum. The survey took about 5-10 minutes to be filled out and was focused on the daily and occasional needs and how their currently used issue tracker and the Scrum Wall fulfill these needs.

Based on the decisions described in chapter 5.1, the survey should give insights on how the different tools issue tracker and Scrum Wall overlap and complement each other.

From the themes considered important in the previous chapter “lessons learned” more concrete needs were extracted. In order to make a reasonable survey we had to identify needs that can be perceived as actions, in order to make sure that the participants of the survey can understand and answer the questions, identifying them with their daily work.

In a first step we divided the whole situation in a problem space and a solution space, the problem space representing the needs in term of information management, and the solution space the features that tools offer to fulfill these needs.

6.3 Problem space: A general set of needs

In the problem space we have different roles or profiles of people – the Scrum Master, the Team Member and the Manager – with different needs and expectations with regard to tool support or that give a different level of importance to needs.

After doing the personal interviews and getting insights into their projects we can assume that they all need…

- ways to interact
- ways to control and drive
- ways to be aware
- to feel active
- to have few constraints

They need ways to interact in order to exchange information and reach a consensus. This allows them to take collaborative decisions and to exchange information internally and externally. A good way of interaction allows them to manage personal issues and to schedule events appropriately.

Furthermore they need a way to control and drive the project, for example to assess the progress. The conservation of knowledge is another important point mentioned often during the different interviews. Goals need to be defined and the teamwork must be improved in order to reach these goals. Also knowledge management can be considered in this point.

In order to do a good daily work, precise and accurate information is needed. People involved in the project must stay up-to-date, but also knowing what the others are doing was considered important as well during the interviews.

Furthermore there are “cross-cutting” needs. The needs mentioned above must be fulfilled in an active and easy way.

For many people it seems to be essential to feel involved and active in the project. Refining the process and adapting it to the team's needs resulted to be a good stimulation of the team's motivation.

Possibly there should be few constraints. That includes for example a strong wish for easy tool solutions that generate a low overhead and a low effort. Also in the research of L.S. Møller et al. this aspect is confirmed: they consider an "intuitive user interface" as an important design requirement for a tool to support Scrum [4].

What they need	**How they need it**
Ways to interact	- active - with little constraints
Ways to control and drive	
Ways to be aware	

Table 1 - needs

For the survey we picked up some key points of these general needs to break them down to concrete user needs, as not all above-mentioned needs could be covered within the survey. The people involved in Scrum projects had to classify these needs with regard to their importance for them.

- Reach consensus / take collaborative decisions
- Exchange information
- Retain long-term knowledge
- Assess progress
- Organize and track time of work/duty
- Know what I should do
- Get accurate and trustable documentation
- Know what the others are doing

Furthermore we wanted to understand better which of those needs – when fulfilled - make them feel involved and active in the project.

With regard to tools, we evaluated how the Scrum Wall and an issue tracker fulfill these needs in order to provide a high-level notion of how successful they are, which is the goal of the research (see chapter 2.2).

6.4 Solution space: A general set of features

The solution space is for the tools. We can find here different tools as for example the Scrum Wall, the issue tracker, a document management system, hybrid platforms and tools for personal communication. In the survey we just concentrated on the Scrum Wall and the issue tracker (see chapter 6.1). Based on the interviews we decided to use the following general feature set for the tools:

The tool offers features to …

- manage detailed "core" information
- synthesize information
- locate detailed information
- track changes / keep history
- notify of changes
- annotate information
- be intuitive
- be flexible

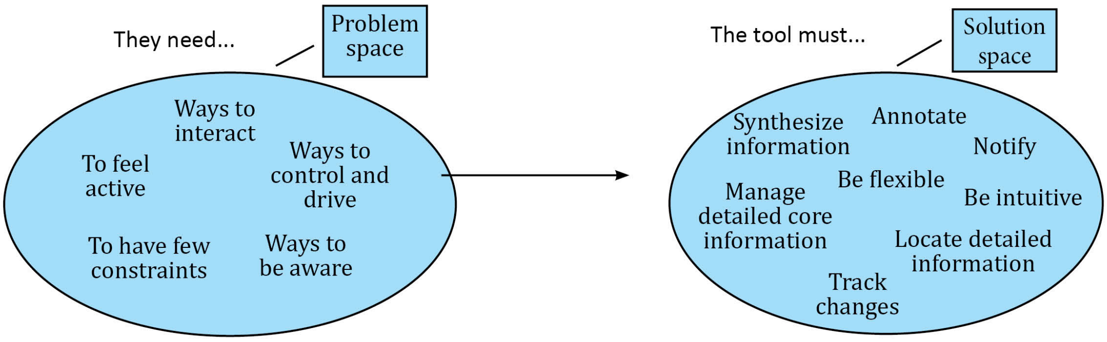


Figure 2 – the problem and the solution space

The management of detailed "core" information consists of work items, pages, status and different operations on those elements.

The synthesis can have different forms: it can be in a dashboard, in a generated report or in different views on the information, depending on the tool.

Another feature provided by the different tools is the search for relevant information. Detailed information must be located and can then be bookmarked or linked with other pieces of information.

Versioning and history might be possibilities in a tool to track changes. Depending on the tool this might have different characteristics.

With notification we mean a feature that notifies a developer when for example a change happens. The work done can be shared with others in this mode.

Adding a comment to an existing issue, highlighting or annotating something on a current user story or other features that allow the user to add additional information to an item are considered here.

All these features should be implemented in a way to fulfill the "cross-cutting" qualities, which are that the tool is intuitive (easy to use and with a smart automation) and flexible. Being flexible means to configure and customize the tool to the specific project's needs.

What qualities they need	How they need them
Manage detailed "core" information	- intuitive - flexible
Synthesize information	
Locate detailed information	
Track changes / keep history	
Notify of changes	
Annotate information	

Table 2 - qualities

These abstract features can be rewritten in a more appropriate manner specifically for the Scrum Wall and the issue tracker.

For the issue tracker:

- Enter new work items
- Organize existing work items
- Generate a dashboard / report about the project status
- Find the relevant information for the daily work available in the issue tracker
- Find the relevant information for the daily work available in other systems
- Browse changes/history for a specific issue
- Update the work items to match with the latest decisions
- Be notified about certain changes (e.g. somebody starts working on an issue)
- Annotate, refine or comment an issue

For the Scrum Wall:

- Enter a new user story
- Organize existing user stories
- Get an overview of the project status (e.g. step back and look at the wall as a whole)
- Find the relevant information for the daily work available on the Scrum Wall
- Find the relevant information for the daily work available in other systems
- See the changes/history of a specific issue (e.g. change of priority)
- Update the user stories to match with the latest decisions
- Be notified about certain changes (e.g. you hear that somebody goes to the wall)
- Annotate, refine or comment an issue (e.g. adding a post-it to the wall)
- Interaction happening around the Scrum Wall

To get an understanding of the importance of the different features of the Scrum Wall and the issue tracker we asked the participants of the survey how often they are actually using them.

Please find the survey questions in the Appendix B.

7 Results

The survey was filled out by a total of 19 persons involved in Scrum projects. Of these 10 people, 3 were from Management, 9 were Team Members and 7 were Scrum Masters. In order to get a clear classification, we asked them to choose the most applicable role when having different roles in the project. Also, we asked them to choose one Scrum project in which they were involved and to answer all the questions with regard to this project.

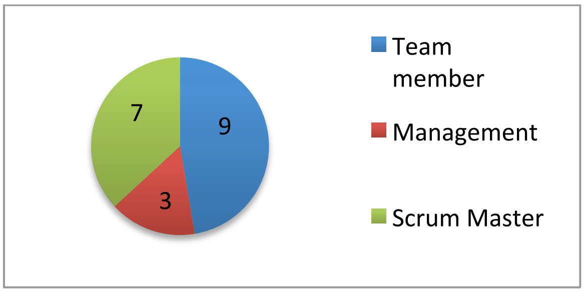


Figure 3 - participants of the survey

7.1 Importance of needs

The exchange of information was considered as the most important need. 12 of the 19 people considered it very important, 4 as important and the rest as of average importance. Also the reaching of consensus/the taking of collaborative decisions, to know what they must do and to know what the others are doing were considered as important. Indeed, the conservation of long-term knowledge, the assessment of progress, the organization of work/duty and the accurate and trustable documentation were considered as less important.

Summarizing, we can identify the following important needs:

- Reach consensus / take collaborative decisions
- Exchange information
- Know what I should do
- Know what the others are doing

and the less important needs:

- Retain long-term knowledge
- Assess progress
- Organize work/duty
- Accurate and trustable documentation

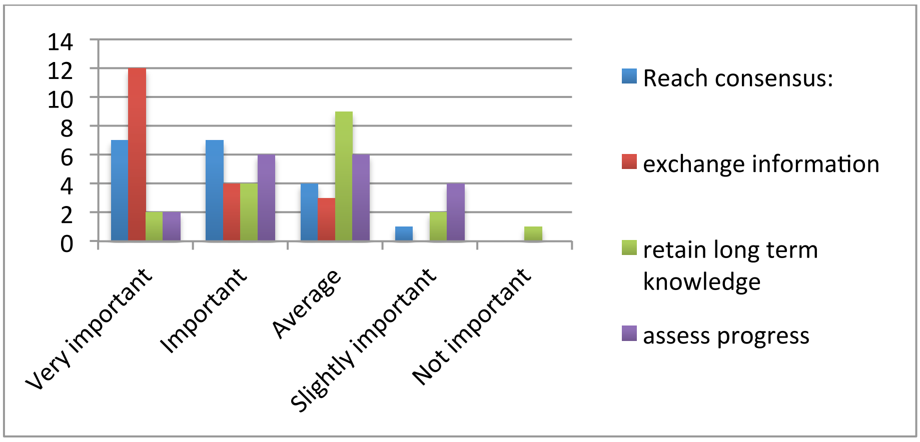


Figure 4 - the importance of needs, part I

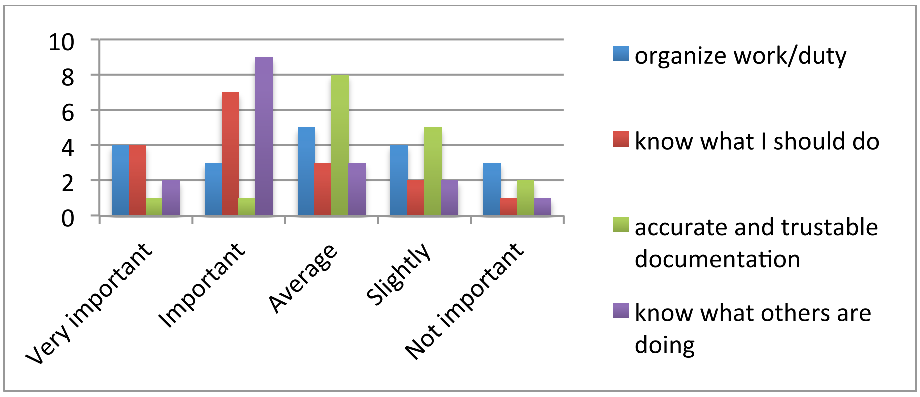


Figure 5- the importance of needs, part II

We wanted to know which needs do make them feel involved, when satisfied. Almost all people (18 of the 19 participants) agreed that they feel involved when they reach consensus within the team and take collaborative decisions. Also the exchange of information and the knowledge about what the others are doing let them feel active and involved within the process and the project. In contrast, just 2 persons felt involved or active by organizing work or duty. Also the conservation of long-term knowledge was considered rather unimportant for feeling active and involved.

The conservation of long-term knowledge and the accurate and trustable documentation were the points considered the most difficult to achieve. Also to reach consensus / take collaborative decisions and the exchange of information were considered rather difficult. Indeed, it seems to be easier for the participants to know what others are doing and to assess the progress.

The conservation of long-term knowledge was considered a problem mainly for people outside the team. One person wrote that for the knowledge within the team there was no issue, but for the conservation of the knowledge for people outside the team that was an issue indeed. As well the step from the project to the maintenance mode was mentioned to be difficult with regard to conservation of long-term knowledge. It seems that in some projects there is often a change of team members. This is as well considered problematic while retaining knowledge. This problem can be compared to the externalization of tacit knowledge proposed by I. Nonaka in “The knowledge creating company” [14].

He illustrates the movement between two different kind of knowledge – the tacit knowledge (highly personal) and the explicit knowledge (formal and systematic, can be easily communicated and shared).

From the answers of the survey we can assume that often the interpersonal issues within the team and the current culture of the context (external to the Scrum team) are highly relevant for the difficulty of fulfilling the needs. For example, in one project they think that tracking is hard to achieve when a manager requires it but the team is not motivated to do so. A lack of time provides another key point of why it is difficult to reach accurate and trustable documentation.

The exchange of information seems often to be difficult between the Scrum Team and the customer. Tools might support the communication in this relationship that is often difficult.

One person noted that “achieving consensus within a team is harder compared to be dictated the common ‘consensus’ by a team leader”.

7.2 How do the different tools satisfy those needs?

In the second part of the survey we verified how the tools (Scrum Wall and issue tracker) satisfy the needs of the participants. In order to have a common base to compare them, spider charts with the weighted average of the level of importance were created.

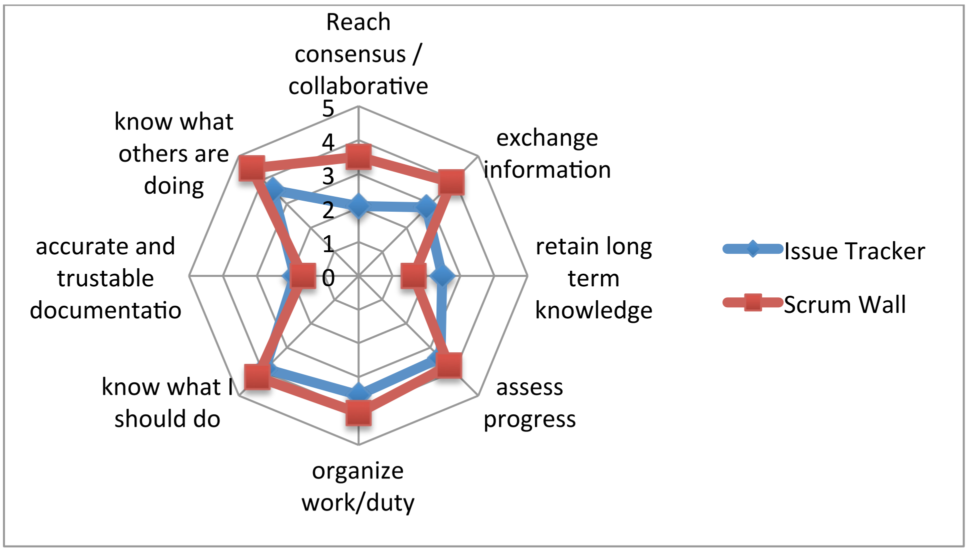


Figure 6 – spider charts

The data received from the survey allows us to assume that the Scrum Wall shines in fulfilling most needs but does not fulfill others. The issue tracker indeed fulfills more or less all the needs, but with regard to quality it is of rather average level. That means that the data for the issue tracker is rather flat, while the data for the Scrum Wall are not flat.

The Scrum Wall for example has great scores for the knowledge of what others are doing and the exchange of information. Therefore it could be considered as an “information radiator”, as defined in “Crystal Clear: A Human-Powered Methodology for Small Teams” by Alistair Cockburn [15]. Also for knowledge about what I should do and the organization of work/duty it seems to work fine. On the other hand it does not fulfill the need for the conservation of long-term knowledge and the production of accurate and trustable documentation. It has to be considered that the Scrum Wall was never intended to do so. It is a means to an end – in order to reach the Sprint goal.

Even though the conservation of long-term knowledge and the accurate and trustable documentation have been considered of average importance by most of the participants (see previous chapter), they are still important enough to justify the use of another tool that is only slightly more adapted to fulfill those needs. We can assume that it should be considered fulfilling both the daily needs (that are considered as very important) and the occasional needs.

7.3 Importance of features

In the previous chapter a feature set for the solution space was created. In order to find out the importance of these features for the Scrum Wall and the issue tracker we asked the participants of the survey how often they use these features.

In order to have a common base to compare them, spider charts with the weighted average of how often they use the features were created (1 – almost never, 2 – 1-5 times per month, 3 – 1-5 times per week, 4 – 1-5 times per day, 5 – more than 5 times a day).

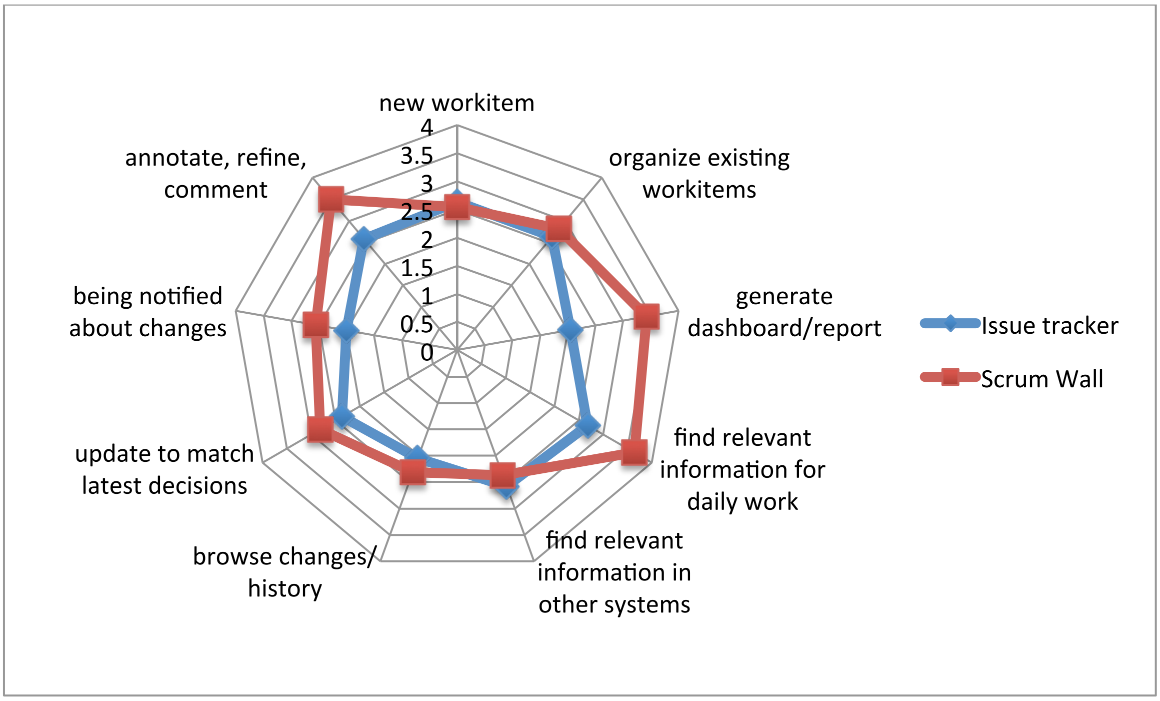


Figure 7 – importance of features

Most participants agreed that adding a new work item, organizing the existing work items and annotating/refining/commenting a work item were the most frequently used features (mostly 1-5 times per week) for the issue tracker. In contrast, most participants used a feature for being notified about changes and to generate a dashboard/report almost never. Also changes were browsed rarely. With regard to finding relevant information within the issue tracker and in other systems there seem to be two groups: one group uses this feature often (1-5 times per week) and others use it almost never.

It seems that the features to support notification and dashboard/report generation are not fulfilling the needs of the team. Getting accurate and trustable information is considered of average importance by many of the participants (see chapter 7.1). This gives evidence that the features mentioned before are not used that often.

They are frequently annotating, refining or commenting a work item. It can be assumed that this helps them to know what others are doing and to know that they must do. Knowledge about their own and other people's tasks is an important need to most of the participants (see chapter 7.1).

With regard to the Scrum Wall the features that were used mostly were organizing existing user stories, getting an overview of the status of the project, finding relevant information for the daily work within the Scrum Wall, annotating/refining/commenting and interaction around the Scrum Wall. Other features instead, as for example browsing history/changes, being notified about changes or adding a new user story, were used less. It can be assumed that the Scrum Wall is not intended to support a history and therefore this feature is used few. Indeed, other research as for example L.S. Møller et al. showed that the lack of a project history results in discussions previously performed during the meetings and therefore to less efficiency [4].

The Scrum Wall is often used to get an overview of the project. This can be confirmed by the strong point of the Scrum Wall with regard to assess progress (see previous chapter). It gives evidence that, as the Scrum Wall is good to assess the progress of the project, it is often used to get an overview.

7.3.1 "Everything important for daily work shall be visible, transparent and available for the team. "

In a free text question of the survey this statement appeared. It seems that the important themes that came out in the evaluation of the interviews appeared also in the mind of the survey participants with regard to important needs in Scrum. The progress must be assessable (visible), everyone should know what the others are doing (transparency) and all this should be available to the Team Members and others. As we can see in the tool spider charts in the previous section, the Scrum Wall fulfills these requirements. We assume that this is a reason why the Scrum Wall is so successful in Scrum projects.

For detailed diagrams of the survey results see Appendix C.

7.4 Synchronization issues

7.4.1 "Tools should solve a problem, and not create other problems"

From the interviews and the survey we can conclude that even if the tools mainly satisfy the needs there are still issues with regard to tools. For example, there are problems with regard to integration of the different tools and with regard to data synchronization between them. In the survey we also asked the participants whether they consider the synchronization and replication of information between different tools a problem. 15 from the 19 participants considered this issue as problematic.

Currently there are some existing solutions in order to avoid synchronization issues. For example there is the Scrum Table from the Hochschule für Technik Rapperswil that provides a Scrum software for the Microsoft Surface in order to work interactive on a touchscreen directly on the Team Foundation Server [16].

In order to understand better the belief of our participants in such advanced technologies, we asked them whether advanced technologies support more natural interaction between people and tools. The same number of people that said “No” said “Yes” with regard to this question. However, many of them answered rather open, as for example that this could help distributed teams in particular or that it does “not help but it is cool”.

We also asked them whether linking of information among the tools is a problem. Here 9 of 19 participants agreed. In a case study done by E.A Engum et al. one of the key lessons learnt is that a tool should organize and specify the items in a transparent way and that it should handle dependencies. It is therefore the tool’s job to support linking of information [5].

7.4.2 Integrated platforms

In the end of most interviews we asked them about their personal opinion with regard to an integrated platform. They answered quite differently, but summarizing, one could say that there seem to be two main groups:

- the supporters of integrated platforms as for example Team Foundation Server
- the supporters of “take the best of each tool category”

Many of them agreed that it depends on the technology: while the Team Foundation Server is often used as an integrated platform when developing with .NET/C# for example, while when developing with Java there are fewer integrated platforms available and from the interviews we learned that it seems that they are used less.

In the survey we asked whether they believe that an integrated platform can help with regard to linking and synchronization issues. 9 from the 19 participants of the survey agreed that an integrated platform could help with regard to these issues.

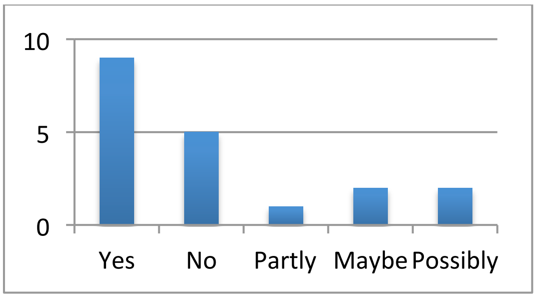


Figure 8 - do you think that an integrated platform can help with regard to linking and synchronization issues?

7.5 Lessons learned

To conclude this chapter, we are going to bring together the information received in the interviews and the results from the survey:

7.5.1 "That’s a problem to have all the documents up to date for everybody."

To be accurate and trustable, the documents must be up-to-date. In the interviews it seems that often this is an issue. Also in the previous section we confirmed this impression with the fact that both Scrum Wall and issue tracker do not fulfill this need very well.

7.5.2 "If one of these guys leave, we are in big trouble.”

In the interviews we often heard that the knowledge is in the heads of the people (5 of 10 interviewees said that it was an issue). Even if they consider the conservation of long-term knowledge as less important, there is still an occasional need for it in case someone leaves the team. As neither the Scrum Wall nor the issue tracker are really intended to address this need additional tools are often used.

We can assume that this loss of long-term knowledge is also caused by a lack of accurate and trustable documentation, as we saw in the previous section. So knowledge is in the people’s heads and when they leave, they take the knowledge with them. Another approach to this kind of issue could be a better request for documentation in the Definition of Done.

7.5.3 "More than a tool issue we have a technology issue."

As we saw in the last section, the feature to find relevant information in other systems is used quite often in the Scrum Wall as well as in the issue tracker. Also in the interviews this was usually confirmed. Often the linking of the information was seen as problematic. More than the integration of different tools, also the integration of different platforms or technologies seems to be an issue. In one project for example they used a Linux and a Windows platform in the same project. Automation of for example test cases became very hard because of that.

7.5.4 “Brain, pin board, paper and a pen”

At the end of the survey we asked the participants what issue tracker they are currently using. Our assumption from the interviews that they sometimes use MS Excel as an issue tracker and the Scrum Wall for the team was confirmed. Some of them indicated that they are using the Scrum Wall also as issue tracker, with no additional computer-related tool. With regard to this a participant wrote: “You can perfectly practice Scrum without any tool (accept brain, pin board, paper and a pen)“.

When it comes to the bigger context, there is instead a need for some computer-related tool. From the answers received during the interviews and the answers from the survey we get confirmation: “The single Scrum project on its own seems not to be a source of trouble”. As described in chapter 5.2, additional tools are often required in collaboration with other teams or departments. In those situations tools can improve the communication and the exchange of information.

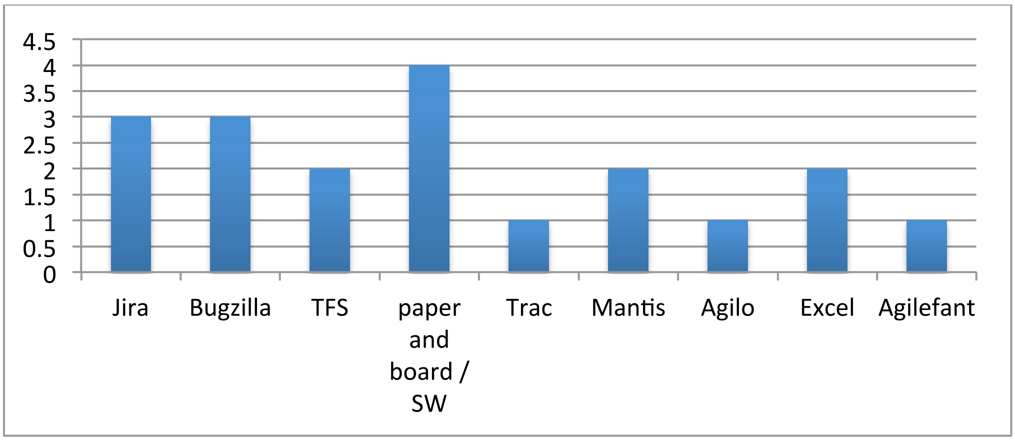


Figure 9 - what issue tracker are you currently using?

7.5.5 "A fool with a tool is still a fool"

In the interview we heard this statement and again it appeared in a free-text question during the survey. People seem to be aware of the fact that often there are problems that cannot be solved by tools. For example, often inter-personal issues within the team caused a lack of communication. These rather personal and individual aspects can maybe be supported by tools but not resolved.

7.5.6 Possible area of improvements

As described in chapter 7.1 there are certain needs that are more important than others. The information gained from the survey on how these needs are satisfied by the Scrum Wall and the issue tracker provide further reference of the status quo. Bringing together the information, we get two matrices, one for the Scrum Wall and one for the issue tracker:

Scrum Wall		Needs	
		+	-
Satisfaction	+	Exchange information, know what I should do, know what others are doing	Assess progress, organize work/duty
	-	Reach consensus/take collaborative decisions	Accurate and trustable documentation, retain long-term knowledge

Table 3 – the needs and the Scrum Wall

Issue tracker		Needs	
		+	-
Satisfaction	+	Exchange information, know what I should do, know what others are doing	Assess progress, organize work/duty
	-	Reach consensus/take collaborative decisions	Accurate and trustable documentation, retain long-term knowledge

Table 4 – the needs and the issue tracker

These matrices lead us to the areas where in particular further efforts can be done in order to improve. As results from the above matrices, in particular the reaching of consensus/taking collaborative decisions, the conservation of long-term knowledge and the accurate and trustable documentation can be supported better than the status quo does.

As mentioned before in the previous chapter the Scrum Wall still supports the reaching of consensus and the taking of collaborative decisions much better than the issue tracker, while it is not intended to be used with regard to long-term knowledge and accurate and trustable documentation.

This raises discussion about whether reaching consensus is inherently a soft skill that no tool can really improve. If so, it is clear that tools will always fail to support it. On the other hand, there are artifacts that can help reaching consensus. An example for that are the cards for the effort estimation of the user stories, provided for example by Zuehlke Engineering AG. These artifacts support the decision-taking process as they make it fun and easier.

The interviews and the survey results give evidence that the status quo could be improved when synchronizing the Scrum Wall to a computer-related structure, such as a spreadsheet or a database. This allows one then to generate more accurate and trustable documentation from the Scrum Wall data. Based on this documentation, a long-term knowledge can be retained.

As part of this project a possible solution approach for the above-mentioned improvement is provided in form of a prototype. In particular, the idea is that the Scrum Wall can be synchronized to a spreadsheet in an easy and automated way. This prototype is presented in the next chapter.

7.6 Threat to validity

The weak point of the survey was the small number of answers and the poor distribution of the answers between the different roles in the project (Management, Scrum Master, Team Member), as most of the participants of the survey were Team Members or Scrum Masters (see chapter 7.1). From the information we received during the interviews, we can make the hypothesis that if there were more participants from Management, the conservation of long-term knowledge and the accurate and trustable documentation would have probably been ranked of higher importance.

In general it can be assumed that the participants understood the questions in the right way – almost all answers were reasonable. Only one participant answered the questions that were not applicable to his project with the lowest level. He remarked this in the comments and his answers with regard to these questions could be removed.

7.7 Future work

It would be interesting to do a similar research for other kinds of tools. For example, a similar research could be conducted for a document management system. It would be interesting to confront the issue tracker and the Scrum Wall to other types of tools.

Furthermore, other solutions for the given problems could be proposed. For example, a solution for the linking of information between the Scrum Wall and the issue tracker could be designed.

8 Proposal of a prototype: Scrum Detector

8.1 Lightweight solutions

As we saw in the last sections, an important requirement for a tool is that it should simplify and not create more problems. Furthermore synchronization of the different tools and the Scrum Wall is an issue often mentioned. Therefore we asked the participants of the survey whether they would consider the evaluation of a lightweight solution for the synchronization of the information from the Scrum Wall to a tool. Most of the participants would do so (9 from 17, two participants did not answer this question), someone would not (6 from 17, two participants did not answer this question) and the other two participants are not sure whether they would do so (perhaps/not yet).

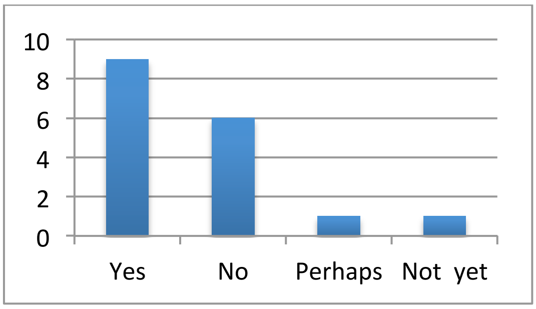


Figure 10 - would you consider the evaluation of a lightweight solution for the synchronization of the information from your Scrum Wall to a tool?

In one of the interviews the team used a template generated in Microsoft Office for the User Stories on the Scrum Wall. Inspired by this idea, an approach of synchronization of such User Stories back to a computer-related structure is proposed here.

8.2 Features

Scrum Detector supports the synchronization between the Scrum Wall and a computer-related tool. This leads to a better consistency of the data and to accurate and trustful documentation. Once accurate and trustable documentation is available, the conservation of long-term knowledge will be easier.

The feature to add a new user story to the Scrum Detector allows adding a new user story easily. When all information for the new user story has been entered, a file in PDF-format will be generated and can be printed. The new user story will contain all the entered information and also a barcode with the ID of the user story. If the entered ID for the new user story was already assigned to an existing user story, the next free ID will be assigned instead. The user story can then be put on the Scrum Wall.

The barcode assigned to each user story is a code 39, which is a variable length, discrete barcode symbology.[1]

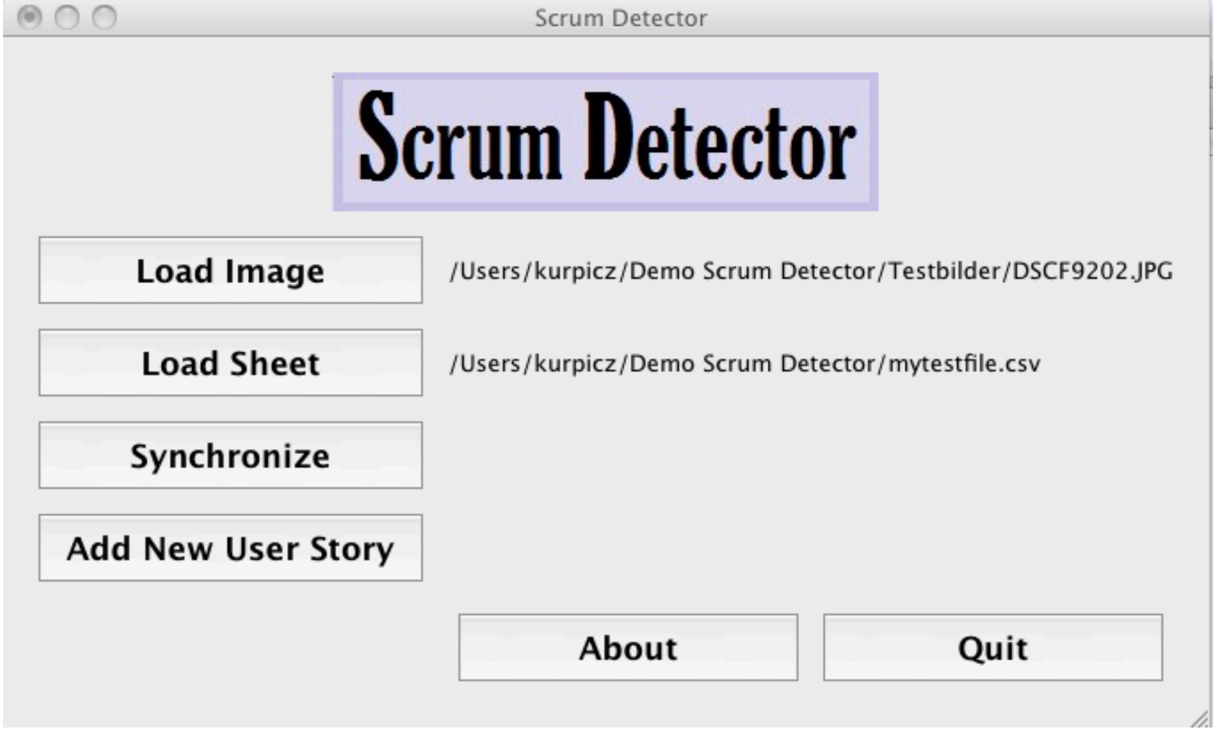
The Scrum Wall for the synchronization with Scrum Detector has 3 columns: open, in progress and done. For the prototype a rudimental Scrum Wall with a black background and white user stories is used.

The priority is set in the lower right quarter of the user story, below the barcode. In the white space post-its of different colors can be added to define the priority. The following priorities are possible in the current prototype:

- pink post-it: high importance
- yellow post-it: medium importance
- green post-it: low importance

For the synchronization, Scrum Detector reads an image in JPEG-format of the Scrum Wall. A spreadsheet from Open Office with comma-separated values will be chosen for the synchronization.

When the synchronization starts, the image will be analyzed and the status of the user stories will be updated in the spreadsheet. Also the priority will be updated if changed and added for new user stories that are synchronized for the first time.



/Users/kurpicz/Demo Scrum Detector/Testbilder/DSCF9202.JPG

/Users/kurpicz/Demo Scrum Detector/mytestfile.csv

Figure 11 – screenshot of Scrum Detector

[1] http://en.wikipedia.org/wiki/Code_39

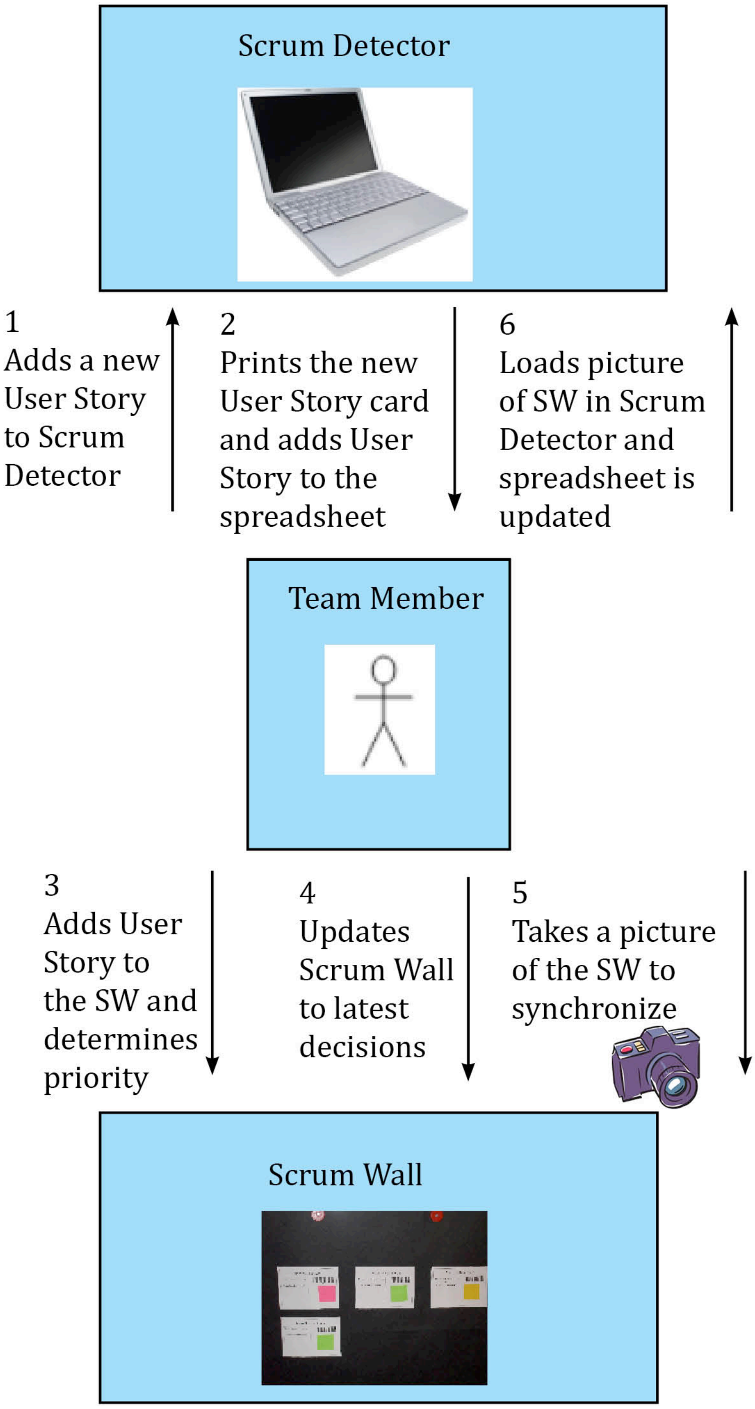

Scrum Detector
1
Adds a new User Story to Scrum Detector
2
Prints the new User Story card and adds User Story to the spreadsheet
6
Loads picture of SW in Scrum Detector and spreadsheet is updated
Team Member
3
Adds User Story to the SW and determines priority
4
Updates Scrum Wall to latest decisions
5
Takes a picture of the SW to synchronize
Scrum Wall

Figure 12 – the Scrum Detector

8.3 What stands behind

Scrum Detector is implemented in C++. The graphical user interface is realized with QT, a cross-platform application and UI framework.[2]

For the image processing a special library was required: the OpenCV framework was used here to support the different image operations required. OpenCV (**Open** Source **C**omputer **V**ision) is a library of programming functions for real time computer vision.[3]

8.4 What can be done

Due to a short development time for Scrum Detector, this prototype can be improved in various aspects. For example, the image recognition algorithm that finds the user stories can be improved. At this moment it is very weak with regard to light conditions (daylight, artificial lights and so on) and it encounters problems when the photo is not taken from an ideal perspective and position.

In general, the design of the classes can be improved. Due to the short development time, there was no time for special effort with regard to refactoring and optimization of the class design.

With regard to functionality, the algorithm to generate and evaluate barcodes can be improved. In the current version it accepts only user stories with ID from 1 to 9. This might be acceptable for the prototype, but surely that are too few user stories for a real project. When enhancing this feature, some attention must be paid to the barcode. The barcode will be longer then the current space and therefore the image recognition algorithm must be adapted.

Furthermore the Scrum Detector must be adapted to the team's needs. For example, the remaining effort in hours could be tracked. Also a strict separation between User Stories and tasks could be implemented, giving the possibility to track both categories.

To learn more about Scrum Detector see http://scrum.kurpicz.ch/scrum_detector.html.

[2] http://qt.nokia.com/products/

[3] http://opencv.willowgarage.com/wiki/

9 Conclusion

This work makes a contribution to provide insights into how Scrum is used in practice and how tools support it. To get information from real-life projects, personal interviews and a 5-10 minute online-survey were done with people involved in Scrum projects.

The personal interviews give evidence that at a first sight there seem to be just some developers who are standing around the Scrum Wall and no tools seem to be needed, but when going deeper things are getting more complex. Often tools are used for example for legal reasons, in distributed teams or because of the non-agile context. The work provides therefore insights in the dynamics of software development.

This work gives insights into important themes mentioned often in different personal interviews and presents abstract needs formulated as actions. Transparency, awareness, flexibility and involvement are just some of the themes.

In order to get more empirical data about the abstract needs mentioned, a short online survey was created. One part of that survey was dedicated to the importance of the needs mentioned above. The results give evidence that there are daily needs such as reaching consensus/taking collaborative decisions, exchanging information, knowing what I should do and knowing what the others are doing and occasional needs such as the conservation of long-term knowledge, the assessment of progress, the organization of work/duty and accurate and trustable documentation.

The daily needs are often considered of higher importance, but nonetheless the use of a tool is often justified by an occasional need. For example, the conservation of long-term knowledge might not be of daily importance, but when a key person leaves the team, it immediately becomes an issue.

The spider charts done for the Scrum Wall and an issue tracker give evidence that the Scrum Wall shines in some areas and does not support well some other needs. It has to be considered that it was never intended for this purpose. The issue tracker has rather flat data. This helps to understand in which areas the issue tracker and the Scrum Wall are working well and can be applied.

The synchronization of information between different tools, in particular between the Scrum Wall and an issue tracker, is seen as problematic. A possible solution would be the automated synchronization of the data from the Scrum Wall into a computer-related form (for example a spreadsheet or a database). Another contribution of this work is a prototype of such a lightweight solution for synchronization.

In the survey results an insight about considerations people make about different solutions is provided: many say that an integrated platform could be a solution; only few see advanced technologies such as big touchscreens as offering a solution for the synchronization issues.

In some cases, interpersonal issues, for example the communication within the team, also caused difficulties. Tools cannot resolve this kind of issue.

Appendix

Appendix A: Interview script

Please see
http://scg.unibe.ch/wiki/students/mkurpicz/ToolsupportforScrum/interview

Appendix B: The survey

Please see
http://scg.unibe.ch/wiki/students/mkurpicz/ToolsupportforScrum/surveyquestions

Appendix C: Survey results

Please see
http://scg.unibe.ch/wiki/students/mkurpicz/ToolsupportforScrum/surveyresults

List of figures

List of tables

Bibliography

[1] John S. Reel: Critical Success Factors in Software Projects. In *IEEE Software*, pages 18 – 23, May/June 1999.

[2] Dave West and Jeffrey S. Hammond: The Forrester Wave: Agile Development Management Tools, Q2 2010.

[3] J. Sutherland, G. Schoonheim, E. Rustenburg and M. Rijk: The secret sauce for hyperproductive offshored development teams. In *Agile, 2008. AGILE '08. Conference*, pages 339 – 344, 2008.

[4] L.S. Møller, F.B. Nyboe, T.B. Jørgensen and J.J. Broe: A Scrum tool for improving Project Management. In *Flirting with the Future, Prototyped Visions by the Next Generation, Proceedings of the Fifth Student Interaction Design Research Conference (SIDeR '09)*, pages 30 – 32, 2009.

[5] E.A. Engum, Z. Racheva and M. Daneva: Sprint planning with a Digital Aid Tool: Lessons Learnt. In *Software Engineering and Advanced Applications, 2009. SEAA '09. 35th Euromicro Conference on,* pages 259 – 262, 2009.

[6] S. Palmer and J. Felsing: A Practical Guide to Feature-Driven Development, Prentice Hall, 2002.

[7] http://www.scrumalliance.org/pages/what_is_scrum, last access: 2.2.2011.

[8] http://en.wikipedia.org/wiki/Scrum_%28development%29, last access: 2.2.2011.

[9] http://www.scrumalliance.org/articles/39-glossary-of-scrum-terms#1116, last access: 2.2.2011.

[10] http://www.mountaingoatsoftware.com/scrum/release-burndown, last access: 2.2.2011.

[11] http://agilemanifesto.org/, last access: 2.2.2011.

[12] R. Jocham: Scrum: Stark im Team. In *Computerworld August 2010,* pages 24-25, 2010.

[13] K. Schwaber: The Enterprise and Scrum. Microsoft Press, page 1, 2007.

[14] I. Nonaka: The Knowledge Creating Company. In *Harvard Business Review November-Dicember 1991*, pages 96 – 104, 1991.

[15] http://alistair.cockburn.us/Information+radiator, last access: 2.2.2011.

[16] http://www.hsr.ch/fileadmin/user_upload/customers/hsr/Aktuelles /ForumInformatik/scrum.pdf, last access: 2.2.2011.