

How Scrum Tools May Change Your Agile Software Development Approach

Matthias Eckhart^(✉) and Johannes Feiner

Internet Technologies and Applications, FH Joanneum, Werk-VI-Strasse 46,
8605 Kapfenberg, Austria

`{matthias.eckhart,johannes.feiner}@fh-joanneum.at`

<http://www.fh-joanneum.at/itm>

Abstract. A major problem for distributed Scrum teams is proper communication between the involved parties to ensure the quality of the final product. This is especially true for coordination issues such as sharing requirements, time schedules, to-dos and code artefacts. Hence, ScrumMasters complain frequently about software tools not suiting their daily needs when supporting agile teamwork, finally leading to the fact of not using a Scrum tool at all. In this paper we describe the extensive interviews held with selected ScrumMasters in which they explained their current tools and the existing gap to their real needs. Within this context, they were able to define the features and aspects they really need. After collecting those requirements for their daily work, we extracted the most wanted ingredients, prioritised them and finally forged them into an Open Source tool called Scrumpy, helping us to present a first solution, which focuses on the agile philosophy of Scrum and the elements needed most by ScrumMasters. Features of Scrumpy include, for example, web-based access to the task board with real-time updates, advanced dashboard visualisation techniques and a sophisticated chat system, which enables effortless communication for distributed teams. Although we already have first anecdotal feedback from users, we plan to improve the tool in a next step by adding more commodity features, perform additional mobile usability tests and systematically evaluate Scrumpy with a large number of end users.

Keywords: Scrum tools · Scrum · Agile software development · Distributed Scrum · Scrum task board

1 Introduction

Understanding potential benefits and risks of Scrum tools is crucial to the success of Scrum in agile organisations, which prefer the use of a software-based tool to assist participants of the Scrum process in handling their daily tasks. Since more and more organisations decide to switch to the agile way of software development [1], the need for a suitable Scrum tool grows substantially [2]. As a result, a vast amount of new products enter the global market, leading to the fact that

Scrum tool producers try to outsell each other. Although the product may fulfil its brand-promise, there is no guarantee that participants involved in Scrum, benefit from features offered by the Scrum tool in the end.

In the last few years there has been a growing interest in reorganising the organisational structure by establishing remote teams at low-cost locations, in order to reduce personnel expenses [3]. As a consequence, further issues in geographically distributed teams may emerge, such as time differences, technical challenges and cross-cultural communication problems [4]. To overcome collaboration barriers across remote teams, the proper use of Scrum tools may be vital to the success of the company in the long run. This actually raises the question what the essence of a valuable tool for supporting efficient agile development is. The study at hand, therefore, aims at investigating flaws regarding the use and development of Scrum tools that needed to be discussed, addressing the correlation between software-based tools and Scrum.

This paper is structured as follows. The next section, Sect. 2, reviews related work in the field. Section 3 describes the applied research methodology. In Sect. 4 we put forward several determined issues concerning the use of Scrum tools and involved risks that might have a negative impact on the Scrum team's productivity. Section 5 proposes concepts for using Scrum tools more effectively. Furthermore, we present new useful features and potential enhancements that can be implemented in existing Scrum tools. In addition, we provide in Sect. 6 a list of minimum requirements that should be implemented to avoid dissatisfaction among users — suggestions put forward by the interviewed ScrumMasters. In Sect. 7, we introduce our developed Scrum tool based on the listed minimum requirements. Finally, in Sect. 8, we draw some concluding remarks and discuss future work concerning this study and the developed Scrum tool.

2 Related Work

Moe et al. [5] indicate that conflicts between team members may emerge when making the switch to agile software development, since additional work can be incurred, in order to foster self-managing teams. Compared to traditionally managed teams, the characteristics of shared leadership embraces a joint decision-making process which requires a strong team commitment. As a result the individuals' ability to work collaboratively with other team members is fundamental to a self-organising team. In addition, Moe et al. reinforce their doubts regarding a team of specialists, because this culture may encourage an individual autonomy which can result in a lack of focus for the team's goal. Furthermore, they describe that this jeopardy may evolve if each software developer is responsible for a separate module, as they may rarely be involved in other components of the system and consequently tend to get side-tracked. According to their study, developing software with Scrum in a remote setup is likely to exacerbate the issues of an individual leadership. Moe et al. also describe that collocation facilitates communication within the team and fosters the ability to achieve a high level of collaboration, which encourages a break of the hierarchical relationship

and as a consequence initiates to build trust and commitment. As stated by Scissors et al. [6], distributed teams may face severe challenges when creating a team culture among individuals from different nationalities. Furthermore, they describe that distributed team members, who rely heavily on nonverbal communication, are particularly prone to conflicts, which may endanger social harmony. In addition, previous research indicates that there is a strong coherence between the communication efficiency and the used communication method. Rayhan and Haque from *Code 71, Inc.* [7] state in their paper that the use of e-mail and Microsoft Excel to manage their remote Scrum team has been proven to be impractical, especially as the team size grew. They also reported that the communication of the daily project status via e-mail posed a distraction for the Scrum team. As a result, they decided to introduce the agile project management software *VersionOne*. However, due to a lack of intuitive user experience design and insufficient support for collaboration, they used *VersionOne* only for time tracking purposes and introduced *Basecamp*, a tool which seemed to fit their needs for managing backlogs. However, transferring artefacts from one tool to another resulted in productivity decrease and consequently engendered the desire to create a seamless process.

A different approach in implementing agile development has been reported by Sarkan et al. [8] who used Scrum tools to support the agile requirements development for software projects at the research and development centre *MIMOS Berhad* in Malaysia. In the initial phase of introducing Scrum, they decided to use *Redmine* for gathering user stories. Although the tool provided an extensive issue tracking system, they recorded bugs in separate tools such as *Rational ClearQuest*. Given that *Redmine* takes full effect of traditional project management, rather than agile methodologies, they decided to substitute the tool with *JIRA*. As a result they could benefit not only from its product backlog features, but also from the task board to monitor the team's progress. In addition, they decided to track issues with *JIRA* which enabled them to link the reported problems to user stories. Furthermore, Sarkan et al. stated in their study that the requirements management effort has been reduced to a minimum degree, owing to the introduction of a Scrum tool. Yet, these findings point towards a tool supported Scrum process, little is known about how ScrumMasters assess the impact of Scrum tools on core Scrum values and principles.

Since there was a shortage in quantitative information related to the needs of a company with regard to Scrum tools, Azizyan et al. [2] conducted a survey to identify the most important aspects of the tools used. They analysed the responses to questions of 120 companies, including collocated as well as distributed team structures. According to their survey results, 65 % of collocated teams and more than 80 % of distributed teams used agile project management tools to support their processes. As a result, we imply that Scrum tools take on an important role in agile software development. Although the study from Azizyan et al. provides comprehensive results about the needs of agile software development companies, further research to determine insights from the ScrumMaster's perspective is still required.

Previous research has indicated that the use of a physical task board in collocated teams is strongly recommended, while the use of its digital equivalent can be regarded as negligible, unless there are substantial reasons which justify its implementation [9]. In contrast, the use of a tool, to support distributed Scrum teams in managing the product backlog, is considered as essential for coordinating the teams at remote locations [10]. Furthermore, the results obtained in [11, 12] suggest that particular attention needs to be paid to the characteristics of the Scrum tool’s digital task board, as well as to the implementation process when migrating from other platforms such as *SharePoint*.

3 Methodology - A Case Study with Interviews

Since extensive research, for example, by Azizyan et al. [2] has addressed various tools used in 120 different companies, we chose a rather qualitative than quantitative approach in which we identified the role of Scrum tools in agile software development, by taking viewpoints of ScrumMasters into consideration. Therefore, we conducted a case study, interviewing five selected Scrum experts for an in-depth analysis of their experiences in (remote) Scrum teams and their work with current Scrum tools. The male interviewees were acquired via the Scrum User Group Graz (XING) and by directly contacting medium-sized enterprises in Austria. At the time of the study, their average expertise with Scrum was 6.2 years, as ScrumMasters 3.4 years and all of the participants were Scrum Alliance® certified. The average development team size was 8.2, whereas two ScrumMasters out of five had to handle two teams at a time.

The scientific method of our empirical observation did not focus on quantitative (online questionnaires, statistical results), but high qualitative data which we received by means of detailed and thorough semi-structured interviews with Scrum practitioners in winter 2014. We used a holistic approach to examine the real-life context, by asking this small group to report about their positive and negative experiences as well as their suggestions for future improvements. Prior to the case study, we prepared a guideline for the interviews which was split into three parts. The first part covered background information about the participants and the company they are working for, such as experience with Scrum, size of enterprise or the office cubicle (workspace) design. The second part focused on a variety of aspects related to the implemented Scrum processes. For instance, we wanted to examine their Scrum team structure, as well as the Scrum activities carried out. To name a few examples, we asked them about the applied effort estimation technique, the development team size or the iteration planning, in order to achieve a broad range of questions. The last component of our three-part interview primarily dealt with the introduced Scrum tool. To encourage the interviewees to talk about the issues faced owing to the used Scrum tool, we asked questions like “*What are the key problems with digital task board usage?*”, “*Did you configure different access permissions for artefacts in the Scrum tool?*” or “*Which reports are generated for the management?*”. As suggested by Seaman [13], we avoided polar questions for collecting qualitative

data, in order to make it easier for participants to expound about the discussed topics. Based on this, we asked 37 open-ended questions in total, within a time frame of 20–60 min. To ensure qualitative research results, we collected data by recording the semi-structured interviews. After the interview sessions were held, we transcribed the recorded audio and subsequently coded our transcript by labelling relevant sections. After we formed a set of codes, we examined the assigned passages in our transcript with respect to repetitions and similarities among the interviewees opinions. Finally, we grouped selected codes which we thought were most important and created themed categories. To gain profound insights into Scrum tools from the perspective of the interviewed ScrumMasters, we interpreted the data.

The limitation of this case study is the small number of participants, which does not really allow to extract statistically relevant data. Nevertheless, based on these findings, we developed a real-time¹ web-based Scrum tool named Scrumpy (see Sect. 7) to examine the link between Scrum and Scrumpy further.

4 Identification of Critical Problem Areas

The following section is devoted to potential deficiencies with regard to Scrum tools and its involved parties. Our aim in this chapter is to raise awareness of several issues concerning the use of Scrum tools, as well as the introduction of the tool in real projects.

4.1 The ScrumMaster’s Role as a Critical Success Factor

According to the interviewed experts, it is still prevalent that IT managers do not acknowledge the ScrumMasters’ work, since they do not produce any code by themselves. As a result, they often do not hire ScrumMasters, but rather retrain existing project managers or developers. Unfortunately, this leads to wrong organisational change, in which an individual occupies two roles simultaneously, e.g. ScrumMaster and software developer or ScrumMaster and Product Owner. Although personnel cost may be saved, Scrum cannot be used to its full extent and continual improvement may not be achieved in the long run, due to the negligence of the ScrumMaster’s duties.

As stated in the agile manifesto, direct communication is the most efficient and effective method of transferring information [14]. One issue that may emerge in this context is the misuse of the Scrum tool as the main way of communicating within a collocated team. Despite the fact that Scrum tools may simplify the act of communicating, especially with distributed teams, they obviously cannot substitute face-to-face communication. This issue is not limited to the live chat features of a Scrum tool, because even the update of a task on the virtual task board will result in a loss of nonverbal communication details such as body

¹ Our application is capable of handling live updates, i.e. data changes will be pushed to clients instantly, without the necessity to refresh the page manually.

language or gestures. Findings of the research conducted by Segers [11] confirm our study results that declare physical task boards as a gathering place for the Scrum team which establishes open communication and subsequently improves collaboration since individuals are able to perceive every single movement or unconscious body signal. As a result, meeting participants should be encouraged to start a discussion if discrepancies among the Scrum team members exist. Therefore the ScrumMaster's task and obligation is to raise awareness regarding the importance of communication. The interviewees past experience suggests that this task requires a lot of effort. For that reason, they prefer a strict observance of the roles, allowing the ScrumMaster to focus only on the tasks defined in the Scrum framework, having no other burden with obligations outside his or her remit.

Furthermore, the data obtained from the interviews indicate that ScrumMasters need to take special care of conflicts caused by obsessive micromanagers. The evaluation of the interviews shows that managers tend to check the progress of the Scrum team based on the burn down chart on a daily basis, even though this tool should be designed by and for the development team only. However, managers should definitely have full access to the Scrum tool including the burn down chart provided by the tool, otherwise the process will become non-transparent and eventually will have a strong negative impact on the collaboration between the Scrum team and other parties, due to feelings of anxiety as managers may feel out of control.

4.2 Inflexible Scrum Tools Limit the Agility of Scrum

One of the major advantages of Scrum in comparison to traditional project management is that the stated agile method offers a certain degree of flexibility with regard to the defined processes. Because of this, the Scrum team is able to maintain continuous improvement and consequently may increase efficiency over time. Therefore, it is even worse if the introduced Scrum tool implicitly requires certain parts of the Scrum process, which cannot be bypassed when using the software.

One interviewed expert exemplified this issue by outlining a workaround which allowed the Scrum team to keep up with the sprint rhythm of the Scrum tool if the development team needed an extra sprint for fixing bugs, especially shortly before a major release. Although all increments were finished according to the Definition of Done (DoD), some builds failed after the integration of critical components. As a result, the Scrum team split the sprint of two weeks in half and used one week for integration tests and one week for fixing bugs. Carrying out integrations multiple times during the sprint is certainly the best-case scenario, but unfortunately this is often not attainable in real-world systems.

Another significant point about the inflexibility of Scrum tools concerns the type of effort estimation, specified by the Scrum tool. Since the preferred effort estimation method may vary from development team to development team, a Scrum tool should not predefine the technique to estimate effort. According to

our interview results, a Scrum tool should provide a feature to record the estimated effort, but any inputs related to the effort estimation should definitely not be designed as mandatory. This request applies to the effort estimation of user stories as well as tasks, because some development teams may skip the estimation of tasks, due to the fact that this may be too time-consuming. Moreover, the selected effort estimation technique affects also the visualisation method, for the simple reason that the measurement of work progress is often deduced from the estimated effort. For instance, the data unit to visualise the work progress can be hours, story points, user stories or tasks completed. Furthermore, even an overlay of multiple data units may be desired. As a result, the development team should have the opportunity to choose the data unit for visualisation methods, provided by the Scrum tool.

Further concerns stem from the fact that Scrum tools are considered to be hard to adapt with regard to process changes. For instance, if the Scrum team decides to record a new attribute for user stories on the digital task board, extensive configurations may apply which could require reading the documentation. In contrast, the physical task board can easily be changed and is not limited to any settings.

Table 1. A comparison of supported characteristics by the physical task board and its digital equivalent.

Criteria	Physical task board	Digital task board
Ease of change	✓	Problematic
Remote capabilities	✗	✓
Social aspects	✓	Problematic
Versioning	Problematic	✓
Haptic	✓	✗

Find in Table 1 a comparison of several physical and digital task board characteristics.

4.3 Agile Organisational Structures Apart from the Traditional Scrum Framework

The expert interviews revealed that there are tremendous discrepancies between the roles which are specified by the Scrum tool and the roles defined in large-scale organisations. One real-world example of a large-scale organisation is presented in Fig. 1. This organisation type extends the typical Scrum roles by a project manager, a Chief Product Owner (CPO), a Product Manager (PM) and the three cross-divisional roles: Chief Software Architect (CSA), one person in charge of writing the documentation (DOC) and the Quality Assurance Manager (QA).

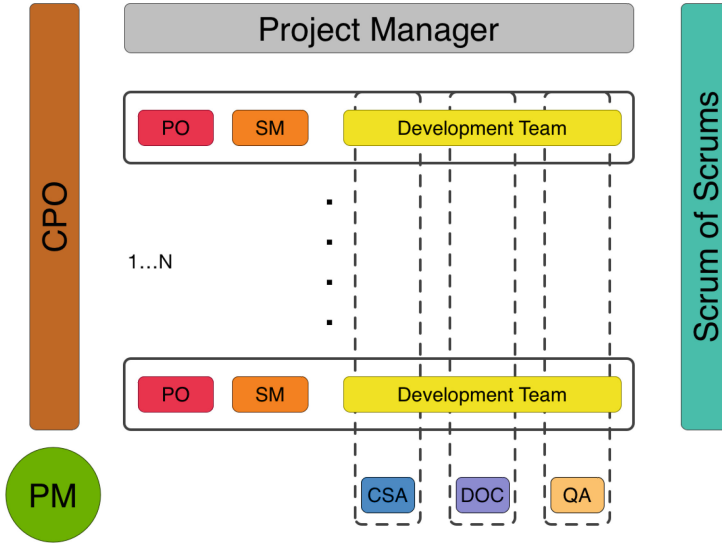


Fig. 1. Organigram of an extended agile organisational structure.

Especially when scaling Scrum in a large enterprise, while remaining agile is desirable, a complete mapping of all the roles in a Scrum tool is often unachievable. After all, the organisation should not be forced to adapt the roles, specified by the Scrum tool but rather the tool should provide a sophisticated role management feature, so that a straightforward mapping of the organisation is attainable or further role separation for the use of the Scrum tool is not needed at all.

4.4 Insufficient Overview of the Digital Task Board Due to a Mass Amount of Information to Display

All interviewed experts complained about an insufficient overview of the digital task board. The main reason for their complaint was that meeting participants may lose the overall context of the discussed tasks when scrolling to the next user story, due to fact that not all of the tasks could be displayed at once, making reading difficult. One interviewee took advantage of this *issue* by forcing the development team to focus only on three user stories at a time. This way, all of the user stories and corresponding tasks in process could be displayed at once and as a consequence, scrolling was unnecessary. However, there is still a demand for an improved design of the task board to display the entire content at once, without compromising readability.

4.5 Scrum Tools Hosted in the Cloud

Numerous producers of web-based Scrum tools, such as *Atlassian*, sell cloud hosting services in addition to the developed product to enable customers a fast

and easy launch of the Scrum tool in their organisation. Although cloud hosting may be a more convenient way than deploying the Scrum tool on a self-hosted server, there are a few things worth taking into consideration when using third-party cloud services for hosting a Scrum tool.

First of all, Scrum tools are used to support the agile software development process and therefore need to handle data which may betray customers' privacy to a certain degree. Unfortunately, it cannot be guaranteed that the service provider will not face any data privacy issues. In addition, they often have their own privacy policies, which may exclude coverage for privacy and security liabilities. Nevertheless, the cloud hosting provider should hold a valid industry-standard security certification which verifies a safe hosting environment at a high level.

Secondly, relying on a Scrum tool which is hosted by a cloud service provider could have a huge negative impact on the productivity of the Scrum team, if the tool is not available when needed. Even more attention should be paid to this potential risk, if the Scrum tool is used as a central information system and business relevant data must be accessible all time. Since hosting issues may not be resolved immediately after occurrence, the web-based Scrum tool can be offline for a longer period of time. However, the existence of this problem could not be verified, since all interviewed experts host the Scrum tool in their internal IT infrastructure, due to the previously stated potential risks.

Owing to both of the mentioned reasons, all of the interviewed ScrumMasters recommend the self-hosted solution, because the use of a cloud hosted Scrum tool will result in an undesirable dependency to the service provider. However, Scrum teams in small companies may benefit more from cloud hosted Scrum tools, since they do not have to care about the deployment of the server and the ongoing periodic maintenance tasks.

4.6 Challenges in Introducing Scrum Tools in an Organisation

According to our study, organisations may face problems related to practicing Scrum if the Scrum tool has been introduced at the same time as the agile methodology itself. One of the main risks related to this approach is that the Scrum team may deduce the Scrum process from the features of the Scrum tool. In this way, they will lose all possibilities for continuously improving the Scrum process from the very beginning. As the Scrum tool may provide ready-made processes, the Scrum team often tends to carry out mindlessly instructions the Scrum tool gives. Two interviewed ScrumMasters compared this pitfall to *SAP* implementation failures due to massive internal organisational changes across a variety of business segments in the company. As a result, it is highly recommended to start introducing a Scrum tool after Scrum has been successfully established in the organisation. Especially at an early stage, the use of a physical task board, to visualise the sprint backlog with sticky notes, may be more effective when practicing Scrum. Only after the Scrum team has understood the importance of face-to-face communication, they can try to make the switch to a digital task board, provided by the Scrum tool. Nevertheless, the physical task

board should be available at any time during the first sprint conducted with the Scrum tool, allowing the team members to switch back to the old method easily. Ideally, a part of the sprint retrospective meeting should be used to reflect on how the Scrum tool was used during the sprint and if it should be used in future sprints. As a consequence, the Scrum tool should provide features to perform a fast migration, so that an implementation later on in the Scrum process is still attainable.

5 Concepts for Success Regarding the use of Scrum Tools

In this section, we present examples for a better understanding of how to use Scrum tools effectively and make suggestions how they could be improved.

5.1 Agile Software Development with Distributed Teams

The coordination of distributed teams requires a lot of effort to keep the collaboration among distributed team members vibrant, ensuring an ongoing productivity on a high level. Considering the economic aspects of distributing teams globally, it may be worth the management effort. However, beside differences in culture and language problems, also technical issues may emerge. One interviewed ScrumMaster reported that they started to use Skype as a video conferencing tool for their daily Scrum meetings on both of the two locations. The interviewee also pointed out that the use of a HD camcorder instead of an off-the-shelf webcam for live streaming is beneficial in terms of video quality and mobility. Furthermore, the ScrumMaster mentioned that the use of a professional studio microphone is more qualified for the stand-up meeting than a built-in microphone, because it can be passed along from person to person. Although the quality of the video calls was acceptable for streaming the daily Scrum meetings, a professional video conferencing system may be superior if more than two distributed Scrum teams participate.

However, according to the interviewees, recording the physical task board is useless, since meeting participants on other locations cannot read the streamed content of the user stories and the corresponding tasks. Even transmitting high-resolution photos of the physical task board has been proven as inefficient. This issue has been resolved by using a Scrum tool and installing an additional monitor on each location in order to view the digital task board and the video conference simultaneously. In addition, meeting participants may benefit from the use of a screen sharing software for mirroring the Scrum tool, because this way they are able to keep track of the discussed topics. Despite that the proposed solution may work to some extent, the ScrumMaster suggested that the Scrum teams should meet at each others location on a rotating basis at the end, as well as at the beginning of the consecutive sprint, if the project enters a critical stage and the travel expenses are affordable.

5.2 Optimising the Digital Tool Landscape

Scattering documents across multiple tools is a known problem for IT companies in general. Especially in companies which specialise in the development of software, this issue may be even more important since they may need a tool to track down bugs. In addition, specifications for the implementation of the developed product may also be managed on data storage systems or with web content management systems such as *MediaWiki* or *TWiki*. In general, there is a need for a revision control system specifically for documents, on the one hand to prevent data inconsistencies, due to concurrent processing of the same file by multiple users, and on the other hand to keep track of the changes made. According to Møller et al. [15], there is also a requirement for an internal logging feature, which tracks the project's progress and in further consequence provides historical data of everything that has happened during the sprint. Their findings suggest that owing to a Scrum tool's project history feature it would be no longer necessary to review extensively the progress of tasks and ipso facto time can be saved in discussions.

As the software maintenance effort increases by every new tool that will be integrated in the existing IT infrastructure, a Scrum tool which could eliminate other tools currently being used in the organisation would be ideal. For instance, a real-time collaborative editor feature like *Etherpad* could replace document processing tools. Moreover, the interviewed experts confirmed the findings of Sarkan et al. [8] that the integration of a bug tracker in the Scrum tool may simplify the bug fixing process, owing to referenced bugs on the digital task board. As a result, the establishment of the Scrum tool as a single environment like a *one-stop-shop* would not only be cost and time efficient, but also may increase the productivity of the Scrum team. Nevertheless, a Scrum tool capable of being used for many purposes in the context of software development may exhibit an undesirable strong dependence and consequently can become a potential single point of failure.

5.3 Integrated Reporting Solution

If the Scrum team works with the Scrum tool on a regular basis, the tool has great potential to increase efficiency by automating complex analysing and reporting tasks. In particular, measured agile metrics are relevant to managers, so that they can address problems and resolve incidents early enough. Typically, measuring and tracking the velocity is significant to estimate the rate of progress of the development team. In this way, managers can determine the project status to make sure that the arranged delivery date can be met. Furthermore, an evaluation of past impediments may increase the team's productivity if recurring issues can be fixed permanently. As Scrum tools may offer features that are able to extract data for reporting purposes or able to integrate a reporting system seamlessly, they could provide support for analysing and evaluating the team's performance. However, as previously stated, the ScrumMaster should give particular attention to prevent the emergence of a command-and-control leadership style by managers that may cause interferences in the development team's work.

5.4 Simulating the Look and Feel of a Physical Task Board

The design of the digital task board can have a huge impact on the usability of the Scrum tool. According to our results of the expert interviews, the layout of the digital task board should be based on an ordinary physical task board, because the Scrum team members may recognise it as a familiar object to work with. Typically, user stories and the corresponding tasks are displayed as virtual sticky notes in a matrix view. Although the overall design of the digital task board is important, the main focus should be put on the interaction with tasks. The interviewed ScrumMasters emphasised that they prefer drag-and-drop functionality for moving the virtual sticky notes to other task status columns, because this way of interacting is more intuitive than updating the task's status by setting a specific value in a drop-down list, to mention one prominent example. Consequently, users may get more engaged with using the Scrum tool.

In contrast to the capabilities of a physical task board, the digital task board provided by the Scrum tool may also be able to display the user's avatar on the virtual sticky note, showing the responsible team member's particular tasks. In this way, social aspects of an online community can be integrated, increasing the interpersonal trust in distributed teams [16]. In addition, the highlighting of tasks, which remain more than one day in progress, is beneficial for identifying problems or indicating that complex tasks should be broken down into sub-tasks.

5.5 Printable User Stories and Tasks

Scrum teams that want to take advantage of the physical, as well as the digital task board should certainly choose a Scrum tool which provides an export feature for user stories and tasks. In this way, the content of the digital task board can be printed and subsequently attached to a whiteboard or wall. The status of the tasks can then be synchronised manually with the digital task board periodically or at the end of each sprint. One interviewed expert recommended the *JIRA* plugin *Agile Cards* which enables an automated synchronisation by importing a photograph of the physical task board to *JIRA*.

6 Minimum Requirements for Scrum Tools

In the following section, we present minimum requirements that a Scrum tool must meet, as reported by the interviewed ScrumMasters. Not meeting these requirements may cause dissatisfaction among Scrum team members. Since the Scrum team's needs may vary, our aim is to provide a brief overview of fundamental characteristics which a Scrum tool should have implemented to its full extent.

As listed in Table 2, the suggested minimum requirements are deduced from remarks of prior sections, but are not ordered by importance or any other criteria.

Table 2. Determined minimum requirements for Scrum tools, based on our study results, including specific quotations made by the interviewed ScrumMasters.

Minimum requirement	Description	Extracts from the interviews
Information hub	The Scrum tool must be established as the primary source for information regarding the agile development process, in order to eliminate multiple subject-specific tools like Wikis and document management systems	“Developers should use the Scrum tool as a daily companion, assisting users in doing their work by providing relevant information concerning the project.”
Usability	According to the ScrumMasters’ opinions, the management may agree on a training for eight hours at the maximum, therefore the Scrum tool must be easy to use and consequently require little or no need for training	“As Scrum is easy to understand, the initial training phase for users of a newly introduced Scrum tool should have a similar learning curve. In this particular context, usability is definitely a factor to consider.”
Availability	The tool must be available during working hours at the minimum. In case multiple teams are employed globally at different locations, the availability of the Scrum tool must be adapted according to potential time differences	“I have been informed that <i>iceScrum</i> ’s datacenter faced network issues several times which caused a downtime of the Scrum tool and affected customers of their cloud service.”
Flexibility	The Scrum tool must preset roles which can be mapped with enterprise organisational structures. Furthermore, the Scrum tool must offer a variety of effort estimation methods and the specified sprint duration must be modifiable	“Whether a Scrum process or any other activity in an organisation is supported by a tool, it should be able to adapt to the given conditions at all times and not vice versa, in the manner of some failed <i>SAP</i> implementations.”

(Continued)

Table 2. (*Continued.*)

Minimum requirement	Description	Extracts from the interviews
Design of the digital task board	The look and feel of a typical physical task board must be simulated by designing the structure of the digital task board in a similar way and by implementing intuitive interactions for handling virtual sticky notes	“The implemented interaction technique of a Scrum tool which we used prior to <i>iceScrum</i> provided a rather inconvenient way of changing the task’s status, because the interaction with tasks was based on basic GUI elements like a drop-down menu. As opposed to this way of interacting with tasks, dragging a virtual sticky note from one column to another seems more appealing to me.”
Traceability	Changes made to artefacts with the Scrum tool must be traceable via a version control functionality, so that modifications are understandable at any time	“A Scrum tool must provide traceability features, otherwise users are not able to understand the changes made to artefacts. Thus, the artefact’s history is fundamental to the comprehension of information involved in Scrum processes.”
Customisability	The Scrum tool must provide an interface to enable adaptations, either by customising the existing code of the Scrum tool or by being capable of integrating third-party plugins	“Since the implemented Scrum processes often vary from organisation to organisation, it should be possible to adjust the Scrum tool to specific needs and different environments.”
Transparency	Artefacts, managed with the Scrum tool, must be visible for anyone involved in the Scrum process	“The Scrum tool must not limit the access to artefacts, otherwise the implementation of Scrum may be classified as a sinking ship and will result in a failed project or cost overrun.”

(*Continued*)

Table 2. (Continued.)

Minimum requirement	Description	Extracts from the interviews
Reporting	Managers must be able to extract useful data concerning the performance of the Scrum team, in order to get a solid source of information regarding the project status	“For now we use an internal website which provides a status report based on various sources of information. Periodically, I export this website as a PDF file and send the document to the management. Since this manual processing is time-consuming, a Scrum tool, which is able to automatically generate detailed reports at regular intervals would be extremely helpful. Furthermore, in my opinion, the Scrum tool’s reporting feature should put emphasis on simplicity, so that the project’s status is easy to grasp. This could be implemented via a coloured illustration of a traffic light which indicates the current status.”
Integrated communication solution	The Scrum tool must provide an integrated real-time one-to-one as well as a group messaging system to simplify communication with distributed teams	“Managing distributed Scrum teams is challenging. However, a Scrum tool can make a significant positive contribution to the collaboration among distributed team members.”

7 Scrumpy – An Agile Project Management Tool Designed to Skyrocket Your Team’s Productivity

The idea to develop a custom-built, Open Source Scrum tool [17] originated from the need of a lightweight solution which is extensible and allows adjustments which are specific to the environment at hand. Initially, the Institute of Internet Technologies and Applications, a department of the University of Applied Sciences JOANNEUM, used *Scrumy* for managing tasks, but the lack of extensibility in terms of LDAP authentication led to the decision to develop a new Scrum tool in-house. As a consequence, we created a web-based Scrum tool named Scrumpy [18,19] based on the results obtained from the expert

interviews. Scrumpy runs on Meteor, a JavaScript framework for building modern real-time web applications. Owing to the integrated real-time web functionality, using the Scrum tool leads to a more vibrant experience, since data changes become immediately visible without refreshing the page.

Our main goal was to build a Scrum tool which can serve as an all-in-one solution, as well as an auxiliary tool which is limited to the digital task board

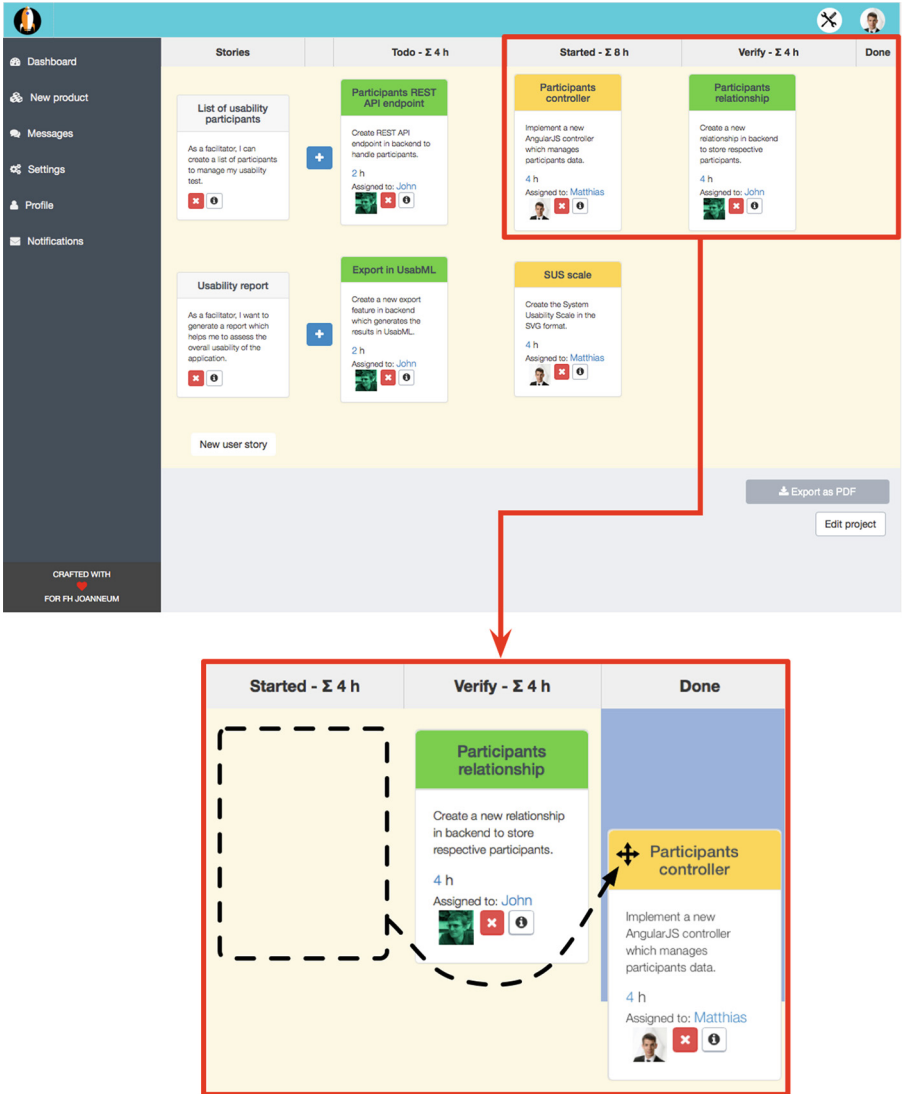


Fig. 2. Users can drag and drop virtual notes in order to update the task’s status.

feature. Therefore we created an advanced and a basic mode² to meet the user's needs. In contrast to the basic mode, the advanced mode also includes a product management feature with detailed statistics on the progress of the product development, an activity stream which enables Scrum team members to easily track changes and a component for managing backlog items. Furthermore, we integrated the library *ShareJS* into the advanced mode, in order to enable users concurrent editing of documents in real-time. Regardless of the selected mode, user stories and the corresponding tasks can be exported as a PDF document, similar to the *JIRA* plugin *Agile Cards*.

According to the assessed needs concerning the usability of Scrum tools in Sect. 4.4, we used a responsive web design approach to take advantage of high-resolution monitors for displaying as much content as possible and not wasting any empty space. Thus, the appearance of the web application is especially beneficial for viewing the digital task board on a modern big-screen as many user stories and tasks can be displayed at once without the need for zooming. Particular attention has also been paid to the design of interactions with tasks and product backlog items. Since the interviewed experts prefer a natural, intuitive way of handling the tasks, we decided to implement a drag-and-drop function-

Table 3. List of minimum requirements and the stage of development, indicating whether the feature is implemented in Scrumpy.

Minimum requirement	Supported by Scrumpy
Information hub	Scrumpy provides a real-time editor which enables multiple users to work simultaneously on the same document. The integration of a file management system in Scrumpy is planned for future releases
Usability	✓
Availability	Depends on the infrastructure of servers and networks
Flexibility	Scrumpy maps the traditional Scrum roles in the advanced mode and provides two different user levels (administrator and team member) in the basic mode
Design of the digital task board	✓
Traceability	✗
Customisability	✓
Transparency	✓
Reporting	✓
Integrated communication solution	✓

² The basic mode is not necessarily intended for Scrum teams, but rather for any team, who wants to manage work visually via a task board.

ality. As a result, users are able to prioritise the product backlog or update the task's status by dragging the backlog item or the virtual sticky note to specific drop zones. Figure 2 shows the digital task board of Scrumpy's basic mode and the way how users interact with tasks.

Although Scrumpy is intended for professional use, we implemented features which should integrate social aspects of a typical online community. For instance, users are able to setup profile pages, upload avatars and chat with other registered members. In this way, the collaboration across globally distributed Scrum teams may be strengthened.

In spite of Scrumpy's huge feature set, we are planning to improve the existing components and continually implementing more functionality. For instance, Scrumpy does not provide a feature to define acceptance criteria for user stories. Furthermore, we plan to implement an internal calendar to schedule appointments and sync it with cloud services like *iCloud*. Last but not least, the integration of *ownCloud* would be desirable.

Find in Table 3 the implemented features according to the identified minimum requirements from Sect. 6.

8 Conclusion and Future Work

The expert interviews revealed problem areas regarding Scrum tools that tend to be crucial to the success of any agile Scrum project. Those issues are not only related specifically to the improper use, but also to the implementation of a Scrum tool in an organisation. For instance, we discovered that the ScrumMaster may face challenging tasks in maintaining effective conversations among team members, since chatting on a Scrum tool is a poor substitute for face-to-face communication. Furthermore, Scrum tools may not be able to map various types of enterprise agile organisational structures completely, which can cause a lack of transparency. This study also investigated problems concerning the introduction of Scrum tools and the hosting of the tool in the cloud, which should be used in caution, since privacy and availability issues may be faced. In contrast, we also analysed how Scrum tools can be used effectively and presented possible feature concepts to improve them.

The practical implications of our study can be summed up as follows. ScrumMasters are sceptical towards the use of Scrum tools in collocated teams, because they appreciate the benefits of face-to-face communication which may be harmed due to the improper use of the tool. Although they think that Scrum tools can be very helpful if they have been introduced in an organisation properly, they certainly prefer a tool-free start of the first agile project, especially if Scrum team members are not familiar with agile-oriented processes. After the first couple of sprints have been successfully completed and the participants respect the agile principles as well as values, they can try to make the shift to a tool-supported Scrum workflow. However, according to the interviewed ScrumMasters, the use of a Scrum tool in a remote setup is almost essential. In this context, we noticed that ScrumMasters who work with distributed teams reconsider their toolset with an emphasis on collaboration features.

Since Scrumpy has not been assessed in real-world contexts, the prototype will be extensively evaluated in the next phase of the development process. This procedure will be done in two steps. First, we want to introduce Scrumpy in a Scrum-based software development course at our university and in further consequence establish the tool as an information hub for students to manage their project work. After that, we want to introduce Scrumpy in local agile software development companies to examine the Scrum tool in a real-world example. Based on this evaluation, we also intend to extend the functionality of Scrumpy in our future research. In addition, future work will also involve usability tests in a desktop and mobile setup, to improve the overall usability of the application and determine if the application can be used on mobile devices.

References

1. The Standish Group International. CHAOS MANIFESTO 2013 - Think Big, Act Small (2013). <http://www.versionone.com/assets/img/files/CHAOSManifesto2013.pdf>
2. Azizyan, G., Magarian, M.K., Kajko-Mattson, M.: Survey of agile tool usage and needs. In: Agile Conference 2011, AGILE 2011, pp. 29–38. IEEE Computer Society, August 2011. doi:[10.1109/AGILE.2011.30](https://doi.org/10.1109/AGILE.2011.30). ISBN: 978-0-7695-4370-3
3. Heller, R., Laurito, A., Johnson, K., Martin, M., Fitzpatrick, R., Sundin, K.: Global teams: trends, challenges and solutions. In: Cornell Center for Advanced Human Resource Studies. Partner Meeting, CAHRS 2010, May 2010. <https://est05.esalestrack.com/eSalesTrack/Content/Content.aspx?file=4578f59e-21b3-4a2c-bbfe-63e53af3f5dc.pdf>
4. Damian, D., Lassenius, C., Paasivaara, M., Borici, A., Schroter, A.: Teaching a globally distributed project course using scrum practices. In: Proceedings of 2nd Workshop on Collaborative Teaching of Globally Distributed Software Development, CTGDSD 2012, pp. 30–34. IEEE Computer Society (2012). doi:[10.1109/CTGDSD.2012.6226947](https://doi.org/10.1109/CTGDSD.2012.6226947). ISBN: 978-1-4673-1818-1
5. Moe, N.B., Dingsoyr, T., Dybå, T.: Overcoming barriers to self-management in software teams. *IEEE Softw.* **26**(6), 20–26 (2009). doi:[10.1109/MS.2009.182](https://doi.org/10.1109/MS.2009.182)
6. Scissors, L., Shami, N.S., Ishihara, T., Rohall, S., Saito, S.: Realtime collaborative editing behavior in USA and Japanese distributed teams. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 2011, Vancouver, BC, Canada, pp. 1119–1128. ACM (2011). doi:[10.1145/1978942.1979109](https://doi.org/10.1145/1978942.1979109). ISBN: 978-1-4503-0228-9
7. Rayhan, S.H., Haque, N.: Incremental adoption of scrum for successful delivery of an IT project in a remote setup. In: Melnik, G., Kruchten, P., Poppendieck, M. (eds.) Agile 2008 Conference, AGILE 2008, pp. 351–355. IEEE Computer Society, August 2008. doi:[10.1109/Agile.2008.98](https://doi.org/10.1109/Agile.2008.98)
8. Sarkan, H., Ahmad, T., Bakar, A.: Using JIRA and redmine in requirement development for agile methodology. In: 2011 5th Malaysian Conference Software Engineering (MySEC), pp. 408–413, December 2011. doi:[10.1109/MySEC.2011.6140707](https://doi.org/10.1109/MySEC.2011.6140707)
9. Perry, T.: Drifting toward invisibility: the transition to the electronic task board. In: Agile, AGILE 2008, Conference, pp. 496–500, August 2008. doi:[10.1109/Agile.2008.62](https://doi.org/10.1109/Agile.2008.62)

10. Berczuk, S.: Back to basics: the role of agile principles in success with an distributed scrum team. In: Agile Conference (AGILE), pp. 382–388, August 2007. doi:[10.1109/AGILE.2007.17](https://doi.org/10.1109/AGILE.2007.17)
11. Segers, J.: Analysis of a paper-and software-based scrum task board. M.A. thesis. Universiteit Twente, September 2012. <http://essay.utwente.nl/62136/>
12. Uy, E., Rosendahl, R.: Migrating from SharePoint to a better scrum tool. In: Melnik, G., Kruchten, P., Poppendieck, M. (eds.) Agile 2008 Conference, AGILE 2008, pp. 506–512. IEEE Computer Society, August 2008. doi:[10.1109/Agile.2008.69](https://doi.org/10.1109/Agile.2008.69). ISBN: 978-0-7695-3321-6
13. Seaman, C.: Qualitative methods in empirical studies of software engineering. IEEE Trans. Softw. Eng. **25**(4), 557–572 (1999). doi:[10.1109/32.799955](https://doi.org/10.1109/32.799955). ISSN: 0098-5589
14. Beck, K., Martin, R.C., Schwaber, K., Sutherland, J., Fowler, M.: Principles behind the agile manifesto (2001). <http://agilemanifesto.org>. Accessed 12 January 2001
15. Møller, L.S., Nyboe, F.B., Jørgensen, T.B., Broe, J.J.: A scrum tool for improving project management. In: Flirting with the Future, Prototyped Visions by the Next Generation, Proceedings of the 5th Student Interaction Design Research Conference (SIDeR 2009), pp. 30–32 (2009)
16. Bente, G., Ruggenberg, S., Kramer, N.C., Eschenburg, F.: Avatar-mediated networking: increasing social presence and interpersonal trust in net-based collaborations. Hum. Commun. Res. **34**(2), 287–318 (2008). doi:[10.1111/j.1468-2958.2008.00322.x](https://doi.org/10.1111/j.1468-2958.2008.00322.x)
17. Eckhart, M.: Scrumpy on GitHub, May 2015. <https://github.com/MatthiasEckhart/Scrumpy>
18. Eckhart, M., da Silva, E.V.: Scrumpy – An Agile Project Management Tool Designed to Skyrocket Your Team’s Productivity, May 2015. <http://scrumpy.meteor.com>
19. Eckhart, M.: Product description of Scrumpy on KMU-goes-mobile, May 2015. <https://kmu.fh-joanneum.at/scrumpy>

Software Quality. The Future of Systems- and Software
Development

8th International Conference, SWQD 2016, Vienna,

Austria, January 18-21, 2016, Proceedings

Winkler, D.; Biffi, S.; Bergsmann, J. (Eds.)

2016, XII, 229 p. 57 illus. in color., Softcover

ISBN: 978-3-319-27032-6