Analysis of a paper- and software-based Scrum task board

Author:
Jan Segers
jan.segers@boombule.com

Business Information Technology MSc Thesis.
Enschede, September 2012.

Graduation commission:
Dr. Klaas Sikkel
Dr. Christiaan Katsma

borisgloger

UNIVERSITEIT TWENTE.

Management summary

The company that hosts this master thesis project is specialized in offering consultancy in terms of an agile software development method called Scrum. One aspect of a Scrum consultant is to advise the client for making Scrum related choices regarding a certain artifact that is called a task board. A task board can be based on plain paper or on a dedicated software tool. The purpose of this master thesis project was to get insight into the advantages and disadvantages of a paper-based respectively software-based solution to form a common ground for Scrum consultants that have to advice their clients in this regard. To do so, I created in this master thesis guidelines that help companies to decide in choosing the matching task board variant.

This master thesis project draws on input from multiple sources. Available literature has been reviewed, company-internal and company-external experts have been consulted, and field research has been performed by visiting companies that use Scrum to investigate their task board usage. Finally, a set of Scrum software tools has been evaluated. Criteria for comparing software with paper have been identified and defined, next to identifying and defining criteria for a software versus software evaluation.

The results of this master thesis project have shown that paper offers more advantages in terms of a task board than software-based solutions. However the latter become more and more important in times of increasing globalization. The evaluation of Scrum software tools backed up the findings of the paper versus software analysis, which showed that software-based approaches currently suffer from shortcomings in various areas.
During the evaluation of the companies that were subject of the field research, an intermediary solution, which combined the advantages of both variants without introducing any crucial disadvantages, was found.
Next to the software versus paper analysis and the evaluation of software tools, company characteristics have been determined based on the companies that were evaluated. Using the results of the software versus paper analysis, the Scrum software tools evaluation and the company characteristics, guidelines in form of a simple decision graph were created. This decision graph is an easy, yet effective tool for consultants who have to advise companies that face such a decision.

The conclusions of this master thesis project are the following:

- A paper-based task board outperforms its software-based pendant in terms of accessibility, motivation, haptic quality, costs, availability, overhead & communication.
- A software-based task board outperforms its paper-based pendant in terms of flexibility, integration, archiving & distance.
- Companies that want to explore Scrum should stick to the paper-based task board, as it is easy to set up and, moreover, cheap.
- Companies that have multiple Scrum teams, which on top of that also might be geographically dispersed, are literally forced to use a Scrum tool and thus a software-based task board to cover other aspects like reporting, administration, etc.
- The intermediary solution suggests using a paper-based task board that is backed up by a software-based task board. The disadvantages of this solution are increased costs and overhead, however it combines all remaining advantages of both solutions and is thus a best-of-bread solution for companies that have multiple Scrum teams as described above.

Thesis structure

The chapters of this thesis should be read consecutively to understand the line of reasoning. However, for readers who are solely interested in specific pieces of information, the following explanation might be helpful:

- To obtain more details about the background of this thesis, the reader should go to chapter 1. This chapter describes the motivation, the research questions and the methodology that is used to answer those questions.
- Chapter 2 can be used to give readers that are not familiar with Scrum a short introduction. This chapter is divided into three sub parts, which describe the Scrum roles, the Scrum events and the Scrum artifacts. Readers who already are familiar with what Scrum is about are free to skip this chapter, with the exception of section 2.6, in which the task board is discussed in more detail.
- The purpose of chapter 3 is to evaluate the paper-based and the software-based task board. The criteria to perform this comparison are described and then applied to both task board types.
- In chapter 4, several major Scrum tools are tested regarding their ability to visualize a Scrum task board. Again, criteria will be defined and used to perform the evaluation.
- Chapter 5 describes different usage scenarios, gives insight into which task board type different companies are using and which company characteristics influence the choice of a certain task board type. Finally, guidelines that support companies to make a choice are given.
- Chapter 6 concludes this master thesis by stating the theoretical and practical implications, the limitations and by describing further research possibilities. Finally, the research goals are reflected.

Preface

When I received my bachelor degree some years ago, I did not know anything about Scrum or Agile Software Development models at all. I studied Engineering and Computing Science and had worked at several companies as a Software Engineer. However, I was not satisfied – something did not feel right. I liked programming a lot, but most of the time I had to write documentation. Besides that, I felt that the developers around me (including myself) could work much more efficient if people from the higher management would hear our suggestions for improvements. However, this only happened occasionally. I was not satisfied how things were going; as a result I decided to follow a master's program to get to know more about IT project management. The master course I followed was Business Information Technology at the University of Twente.

During one of my courses, I got a brief introduction to Agile Software Development methods including Scrum: that was the moment when I felt that this might be what I was looking for. Unfortunately, the topic of most of the courses was not related to Scrum so I bought myself some books about it and read them in my spare time. From then on, I was convinced that agile methods would make programming software more efficient and whenever a course gave me the freedom to apply Scrum to other fields I did so.

When it was time to do my master thesis project, I knew that I had to do something Scrum related. I applied for master thesis projects at various companies, one of them being bor!sgloger consulting in Germany. Within a week I was asked for a job interview and directly after that interview I got an assignment – the result of it is this master thesis.

I am deeply indebted to various people. At first I would like to thank both Christiaan Katsma and Klaas Sikkel of course for being my supervisors as well as for telling me something about Agile Software Development methods during their courses – otherwise I could not have written this thesis. This whole master thesis project was quite enjoyable thanks to their great support, including suggestions and feedback on my research.

I also would like to thank Boris Gloger for offering me the possibility to write this master thesis as well as for the topic suggestion and his feedback. I am very grateful to all the people at bor!sgloger consulting, especially to my internal company mentor Bernd Krehoff, who always had a willing ear for all of my questions and issues. The same is true for Kristina Klessmann, who is also writing her master thesis at bor!sgloger consulting and therefore was a supportive co-worker to discuss master thesis related topics. I also want to say thanks to Olga Repnikova who inspired me to think about usage scenarios, one of the key aspects of my master thesis project.

Jan Segers

Table of contents

List of figures

List of tables

1 Introduction

This chapter defines and explains the master thesis project. First, background information is given to understand the context, followed by the actual motivation to do this master thesis project as well as its constraints. As a company hosts this master thesis project that company will be introduced as well. The second part, the problem statement, defines the resulting research goal as well as the research questions and the corresponding methodology to answer them.

1.1 Background

When doing a software project for a customer, usually the customer has some kind of idea he wants to get realized. He then has sales conversations with different companies and eventually chooses one that he thinks suits best for realizing his idea. After setting up the formalities, the customer will explain his idea to the company in detail to create the requirements, and after the company has the feeling it has enough knowledge to build the software (based on the customers idea) they will usually start right away. On day X, the day when the software has to be ready, the company will see to have finished the software and to deliver it to the customer. The customer receives the software and is pleased, as it matches exactly what he wants.

Unfortunately, this is only true in theory; in real life such outcomes for projects do not exist. Far too often the problem will be that the company was unable to finish the software in time, that it did involve more money than expected or, even more troublesome, that the software performs in a way the customer does not expect or isn't even sure to want at all.

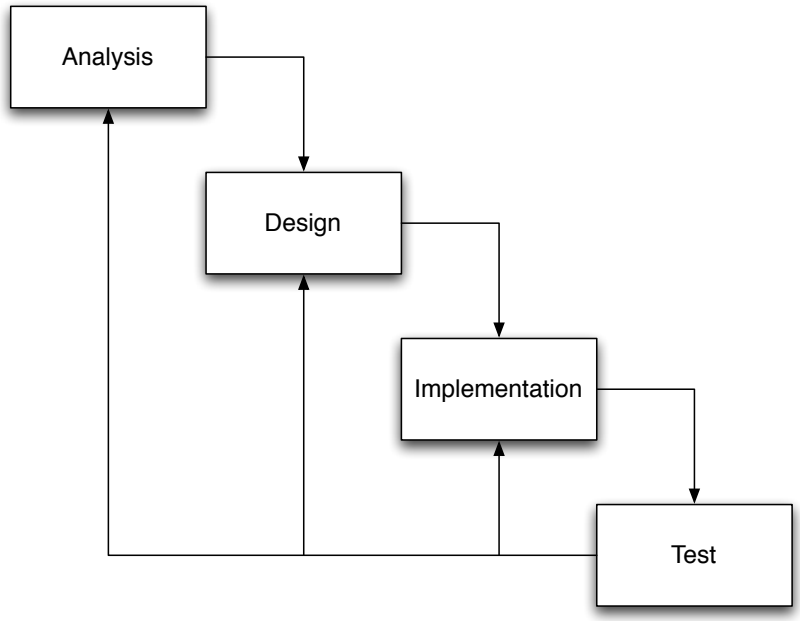

Figure 1: The waterfall model

In order to overcome such problems, certain project methods were introduced. One of the most famous (and still used nowadays) methods is the so-called Waterfall model (Royce, 1970). The major advantages in contrast to using no project method at all were that it encourages the developers to specify what the software is supposed to do before developing the system and that it divides development into several steps with intermediate deliverables that lead to the final piece of software. Still, the waterfall model is not a perfect model as it comes with certain disadvantages, namely assuming that all requirements are known and as soon as the design is created, requirements cannot be changed anymore (Kan, p. 9-13, 2002).

In the early 2000's the field of Software Engineering experienced a radical change when the so-called Agile Manifesto was published. This manifesto was a reaction based on emerging lightweight software methods that had their origins already in the mid 90's. Those methods promised to lift the heavy burden of the waterfall model from the software engineers by focusing on what was really important: making high quality software in the shortest amount of time possible, without projects that

were always overdue and without the need for extensive documentation. Agile software development has produced many different forms; forms sharing the same principles, but using different approaches. One of these methods is called Scrum. It is the most often used agile method (Azizyan, Magarian & Kajko-Mattson, 2011).

Scrum defines certain roles (e.g. customer, development team, etc.), events (e.g. a 1-4 week lasting development session called sprint, a daily meeting called daily scrum, etc.) and artifacts (a list containing all aspects the final product should have called product backlog, a list of current tasks called sprint backlog, etc.). Instead of involving the customer only during the analysis phase as is done within the waterfall model in order to map customer wishes to requirements and excluding him after that, with Scrum the customer visits the team at least once each sprint. Besides intensive customer involvement, Scrum also makes sure that after each sprint, a working software product is delivered so that the customer can test it and give direct feedback. Figure 2 visualizes the Scrum flow, which is explained in more detail in chapter 2.

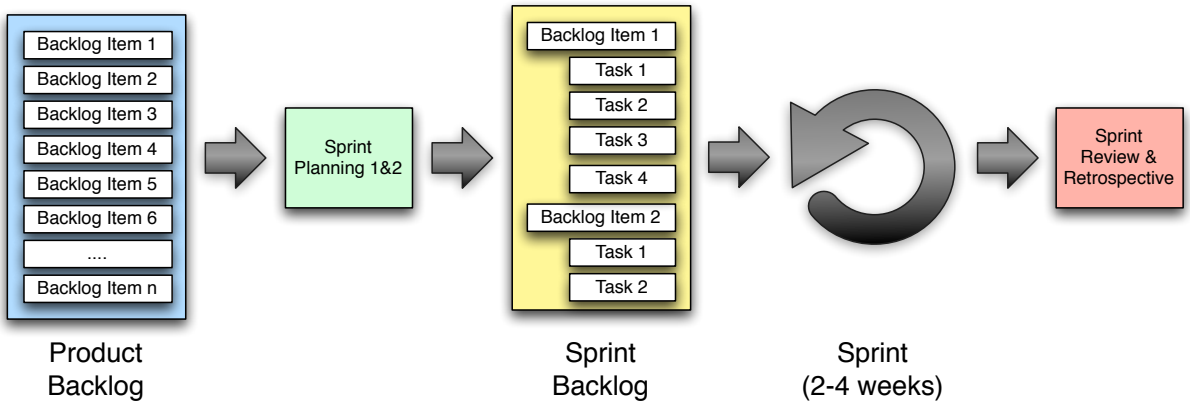

Figure 2: Scrum is an iterative model

In contrast to the waterfall model, Scrum does not divide the project into four subparts. Instead, the project is divided into so-called user stories. A user story is basically a small slice of the product (all user stories are stored in the product backlog) that is more tangible for the customer by providing a concrete example, e.g. how he will experience the component when it is implemented into the product. User stories in turn are divided into tasks, which are generally solvable by one person within a day. By doing so, every member of the team can reflect each day on the tasks he has to do, he is currently doing or he already has accomplished during the Daily Scrum standup meeting. The benefits of the user story - task idea is that the work that has to be performed is more concrete to the team and to the customer *and* that it is less complex at the same time.

Regarding this master thesis, one particular Scrum artifact is of importance, namely the sprint backlog. When Scrum was invented, the sprint backlog was visualized using a task board with Post-Its. The task board lists the user stories and their corresponding tasks (user stories split into smaller pieces), which are planned for the current sprint. It is one of the most important Scrum artifacts as the task board takes a central position. It is used to support the team to track its work, but it also makes the process of software development more accessible for everyone who is interested in it. As time goes by, software tools were invented that ought to replace the original method of visualizing the task board as the original method is limited when it comes to geographically dispersed teams: A paper-based task board is only available at one single location due to its nature. As a result, a paper-based task board is not suitable for teams working at different geographical locations.

However, this does not mean that a software-based task board outperforms a paper-based one on each level: next to the geographical aspect several other aspects exist for which a paper-based task board might still be the better choice. An example are the costs of a task board: the costs of acquiring a sheet of paper and a bunch of Post-Its cannot be compared to the costs of buying servers as well as a Scrum tool license. It is the purpose of this master thesis project to identify all aspects, to analyze and eventually to evaluate them so that both types of Scrum task boards, paper-based as well as software-based are understood. By doing so, one is able to understand why both types exist and what their advantages and disadvantages are.

To perform this analysis one has to evaluate a set of Scrum software tools that are capable to represent the task board as all tools have their advantages as well as their disadvantages. Thus, besides comparing the task board types it is also necessary to compare these Scrum software tools in order to find out to which extent they are able to represent the task board.

Companies that are using Scrum have difficulties when it comes to choosing a task board type, be it paper or be it software as both types differ on many levels. Therefore, the evaluation will also include findings that have been made during field research, e.g. by visiting companies that are using Scrum. Practical as well as theoretical input is of importance to create a complete and thorough evaluation.

Eventually, the results of these evaluations are used to support companies in making the right choice in terms of the type of a Scrum task board. The different areas that are covered by these evaluations will make sure that the outcome of this master thesis project will make it as easy as possible to support different types of companies.

1.2 Motivation

As stated before, software-based task boards are not superior to paper in every aspect – they both have their right to exist. To put it simply, most of the advantages of the one are the disadvantages of the other and vice versa. After following the hype of software-based task boards, companies became aware that not everything is as good as it seemed to be (Perry, 2008). As a result, companies struggle nowadays when they have to choose between the one and the other. Although the topic is known for years (Perry, 2008), countless expert discussions still take place whether software or paper is superior. The motivation behind this master thesis project is to advise companies in solving that question. Therefore, by providing certain guidelines, this study aims to help those companies in making a choice between paper and software and if decided for software, which type of Scrum tool is best.

1.3 Constraints

As described in section 1.1 Scrum features different types of roles, events and artifacts. Depending on the functionality of a Scrum software tool, most of the events and artifacts can be represented electronically. The focus of this master thesis is on one single Scrum artifact, the task board.

Second, although the Agile Manifesto (Agile Manifesto, 2001) is the basis for all lightweight software development methods, only Scrum will form the basis of this master thesis project. Giving an overview of all the available alternatives to Scrum within the agile context is therefore out of scope.

Although multiple companies are subject of the field research that will be conducted, it cannot be guaranteed that the scenarios experienced will match with the ones that every company has to face in making a choice regarding the nature of the task board artifact. As a result, it is not possible to state that the guidelines will be applicable for all existing software development companies as each company is more or less unique. Nevertheless, it is the purpose of this master thesis project to define guidelines that assist as many companies as possible.

1.4 Company

bor!sgloger consulting GmbH is a German consulting company that is specialized in offering trainings, consultancy and support for companies that want to implement Scrum. To do so, consultants visit the client (= the company) and continuously help him to implement Scrum by coaching and training the employees. Initially the Scrum consultants perform the role of the ScrumMaster within those companies; trained employees will replace them by and by. bor!sgloger consulting GmbH constantly aims to improve its service and to match their offerings with the requirements of their customers. One aspect that is often demanded by clients is the question regarding software tools that help to perform Scrum, e.g. tools that can represent electronic versions of Scrum artifacts and events. Therefore, the focus of this master thesis project will be on researching the possibilities of software tools that can represent the task board and comparing those tools with the original paper-based method to create guidelines for customers of bor!sgloger consulting GmbH.

1.5 Problem statement

Companies that have chosen an agile manner in order to develop software often face similar difficulties. When using Scrum, employees have to learn and understand a new method, the management has to accept a new way of thinking and all of a sudden, hierarchies are not so important anymore at all – only to name a few of the challenges. Scrum also introduces other challenges, which are invisible on first sight. One of these challenges is the choice between paper or software as a basis for communication and as a source for information. Often treated as a matter of personal taste, this could not be further away from reality. Choosing the former or the latter introduces some specific advantages as well as disadvantages. Very often, the advantages of the former are the disadvantages of the latter and the other way round. Companies need consultancy in order to understand, which of both methods are more suitable. Experiences of consultants and project results are often not documented; as a result, choices made are often based on limited, personal experience. Depending on the practical experience of such a consultant, guiding the client in making a choice can be a difficult thing to do, as consultants cannot rely on extensive case documentation. Still, making a choice has to be well thought over to avoid making critical errors – guidelines that help making that choice are thus of major importance.

1.6 Research goal

The research goal of this master thesis project is twofold. The first half of the research goal of this master thesis is to perform a thorough analysis of both task board types. The essence of this analysis is again twofold: first, paper-based and software-based task boards are evaluated in general e.g. what are the advantages and disadvantages of paper and software in terms of a task board. Secondly, a set of current Scrum software tools that are able to represent a software-based task board are evaluated.

Software tools have individual strengths and weaknesses; the same is true for companies: when comparing them, in some areas they are equal whereas in other areas they are completely different. Therefore, this evaluation is done in order to give an indication which of the current available software tools offers the best solution for each different type of company. See Figure 3 for a graphical representation of the research goal.

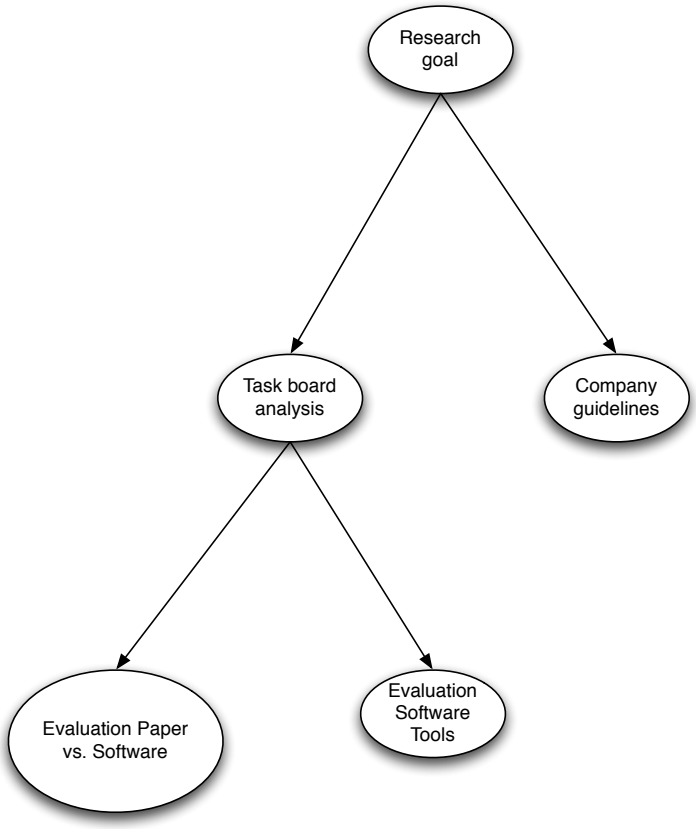

Figure 3: Research goal

However, the emphasis of the first half of the research goal is on the analysis of paper-based and software-based task boards. The reasoning behind this is that the author is well aware of the fact that the status quo of today's technology can be outperformed easily by the development of tomorrow's technology. Therefore it is important to perform this analysis so that one is able to understand the line of reasoning behind the general comparison of paper versus software. As a result, the evaluation of the Scrum software tools is merely a snapshot of what is possible today – no one can guarantee that they are valid for any fixed amount of time in the future.

The second half of the research goal is to create guidelines for companies that have to choose a Scrum task board variant. These guidelines have to be common enough so that they are useful for different types of companies. They will be based on derived company characteristics that in their turn are based on aspects e.g. how many developer teams does the company have, how many developers are within one team or whether these teams are geographically dispersed, only to name a few. Based on several research methods (see upcoming section 1.8), positive as well as negative experiences and aspects will be used to form these guidelines.

1.7 Research questions

In order to address the problem statement, the following research questions have to be answered to reach the research goal. A total of five research questions will be defined, which are used to get to the research goal step-by-step.

Research question 1

What is the role of a task board in Scrum?

This research question has four sub-questions:

1. *What is Scrum?*
2. *What is a task board?*
3. *What is a paper-based task board?*
4. *What is a software-based task board?*

Before the actual analysis can be performed, first the Scrum framework has to be explained and described so that the reader is able to understand the context of this master thesis project. Second, after explaining the Scrum framework, the task board Scrum artifact has to be explained in more detail regarding what it exactly is about, which purpose it serves and why it is so important.

Third, the task board variants, namely a paper-based and a software-based task board have to be explained. By answering this research question, it will become clear what is meant by a task board, how it looks like, what it provides and why so many different variants exist.

Research question 2

Which criteria can be used to compare a paper- and a software-based task board?

Now that research question 1 has set the prerequisites, the actual comparison is split in two research questions. Before the comparison can be performed, the purpose of research question 2 is to define criteria that are necessary to compare both variants.

First, the origin of these criteria will be described and second, each single criterion will be explained thoroughly so that it becomes clear to which extent each criterion is used to answer this research question.

Research question 3

What are the advantages and disadvantages of paper-based and software-based task boards?

As described previously, research question 3 uses the outcomes of research question 2 to reach one part of the research goal. After defining criteria to compare both task board variants, the purpose of this research question is to make the actual comparison. To do so, the criteria will be used to unveil the advantages and disadvantages to get a complete image. The analysis is not limited to functionality; also other domains (e.g. usability) have to be explored to obtain a holistic view.

Answering research question 2 and research question 3 will lead to reaching the evaluation of the software versus paper part of the research goal that has been described in section 1.6.

Research question 4

Which software-based task board tools exist and how well do they perform?

This research question has three sub-questions:

1. *Which criteria can be used to evaluate task board tools?*
2. *Does the evaluation back up the results of the previous analysis?*
3. *To which extent are the tools able to represent a Scrum task board?*

In section 1.6, the research question was split into two halves of which the first half was split again into two parts. The first half (task board analysis) of the research goal will be reached by giving an answer to research question 2, 3 and 4.

The purpose of this research question is threefold. First, criteria will be defined to perform the evaluation. This process is comparable to the one of research question 2. Second, the software tools

will be evaluated. The outcome of this evaluation will be used to research whether these findings back up the analysis of the paper versus software that has been done by answering research question 2 & 3. Finally, the tools will be compared to each other in order to determine the extend in which they are able to represent the task board.

Research question 5

Which Scrum task board-related company characteristics and guidelines can be defined?

This research question has four sub-questions:

1. *Which usage scenarios regarding the task board do currently exist?*
2. *Which task board type do companies use and how satisfied are they with them?*
3. *Which company characteristics can influence the choice of a task board type?*
4. *How can these company characteristics be used to create guidelines?*

This research question has to be answer in order to reach the second half of the research goal, namely to create guidelines that support companies in making a choice between both task board variants. As the purpose is to create guidelines, it is necessary to know for whom these guidelines are created. Therefore, company characteristics have to be determined that can be linked to certain outcomes of the previous research questions, which in its turn will result into guidelines that are general enough to be applicable by other companies.

The first sub-question aims to collect all current usage scenarios. Two of them are obvious: using a paper-based task board and using a software-based task board. Still, intermediary solutions exist, e.g. teams at different locations could make a photo of their task board every day and send it to the other team so that they can synchronize each others task board. Different companies use different scenarios, therefore it is necessary to summarize them with their positive and negative aspects.

After collecting all possible usage scenarios, the second sub-question is used to determine how satisfied or dissatisfied companies are with their current scenario. To do so, several companies will be visited to evaluate their types of task boards.

As soon as the task board usage scenarios and the outcomes of the companies that were visited are known, the third sub-question is used to evaluate whether certain characteristics exist that influence a company's task board choice. To do so, potential candidates for such characteristics are analyzed and evaluated whether they are applicable or not.

Finally, now that a set of applicable company characteristics has been determined, the fourth sub-question is used to combine these characteristics with the outcomes of the previous research questions to create guidelines for companies that have to choose a Scrum task board type.

1.7.1 Research question flow

When reading the research questions it becomes clear that they all are somehow connected with each other. The rationale behind this connectivity is the research goal. The research questions follow a hierarchical principle, e.g. research question 1 has to be answered before research question 2 can be answered and research question 2 has to be answered before research question 3 can be answered and so on. The research questions are thus related and this relationship is summarized in Figure 4:

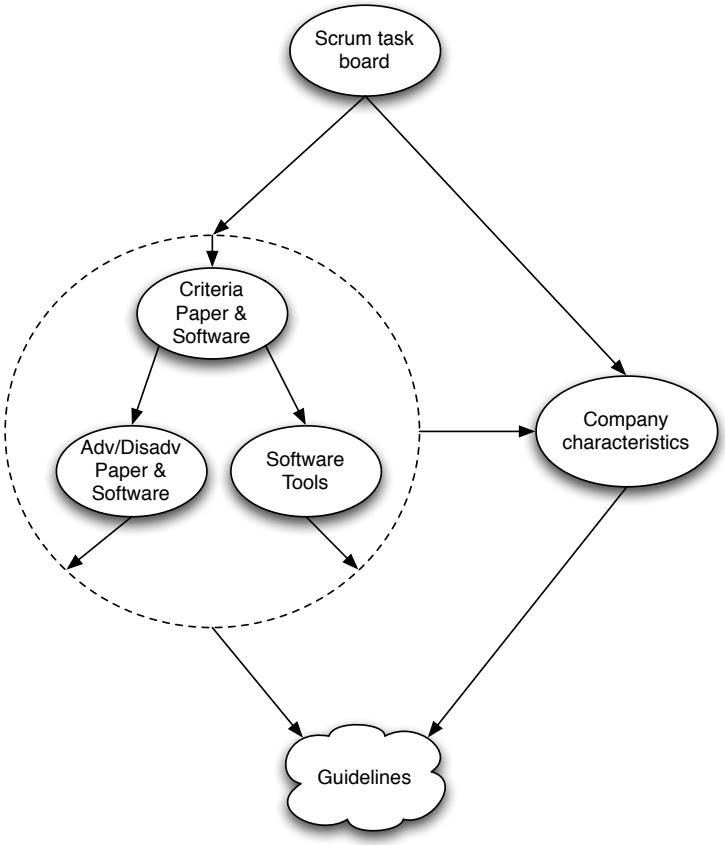

Figure 4: Research question flow

This figure can be read the following way: At first, Scrum and its task board have to be explored and described (research question one). Based on these prerequisites, criteria are defined (research question two) for comparing paper- and software-based task boards. Based on these criteria, the actual analysis can be performed by identifying the advantages and disadvantages of paper- and software-based task boards (research question three). These criteria also serve as input for evaluating Scrum software tools that are able to represent the task board (research question four). The outcomes of exploring the Scrum task board (research question one) as well as the outcomes of research question two, three and four are used to define company characteristics as well as the guidelines (research question five) that support companies in choosing the right task board variant.

1.8 Research methodology

To reach the research goal of this master thesis project, four different types of research methods will be used: a literature review, expert opinions, field research and software evaluation. The following four sections will explain the usage of these methods.

1.8.1 Literature Review

The literature review process was twofold. First, literature was searched to give an introduction to Scrum, e.g. what are its origins, why has it been developed, how does it work and what does it exactly mean – in order to create a common basis for the following sections. To do so, mainly Scrum-related books were taken into account. This literature review is used to write the Scrum-related background in chapter 2.

Second, another literature review was conducted to research what has been written about both task board variants. Other aspects also had to be covered: how do users feel themselves when they work with the task board? Do they like/dislike it? Why do they like/dislike it?

Different search terms were used in order to create an overarching view:

- Scrum taskboard
- Scrum task board
- Taskboard
- Task Board
- Scrum sprint backlog
- Sprint backlog

The yield was rather poor; only around 25 articles were found that were applicable for this master thesis project. Most results were received by using the plain “taskboard” / “task board” keyword, but many of these articles described totally different task boards that had nothing in common with Scrum.

The result was split among three different categories:

1. Experiences regarding the usage of a task board
2. Experiences regarding Scrum software tools
3. Agile Tool surveys

The first category is used for deriving criteria regarding the evaluation of software and paper in chapter 3; the second and third category are used to define criteria for evaluating the task board capabilities of Scrum-related software in chapter 4. The third category is also used as input for chapter 5: the company characteristics & guidelines.

However, the results are not used for the before-mentioned chapters exclusively; for example the first category is also used in other chapters besides chapter 3.

1.8.2 Expert Opinions

The second research method used in this master thesis project are expert opinions. There are two types of experts that were consulted: company-internal experts, thus employees of bor!sgloger consulting and company-external experts, who are employees of clients of bor!sgloger consulting.

Nine company-internal experts were consulted; they helped to identify the problem, they gave input for criteria that were found during the literature study and were used as source for additional criteria that were not mentioned in literature.

Also five company-external experts were consulted as these experts are using the task board in their daily working lives; they know therefore from experience what works well and what doesn’t. Therefore, they were a valuable source for giving practice-relevant information.

1.8.3 Field Research

The main purpose of doing field research was to get information for determining company characteristics, understanding task board usage scenarios, as well as creating guidelines.

Four companies that are using Scrum in combination with either a paper-based or a software-based task board (or even an intermediary solution) were visited. During a period of two to four days, ten Scrum teams were studied in terms of how they were using their task board, be it a paper-based task board or a software-based one.

Additionally, the teams were asked whether they were satisfied with their current task board or not and what they liked or disliked when using it. If available, also the person in charge regarding the choice of the Scrum tool was asked.

1.8.4 Software evaluation

Finally, several software-based task board tools that are currently available on the market were evaluated. In order to do so, the results of the paper versus software analysis in chapter 3 were used in combination with new criteria that were more software-related, like the user interface, performance, etc. The outcome provided an evaluation for each tool based on the defined criteria.

1.9 Putting it all together

The previous section has introduced different research methods that are used to answer the research questions, that are described in section 1.7. The purpose of this section is putting all those aspects together to create a complete view. The following table shows how these aspects are linked to each other:

Chapter	Research goal part	Research question	Research Method
1. Introduction	-	-	Literature Review
2. Scrum	Task board analysis (Evaluation paper versus software)	1. What is the role of a task board in Scrum?	Literature Review
3. Task board Evaluation	Task board analysis (Evaluation paper versus software)	2. Which criteria can be used to compare a paper- & software-based task board? 3. What are the advantages and disadvantages of paper-based and software-based task boards?	Literature Review, Expert Opinions
4. Task board Tools	Task board analysis (Evaluation software tools)	4. Which software-based task board tools do exist and how well do they perform?	Literature Review, Software Evaluation, Expert Opinions
5. Company Characteristics & Guidelines	Company guidelines	5. Which Scrum task board-related company characteristics and guidelines can be defined?	Literature Review, Expert Opinions, Field Research
6. Conclusions	-	-	-

Table 1: Putting the research aspects together

This master thesis has a total of six chapters and this table shows for each chapter the corresponding aspects that have been used.

Chapter 1 is the current chapter and introduces the master thesis project to the reader. It does not answer a research question, as this chapter is used in setting up those questions.

Chapter 2 is used to give an introduction to Scrum with its roles, artifacts and meetings. The last section of chapter 2 analyzes both task board variants. Research question-wise, the first research question is answered by explaining what Scrum is, how a task board looks like and which task board variants are available. This research question is answered by means of a literature review.

The third chapter sets up criteria for comparing both task board variants with each other, next to stating the advantages and disadvantages of both types. By doing so, the second and third research questions are answered by using the literature review as well as expert opinions as research methods. The first sub-part (evaluation paper versus software) of the task board analysis part of the research goal is reached by providing an answer to the first three research questions.

Chapter 4 zooms in on various Scrum software tools that are able to represent the task board artifact. This chapter is mainly based on software evaluation combined with the literature review and the expert opinions to define evaluation criteria. By doing so, chapter 4 provides an answer to research question four and so the second sub-part (evaluation software tools) of the task board analysis part of the research goal is reached.

Chapter 5 lists various usage scenarios, describes the companies that were visited and evaluates the company characteristics.

Finally, the guidelines are given by using the outcomes of chapter 2, 3, 4 & 5. The research methods used within this chapter are the literature review and the expert opinions next to the field research described previously to answer the last research question. As a result, the second half of the research goal (stating company guidelines) is reached.

Chapter 6 concludes the findings of this master thesis project. Since a conclusion does not introduce new information, neither a research question is answered nor a research method is used in this section.

1.10 Wrap Up

In this chapter, the master thesis project has been introduced to the reader. The background was given, followed by the motivation to do this master thesis. After that, the company that hosts this master thesis project has been described and the problem statement with the corresponding research goal has been given. The research goal was divided into two parts (task board analysis & company guidelines). The first part was divided again into two parts (evaluation paper versus software & evaluation software tools). To reach this research goal, five research questions have been described and an explanation of the research methods that are used has been given.

2 Scrum

This chapter will provide an answer to research question one by giving the reader a quick introduction to Scrum regarding its historical background, its roles, events and artifacts and how Scrum works. Finally, a more detailed description of the task board will be given. Readers who are already familiar with Scrum can skip this chapter with the exception of section 2.6, which describes both task board variants in a detailed manner.

2.1 Background

When researching the origins of Scrum, one becomes aware of the fact that the framework behind Scrum was developed way before this framework was actually called Scrum. Already back in 1986, Hirotaka Takeuchi & Ikujiro Nonaka (Hirotaka & Nonaka 1986) defined a new approach for developing products that has similarities with the characteristics of Scrum.

In the early 1990s, Ken Schwaber and Jeff Sutherland both developed simultaneously a software development approach that was practically the same although they were unaware of each other at first. When becoming aware of each other, they combined their ideas, resulting into a paper that was presented to the public at the OOPSLA (Object-Oriented Programming, Systems, Languages & Applications, a research conference) in 1995 (Sutherland & Schwaber 1995). After the first publication regarding Scrum during that conference, their collaboration continued over the years to collect best practices, findings and experiences.

All their efforts resulted into a series of publications in 2001 – one of them was the so-called Agile Manifesto (Agile Manifesto, 2001), which is a set of rules that were created to mark the begin of a new age in software development, namely the rise of the lightweight, agile software development frameworks. This Agile Manifesto was the result of a discussion of 17 software developers that had gathered at one place. It defines four principles:

Individuals and interactions over processes and tools

Working Software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

Those four principles were in contrast to what was common in these days. Instead of spending months to create an architecture that ought to cover each possible aspect, only a short amount of time was used to create an architecture that features the aspects that were needed when starting the project – new aspects that are needed later on are added dynamically, which results into an evolving architecture. Suddenly, the customer was embedded into the project instead of fighting with him regarding arbitrary contracting details. Thus, these principles introduced a fundamental change to the field of software development. Based on the Agile Manifesto, many other agile software development methodologies were created; one of the most popular ones next to Scrum is called eXtreme Programming (XP).

Another important publication in 2001 was the one of Ken Schwaber and Mike Beedle – "Agile Software Development with Scrum" (Schwaber & Beedle 2001) which is considered as the breakthrough of Scrum (Gloger, p. 20, 2011).

Ken Schwaber is a founder member of the Agile Alliance and founder of Scrum.org. Jeff Sutherland is the Chairman of the Scrum Foundation. Recently, Ken Schwaber and Jeff Sutherland have teamed up and published a new book called "Software in 30 Days" (Sutherland & Schwaber 2012) that summarizes the Scrum method and explains how to develop serious software in just 30 days. Nowadays, both are giving courses how to become a certified ScrumMaster (a Scrum role that will be explained in the following section).

2.2 Roles

The Scrum framework is build upon three pillars: roles, events and artifacts. Using those three pillars will enable one to understand how Scrum works (see 2.5). This section explains the different roles available in Scrum, namely the team, the ScrumMaster, the product owner, the customer, the management and the users. Note that within Scrum, roles are not positions and vice versa. A role is linked to a responsibility – which means that someone, who is taking a certain responsibility, can perform a certain role. As a result, Scrum ought to remove hierarchies, which is one of the reasons why it is quite hard to introduce Scrum in companies, especially if the companies are large. People are afraid of loosing their status - but in the end, status has no meaning at all when Scrum is introduced successfully. Coming back to the roles, one can differ between roles that belong to the project team (the team itself of course and the ScrumMaster), roles that affect the project team (the product owner) and more external roles (the customer, the management and the user).

2.2.1 Team

Within Scrum, one strives to create a cross-functional team, where cross-functional means that the team is built up on team members with different specialties. In the context of software development this means that for example testers form part of the team, which is often not the case, but also user interface designers. It is the team that has to deliver the actual project, e.g. the software. In Scrum, the focus is therefore on the team: they have to deliver the product, so they should be supported at its best. As a result, motivation and commitment play an important role. Nowadays, software developers often feel like their work is not valued, they don't have the feeling that their work contributes to a greater cause. Scrum counters this by providing a clear goal that is formulated by the product owner (see below). Connected to motivation is commitment. The team has to commit to deliver the product, but it will only do so if they get the feeling that the goals are within reach of what is possible. Getting commitments is not easy as single persons are easy to blame when something goes wrong. Within Scrum, the commitment holds true for the whole team. If the team fails to hold the commitment, each team member is accountable; single team members cannot be blamed. Last but not least it is important to let the team perform their work in their own way. This does not mean that the team can do whatever they want to, but within defined boundaries they are free to choose how to work as long as they can hold their commitment. The team therefore organizes itself, every member may choose a piece of the product he or she likes to work on, there is no one who assigns certain tasks to certain team members. The ScrumMaster and the product owner (see the next two sections for both roles) support the team to make sure that they can reach their goals.

2.2.2 ScrumMaster

The ScrumMaster is a dedicated Scrum role. His job is to make sure that the team has everything it needs to do its work and to absorb any external issues that might affect them. For example, if a team member lacks a certain piece of hardware, it is the job of the ScrumMaster to make sure that the team member receives that piece of hardware as soon as possible. The job of the ScrumMaster can be partly compared to the one of a bodyguard: he tries to make sure that no one harms the team, in other words that the productivity remains stable. As an example: if a (project) manager wants one of the team members to work for another team, it is the job of the ScrumMaster to tell the manager that this is not possible. The job of a good ScrumMaster comes thus at a risk; he is always in the line of fire. The ScrumMaster also ensures that the communication between the team and other external parties, like the product owner (see below) is as good as possible. Any surfacing issues are called impediments. The ScrumMaster collects those impediments and tries to solve them one by one - the team can help him by stating which impediments are more time critical than others. To resolve the impediments, a ScrumMaster needs authority. Management has to understand the role of a ScrumMaster in order to provide the ScrumMaster with the authority he or she needs. The job of a ScrumMaster is often underestimated; companies tend to see this role merely as an addition to an existing position. This is quite harmful, since the role of a ScrumMaster is a fulltime job. Also, if he or she performs another role next to being a ScrumMaster, decisions will likely to be biased.

2.2.3 Product owner

The product owner does not directly belong to the project team like the ScrumMaster. Instead, the project team forms together with the ScrumMaster and the product owner the so-called Scrum team (Gloger, p. 108-109, 2011). The Scrum team can be seen as a mini start-up that is hosted under the roof of the main company. As a result, the product owner can be seen as an executive who has an idea that the team has to build and which he eventually wants to sell. As an example take Google. Google has a big product portfolio; think of the search engine, Gmail, Google Earth or the social network Google+. Each of these products can be seen as mini companies. The existence of such a mini company is dynamic – as long as the product generates profit it continues to evolve, but can be stopped at any day, as happened with Google Wave, for example. Therefore, the product owner works as an intermediary between the company and the project team. He creates projects that generate value for the company. At the beginning of such a project he creates what is called a vision. Such a vision describes the project and has to motivate the project team at the same time. Thus, the team has to be able to feel the vision, it has to identify itself with developing the vision and the team should get the feeling that it is developing something meaningful, something important. Writing a good vision is a crucial skill of the product owner. When the vision is clear to everybody, the product owner starts to make it more tangible by writing so-called user stories (see section 2.3.5). By doing so, the product owner makes sure that the team is able to transform his vision into a product – a product that generates a positive return of investment (ROI).

2.2.4 Customer

The customer is the person that funds the project and this is one of the key differences to the role of the product owner. In essence, the customer wants to get a certain problem solved and agrees to trade his money for a solution to his problem. In contrast to the product owner, the customer often does not really know what he wants. Although this sounds contradicting, it explains the role of the product owner, being an intermediary between business (the client who offers money) and the project teams (who can create a solution). One can distinguish two types of customers, internal customers and external customers. An internal customer is for example the navigation systems department of a car manufacturer that needs certain circuit boards from the electronics department. Therefore, the navigation systems department is the client who needs a solution to its problem, thus getting a circuit board that fulfills some requirements. Besides, also external customers exist; a matching example would be a shop owner who asks a software company to create a web-shop system according to his or her needs.

2.2.5 User

The user is the one who will work with the product in the end. To continue the example of the shop owner who wants to have a web-shop, it is unlikely that this shop owner will input all the articles he or she wants to sell into the web shop system. Most likely he or she will ask an employee to enter the details of the articles into the system. Using this example it becomes clear why the customer is not necessary the same person as the user. As the user is the main target of the product the team is developing, it is one of the principles of Scrum to embed the user into the project team, matching the approach of cross-functional teams as described above. However, this is often not possible. Still, the team has to get frequent feedback from the user as his or her point of view is of huge importance. One has to keep in mind that the user is the one who is using the product in the end.

2.2.6 Management

Managers are important people. Also within the Scrum environment, this remains true. One of their main tasks is to make sure that the right facilities are available and that employees have to be well trained to be able to deliver high quality work. In this context, facilities means that the team has a room where they can work and that this room is equipped with everything they need. Management has to support the ScrumMaster in solving impediments next to making sure that the right people are hired.

2.3 Artifacts

The second pillar of Scrum is about the artifacts. Artifacts are a medium of information; some of them are called information radiators, where others are more tangible and thus are called physical information radiators (Cockburn, p. 76-79, 2001). The following sections will explain the standard artifacts.

2.3.1 Vision

Briefly touched in the previous section, the vision is the starting point of all Scrum projects. The product owner forms the vision and its main purpose is to tell the team what has to be done in such a way that it is motivated. The team has to feel like it is on an important mission, a mission that is crucial for the company's welfare. As a result, it is very important that the product owner is capable in creating good visions, since a vision's quality decides whether the project might become successful or not. A good vision is short, but concise; some sentences are already sufficient. A vision can also be spread by using a presentation; the important aspect is that the team is aware of what has to be done.

2.3.2 Product Backlog

To create the product that is described by the vision, it is the job of the product owner to create slices of work. Such a piece of work is called a user story (see for explanation below). Different user stories represent different functionalities of the product. Next to creating those user stories to fill the product backlog (by product backlog a list of features is meant) the product owner has to prioritize them according to their business value. For example, when creating a web shop, it is much more valuable for the owner that the users can pay the products they buy instead of being able to print a certain product description. Therefore, the more important user stories are on top of the product backlog.

2.3.3 Sprint Backlog

Scrum is an iterative development approach. As a result the team has to work on different user stories during each iteration. In Scrum, such an iteration is called "sprint". For each sprint, the product owner chooses a certain amount of users stories that are on the product backlog. These chosen user stories are put in the sprint backlog. Thus, a sprint backlog is a selection of user stories the team has to complete within the current sprint.

As stated in chapter 1, the sprint backlog is the Scrum artifact, which is the subject of this master thesis project – it is visualized using a task board. The purpose of section 2.6 is to get into detail regarding the task board.

2.3.4 Impediment Backlog

The impediment backlog is mainly used by the ScrumMaster to list all the impediments that influence the performance of the team. Again, this list is prioritized according to the needs of the team, e.g. the team tells the ScrumMaster which impediments are more critical than others. It is the main job of the ScrumMaster to make sure that all impediments are solved as soon as possible.

2.3.5 User Story

A user story is another representation of a product feature. In essence, a user story is one sentence that is based on a predefined structure:

*As a user **role,** I need a **functionality** so that I get **business value*** (Cohn, p. 25-26, 2005)

e.g.:

*As a **shop owner**, I need a **payment option** so that I can get **the money of my clients**.*

By using this structure, the product owner tells the team a story, a story that helps the team to understand what this functionality exactly is about. Thus, a user story puts a product's functionality into a real life context. The amount of work to implement such a user story is measured in story points, which is a Fibonacci-like scale ranging from 0 to 100 (Cohn, p. 52-53, 2005). The higher the amount of story points, the lesser is known about how to fulfill the story (Gloger, p. 141, 2011). For example,

for a software developer it is easier to write a "Hello World" application than to write a complex algorithm. Therefore, a "Hello world" application would have less story points than a complex algorithm, simply due to the fact that at the moment the story is written, the team knows better how to complete the former than the latter.

2.3.6 Task

Although a story can be classified by using story points, it is usually too big to complete without further splitting – work has to be refined (Parnin, Görg & Rugaber 2009). Therefore, a user story has to be split into several tasks, where the idea of a task is that it can be solved by one software developer within one day. Tasks help the team to grasp the aspects behind a story. Picking up the web shop example from the previous section some example tasks could be:

- Identify different types of payment
- Link a credit card to an user account
- Charge the credit card as soon as the payment process is complete
- ...

A finite amount of tasks belong to one user story. If all tasks belonging to a user story have been completed, the story is completed, too.

2.3.7 Burn Down Chart

The burn down chart is the main measurement metric that is used within Scrum. It is two dimensional, showing the amount of story points left of the current sprint on the y-Axis and the amount of days of the current sprint on the x-Axis. After each day, a line is drawn horizontally. As soon as a story gets completed, a line is drawn vertically – this line is as long as the amount of story points that have been completed. Ideally, that line reaches zero at the end of the sprint, indicating that all user stories have been completed within the estimated time.

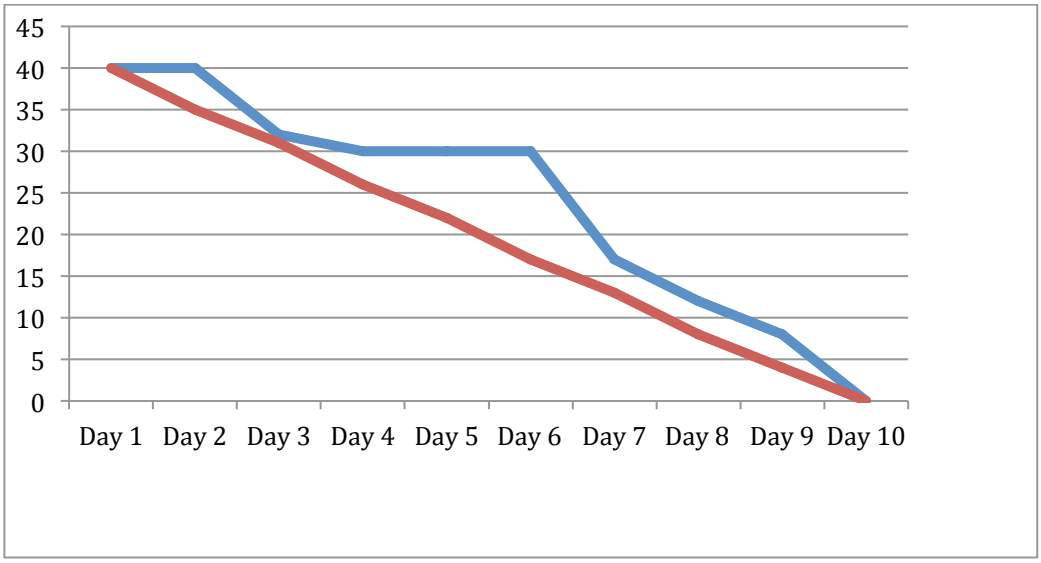

Figure 5: Example of a burn down chart

Figure 5 shows an example of a burn down chart for a typical sprint, where the red line is how the burn down chart should look like in an ideal case and where the blue line is an example of a more realistic case.

The burn down chart can be used to determine the speed of a team (called velocity in Scrum), e.g. how many story points it can fulfill within one iteration. By using this outcome, the product owner can get a better estimation in terms of how much the team is capable to do for the next sprint.

2.4 Events

Events are the last pillar that have to be explained, before Section 2.5 will show how all three pillars are combined to explain the Scrum flow, e.g. how it works.

2.4.1 Sprint

Briefly touched in the previous section, a sprint is the Scrum terminology for an iteration. Commonly, a sprint lasts two weeks, but three or even four weeks are also possible. Before actually starting to work on the stories, the team has to do some planning that is done in Sprint Planning Meeting 1 & 2, see section 2.4.3. On each day, the team has to perform a little standup meeting that is called Daily Scrum (section 2.4.4) and at the end of each sprint, a sprint review (section 2.4.5) and finally a retrospective (section 2.4.6) has to be done. During the sprint, an estimation meeting (section 2.4.2) should be performed at least once, better twice.

2.4.2 Estimation Meeting

The idea of the estimation meeting is that the team gives the product owner some feedback regarding the amount of story points each story should have. The product owner collects that information and updates the product backlog accordingly. Different techniques (Grenning 2002, Gloger, p. 143-147, 2011) exist that support the team in estimating the stories. After having all stories estimated, the product owner has a perfect estimation, as only the team is able to tell him what they are capable of. By using that information in combination with the teams velocity (see burn down chart, section 2.3.7), the product owner knows how many finished stories he is most likely to receive at the end of a sprint.

2.4.3 Sprint Planning Meeting 1&2

Sprint planning meeting 1 and 2 are used to plan the work for the next sprint. For sprint planning 1, the team meets the product owner, who in turn presents the stories he would like to see finished after the end of the sprint. The team asks questions about everything that is unclear to them. At the end of this meeting, requirements, user acceptance criteria and tests are defined for each user story. Finally, the team has to commit themselves to the amount of stories they think they are able to finish within the sprint. The product owner asks whether the team wants to commit themselves to the first story, to the second story, to the third story and so on, until one team member cannot give his commitment anymore. The team has then created a deal with the product owner; it is their task to finish the user stories they committed to in time.

Sprint planning meeting 2 takes usually place without the product owner. The team comes together and discusses the user stories. For each user story, tasks will be created that help the team to make the user story more tangible and less complex. Next to creating tasks, architectural decisions will be made, classes will be designed and database tables discussed. At the end of this meeting, every single team member knows exactly how all user stories can be solved.

2.4.4 Daily Scrum

The daily scrum is a daily stand up session that lasts a maximum of 15 minutes. During this session, every team member will state what he or she has done, on what he or she is working on and what he or she is planning to do. Furthermore, if a team member struggles with a task, the daily scrum is the moment to get help from other team members.

2.4.5 Sprint Review

At the end of each sprint, a sprint review takes place. The team meets up with the product owner and, if possible, with the customer to present the finished user stories. It is important to note that by presentation is meant that the team will demonstrate the user stories on their computers. It is quite common that each team member is able to show some of the finished functionalities; therefore, the product owner and the customer go from machine to machine to get an idea about what the team has accomplished. It is very important that only fully tested, thus working software is presented. Software that hasn't been tested or only works in a special environment cannot be demonstrated during the sprint review.

2.4.6 Retrospective

After having demonstrated the outcomes of the sprint, the team meets to do a retrospective, where each team member is asked to reflect on what has happened. The retrospective is divided into three parts:

1. Each team member tells the rest of the team which important events (for him or her) have occurred. By doing so, the team gets a different grasp on the whole situation; it is now able to understand the feelings and actions of each other team member.
2. After that, each team member has to state what went well during the sprint. The data is usually collected on Post-Its that are pinned on a board so that the whole team knows what they have accomplished.
3. Finally, the team is asked about what could be improved. This is the moment where the ScrumMaster can update his impediment backlog. Impediments have to be discussed whether they can be solved by the team itself or whether the ScrumMaster has to solve them.

By performing a retrospective, the team gets to know more about each other, and the ScrumMaster is able to uncover certain problems that have to be solved.

2.5 How Scrum is put together

With the help of the three pillars of Scrum described above, one is now able to comprehend the way of working when using Scrum.

At first, a vision is created and shared among the team that is now able to grasp the idea of the product they have to build. The product owner creates a product backlog with the corresponding user stories.

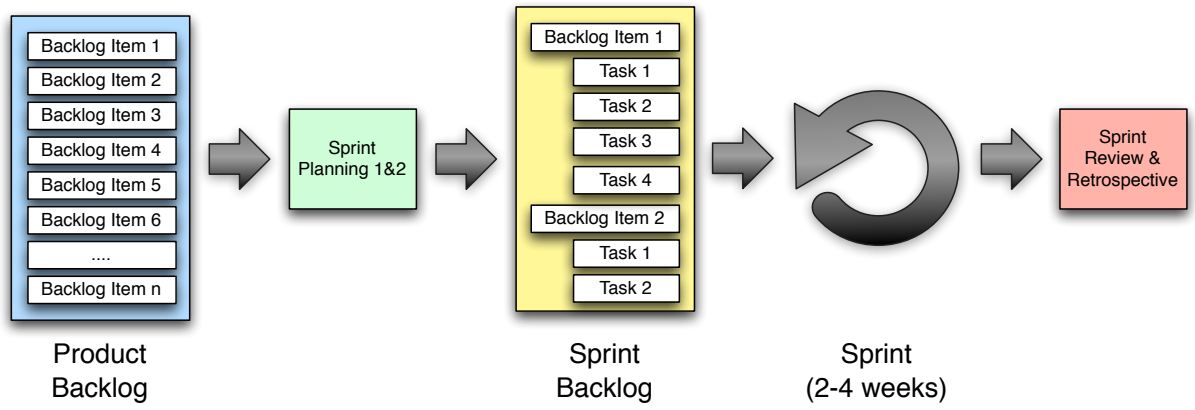

Figure 6: The Scrum flow

When starting the first sprint, the product owner presents some stories and the team commits to as much stories as it thinks it is capable of getting done. Those stories are then put onto the sprint backlog, which represents all the stories of the current sprint. In the sprint planning meeting 1 & 2, the team discusses the stories and creates tasks that are also added to the sprint backlog. After that, the actual development of the product starts.

The team members take the tasks they like to work on and at the beginning of each day, they perform the daily scrum standup meeting to discuss each others progress. While developing the product, the team as well as the product owner might get new ideas, which are formulated as stories and put into the product backlog for upcoming sprints.

After 2-4 weeks (the length of a sprint is a matter of choice), the sprint is at its end and the delivered user stories are demonstrated during the sprint review. As only fully working software is demonstrated, the customer gets a working increment of his or her software at the end of each sprint. Of course, the software has not all the functionalities that are required by the customer, but the functionalities that are delivered are fully working and can thus be used in a productive environment.

As soon as the sprint review is finished, the team gets together to perform the retrospective, where they reflect on what has happened during the last sprint, what went well and what could be improved.

This is also the very moment where the ScrumMaster will collect impediments that he has to solve as soon as possible. Those impediments are listed and prioritized in the impediment backlog.

Once, at least twice each sprint, the team gets together with the product owner to estimate upcoming stories. After each sprint, the team's velocity is measured using the burn down chart which serves as input for the product owner – he gets a better idea of what the team is capable of and updates the amount of stories for the next sprint accordingly.

2.6 Task board Types

The main research goal is to perform a thorough analysis of paper-based and software-based task boards. In section 2.3.3 the sprint backlog was already roughly introduced. This section aims to give a more detailed explanation.

A task board is a tabular representation of user stories with their corresponding tasks of the current sprint. Scrum defines four columns (Wirdemann, p. 144, 2011):

1. "Stories", which lists one user story per row.
2. "To Do", which lists all tasks belonging to a particular user story that have not been worked on yet.
3. "In Progress", which lists all tasks belonging to a particular user story that are currently processed.
4. "Done", which lists all tasks belonging to a particular user story that are finished.

Each row represents one user story and an arbitrary number of corresponding tasks. A simple graphical representation of a task board is shown in Figure 7.

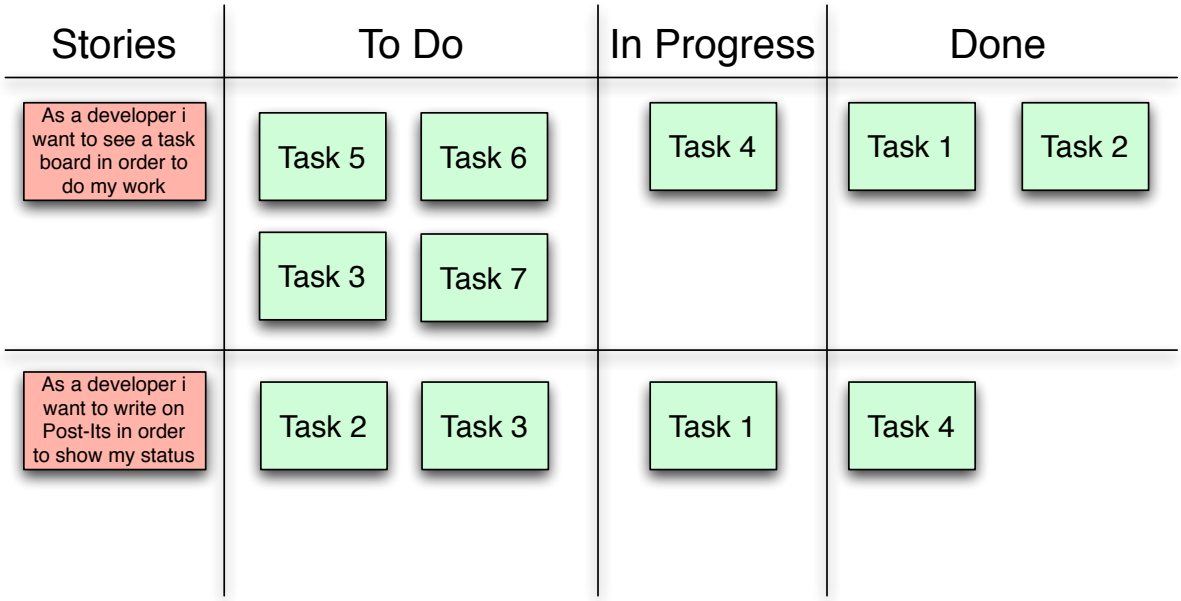

Figure 7: Simple representation of a task board

Based on its structure, a more common name for the sprint backlog is the Scrum task board. User stories are sorted based on their priority; that means that in the example of Figure 7 the first story has a higher priority than the second story. Under ideal circumstances, the team has to complete the first story before beginning to work on the second one. However, in a real world scenario, this is difficult. In the case of the first story, seven tasks are listed, one of them is currently in progress and two are already done. Imagine a team with more than seven developers: at least one of them would not be able to work, as the remaining developers did not finish their tasks yet. As a result, the developers that have already finished their tasks will start to work on tasks of the second story and so on.

By using a task board, the status of the sprint can be tracked. Team members are able to decide autonomously which tasks they want to do – no one assigns a task to a particular person. As soon as they decided to work on a task, they move it to the "In Progress" column. As soon as the task is

finished, they move the task again, this time to the “Done” column. When all tasks are done, the user story should also be finished. If this is not the case, the team forgot to add some tasks. Finished stories should be ready for demonstration and fully functional. The product owner could come by, review the user story and should be satisfied with the work of the team.

Currently, two types of medium are available to represent a task board. When Scrum was invented, the medium of choice was paper. However, as time goes by the paper medium was digitized due to various reasons that will be discussed. As a result, software is also a medium of displaying a task board, though an electronic one.

2.6.1 Paper

The paper variant of the task board looks quite similar to the representation given by Figure 7 (Perry 2008, Gloger, p. 167, 2011). The board itself is often created by using a large, plain sheet of paper. The stories and tasks are attached to the board by using Post-Its, see Figure 8.

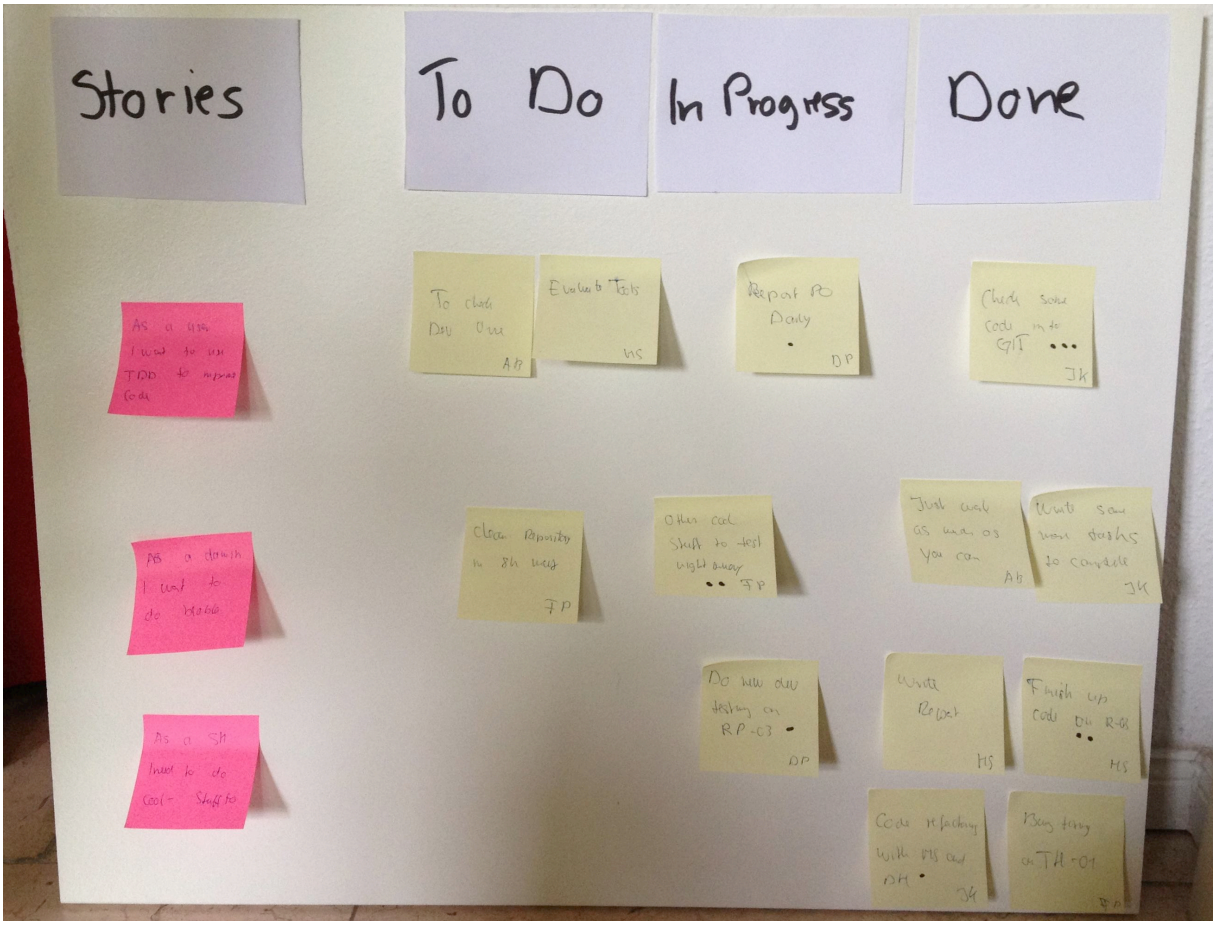

Figure 8: A paper-based task board

By using an ordinary sheet of paper, one is very flexible in crafting a task board according to his or her needs. Paper can be simply extended by using sticky tape. As a result, to create a task board with several meter length or height is no problem at all.

Instead of using sheets of paper that are fixed on a wall, teams often tend to use whiteboards or they put sticky tape directly onto the wall in order to create the necessary rows and columns. Usually, the Scrum task board can be found in the team’s room, so that every team member can see the current status quo. On each day the daily scrum standup meeting takes place and the team gathers around the task board to explain the current status of his work by using the Post-It tasks, e.g. what everyone has done, what everyone is working on and what everyone plans to do. When a team member wants to assign a task to himself, he simply puts his name or his initials on the task so that everyone is able to see who is doing which task. Moving the tasks is equivalent to starting or finishing the piece of work. Such paper-based task boards are still preferred nowadays (Gloger, p. 314, 2011). However, paper-

based task boards also have their disadvantages and these disadvantages have resulted into a second medium that is available to visualize the task board: software.

2.6.2 Software

If one wants to run a successful company nowadays, one has to get used to the idea of globalization, due to various reasons – one being able to get cheaper employees. Also in the field of software engineering, globalization is well known. Therefore, Global Software Development (GSD) is a much discussed subject, as it claims to reduce time to market, an increase in quality and productivity as well as saving costs (Jimenez, Piattini, & Vizcaino 2009). This topic has also affected Scrum, although spreadsheet tools were already used when Scrum was in its early phases (Perry, 2008). Suddenly, teams have to be distributed among different locations. As a result, only a part of the team is able to have a look at the board, can create new tasks or move tasks. This has led to the creation of software tools that emulate the paper-based Scrum task board on the screen. Thus, instead of moving tasks by hand, one moves tasks by using the mouse or the keyboard.

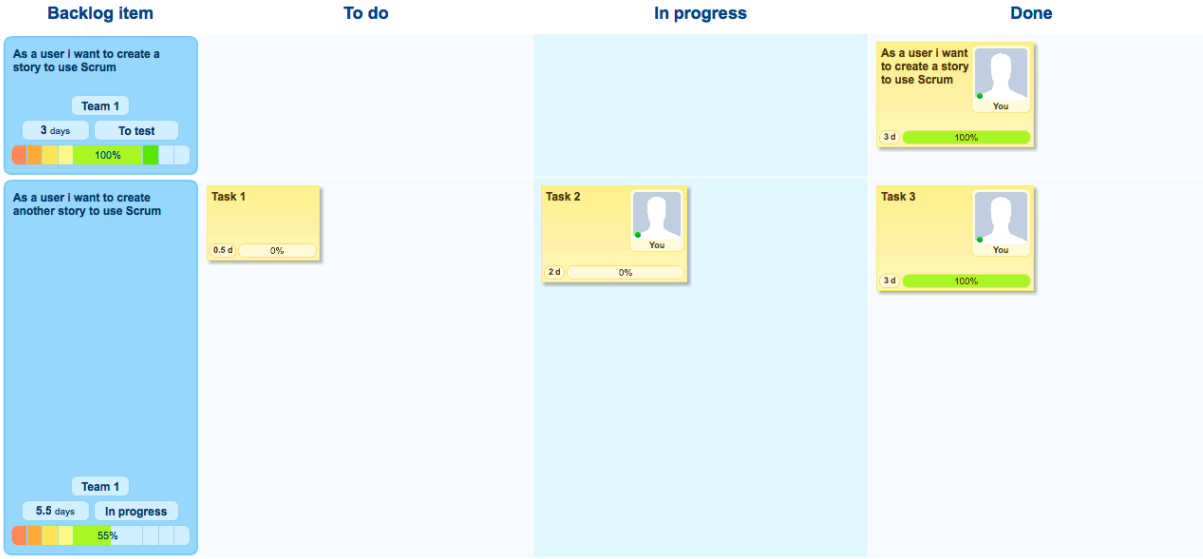

Figure 9: A software-based task board

Recently, most of the tools are browser-based – i.e. a server hosts the Scrum task board and every team member can access it regardless of his or her location. Scrum tools go beyond of just displaying the task board: they also have integrated the product backlog; they can display all kinds of graphs (like the burn down chart) and reports. However, examining the functionalities of the Scrum tools beyond their task board capabilities is out of scope for this master thesis project.

2.7 Wrap Up

In this chapter, the first research question has been answered by giving an introduction to Scrum and by showing how it works, by explaining what a task board is and by discussing the paper-based and software-based task board variants in section 2.6.

The reader is now able to grasp the key differences between a paper-based and a software-based solution. In combination with chapter 1, the reader has now a sufficient background to understand the upcoming chapters that will follow.

3 Task board Evaluation

Now that one knows how Scrum works, what a task board is and what it looks like, the actual analysis is performed. Section 2.6 has explained both task board types, which form the basis for this chapter and already briefly touched the differences. Within this chapter, those differences will be further analyzed.

The purpose of this chapter is to give an answer to research question two and three by providing a thorough evaluation of the task board.

First, section 3.1 is used to answer research question two by identifying criteria that will be used to evaluate the paper- & software-based task board.

Second, research question three will be answered by performing the actual comparison of both task board types. This will be done in section 3.2.

Finally, an overview regarding the analysis of both task board variants is given in section 3.3.

3.1 Evaluation Criteria

To perform the actual evaluation of paper- and software-based task boards, certain criteria have to be defined. The sources for these criteria are the following:

- Literature (see the following sections)
- Three external company experts.
- Seven internal company experts.

Criterion	Literature Review	Internal Expert Opinion	External Expert Opinion
Accessibility	✔	✔	✔
Flexibility			✔
Motivation	✔	✔	✔
Haptic Quality	✔	✔	
Integration			✔
Costs	✔	✔	✔
Availability		✔	✔
Archiving			✔
Overhead			✔
Distance	✔	✔	✔
Communication	✔	✔	

Table 2: Criteria found in stages

Table 2 shows the criteria found within the first column and the research methods that back these criteria within the remaining columns. The table underlines the rather poor literature review results that were described in section 1.8.1: only about ten papers were found that contributed to these criteria. Most of the criteria were found by consulting internal as well as external experts. Interestingly, only three out of eleven criteria were found during both research methods. The difference between both expert groups underlines the different views of those experts, whereas the internal experts are consultants of bor!sgloger and the external experts are employees of companies that were subject of this master thesis project.

The following sections will describe the criteria that will be applied when comparing a paper-based task board with its software-based pendant. Each criterion will be discussed to create a common understanding.

3.1.1 Accessibility

In German, accessibility is translated to "Flurtransparenz" in the context of Scrum – which roughly means "floor transparency". The main aspect of this criterion is the fact that the development process should be made visible and open - not only for the team, but also for everyone who visits the team. This criterion is quite crucial, as transparency is very important for the team's mindset (Perry 2008). If the task board is not available for visitors, a manager for example is not able to get an overview of the team status. It is very important to note that the task board should not be seen as a controlling item, but more like an information radiator (Cockburn, p. 76, 2001) that immediately shows if something within the team is wrong (Gloger, p. 174, 2011).

Next to the physical presence of a task board it is also important that the information it contains is readable. Although readability is not directly important to a visitor who is only interested in the team's progress, it is necessary for the team itself. If the text is not readable, then people will not use the task board.

3.1.2 Flexibility

The aim of the flexibility criterion is to investigate to which extent a task board can be shaped according to the needs of the team. As described in the previous chapter, a standard task board has a column for the stories and three status columns for the tasks, e.g. "To Do", "In Progress" & "Done". Yet, it is possible that the team might have a need for an extra column that helps it to do its work better.

3.1.3 Motivation

Motivation is another important criterion. Scrum tries to make the whole process of creating software more joyful by putting the focus on the team (Gloger, p. 57-61, 2011). As a result, it is the team who has to adapt Scrum (next to others of course, like the ScrumMaster or the Product Owner). Acceptance of the task board type is thus essential, as the team works with it all the time. The task board should encourage the team to do their work in order to increase the team's progress. However, choosing a task board type is not a team decision. Instead, the choice for a particular task board type depends often on global company principles.

3.1.4 Haptic Quality

Linked to the motivation criterion, it has to be evaluated whether the task board is able to give the user a haptic response. Literature has shown that achievements, no matter how small, can stimulate us to enjoy doing our work (Amabile & Kramer, p. 76-84, 2011). A practical example is a To Do list: everyone likes to cross an item out as soon as it is finished. It has thus to be researched if both types of task board can provide this feeling of joy.

3.1.5 Integration

Companies that are using Scrum are mostly companies that develop software. For such companies, Scrum is only a part in their whole workflow. For example, software-engineering companies have bug-tracking systems, source code check-in tools or automatic build-systems that are able to build the current version of the software over night. These systems are connected with each other and form the workflow of a company. As a matter of fact, embedding Scrum tools into such a company workflow is of high interest. The evaluation of both task board types has to investigate which level of integration if possible.

3.1.6 Costs

Costs are always a point of discussion (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009). Therefore, the task board types have to be evaluated

regarding their needs of resources. If one task board type needs more expensive resources than the other, then this might influence the decision for choosing one.

3.1.7 Availability

As stated in the "Motivation" criterion section, the team relies on the task board, resulting into the fact that the task board should be always available. If it is not available, no matter due to which circumstances, the team is not able to make progress on finishing their tasks. As one might imagine, the availability of software and paper is of course different.

3.1.8 Archiving

This criterion is used to research to which extent the task board types are able to preserve their data. Sometimes, it is necessary to be able to dig into the past in order to check back on something, e.g. the flow of a particular story or task.

3.1.9 Overhead

Benjamin Franklin has already stated back in 1746 that "time is money". Therefore it is important that using the task board - no matter which type - should be easy. The amount of time needed for training and for using the task board has to be as low as possible to avoid unnecessary overhead.

3.1.10 Distance

As explained in the previous chapter, two types of teams currently exist: the collocated ones, which means that the whole team is at one location and the distributed ones, where the team is divided over multiple, geographically dispersed locations (Herbsleb & Moitra, 2001). Those types of teams have different requirements regarding the capability of their task board. The evaluation needs to identify those requirements and compare them with the capability of both task boards.

3.1.11 Communication

Linked to the "Accessibility" criterion as well to the "Availability" criterion, communication within the team is essential. Usually, team members will have spots where they meet and where they will discuss whatever they want. An example of such a spot might be the coffee-kitchen. Also the task board is such a spot, where team members are likely to get into touch with their colleagues while they update tasks according to their status. It is therefore important that such spots exist (Cockburn, p.76-79, 2001); otherwise the communication and discussion within the team will decrease and might lead to the isolation of team members.

3.2 Paper vs. Software Evaluation

The previous part of this chapter has given an answer to research question 2 by defining criteria for the analysis.

This section and the upcoming section 3.3 are used to give an answer to research question 3 by performing the actual evaluation of paper-based and software-based task boards and concluding the findings. For each criterion that has been found in section 3.1, the situation is analyzed for both task board variants. However, each criterion has its own set of properties that are used to evaluate task board variants. The properties were defined with the help of both external as well as internal company experts. Since most of those properties are not the same, the following table lists the properties for each criterion.

Criterion	Property
Accessibility	• Size (small / large) • Readability (bad / good)
Flexibility	• Degree of customization (low / high)
Motivation	• Amount of feedback (low / high)

Criterion	Property
Haptic Quality	• Tangible reaction (no / yes)
Integration	• Programming interfaces (no / yes) • Degree of automation (low / high)
Costs	• Expenses (cheap / expensive)
Availability	• Uptime (low / high) • Dependency on others (no / yes)
Archiving	• Amount of time (low / high)
Overhead	• Amount of time (low / high)
Distance	• Applicability (low / high)
Communication	• Collaboration with others (low / high)

Table 3: Criterion properties

During the next sections, the properties will form part of the analysis of each single criterion.

3.2.1 Accessibility

A task board is of major importance to the team, as it is an artifact the team is interacting with all the time. So, next to having a task board, one has to make sure that it can be seen by anyone who wants to see it, be it the team that works with the task board or be it a casual person walking by that is curious about the teams' status (Perry, 2008). To enable this visibility, the task board needs to exist within the team room and it has to be big enough so that people can actually work with it and read its content also from a distance. The task board is, like the coffee-kitchen, a central meeting point for the whole team. It is the place where they most likely will discuss solutions to problems and other things that are important for them – not necessary project or work related. A paper-based task board is based on large sheets that are hanging on the wall with Post-Its, which represents the stories and tasks. Such task boards are big, usually several meters long and high – they are very present and immediately catching one's eye (Wirdemann, p. 144-145, 2011). Also, research has shown that without a visible reminder, people tend to forget what they have to do (Perry, 2008). In contrast, software-based task boards only exist on a machine; they can be made visible by using a display. Such displays are smaller than those large sheets of paper; even a big 50" HDTV screen is much smaller. As a result, the text is also smaller and thus worse to read from distance. However, this stands in contrast to the fact that Scrum needs openness – when people cannot see at a glance what is going on, the task board looses its purpose. Using a software-based task board and displaying it on a large screen is a first step in the right direction but cannot stand the comparison with a paper-based task board.

3.2.2 Flexibility

By flexibility is meant to which extend the task board is customizable according to individual needs. A paper-based task board is quite flexible; it can be extended in height and width very easily by simply attaching another sheet of paper onto the wall. By doing so, enlarging a certain column or adding a new one becomes very easy. However, in the end, also a paper-based task board is limited; for example it cannot be made bigger than there is space on the wall. Software-based task boards are not bound to that limitation, they can be extended in any way one likes. Although the screen is smaller than a paper-based task board (as described above), one is able to scroll horizontally and vertically. This freedom comes though at the cost of usability; see section 3.2.9.

3.2.3 Motivation

Above was already explained why accessibility is so important. In this section it will be shown that accessibility can be linked to motivation. Motivation is necessary – a motivated team enjoys doing its work more than a team that is not motivated; a motivated team is thus more focused and efficient. Several authors describe how being motivated helps us to improve. One of them claims that skill is not

something what humans receive when born, but that it can be mastered if one is motivated by one's own failures (Krakovsky, 2007). If one accepts failures and if one sees them as a motivation to become better, than one is actually able to improve. To do so, one has to make errors public. Combining this information with the knowledge from the previous section, a paper-based task board supports this personal development due to its transparency, where as a software-based task board doesn't. Of course, the team is able to see if something is going wrong when using a software-based task board, but this discovery process is triggered much later in time, due to software complexity and an overwhelming set of features that distract from what is really important. This point of view is also supported by the result of another study (West & Grant 2010), which basically states that there should be room for making errors, as the team can reflect on them and improve eventually.

A different approach is the one of "Flow" (Csíkszentmihályi, 1996). By Flow, the mental state of a person being joyful is meant. Such a person is highly motivated, enjoys its work and works more efficient. In order to reach such a state, certain prerequisites have to be fulfilled, important for this evaluation are the following ones:

- Finishing a task requires direct feedback
- A task must be rewarding
- Visualization increases efficiency

Mapped to the task board, only a paper-based task board supports at least indirect feedback. The task board itself cannot give feedback – but to finish a task, a person has to walk to the board and move the Post-It to its new location. This action attracts the attention of the rest of the team, where moving a task on the screen in front of one's own machine does not, as the rest of the team is not able to see it. Such a lack of interaction between the team members could lead to reduced team awareness (Hossain, Bannerman & Ross Jeffery, 2011). Of course, software could be improved in this area, e.g. by playing a jingle as soon as a status gets changed, etc. However, this is not available currently.

Linked to the first prerequisite, a task is rewarding when one receives positive feedback from others. If the attention of the others is not called, they do not know if a person has finished a task. Currently, some software tools are able to create this attention by including social networking aspects like a timeline, where updates are posted. By doing so, other team members are able to see that a task is finished. However, this does not necessary mean that they give positive feedback to that person, but the same also holds true for the paper-based variant.

Especially in groups, Flow is stimulated when information visualization is available. Again, paper-based task boards have the advantage of catching attention. However, if a large screen is used to visualize the software-based task board, that lead is shrinking.

3.2.4 Haptic Quality

The haptic quality criterion is quite an underdog: only experts that have knowledge in the field of psychology have stated the outstanding importance of a haptic response. As described in the criteria section (3.1.4), an example that most people can confirm is that crossing-out items on a paper To-Do List is more joyful than doing the same thing with a virtual To-Do list that is stored on a computer. This example is adaptable: team members feel the same way as soon as they put a finished task in the "Done" column on a paper-based task board, whereas team members who solely use a software-based task board do not. The reason for this is insufficient somesthetic feedback (Robles-De-La-Torre, 2006). Not receiving adequate tactile feedback is comparable to not receiving reaction from one of our five senses, in this case the touch sense. As a consequence, if a software-based task board is used, users do not feel the same joy as when a paper-based task board is used that provides them with tactile feedback. To counter this, a touchscreen would be an option. However, large HDTV screens that come with a touchscreen are very rare and cost a multitude of their touch-less pendants.

3.2.5 Integration

The purpose of integration is to which extend a task board can be integrated into the company workflow. This integration is of high importance nowadays, as companies want to increase the

automation of processes (West & Grand, 2010). The idea of integration is also linked to the rise of geographically dispersed teams, which are dependent on certain tools in order to work with each other.

Software-based task boards have an advantage; they are part of an agile application life-cycle management (ALM), which can be connected with other tools like issue-ticketing systems or build platforms. Most of the software-based task board vendors offer complete packages, which embed other Scrum-related functionality into a series of tools that can be connected to other software by using an application programming Interface (API).

Paper-based task boards cannot be integrated into the tool workflow of a company, due to the fact that a paper-based task board does not exist electronically. Of course, companies can and actually use certain scenarios (see section 5.1.5), which allow the co-existence of paper next to software to avoid such problems.

However, it is questionable whether such integration is necessary in the case of a task board since it is only a tool to track the status of the team. It does not output data that are needed as input for other tools, in contrast to e.g. the product backlog, which holds all the upcoming functionalities of the product.

3.2.6 Costs

When comparing costs of a paper-based and a software-based task board the result is quite obvious: some sheets of paper and a pile of Post-Its cannot be compared cost-related to a software-based task board system, which costs at least $6000 for small companies with 40 users up to $400.000 for a big company with 700 users – valid for the Top10 agile tool vendors (West & Hammond, 2010). On the other hand, such software tools feature much more functionality than just a task board; still this is out of scope for this master thesis.

3.2.7 Availability

As the team uses the task board very frequently, it has to be available at all times. If the task board is not available, the team is unable to continue their work. Important for task boards is thus their uptime – the time they are available. In the world of software, the uptime is measured by stating the downtime per year. For example, an availability of 99% would result into a downtime of 3,65 days a year – the software is thus not available on 3,65 days. Around four days of a whole year does not sound that much – for example, the software could be not available on four Sundays – still, it is unpredictable when this downtime does occur. Unfortunately, software-based task board vendors do not provide any numbers regarding their availability. In contrast to software, a paper-based task board is likely to have less “downtime” – the worst thing that could happen would be if the paper falls down and has to be put back on the wall. The team could do this themselves, but when a software-based task board is down, the team is depended on the system administrators. Thus, they cannot fix the problem themselves.

3.2.8 Archiving

However, availability could also be related to archiving, e.g. how long one is able to go back in time to check on something that has happened in one of the previous sprints. Archiving sprint data is an aspect that is needed by companies (Perry, 2008), although barely mentioned in literature. When using a paper-based task board, all the Post-Its are thrown away at the end of each sprint or stored in a box. Either way, historical sprint data are not available. Someone who wants to go back in time in order to research a specific thing is not able to do so when using paper. Taking a photo of the task board does not help, as this photo only shows a particular moment in time, not the whole progress. Also, when making a photo at the end of a sprint, the task board should be empty: in ideal circumstances, all tasks are in the “Done” column – one can thus predict this status without the support of a photo. When using a software-based task board, things are different: one is able to track the status of every available story or task at any moment in time and one is able to read comments that were made on any story or task (Sarkan, Ahman & Bakar, 2011).

3.2.9 Overhead

In this context, by overhead is meant the amount of training needed and usability experienced, which both are connected to each other.

Software can be very complex and software can be made so that it is easy to use, but the simplicity of a paper-based task board cannot be beaten by software. A paper-based task board needs a minimum amount of training; it only requires a few sentences to explain that the status is updated by moving the Post-Its and to explain the purpose of each column. When one wants to do the same with a software-based task board, several steps have to be made before one is actually able to move tasks: the machine has to boot, the user has to login into the system, the software has to be started, the user has to login into the task board software and not until then he is finally able to do the same.

Next to the lower usability, humans have to be trained in order to use software. They have to remember their credentials, they have to know where to click, they have to understand the nature of the software and they need to learn some tricks to improve their usage – these are all aspects the user does not need to know when using a paper-based task board.

3.2.10 Distance

Distance is a key criterion when it comes to making a choice between a paper-based or a software-based task board. As already described, originally Scrum did not propose to use any software tools. However, since a decade, globalization gets more and more important for software engineering teams (Hossain, Bannerman & Ross Jeffery, 2011). As a result, nowadays only 17% of the teams are at one, single location, the remaining 83% are teams that are distributed over at least one location (West & Grant, 2010). Even more important, 22% of these distributed teams have stated that more than 50% of the team members are located in foreign countries. Thus software tools like a task board can help these majorities to perform Scrum in real time. Teams heavily rely on Scrum tools that work over the Internet that are split in such a way. In Section 5.1 will be described that it is possible to use a paper-based task board within distributed teams, but also that the advantages succumb to its disadvantages, so that teams need therefore a tooling strategy (West & Grant, 2010).

Using distributed teams introduces problems at three different levels (Hossain, Bannerman & Ross Jeffery, 2011):

1. Temporal distance

2. Geographical distance

3. Cultural distance

The temporal distance issue comes into play if the sites where the team members are located are far away from each other, e.g. a team that is distributed over Germany, India and Vietnam are working at very different hours due to the time-shift: when the German team starts working, the Indian team is already having lunch for example.

Connected to the former issue, the geographical issue holds true if the teams are at different locations as this makes it very hard to work with each other, especially if there are important meetings to absolve.

The cultural distance issue is explained by the fact that human people coming from different countries have different backgrounds, be it religion, principles or something comparable.

Software-based task boards offer a solution for the geographical distance issue, as they provide the team with a working task board at any location that synchronizes within seconds – a paper-based task board is not able to do so. Yet, using any of the two task board variants cannot solve the temporal distance issue; it is the job of a ScrumMaster to make sure that the team can meet to discuss important things, although their time zones might be different. The cultural distance issue is easier to solve when the team works together at one location. If that is not possible, the team should have met at least once face-to-face (Paasivaara, Durasiewic & Lassenius, 2008).

An advantage of software-based task boards is the fact, that the customer can get an idea regarding the status of the team, even if he is not able to visit the team. However, this also turns out to be a

disadvantage: most of the customers are not experienced in working with Scrum software tools and might misinterpret the data (Perry, 2008). If the team is located at one place, it is free to choose whether it wants to use a software-based task board or a paper-based task board. Of course, also a distributed team is able to do so, but will face many challenges when choosing for a paper-based task board.

3.2.11 Communication

Scrum heavily relies on face-to-face communication, as it is the most efficient method of exchanging knowledge (Sarkan, Ahman & Bakar, 2011). The benefit of face-to-face communication is that humans are able to use all of their five senses and are thus able to detect nonverbal communication. However, this becomes impossible if teams are geographical dispersed, as team members are limited to the sense of hearing when using an audioconference system and also to the sense of sight if a videoconference system is available. The first criterion, accessibility, showed that it is important to have a physical task board in the team's room available. The task board is a place to meet each other – if such a place is not available anymore, people will have less opportunities to talk with each other, which should be avoided at all costs, as can be shown by using the following example (Reinertsen, p. 185, 2009): Many years ago, Hewlett-Packard had fixed coffee breaks, thus certain points in time where employees met with each other. Someone who had worked on a certain product just before the coffee break would use that break to discuss his or her current status with the others. Thus the coffee break was used for information sharing. However, to save costs, Hewlett-Packard decided to cancel these free coffee breaks and with them the fixed points in time where everyone met with each other. By doing so, the process of drinking coffee went from being synchronous to being asynchronous - as a result, there was no more exchange of knowledge.

Not having a task board is one issue; another issue is when the task board is just merely displaying information without the possibility of altering it. A HDTV screen is thus strictly seen not a task board, it is more like an information panel – nice to have, but it does not stimulate communication. In fact, the opposite is true: when team members want to add or update anything on the board, they have to do it on their own machines, or let the ScrumMaster do it (Perry, 2008). As a result, they do it alone, not together with the team. Teams that want to use a software-based task board in a collocated environment have to be made aware of these facts so that they can think of methods to counter these problems, e.g. using a touchscreen, or adding a dedicated machine that is used to add & update tasks next to the HDTV screen.

3.3 Paper vs. Software Overview

In the previous two sections, the criteria for the analysis have been defined and applied to the paper-based and software-based task board.

The purpose of this section is to conclude this analysis by identifying in which situation a paper-based task board should be used and in which situation a software-based task board should be used. Table 4 provides a summary of the comparison (section 3.2) based on the criteria that were defined in section 3.1.

Criterion	Paper-based Task boards	Software-based Task boards
Accessibility	✔	✗
Flexibility	✗	✔
Motivation	✔	✗
Haptic Quality	✔	✗
Integration	✗	✔
Costs	✔	✗
Availability	✔	✗
Archiving	✗	✔
Overhead	✔	✗
Distance	✗	✔
Communication	✔	✗
Total	**8**	**4**

Table 4: Evaluation comparison

If one compares the colors, the paper-based task board scores better than the software-based task board. Yet, the criteria weights are not equal; the outcome of each single criterion can influence or even make a decision for choosing one of the two task board types, as will be shown in the two following sections.

3.3.1 Paper-based task boards

Paper-based task boards have several advantages when it comes to accessibility, they can be very large, they are always present and they catch everyone's attention as described in 3.2.1. Accessibility is very important and solely achieved by the paper-based variant. The purpose of a task board is to be a tool that helps the team members to do their work by visualizing the progress (Perry, 2008). It makes the progress accessible for everyone who is either in the team's room or who visits the team's room. Due to the reason that paper-based task boards are usually several meters wide and tall and that tasks have to be written by humans which tend to write larger letters than a computer, readability is much better than on a big screen.

Since a task board is an information radiator (Cockburn, p. 76, 2001) that shows the status of the team it is important that this status is tangible next to being accessible. Paper-based task boards are flexible, the team can choose different kinds of Post-It colors to indicate various types of tasks, be it a plain task, a bug, a planned task or something else. However, this is only a fragment of the flexibility criterion: although paper is flexible, it has certain boundaries, e.g. physical limitations: as stated in 3.2.2, a paper-based task board cannot be larger than the particular wall it is attached to. Next to physical limitations also functional limitations exist like not being able to attach a file (e.g. a spreadsheet) to a certain story onto the task board due to obvious reasons.

When it comes to motivating the team to do its work, a direct link to the accessibility criterion is found. Section 3.2.3 has shown that a task board must be always present in order to support the team members to learn from their mistakes. As a paper-based task board is attached to a wall or is pinned on a bulletin board, it is always available within the team's room. Therefore, it attracts the attention of the team – when a team member goes to the task board to move a task, he or she gets immediate feedback from his or her team members as they see the action.

Working with a paper-based task board provides the user with haptic feedback, which encourages and supports him or her in doing the work. Section 3.2.4 has proven that working with a paper-based task board is a more joyful process. Literature (Amabile & Kramer, p. 76-84, 2011) has shown that small pieces of work can create a feeling of success; mapped to the task board space this means that if a team member moves a task from one column to another, he or she feels being successful. This feeling of success is yet again linked to the previous mentioned accessibility criterion as this achievement feeling can be increased by feedback from other team members, which are only able to do so if they are notified about the change. Once again a paper-based task board supports such a notification since each team member has to go to the board to change something.

In terms of integration, paper-based task boards can't hold a candle to its software-based pendant, as shown in section 3.2.5. The reason for this is evident since a paper-based task board has no connection to software as long as the coexistence scenario described in section 5.1.5 is not given.

Cost-wise, a paper-based task board cannot be beaten by any kind of software. To start with a paper-based task board, only a few sheets, some glue and Post-Its are necessary - see section 3.2.6. Even if the software has no license fees at all, a dedicated server is necessary that comes with some changes to the IT landscape. Costs are thus always lower if a paper-based task board is used.

Next to being cheap, a paper-based task board has also a high availability as paper has no downtime in contrast to IT systems (3.2.7). A paper-based task board is always available; the only thing that can happen to a paper-based task board (and what really happened once during field research) is that it can fall down due to dissolving glue. However, as such an event becomes immediately visible, the team members themselves can fix it in no time.

A criterion that has been found when consulting external experts was the need for archiving sprint data, e.g. the possibility to check back on previous sprints to track a certain item. Paper-based task boards do not offer any kind of archiving (3.2.8), although pictures of the task board can be taken. Yet, taking pictures is merely a workaround, as the information is static and not interactive as the picture is just a snapshot of the status quo.

A recent survey (Azizyan, Magarian, & Kajko-Mattson, 2011) has shown that companies indicate that ease-of-use is the most important aspect of a Scrum tool. The overhead criterion grasps this aspect and the amount of training that is needed. The paper-based task board is very easy to use and it can be explained to people who are unfamiliar with it within a few minutes as described in section 3.2.9. In fact, the team members only have to do two things: write new tasks and update the status of the current ones, there are no prerequisites that have to be fulfilled before being able to do so.

The biggest issue with paper-based task boards is that they are only suitable for teams that are located at one site. If a team is distributed and geographically dispersed, it becomes much more difficult to work with a paper-based task board. In fact, to do so one has to make sure that at each location copies of the same task board are available. Yet, there is still no real time synchronization between the task boards, which can lead to confusion within the team regarding who is currently working on which task (see section 3.2.10). Different scenarios exist (see sections 5.1.2 & 5.1.5) which try to enable the usage of paper-based task boards at multiple locations, but they are merely workarounds.

In terms of communication (3.2.11), paper-based task boards encourage the team to communicate with each other. In fact, a paper-based task literally forces the team to communicate, as the task board acts as a central meeting point. With the accessibility and availability criterion in mind, the paper-based task board shows it strength: it is always present and since it has only one single interface, it enables discussions and communication.

3.3.2 Software-based task boards

Software-based task boards are fundamentally different when it comes to visualization options: instead of being glued onto a wall like a paper-based task board, the software-based variant is a representation of a task board by using software. Therefore, a software-based task board can only be shown on a monitor, which are usually much smaller than a paper-based task board. As a result, also the items on the task board like stories and tasks are much smaller; the same is true for the size of the text. Accessibility is thus limited to the screen size of each team member (3.2.1). Of course, companies try to compensate this disadvantage by placing big HDTV screens in the team's room (see 5.1.4) but they are still small compared to a standard paper-based task board.

Flexibility-wise, software-based task boards show their strengths as described in section 3.2.2. Software is not bound to physical constraints other than computer hardware, which is scalable. Current Scrum tool vendors have utilized that advantage as is shown in chapter 4.

As stated in the previous section, the accessibility criterion is related to the motivation criterion. One of the strengths of a paper-based task board - namely its great accessibility - cannot be countered by software. Since team members change the progress of their work by using a piece of software on their own machine, other team members do not get directly informed when a change has occurred and are therefore not able to provide direct feedback (3.2.3). Yet, literature (Krakovsky 2007, Csíkszentmihályi 1996) has shown that making progress transparent is the key in motivating each other to perform better. Thus, as long as the accessibility of software cannot be matched to that one of paper, the same stays true for the motivation criterion.

The haptic quality criterion is yet another problem of software-based task boards: a software-based task board does not match the haptic response a paper-based task board provides (see section 3.2.4). The key difference between both task board types is the way data are entered and altered within the system. Whereas one can write on Post-Its and move them on the board when using a paper-based task board, one has to use a computer to enter the text and to change the status of a task. The above-mentioned large HDTV screens do not help to solve this problem, as they usually do not have any way to input data. Software-based task boards thus need to develop a solution that allows inputting and changing data in a similar way as paper-based task boards do.

Integration is one of the strong points of software-based task boards as they are usually a part of a Scrum software tool that covers several other artifacts of Scrum like the product backlog or a Burn-Down-Chart. The need for this functionality is again given by the recent survey mentioned above (Azizyan, Magarian, & Kajko-Mattson, 2011) and backed up by the companies that were subject of the field research. Nowadays, companies have defined global workflows that support the process of software engineering. These companies request that the tools they use can be integrated in such a workflow, which is not possible when using paper-based tools as they do not provide input or output that can be used by software. As indicated in section 3.2.5 it is questionable whether a task board needs to be integrated in such a workflow – yet again, a software-based task board is just a small part of a Scrum tool.

In terms of cost, a software solution cannot compete with a few sheets of paper and some Post-Its (see 3.2.6 for the cost criterion analysis). However, this is only partly true as Scrum software usually offers much more than just a plain task board. Evaluating this functionality is out of scope for this master thesis project. Still, one has to keep that in mind when choosing a Scrum software tool.

The uptime of a paper-based task board cannot be beaten by a software-based solution. As described in section 3.2.7 as well as during the evaluation of the Scrum tools in chapter 4, vendors offer certain guarantees regarding the availability of their software. Still, this guarantee cannot be compared to the one of paper-based task boards. The reason for this lies in the fact that unavailability of a paper-based task board can be fixed by the team itself, whereas unavailability of a software-based task board is outside the control of the team.

One of the advantages of a software-based task board is the ability to archive Scrum related data (see section 3.2.8). By storing the data in a database, Scrum software tools are able to go back to any moment in time. With this advantage, issues can be tracked and actions performed in the past can be examined to learn from errors and to improve the Scrum process.

As stated above, by overhead the amount of training needed to use a task board is meant as well as its usability. Again, software cannot compete with paper, as is described in more detail in section 3.2.9. Using a software-based task board requires more user training and is more complex to use. Team members have at least to login into the system and get to know the software before they are able to use it. The ease-of-use of a paper-based solution is thus unmatched.

The distance criterion is next to the integration and the archiving criterion the key factor for using a software-based task board. Due to its nature and by making use of the Internet, software-based task boards are available at multiple locations and information is synchronized within seconds. No workarounds are thus needed in order to have a task board at each site of a company. Making the task board accessible via the Internet also enables the customer to check back on the teams' progress. Whether this advantage can also result into a disadvantage depends on the customer (Perry, 2008). Yet, software does only offer a solution for one of the distance problems described in section 3.2.10, namely for the geographical distance issue. Temporal as well as cultural distance issues keep staying present, problems that have to be solved by the ScrumMaster. Still, it is arguable that cultural barriers can be more easily resolved when people are working together at the same site by using a paper-based task board (Paasivaara, Durasiewicz & Lassenius, 2008).

Software-based task boards perform worse in terms of communication, when being compared with its paper-based pendant. Whereas paper-based task boards support all human senses, software-based task boards only provide the sense of sight. Next to this discrepancy, face-to-face communication is not possible. Teams have to use text-, audio- or video-based systems, which only allow a maximum of two senses and can lead more often to misunderstandings. Again, as a software-based task board in combination with a large, standard HDTV screen does not offer any input possibility (see section 3.2.11), the task board is merely an information panel instead of a central point of communication - which is in effect the same when using a paper-based task board.

3.4 Wrap Up

In this chapter, research question two was answered by defining criteria to perform the paper versus software analysis within section 3.1.

In section 3.2 the actual analysis was performed and in section 3.3 an overview was given. Both sections together gave an answer to research question three.

With the findings presented in this chapter, companies are able to decide whether a software-based task board suits them better than a paper-based one or the other way round. As a result, by giving an answer to research question two and three, the first half of the first part of the research goal has been reached.

4 Task board Tools

Now that the analysis of paper-based versus software-based task boards is completed, the purpose of this chapter is to reach the first sub-research goal and to answer research question four.

To do so, this chapter will evaluate a set of tools that are able to represent a software-based task board. Before the review-process of these tools can be performed, two aspects have to be dealt with.

First of all, a not even remotely comprehensible amount of Scrum tools exist. Reviewing every single tool is outside the scope of this master thesis project. Therefore, some requirements will be defined to create a subset of these tools that are then subject of this master thesis. This will be done in section 4.1.

Second, criteria for performing this evaluation of Scrum tools have to be defined, similar to the criteria selection process in the previous chapter. The reason for defining new criteria is that after having consulted various experts and having checked the outcomes of agile tooling surveys, it became clear that if one wants to compare software among each other, the aspects of importance are different when comparing the aspects that are important for the paper versus software analysis. For more details and for a definition of criteria see section 4.2.

Finally, in section 4.3, the actual evaluation of the tools will be performed, followed by an overview in section 4.4.

4.1 Criteria for selecting Scrum tools

Nowadays, a wide variety of Scrum tools that feature a software-based task board exist. Evaluating all of these tools would be outside the scope of this master thesis project. Therefore, criteria will be defined that allow creating a subset of tools that will become subject of this chapter. The following three sections will explain these selection criteria; section 4.1.4 will summarize the results.

4.1.1 Vendors

In order to create a selection out of all available vendors, the Top 10 vendors of Scrum tools according to Forrester Research were considered (West & Hammond 2010), which is comparable to the list of vendors of a tooling survey (Behrens, 2006). These vendors have several years of experience and their products can be considered to be mature. Included in this list are:

1. Atlassian – JIRA
2. CollabNet – ScrumWorks
3. Hewlett Packard – Quality Center Agile Accelerator
4. IBM – Rational
5. Micro Focus – Star Team
6. Microsoft – Visual Studio Team Foundation Server 2010
7. MKS – Integrity
8. Rally Software – Rally
9. Serena – Serena Agile Planner
10. VersionOne - VersionOne

4.1.2 Costs

From this list of vendors, only those were selected that allowed a free trial to evaluate their products due to the fact that for this master thesis project no financial resources were available.

4.1.3 Platform

Finally, only those vendors were selected that offered a hosted, browser-based product. The reasons for this were twofold: first, there were no additional resources for a dedicated server to install the product. Second, experts stated that it is important for a Scrum tool to be platform independent and that the tool does not require any additional installation on the client side as this would lead to problems with external customers and temporary workers (Berczuk, 2007).

4.1.4 Selected Tools

Table 5 evaluates the vendors according to the defined criteria above.

Vendor	Costs	Platform
Atlassian	✔	✔
CollabNet	✔	✔
Hewlett Packard	✔	✗
IBM	✔	✗
Micro Focus	✔	✗
Microsoft	✔	(*)
MKS	No response regarding the evaluation request.	
Rally Software	✔	✔
Serena	✔	✗
VersionOne	✔	✔

Table 5: Tools matched with criteria

Note: The asterisk (*) in the Microsoft row indicates that the Microsoft product does not match the platform criterion. However, a third party tool (Urban Turtle) is available, which enables Scrum for the Microsoft product. Additionally, Urban Turtle matches the platform criterion. MKS did not reply on the request for evaluating their product.

As one can see, 5 out of 10 vendors matched the criteria. After defining the evaluation criteria for those tools, each tool will be evaluated in a dedicated section.

4.2 Evaluation Criteria

As indicated above, some of the criteria defined in chapter 3 cannot be applied for comparing software among each other due to the different setting (software versus paper in contrast to software versus software). For example, the haptic quality criterion described in section 3.2.4 is unimportant in terms of comparing software, since the prerequisites for using software (monitor, keyboard, mouse) are essentially the same. Therefore, next to those criteria that are still applicable, some criteria have been stripped and others have been added.

Fortunately, when looking for literature regarding Scrum software tools to define criteria, several surveys can be found in which companies were asked which tool they are using, which aspects they like / dislike, what type of company they are, etc. (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009). However, those surveys could only partially be used for this master research for the following reasons:

- Some surveys conducted by tool vendors themselves are potentially biased (Dubakov & Stevens 2008, VersionOne 2009)
- Lack of questions regarding which tool criteria are important for companies (Behrens 2006, Dubakov & Stevens 2008, VersionOne 2009)
- Some surveys had no specific focus on task boards (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009)
- Surveyed persons were mostly managers and not Scrum team members (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009)

Still, one survey mentions some criteria that are useful when evaluating software-based task boards. Also 12% of the survey respondents indicated that they “wished for tools to adequately replicate

physical task boards" (Azizyan, Magarian & Kajko-Mattson 2011) and 17% of the surveyed persons were developers, thus Scrum team members. This matches with the fact that some of the respondents indicated the need for a proper, electronic task board replication.

The following table lists the criteria found for evaluating the Scrum software tools. The criteria are based on the following sources:

- Criteria from chapter 3.
- One survey that was applicable (Azizyan, Magarian & Kajko-Mattson 2011).
- Two external company experts.
- Three internal company experts.

In the next sections each of those criteria will be defined and explained.

Criterion	Literature Review	Internal Expert Opinion	External Expert Opinion
Usability	✔	✔	
Resources			✔
Quality			✔
Integration	✔	✔	✔
Functionality	✔	✔	✔
Motivation		✔	
Costs	✔	✔	✔

Table 6: Criteria for evaluating tools

Like in chapter 3, each criterion has its own set of properties, which are used in turn to evaluate the software tools. This time, the properties were mainly defined with the help of the two above-mentioned external company experts (except for the motivation criterion, for which the internal company experts were consulted). These experts were either the persons in charge for choosing a Scrum software tool at their company or persons that had worked for a certain period of time with Scrum software tools. For the defined set of criteria above, the following table lists their properties:

Criterion	Property
Usability	• Dashboard functionality (low / high) • Context-based help functionality (no / yes) • Drag & drop (no / yes) • Number of task data shown (low / high) • Keyboard shortcuts (no / yes) • Ability to create tasks (no / yes) • Ease-of-use (low / high)
Resources	• Minimum screen resolution (low / high) • Browser requirements (low / high)
Quality	• Availability of phone support (low / high) • Support response time (slow / fast)
Integration	• Number of products in vendor portfolio (low / high) • Programming interfaces (no / yes) • Number of connectors for third party products (low / high)

Criterion	Property
Functionality	• Full screen mode (no / yes) • Columns customizable (no / yes) • Font size (small / large) • Degree of story estimation (low / high) • Degree of task estimation (low / high)
Motivation	• Comments (no / yes) • Task dots (no / yes) • Automatic page refresh (no / yes) • Acoustic / visual trigger (no / yes)
Costs	• Expenses hosted (cheap / expensive) • Expenses download (cheap / expensive)

Table 7: Software tools criteria with their properties

All above-mentioned criteria with their set of properties are subject of this evaluation. However, to evaluate some of the properties in a thorough manner would consume more resources (time-wise and person-wise) than available, for example the "programming interfaces" property of the "Integration" criterion: for this evaluation, it is checked whether such an interface exists or not. However, a more thorough evaluation of Scrum software tools should include tests of that interface in terms of functionality, usability, etc. The same is true for the "number of connectors for third party products": also these connectors should have to be tested to evaluate the "Integration" criterion properly. Still, also properties exist which are fully covered by this evaluation, for example the "Ability to create tasks" of the "Usability" criterion or the "Task dots" of the "Motivation" criterion.

To sum it all up, this evaluation is a thorough as possible within the scope of this master thesis project, but further research would be able to could cover some of the criteria in a more detailed manner.

Now that the criteria have been defined and their corresponding properties are listed, the criteria will be explained in the following sections.

4.2.1 Usability

The usability criterion indicates the user-friendliness of a Scrum software tool. This comprises several aspects like ease-of-use (Uy & Rosendahl, 2008), context-based help functionality, support for drag & drop of tasks, etc. (Azizyan, Magarian & Kajko-Mattson 2011). Experts and field research have indicated that usability is one of the most important criteria. No matter how many features a tool offers, if it is not user-friendly, the team is not going to enjoy working with it.

4.2.2 Resources

Field research has indicated a trend towards browser-based applications. All of the visited companies (see section 5.2) stated that installing local client software is undesired; this is especially true for Scrum software in particular. Reasons for this trend are the rise of distributed teams due to globalization: external customers are becoming quite common (Perry, 2008) and some foreign, distributed locations have strict IT policies that do not allow the installation of software. As a result, companies demand browser-based Scrum software – as offered by the majority of all tool vendors. When programming websites, techniques (e.g. Asynchronous JavaScript and XML (AJAX)) exist that give websites the look and feel of local-installed software. However, using such techniques requires specific browser versions. On top of that, each browser renders pages differently. Thus, tool vendors have to make sure that their software works with common browsers to make sure that clients are satisfied.

4.2.3 Quality

By quality is meant whether the tool vendor provides its customers with certain guarantees. The reason for that is that some of the vendors allow to rent their software instead of selling it. These two

options are quite different: if the vendor sells its software, the customer has to install it on his own server and has to make sure that this server is administrated well. However, if the vendor rents his software to clients, it is the vendor who has to make sure that his servers are available as much as possible, because a downtime means that his customers cannot use the software. The client requires a kind of guarantee of the vendor, which is called Service Level Agreement (SLA). Therefore when a tool is evaluated that offer a hosted option, the vendors SLA has to be taken into account.

4.2.4 Integration

Although the integration criterion is not directly linked to the task board functionality of a Scrum software tool, it is one of the main advantages of using such a tool (see section 3.2.5). Integration can be evaluated on two levels:

1. High level, which means that the tool vendor offers a product portfolio, which allows the tools inside that portfolio to connect to each other.

2. Low level, which means that the tool vendor provides an Application Programming Interface (API) that allows other, third party tools to communicate with the vendor's tool.

The effort for evaluating the levels of integration is low, but important for companies at the other hand (Azizyan, Magarian & Kajko-Mattson 2011). As a result, this criterion will be included.

4.2.5 Functionality

By functionality, basically the availability of specific features (see below) is meant. Functionality is important (Uy & Rosendahl, 2008), as it is one of the possibilities to outperform a paper-based task board.

Within the Scrum-context, stories have to be estimated. Different approaches exists, the suggested one is the measure of so-called story points, which are used to indicate the size of a story. Some companies however like to estimate stories based on time, e.g. the amount of hours that needs to be spent in order to finish the story. Explaining the advantages and disadvantages of these methods (Gloger, p. 140-150, 2011) is out of scope for this master thesis. In ideal circumstances, the tool allows both measures so that one is able to choose freely between them.

Another feature would be the ability to create new task board columns (Azizyan, Magarian & Kajko-Mattson 2011), next to the default ones ("To Do", "In Progress" & "Done"), for example a "To Test" column which represents tasks that can be tested by someone from quality assurance (QA) or a "Documentation" column, if a certain task needs special documentation.

Next to such features, other features that go beyond the ones known from a paper-based task board will be taken into account.

4.2.6 Motivation

Field research and experts share the same opinion when it comes to motivating the team to work with a task board: a big HDTV screen that does not allow any form of input is ignored after a short period of time. The reason for this is that one is not able to interact with the device; it is merely a source of information that is mostly static. Ignoring a task board is also called wallboard-blindness. Several possibilities exist to counter such behavior:

- Triggering the team visually, e.g. by changing the color in order to attract the users
- Triggering the team acoustically, e.g. by playing a sound as soon as a change occurs

Next to stimulating the team in the above-mentioned ways another interesting way of motivating the team is by embedding certain social network features, e.g. being able to comment on events (e.g. creating a task) or actions (e.g. changing the state of a task). By doing so, the team gets more involved in the work of the others and communication is stimulated. Last but not least, field research has shown that teams like to mark their tasks with a little dot to indicate that finishing it takes longer than expected. For each day that the task is in progress, a dot is added. By doing so, potential impediments can be easier identified. A software-based task board should thus be able to put virtual dots on its tasks.

4.2.7 Costs

The same what was said in section 3.2.6 is valid for this criterion. Costs are always an important factor for companies. Tool vendors often offer different types of packages. For comparing the costs of each tool, three different team sizes will be used (West & Hammond, 2010):

- Small team (10 daily, 30 casual users)
- Medium team (50 daily, 80 casual users)
- Large team (200 daily, 500 casual users)

By doing so, a common base for checking the costs of each tool will be created.

4.3 Tool Evaluation

This section will present a review of Scrum tools that are listed in section 4.1.4. For each following tool, the criteria described in section 4.2 will be used to perform an evaluation.

4.3.1 Atlassian JIRA

JIRA is an issue tracking system made by a vendor called Atlassian, which is also the company behind many other well-known pieces of software like its wiki called Confluence. Some time ago, Atlassian has acquired a company that has developed the so-called GreenHopper plugin for JIRA, a plugin that introduces certain agile features. Field research has indicated that more and more companies that use Scrum for developing software have made the transition to JIRA or are planning to do so. Within the following section, the above-defined criteria will be used to evaluate JIRA. Whenever JIRA is mentioned, the GreenHopper plugin is meant to be included.

Usability

Once JIRA is set up and working, the user experience as well the look and feel of the task board is quite good. As soon as the user is logged in, JIRA presents the user his personalized dashboard. This dashboard features multiple boxes that in turn give the user a status overview. By default, four boxes exist:

- Introduction, which provides links to the JIRA user guide and training courses.
- Activity Stream, which lists all recent changes made by the project team. Examples of these changes are a new task that was created or that someone started working on a particular task.
- Assigned to Me, where a table of tasks is presented that are assigned to the user. This table is sorted on the basis of the task's priority.
- Favourite Filters, a powerful feature that allows the user to create his own filters to display information. The above-mentioned "Assigned to Me" is an example of a filter, which only lists those tasks that are assigned to the user. Next to that, the user is enabled to list all open tasks of the current sprint or all stories that are not yet finished – only to name a few possibilities. By using this feature, the user gets the information that is most helpful for him or her at a glance.

When using the actual task board provided by JIRA, the experience is mixed. The reason for this is, that a recent version of the GreenHopper plugin introduced a new style of representing the task board, without removing the old layout. As a result, the user is confronted with two representations of the task board. On the one hand, the user can use the original type by clicking on "Task board", on the other hand the user can choose a "rapid board", which present the newer variant of the task board by clicking on "Work" in the upper right corner. Even more troublesome, both types of task boards rely on the same data, e.g. the defined stories and tasks – however, it appears that the new rapid board adds a custom field so that it only shows the stories and tasks that use that field. This issue results into different representations of the status quo, although they should be the same, as can be seen when comparing Figure 10 with Figure 11.

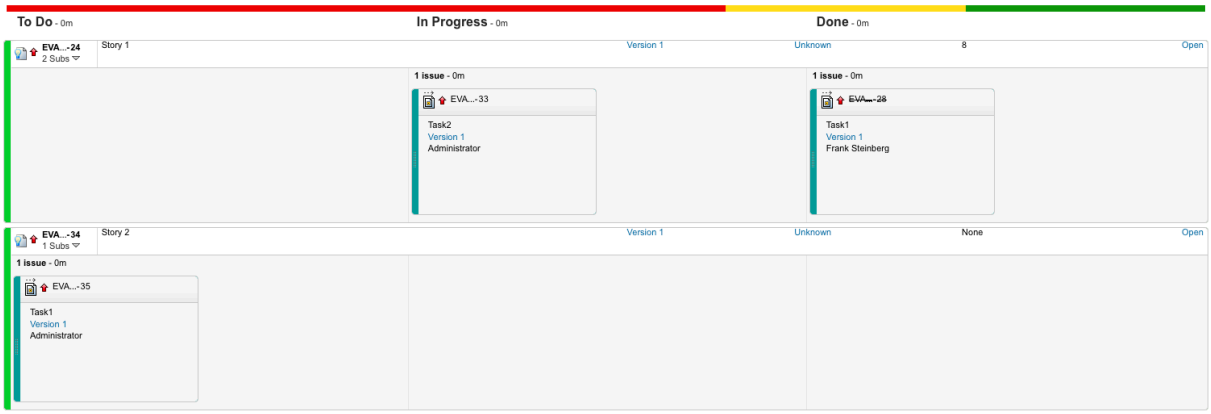

Figure 10: JIRA Task Board

As one can see, Story 2 does not appear on the rapid board although it is shown on the original task board. Also, the data that are displayed by the rapid board are different; see the functionality section below for a more detailed explanation. This all leads to an inconsistency that will confuse the user. When using JIRA, the project team has to decide whether to use the former or the latter, but both at the same time seems not to be possible.

As Atlassian is pushing the rapid boards recently, it seems that the original task board will be removed in an upcoming version. As a result, for the reminder of this evaluation, the rapid board will be used.

Figure 11: JIRA Rapid Board

The look and feel of the rapid board is like one would expect from a Scrum software tool. Tasks do not display the name of the assigned team member; instead JIRA has chosen to display the avatar. This has the advantage that users can identify themselves better by using images. On the other hand for every team member who is not using an avatar, JIRA will show a default one as can be seen in Figure 11. If multiple users do not choose a custom avatar, team members will get confused as no one knows who is assigned to which task. Still, users are able to click on the ID of each task or story – JIRA will then display a panel on the right side with more detailed information including the name of the assigned team member.

The rapid board supports drag & drop; user are thus able to change the status of a task by using the mouse. Next to using the mouse, JIRA offers a great amount of keyboard shortcuts to perform certain activities, which is a well-liked feature especially for developers.

Context-based help is not available when using the task board (the dashboard offers a user guide as described above). As a result, actions have to be intuitive, like dragging & dropping a task from “To Do” to “In Progress” which works the way it should. However when it comes to creating a task, things get more complicated: to do so, the user has to click on the corresponding story ID, then on the gearwheel at the right and choose the “Create Sub-Task” option which confronts the user with a great variety of options. Alternatively, one could try to use the always-present “Create issue” link on top of the task board, but there is no possibility to link a task to a certain story. This problem illustrates that JIRA is in its origin a ticketing system and not a dedicated Scrum tool; there are certain relicts that have to be corrected in upcoming versions to increase the usability of this tool.

Resources

As JIRA uses AJAX for drag & drop support, the browser should be an up-to-date one. However, Atlassian states on one of their support sites that all browser types (except Opera) are supported[1]:

[1] https://confluence.atlassian.com/display/GH/Supported+Platforms (accessed at 01.10.2012)

- Microsoft Internet Explorer 8.0 and higher
- Mozilla Firefox 4.0 and higher
- Apple Safari 5 and higher
- Google Chrome 12 and higher

Most of the browser requirements (especially Mozilla Firefox and Apple Safari) are quite low – chances are high that one of these browsers is used within a typical company.

Resolution-wise, a XGA (1024 pixels * 768 pixels) screen is sufficient which is also low compared to standard resolutions in companies nowadays, especially for developers who cannot have enough space on their screen.

Quality

Atlassian has defined multiple SLA's for their products. Three categories are available:

Support type	Hosted	Download (Standard)	Download (Enterprise)
Phone support	8 hours / Mo. – Fr. *	8 hours / Mo. – Fr. *	8 hours / Mo. – Su.
Response time	1 - 24 hours	1 - 24 hours	1 - 24 hours

Table 8: Atlassian Service Level Agreements

* Only for system outages.

Next to phone support & response times, various other features are described, like how many technical contacts are allowed and what support does / does not include[2].

Integration

When it comes to integration, Atlassian shows its strength. The product portfolio includes 13 different applications that can be connected with each other. The portfolio satisfies every need a developer could have, from its Scrum Software GreenHopper over a Chat System called HipChat to continuous integration management systems like Bamboo. On top of that, most of the Atlassian products feature a plugin system, including JIRA. In fact, GreenHopper is nothing else than a plugin for JIRA. Next to GreenHopper, more than 400 plugins are available that try to fulfill every need that is not covered by Atlassian's own portfolio. To complete the picture, Atlassian offers also an API, which allows interested developers to program scripts to extract or insert data into JIRA.

Functionality

Functionality-wise, JIRA has plenty to offer. However, only a fraction is of interest for this master thesis project, as the focus is on the task board. Sadly, the briefly discussed inconsistency continues to exist when one compares the features of both task board variants. The original task board shows the exact amount of information one knows from a paper-based task board: for each story, JIRA creates a swim lane in which the tasks are stored. On top of the task board, a bar indicates the progress of the team by using traffic light colors. For example, in Figure 11 most of the bar is colored in red due to the fact that JIRA generates this bar based on task *and* story state, which can be confusing at first. However, in a real world scenario each story has approximately ten tasks – the bar will then be less confusing as the steps are much smaller. The user is able to choose between three different styles that differ in the amount of information that is displayed per task. Besides, the user is able to blank out each column, which might be handy if one only wants to see the "To Do" and the "In Progress" column to save screen space. Last but not least, the original task board also offers a full screen mode, which disables the controls and puts the focus on the task board.

All of the above mentioned features are not available when using the new rapid board right now. Most of the features are nice to have but not essential, only the full screen mode should be included as soon as possible, as it makes sense for large HDTV screens where browser controls are only wasting screen space. However, the lacking features are compensated by the less overloaded interface. Although there are no swim lanes, a little bar on top of each task indicates to which story it belongs. Instead of

[2] https://confluence.atlassian.com/display/Support/Atlassian+Support+Offerings (accessed at 01.10.2012)

displaying the team members' name, an avatar is used, the advantages and disadvantages of this choice were already described in the usability section. Font sizes are larger and thus easier to read, especially on a HDTV screen. However, both task boards are far from being readable if one is several meters away from the screen. Besides, one cannot see at a glance if a task has some user comments.

Yet another difference is the estimation of stories; the original task board only supports time-based estimations, where the rapid boards focuses on story points but also allows to enter a time-based value.

The true strength of the rapid boards is that they are very customizable. For example, one can create a rapid board that spans several projects – ideal when team members have to work on multiple projects or if a project is spanned over multiple teams. Doing so is however not trivial and one has to spend quite some time in order to tailor those boards according to one's needs.

Motivation

Motivation is, as described extensively in chapter 3, one of the weak spots of software-based task boards. Without providing any haptic feeling, the process of using a task board becomes much less joyful. JIRA tries to compensate this by embedding social network elements, like the activity stream explained in the usability section. By informing the user about the activity of his teammates, he gets more in touch with the others and perhaps he tries to compete with them, which helps to create a social bond. JIRA has mastered this functionality very well; the user is not only able to see what the others are doing, he is also able to comment on it to get in touch with the other team members.

However, JIRA does not provide anything to counter wallboard-blindness. If the status of a task gets changed, there is no visual or acoustic signal that notifies the user. On top of that, he has to reload the task board page to see if there are any changes made, the page does not automatically update itself in case a change had occurred. There is also no possibility to add a dot to a task other than using a different field (e.g. the commenting function) as a workaround.

Costs

Table 9 gives an overview about the costs of JIRA according to the defined team sizes in section 4.2.7. The costs include JIRA and the GreenHopper plugin and feature both the hosted and the download option.

Software type	Small Team	Medium Team	Large Team
Hosted (per month)	200 $	500 $	1.000 $
Download (per year)	3.300 $	12.000 $	20.000$

Table 9: JIRA costs

Note that Atlassian does not offer a discount for casual users. Compared to other tool vendors, JIRA is one of the cheaper options, which is one of the reasons why many companies nowadays are switching to JIRA. Although there are usability inconsistencies, JIRA offers much value for its price.

Conclusion

Although JIRA has the most available plugins, its real strength is its cheapness. Next to its customizability and its great amount of features, the relatively low costs are the main reason why many companies are switching to JIRA, as the outcome of the reviewed companies in section 5.2 will show. Yet, the agile extension of JIRA is not yet as mature as the ones of other tools, but Atlassian is very active, newer releases are likely to catch up with the other tools within a finite amount of time.

4.3.2 CollabNet ScrumWorks Pro

ScrumWorks Pro is the Scrum tool of a company called CollabNet. CollabNet has put its focus towards the cloud; their applications are thus hosted. According to Forrester Research, CollabNet has got a strong strategy, but its product ScrumWorks Pro is not as strong as the strategy suggests (West & Hammond, 2010).

Usability

After logging in, ScrumWorks welcomes the user with a dashboard, like JIRA does. However, the information provided is significantly less: the user only gets an overview about all the tasks that have been assigned to him or her. In fact, the information is visualized using a table that indicates to which stories the tasks belong, whether there are any comments on the stories or tasks, the amount of hours remaining per task and the status of each task.

The interface is simplistic, but sufficient. The tool features a horizontal header that lists tabs, which route the user to their specific content. The one that redirects the user to the task board is called "Sprints".

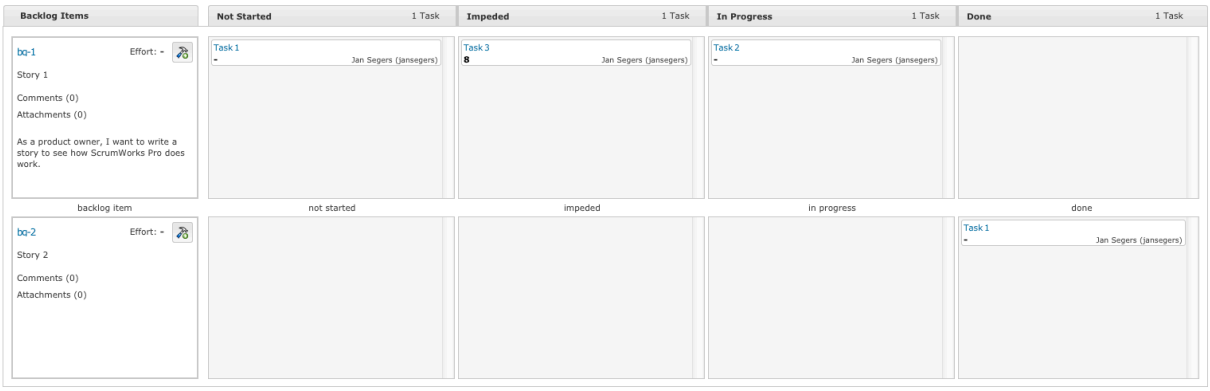

Figure 12: ScrumWorks Pro Task Board

The thing that immediately catches the attention of the user is an extra task board column next to "To Do", "In Progress" and "Done" – the "Impeded" column, which can be used for tasks that cannot be worked on due to an external impediment. Two other differences appear when comparing the ScrumWorks Pro task board with a paper-based task board. The first difference is a small representation of the Burn-Down-Chart in the upper right corner, which provides the current team status at a glance. One is able to click on that Burn-Down-Chart representation in order to get a much larger variant with more detailed information. Secondly, in the upper left corner there is an option to highlight tasks for team members. When enabled, each team member gets a dedicated color and all the tasks he or she is assigned to will get a border with the very same color – very handy to see at a glance who is doing what.

By clicking on the icon within the story in the upper right corner, one is able to add new tasks. Tasks have a very small font size but display the necessary information, namely its description, its assignee and the amount of hours left to complete the task. ScrumWorks supports time-based management, but an option for story points is nowhere to be found. Unfortunately, there is no indication on the task board when a task gets commented; the user has to click on each task to get to its details, where the comments are mentioned. For stories however, it is different: they have a dedicated counter for the amount of comments and the amount of attachments.

Like JIRA, there is no context-based help functionality. Instead, a user guide and general documentation are available. There are plenty of keyboard shortcuts, however they only work when using the desktop client of ScrumWorks, see the functionality section below for a more detailed explanation.

ScrumWorks supports drag & drop so that tasks can be easily moved around. Last but not least, ScrumWorks allows choosing a sprint for which the task board should be displayed. This could also be a sprint that has been finished in the past which matches the archiving benefit of software-based task boards as described in section 3.2.8 quite well.

Resources

As for the resources, ScrumWorks is a tiny bit more frugal[3] than JIRA:

[3] http://www.danube.com/docs/scrumworks/pro/latest/webuserguide.html (accessed at 01.10.2012)

- Microsoft Internet Explorer 8 and higher (Microsoft Internet Explorer 7 should also work, but will lack features and performance)
- Mozilla Firefox 4.0 and higher
- Apple Safari 5 and higher
- Google Chrome 9 and higher

Due to the fact that the amount of AJAX that is used by ScrumWorks is comparable with the amount used in JIRA, it is not surprising that the browser requirements are roughly the same.

The documentation does not state any screen resolution requirements, however tests have indicated that a resolution below XGA will cut off the task's assignees.

Quality

CollabNet offers three types of SLA for ScrumWorks, which differ of course price-wise:

Support type	Download (Silver)	Download (Gold)	Download (Platinum)	Hosted (Business)
Phone support	8 hours / Mo. – Fr.	24 hours / Mo. – Fr.	24 hours / Mo. – Su.	24 hours / Mo. – Su.
Response time	4 - 72 hours	2 - 48 hours	1 - 24 hours	4 hours

Table 10: CollabNet Service Level Agreements

The advantage of the CollabNet service level agreement is the 24 hours telephone support (Gold & Platinum) where Atlassian does not go beyond 8 hours. However, response time is worse than that of Atlassian.

During evaluation of ScrumWorks, an error was encountered – therefore there was a need to contact support and test the SLA of CollabNet. Although CollabNet does not indicate which type of SLA is provided for the evaluation version, the problem was escalated and solved in less than 24 hours.

Integration

The product portfolio of CollabNet includes four different products, ranging from a source-code check-in tool to an application lifecycle management tool called TeamForge.

Next to that, CollabNet offers a serious amount of integrations with tools of other vendors like Microsoft or IBM. Like Atlassian, CollabNet also provides an API so that developers are able to exchange data with ScrumWorks.

Functionality

In terms of functionality, one has to bear in mind that ScrumWorks is based on two components, a desktop client and a web client. When using the ScrumWorks web client for the very first time, one is not able to do anything at all: there is neither a possibility to create stories nor is there one to create tasks. The reason for this is that the project has to be set up first and only the desktop client is able to do so. Luckily ScrumWorks provides a small link to this desktop version in the upper right, which starts the java-based desktop client. As soon as a backlog, a sprint and a release are created and a team is administrated, one is able to go back to the web client, which is now filled with data.

At first one might get the idea that ScrumWorks is originally desktop software and has to follow the rising trend of browser-based software. However, the web client is not just a viewer of data, but one is also able to create new tasks and even stories. That is a bit confusing because it does not become clear why the desktop client is needed to create stories and tasks when the web client is actually capable of doing so, too.

Feature-wise, ScrumWorks cannot compete with JIRA. However, this does not mean that the task board is worse – in fact the features that are important are implemented: displaying the assignee name for each task, support of drag & drop or displaying a Burn-Down-Chart on the same page, only to name a few. Points of criticism are that comments on tasks are not displayed on screen and that stories cannot be estimated using story points.

The handling of tasks works the way it should. Yet, when all tasks are placed in the "Done" column it implicates that the story is finished as well. To do so, one has to click on the story and set it manually to "Done". After that, a green check mark indicates that the story is finished. How to handle finishing stories is a matter of choice. On the one hand, one could argue that if all tasks are finished, the story should also be finished. On the other hand, within Scrum there is a Product Owner, who has to check all finished stories in order to determine whether the story satisfies his or her needs of whether the team has to correct something. In this case, the Product Owner would be the one who marks the story as "Done" to give the team an indication that they have done a good job.

A story, unlike a task, can have an attachment in form of a file and, quite unique, one can calculate certain business weights in form of return on investment (ROI) or the resource-based view (rBV). As a result, this is a huge advantage for Product Owners that rely on this data.

Besides the additional "Impeded" column, the task board cannot be customized in terms of columns. Like JIRA's rapid board, ScrumWorks is missing a full screen mode and is thus not suitable for big screens.

Motivation

Social network elements are available within ScrumWorks, but are less distinctive than in JIRA. The user can comment on stories and tasks, but not on transitions, e.g. when a task moves from "To Do" to "In Progress". Comments are only visible when they belong to a story; if they belong to a task, the user has to click on the task's details to see whether there are any comments or not. The dashboard is limited, it only shows the tasks that are assigned to the user, whereas JIRA also gives information about the whole project activity stream and allows creating own filters.

ScrumWorks does not encounter wallboard blindness; there are no acoustic or visual signs when a task board is changed and there is no possibility to add a virtual dot to a task. But in contrast to JIRA, the task board gets refreshed every 30 seconds, thus if a team member makes a change, that change is visible for the other team members within half a minute (or as soon as one refreshes the page automatically, of course).

Costs

Linked to the SLA of CollabNet, three different prices are available if one wants to run ScrumWorks on a dedicated server and one additional price for the hosted option. As with Atlassian, there is no discount available for casual users.

Software type	Small Team	Medium Team	Large Team
Hosted (per month)	1.040 $	3.380 $	18.200 $
Download (Silver) (per year)	11.040 $	35.880 $	193.200 $
Download (Gold) (per year)	11.520 $	37.440 $	201.600 $
Download (Platinum) (per year)	12.000 $	39.000 $	210.000 $

Table 11: ScrumWorks Pro costs

Compared to JIRA, CollabNet is much more expensive than a comparable license type of JIRA.

Conclusion

The approach of CollabNet ScrumWorks Pro, namely using a local software client to administrate the Scrum teams, the product backlog items, the releases, etc. is outdated. All other tools have shown that it is possible to do everything within the browser. Still, the web client interface that is accessible by using a browser features a task board that works without distracting the user with special features. The

huge plus of ScrumWorks Pro is its quality in terms of support. The availability of the support is unmatched and also the response time is better than the ones from other vendors. In combination with the second lowest costs (still much more than JIRA), it is an attractive solution for Scrum teams that don't need any special functionality.

4.3.3 Microsoft Visual Studio 2010 Team Foundation Server / Urban Turtle

Team Foundation Server is a Microsoft product, which falls in its famous Visual Studio category – an integrated development environment (IDE) tool, originally made for software engineers. As time goes by, Microsoft continued to extend the Visual Studio line - the Team Foundation Server is the ALM product of Microsoft. Team Foundation Server is browser-based but unfortunately, Microsoft does not offer a hosted evaluation. However, it is still subject of this evaluation. The reason for that is the fact, that if one wants to use Scrum in combination with Team Foundation Server, one is forced to use Urban Turtle, which is basically a Scrum extension for the Microsoft product. Luckily, Urban Turtle does offer a web-based evaluation. Thus strictly seen, this section will not evaluate Microsoft Team Server. Instead, Urban Turtle will be subject of this evaluation.

Usability

If one has used a web-based Microsoft product like Outlook Web-Access before, the user interface feels familiar. The dashboard of Urban Turtle is a bit modest, yet one has to keep in mind that Urban Turtle is not in charge for the dashboard. Instead, Urban Turtle adds two more tabs to Team Foundation Server: the "Planning Board" and the "Task board".

Still, the dashboard displays the recent activity, comparable to the activity stream in JIRA but without additional information like comments. Above the comments, a summary is given about the tasks the user is assigned to, e.g. how many of them are "In Progress" or "Done". One is also able to create a new backlog item out of the dashboard – this might be handy if something came up to one's mind and one wants to capture that thought quickly. What is known as filters in JIRA is named queries in Urban Turtle. These queries can be customized so that the dashboard in turn can be changed according to one's personal preferences.

The task board itself suffers from information overload, see Figure 13.

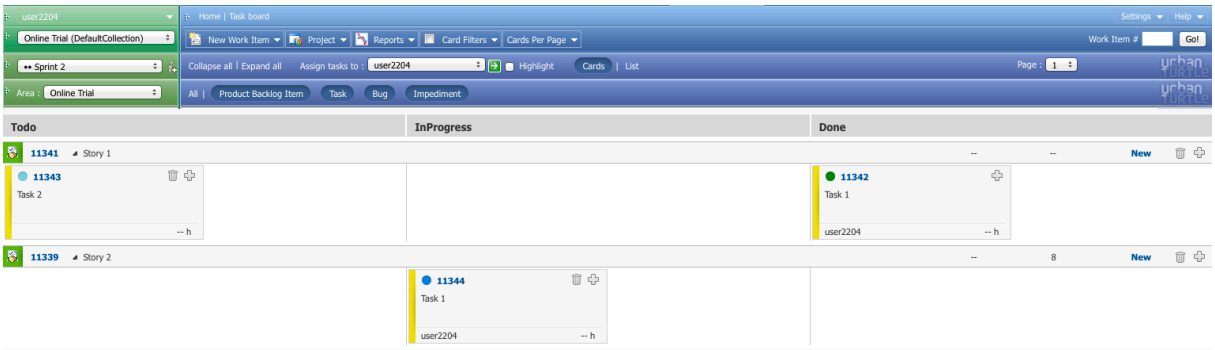

Figure 13: Urban Turtle task board

Although the font size of the columns is large, they are not visible at first because the user is distracted by the huge green/blue horizontal bar. As one can see, most of the space inside the blue bar is unused; therefore it would have been easy to put all items in one of those three sub bars in order to save vertical space. By not doing so, one is forced to scroll more often as soon as the task board is getting bigger. Not without a reason, the task board has a page indicator, which leads the user to the next task board page. This is in contrast to the requirement of a task board to see everything at a glance. However, Urban Turtle seems to be well aware of this fact as it offers alternatively a flat view, which saves a lot of screen space.

Moving the tasks is done by drag & drop and it works without any delay, in contrast to the other tools. The tasks show by default the right amount of information (the description and the assignee). Additionally, the default card view also shows the remaining hours per task; the list view does not. Like ScrumWorks, Urban Turtles also comes with alternative methods to indicate which tasks belong

to which user: one is able to choose a fellow team member, check the "Highlight" field and only the tasks that are assigned to that team member are shown, the other tasks are nearly invisible.

Context-based help functionality is not available, a link in the upper right corner leads to the documentation and support forums. In case of Urban-Turtle, context-based help is not really needed. The buttons and links are quite self-explanatory and work the way they should. Tasks can be created in two different ways: by clicking on the (+) sign at the right end of a story or by clicking on "New Work Item" in the blue bar above. The former way creates a task that belongs to the story, the latter creates a task that does not belong to any story; it makes sense to do so if one wants to create a planned task, which is a general task like "knowledge transfer" that is not related to any story in particular.

Resources

When it comes to browser requirements, Urban Turtle needs roughly the same as the other tools[4], with the exception that also earlier versions of Microsoft Internet Explorer are supported, which should not be surprising as the Team Foundation Server is also a Microsoft product. The following browsers are thus supported:

- Microsoft Internet Explorer 6 and higher
- Mozilla Firefox 3.0 and higher
- Google Chrome 1.0 and higher

As one can see, Apple Safari is not listed in the supported browser. However, a quick test with Safari has not resulted in any errors, so that one can assume that Safari also works well.

A minimum display resolution is not stated, yet, the Urban Turtle site scales quite well so that also a resolution below XGA would be possible. Whether one wants to work with such a resolution is of course a different story.

Quality

Unfortunately, no statements regarding the service level agreement of Team Foundation Server or Urban Turtle were found.

Integration

If one relies on Microsoft products, integration is one of the strongest advantages of Urban Turtle in combination with Team Foundation Server. It offers a seamless integration and features like a dedicated section where one is able to start building the project from the source code the team is working on. Next to triggering the build system within the browser, one is also able to check on the source code itself, see recent changes and different versions.

Functionality

Urban Turtle has quite some interesting functionalities to offer. First, the basics: stories can be estimated using story points and tasks can be estimated on an hourly basis. The task board can be configured in such a way that it only shows the stories, the tasks, the bugs, the impediments or any combination of the four.

A feature that is used quite often is the ability to display planned tasks or bugs in a special area of the task board. Within Urban Turtle, this area is called "Other Work Items". When adding a new item, one is able to choose between a story, an impediment and a bug. Bugs, impediments and tasks without a corresponding story are shown in the "Other Work Items" and on top of that, bugs have a dedicated swim lane that is colored in red and has a warning sign that states, "These work items need your attention" which makes sure that team members will not forget to fix these bugs.

Like with ScrumWorks, one is able to switch between the different sprints and their corresponding releases. The task board is then refreshed to show the matching stories, tasks and other cards like bugs & impediments. Every item type has a dedicated color stripe on the left side for a better recognition, e.g. stories have a green stripe, tasks a yellow one, bugs a red one etc. Next to that, every item also has

[4] http://msdn.microsoft.com/library/ee523998.aspx#Browsers (accessed at 01.10.2012)

a colored circle in the upper left corner that indicates the state of the item, e.g. if a task is “In Progress” the circle is blue, if a task is in “Done”, the circle is green. Although distinct colors are useful, Urban Turtle is currently not suitable for big screen usage, as a full screen mode is missing. In combination with the large, blue header, one has to scroll down as soon as one sprint has more than 5 stories, which is in a real world setting often the case.

A task cannot be moved from “To Do” to “Done” without moving it into “In Progress” first. Again, it is arguable whether this is an advantage or a disadvantage. On the one hand, a task should be processed in such a way that it matches the Scrum flow. On the other hand, there are situations where a direct transfer is a valid option, for example if the team wants it that way.

Each story has its own swim lane and at the right side of that lane, one can click on a link called “Committed”. The functionality is similar to the check mark in ScrumWorks; it is a tool for the product owner to let the team know if he or she accepts a finished story.

Although the user interface uses AJAX, the performance experience was mixed. Sometimes it took several seconds to do the same action that normally is finished within a fraction of a second. The reason for this unstable behavior might be due to the fact that everyone who wants to evaluate Urban Turtle seems to use the same installation; at least this would explain the hundreds of users available.

Motivation

Motivation is the weak spot of Urban Turtle. For starters, Urban Turtle does not feature a dedicated commenting system. One is able to click on each task and on each story to enter a description, but that is only a workaround at its best.

Visual and acoustic signs are not available and the task board does not automatically refresh as soon as it gets changed. Again, also Urban Turtle does not offer a possibility to add a dot to a task in order to identify a possible impediment.

Costs

The licensing model of Urban Turtle is easy to understand - in contrast to the one of Microsoft. Neither Urban Turtle nor Microsoft offers a hosted option, so one is forced to setup a dedicated server. However, in the context of Team Foundation Server, it would not make sense to use a hosted variant, as the team would not be able to build their source code within the browser, since companies are not inclined to put their source code on external servers. Again, casual users do not get a special license; they are treated as normal users.

Software type	Small Team	Medium Team	Large Team
TFS + Urban Turtle (per year)	14.010 $	40.290 $	200.000 $

Table 12: Team Foundation Server & Urban Turtle costs

The prices are comparable to the ones of ScrumWorks, although it remains unclear which service level agreements are offered by Microsoft / Urban Turtle.

Conclusion

The Urban Turtle extension adds Scrum functionality to Microsoft’s Team Foundation Server, in essence by creating a task board and a product backlog. Feature-wise, Urban Turtle has a lot to offer, but in terms of usability, Urban Turtle scores quite low. One reason is that the header bar is the largest of all the evaluated tools and therefore consumes much screen space, which could have been used to display more task board items. Next to the above-mentioned amount of features, another strong point for Urban Turtle is the perfect integration into the Microsoft environment. If the company is using Microsoft products, this integration is certainly a convincing argument to use Urban Turtle.

4.3.4 Rally Software

Rally is a Scrum tool made by a company called Rally Software. According to Forrester Research, it offers the best strategy and has the best current offering (West & Hammond, 2010). Indeed, customer feedback is very positive (Scharff, 2011).

Usability

After logging in, Rally offers the user the most-detailed dashboard of all tools that are part of this evaluation. The dashboard comprises of eight different sources of information:

- Iteration summary, which states the amount of days left before the current sprint is finished, the percentage of the accepted work, the amount of active defects and the percentage of tests that are passing.
- Iteration Burnup, which is useful if new items are added during the sprint, as the Burn-Down-Chart does not visualize this.
- Iteration Burndown, which provides the Burn-Down-Chart for the current iteration.
- Recent activity, which is comparable to the activity stream of JIRA.
- Blocked Work, a list of user stories that cannot be worked on due to an external problem.
- My Tasks, the tasks that one is assigned to.
- My Defects, the defects that one is assigned to.
- Recent Retrospective, which lists the result of the last retrospective to remind the team.

Although the amount of items that provide the user with information is considerably more than the ones of the other tools, the dashboard does not feel overloaded. The most important parts of the dashboard (Iteration Summary & Iteration Burndown) are also larger than the rest. Each item (called "App" in Rally) can be configured or removed from the dashboard.

The first thing that catches the attention of the user when opening the task board is the fact that it only consumes the upper half of the screen whereas the bottom half has plenty of white space. Fortunately, one can enlarge the task board so that it also can make good use of the available space.

After having done so, the second thing that draws the attention of the user is the fact that Rally uses a large font size for the description of the stories and tasks. As a result, the readability is very good, also from a distance. The disadvantage of using a large font size is that users have to scroll to see all stories and their corresponding tasks. A list-based view, like the one of Urban Turtle, would be a nice addition.

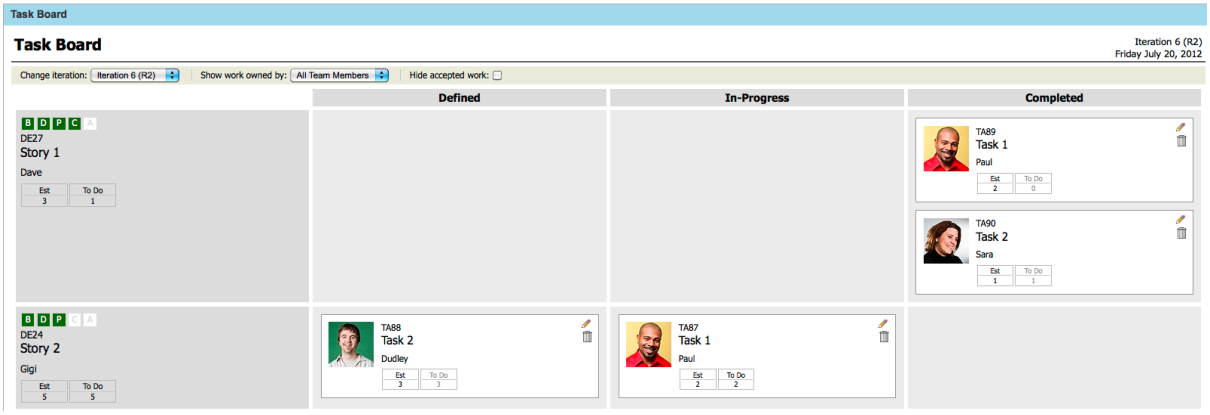

Figure 14: Rally task board

The task board visualizes the most important data, e.g. the description of the tasks and stories, as well as the assignee for each task. Interestingly, next to displaying the assignee, Rally also shows an avatar of the corresponding user. The advantage of doing so is that team members do not necessary have to use a portrait photo, but can choose to use any picture they want.

Each task has also two icons, one shows a pen and the other one shows a recycle bin. By clicking on the pen, the user is able to edit the task. Yet, the edit possibilities are limited to changing the estimated / remaining hours, the assignee and the state. The latter one offers a quite unique feature: by clicking

on the corresponding state icon, one is able to mark the task as blocked, indicated by a red "STOP" sign. On the task board, such a task gets a red boarder and also shows the red sign. Team members are now able to see any impediments that prevent them from finishing their tasks immediately.

All tasks list the amount of hours that are estimated and the amount of hours that are left before the task should be completed. The stories show the same set of information, as it sums up all values of all corresponding tasks. However, one is not able to add a new task when viewing the task board; if an extra task is needed it has to be added using the so-called "Iteration Status" page.

Although Rally includes a large amount of keyboard shortcuts, they are unfortunately not available when working with the task board. Tasks can be moved by using drag & drop, context-based help is also not available. However, it is not needed, as most of the functions are self-explanatory.

Resources

The resources that Rally requires are comparable to the ones of the other tools. Yet, Rally states that only the latest two versions of each browser are supported[5], thus only the following browsers are supported:

- Microsoft Internet Explorer 8 & 9 (Rally states that they provide limited support for Internet Explorer 7)
- Mozilla Firefox 13 & 14
- Apple Safari 4 & 5
- Google Chrome 19 & 20

So far, Rally has the highest requirements of all tested tools. Still, browsers are updated frequently, as most fixes are security related. As a result, the IT department of a company has to be up to date when it comes to browsers to avoid possible security breaches.

Rally does not state any display resolution requirements, however the monitor should have XGA resolution in order to avoid unreadable task descriptions.

Quality

In terms of service level agreements, Rally does not offer different levels for their products; the service level is thus always the same.

Support type	Enterprise Edition	Ultimate Edition
Phone support	13 hours / Mo. – Fr.	13 hours / Mo. – Fr.
Response time	max. 24 hours	max. 24 hours

Table 13: Rally Service Level Agreements

When comparing the availability of the phone support, Rally is average – it is above the service level of Atlassian and below the one of CollabNet.

Integration

Integration-wise, Rally Software's approach is similar to the one of VersionOne. The company offers a great amount of so-called "Rally Integrations" which are connectors to other products from third party vendors. The type of connectors range from CRM systems over IDE's to build systems or wikis.

Next to connectors, Rally Software also offers small apps, which extend the functionality of Rally. Users can choose to install different types of boards and quality assurance tools only to name a few possibilities.

Rally Software encourages developers to create Apps for their Rally application by providing an SDK, as well as a web services API.

[5] http://www.rallydev.com/help/supported-web-browsers (accessed at 01.10.2012)

Functionality

The task board offers various possibilities in terms of viewing perspectives: one can see the current iteration, but one is also able to select from a list of previous iterations. By default, the task board shows the work of all team members, but it is also possible to only show the tasks of a single user. As a last viewing option, tasks and stories that have been accepted by the product owner can be hidden from the task board.

As for the tasks, each story has multiple icons that indicate its current state. Five states are available: "Backlog", "Defined" (same as "To Do"), "In-Progress", "Completed" & "Accepted". As soon as a state is completed, it is colored in green. Thus, if all tasks are "In Progress", the "Backlog" icon and the "Defined" icon are green. When a team member blocks a certain task as described above, the icon of the current state gets red. Of course, one is able to see the current state of a story by looking on the complete task board, but by showing the state with the help of icons, one is able to browse through the whole list of stories to grasp the progress.

Estimation-wise, Rally uses the story point format for stories and hours for tasks. Like with VersionOne or ScrumWorks, stories are represented using swim lanes, which makes it easier to identify which task belongs to which story.

Although the user guide claims that a full screen mode is available, it does not work for the task board. However, due to the large font size of the descriptions, this is not a necessity when using Rally on a big HDTV screen.

Motivation

Rally comes with a commenting functionality; however, users cannot place a comment when they are working with the task board. Instead, they have to go to the "Iteration Status" page if they want to add a comment. Comments can be added to all available items, and they are immediately displayed on the recent activity app, which can be seen when opening the dashboard.

Using the dashboard motivates the user by providing information that encourages the team to finish their sprint successfully. Rally achieves this by giving the above explained iteration summary and by listing the results of the recent retrospective.

Comparable to the other evaluated tools, Rally does not trigger the user visually or acoustically as soon as a change occurs and offers no functionality for adding a virtual dot. Next to that, the task board does not automatically refresh when someone has made a change. Users have to refresh the task board page manually.

Costs

Rally features three editions, a community edition, an enterprise edition and an unlimited edition. The community edition is free, but is restricted to one project with a maximum of ten users. The remaining two editions are hosted only and differ in functionality – Rally is not available for download.

Software type	Small Team	Medium Team	Large Team
Enterprise Edition (per month)	1.400 $	4.550 $	24.500 $
Ultimate Edition (per month)	1.960 $	6.370 $	34.300 $

Table 14: Rally costs

In terms of cost, Rally is a bit more expensive than the other tools that are subject of this evaluation.

Conclusion

Rally is the most mature Scrum tool compared to the other tools that were evaluated. It offers a lot of useful features; Rally Software has obviously spent some time to bring them to perfection. The task board of Rally is the only one that uses a big font size as well as showing a user avatar next to its name for each task. Besides the task board, Rally Software also offers the best dashboard that provides lots

of useful information when logging in into the system. Yet, these features come at a cost: Rally is the most expensive tool that was evaluated – in terms of hosted Scrum tools, as Rally does not offer a version that one is able to install on a dedicated, local server.

4.3.5 VersionOne

VersionOne is an agile project management tool from the correspondent vendor. Next to Rally and ScrumWorks Pro, VersionOne was one of the first agile tools on the market.

Usability

VersionOne looks different when comparing its product with other tools from vendors like the ones described previously. All user interface elements are customized and have their unique look & feel which gives the tool a professional touch.

The dashboard of VersionOne is spread over five sub pages, where each sub page expresses a particular piece of information:

- My Work, which lists all stories and tasks a team member is assigned to including the state and the estimate of the task.
- My Dashboard, which gives information regarding the status of each project by displaying a progress bar that indicates how many story points are already done and how many are remaining.
- My Inbox, which is an embedded messaging system within VersionOne.
- Subscriptions, which lists all items a team member is subscribed to.
- Filters, which are roughly the same as the filters in JIRA; they are giving the user the possibility to define which data he or she wants to see.

Although the dashboard is very detailed, it does not give the user all information at once like it is the case in JIRA. Instead, the user has to click on each sub page to get the necessary information, which is time consuming.

The task board itself looks somewhat different in respect to the known ones. It has five columns, one for the stories and three for the different task states. The last column is titled “Summary” and is used to display the current test results by indicating the amount of work that needs to be completed as well as a progress bar. On larger screens, this “Summary” column scales with the screen resolution, which means that it can happen that this column takes up to 50% of the whole task board, as the story column and the task columns do not scale.

Yet, VersionOne manages to present a task board that is designed in such a way that its readability remains good. Tasks have different colors but unlike ScrumWorks, the whole task item is colored, not just the border around the item, see Figure 15.

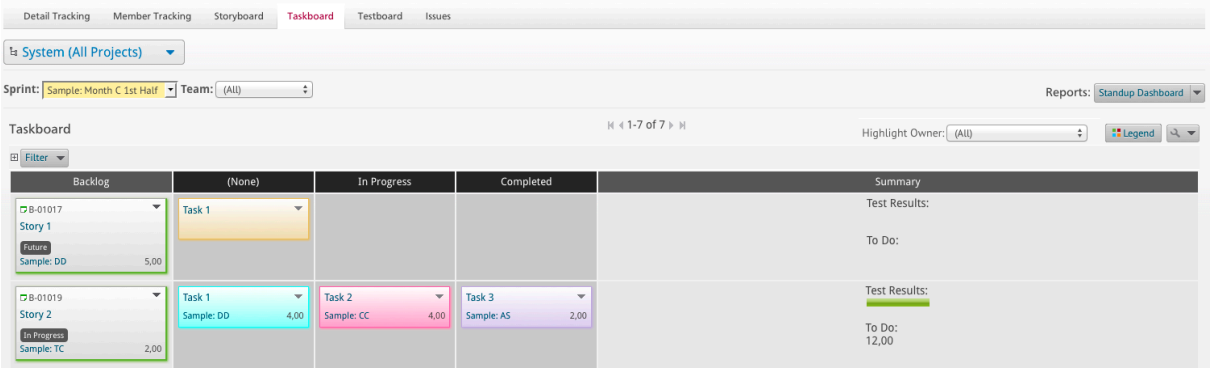

Figure 15: VersionOne task board

Another difference is the way tasks are colored: colors do not indicate which task is assigned to whom, instead, they indicate the type of task which can be “None” (yellow), “Design” (purple), “Code” (blue) and “Admin” (red). Stories are not completely colored like tasks; instead they have a colored border that indicates to which team they belong. Finding out what the different colors express

was not trivial. Since there is no context-based help available and also the user guide does not state the meaning behind the different colors, much trial and error was necessary.

The context menu of a story and the one of a task can be reached by clicking on the small arrow in the upper right corner of each item. Story-wise, it enables the user to edit or to add new tasks. If one clicks on the arrow of a task, one is able to assign the task to someone.

Comparable to ScrumWorks, the user can access the Burn-Down-Chart when working on the task board. Yet, instead of offering a clickable thumbnail of the Burn-Down-Chart, VersionOne includes a dropdown menu where the Burn-Down-Chart can be selected. Both tools share the ability that the Burn-Down-Chart can be maximized.

Resources

The browser requirements of VersionOne are stricter than the ones of other tools (Rally is an exception). In general, the browsers have to be more recent[6]:

- Microsoft Internet Explorer 8 and higher
- Mozilla Firefox 12 and higher
- Apple Safari 5.1 and higher
- Google Chrome 19 and higher

VersionOne does also specify minimum requirements regarding the screen resolution: displays should have at least a XGA resolution; some pages may require the higher SXGA (1280 * 1024) resolution. Still, most of the computers nowadays will have a display that supports SXGA and up.

Quality

VersionOne does not offer different service level agreements for its product – the variants differ in functionality, but not service-wise. Therefore, the SLAs are the same for all product variants see Table 15.

Support type	Download (Enterprise)	Hosted (Enterprise)	Download (Ultimate)	Hosted (Ultimate)
Phone support	10 hours / Mo. – Fr.	10 hours / Mo. – Fr.	10 hours / Mo. – Fr.	10 hours / Mo. – Fr.
Response time	max. 24 hours	max. 24 hours	max. 24 hours	max. 24 hours

Table 15: VersionOne Service Level Agreements

Compared to other tools and based on the costs of VersionOne, the service level is rather poor. Where other tools offer a 24-hour support 7 days a week, VersionOne does not exceed 10 hours a day at a maximum of 5 days a week.

Integration

VersionOne is the only tool of its vendor, yet it provides an agile application life cycle management platform. To connect with other products from third party vendors, it offers several connectors[7] that can be used to interact with these tools. Next to such connectors, developers can also make use of the available API and of two software development kits (SDK).

Functionality

VersionOne is not what one would call the user-friendliest tool, however it has an interesting set of features. One of these features in VersionOne is that not only to a task board, also a story board and a test board exist.

A story board is the same as a task board, but more high level. It is used for release planning where usually a bunch of stories are linked to a bigger user story, which is called “epic”. Such an epic could describe a whole component that is then divided into stories, which in turn are split into tasks.

[6] http://community.versionone.com/KnowledgeBase/FAQs/Q10042.aspx (accessed at 01.10.2012)

[7] http://www.versionone.com/Platform/Integrations.asp (accessed at 01.10.2012)

The test board shows the corresponding tests for each story. The status of the test board feeds the "Summary" column of the task board, which indicates the state of the tests. Whether one needs a dedicated test board is project-specific and / or a matter of personal preference. Commonly, when a task describes source code that has to be implemented, developers automatically add some test cases to ensure that the code works the way it should. However, if a project requires extensive testing, for example additional tests that are not task specific, such a test board makes sense. Yet, the agile mindset does not want to separate testers from developers. Instead, developers should also have testing knowledge, so that both groups do not have to be separated. In ideal circumstances testers should also program and programmers should also test.

Stories and tasks have many more features that go beyond the standard functionality, e.g. assigning a task to a user or adding a task to a story.

Story-wise, one is able to track the state by receiving event-based notifications. Next to tasks, also tests can be added; a story can be converted into a bug (called "Defect" in VersionOne). A story can also have a certain complexity value, a priority and a type (e.g. enhancement or new feature), a product owner as well as a team can be assigned and the sprint in which the story should be finished.

Task-wise, one is also able to track its state next to assigning multiple persons to the same task which is quite handy if the team relies on pair programming, which means that two developers use one screen, one keyboard and one mouse. Like with stories, tasks can have a certain type and of course they can be estimated.

Yet another feature of VersionOne is the ability to filter out certain elements on the task board. Options are endless, ranging from only displaying tasks that are assigned to a specific user extending to tasks with a certain priority or a certain product owner.

Last but certainly not least, the feature of showing what is called a "standup dashboard" is perfectly suited for the daily scrum standup meeting. This standup dashboard shows the task board, but on the topside it also displays the Burn-Down-Chart in a reasonable size as well as the cumulative flow, which is sorted by the status of the tasks. By embedding both diagrams, the team perfectly knows about the current overall status of its work.

Motivation

In terms of motivation, VersionOne offers a feature-rich tracking system that enables users to track the status of stories, tasks and other items. Next to tracking items, users are also able to comment on them. On top of that, VersionOne offers an own messaging system that enables communication between the team members.

Regarding the task board, VersionOne offers the above-mentioned standup dashboard, which supports the team when performing their daily scrum routine. However, neither automatic refresh of the task board nor the possibility to add a dot to a task is available.

A dedicated full screen mode is lacking. Similar to the other tools, VersionOne does not offer any visual or acoustic triggers to indicate that a change has occurred.

Costs

VersionOne offers four different versions of its product that differ in functionality. The two cheapest options are not subject of this review as they are limited to 10 respectively 20 users. The remaining two versions can be hosted on the servers of VersionOne as well as downloaded and installed on a dedicated, local server.

Software type	Small Team	Medium Team	Large Team
Enterprise (hosted, per month)	1.160 $	3.770 $	20.300 $
Enterprise (download)	23.800 $	77.350 $	416.500 $

Software type	Small Team	Medium Team	Large Team
Ultimate (hosted, per month)	1.560 $	5.070 $	27.300 $
Ultimate (download)	31.800 $	103.350 $	556.500 $

Table 16: VersionOne costs

VersionOne is, compared to other tool vendors, more expensive. The hosted option is in terms of pricing comparable to ScrumWorks Pro. The download variants are the most expensive ones, however their prices are not per year. Yet, one has to add 119$ per user/year for the Enterprise edition to the calculation for annual support & maintenance. For the Ultimate Edition it is 159% per user/year.

Conclusion

VersionOne does not perform better in any aspect than the other tested tools. The software is very mature and offers a lot of features; especially the story & test board are an interesting option next to the information dashboard, which supports the daily scrum routine perfectly by providing additional data at a glance. However, the tool is not very user friendly – a reasonable amount of features is not self-explanatory. Like with ScrumWorks Pro, the font size of the task board items is very small and stories cannot be estimated by using the Story Points metric. Finally, the tool is the most expensive one when a company wants to use it on its own, dedicated server. The hosted option is the second most expensive one; only Rally is even more expensive.

4.4 Software Tools Overview

A total of five Scrum tools were evaluated that feature a task board. Criteria for evaluating those tools were defined in section 4.2. Table 17 summarizes the outcomes of these criteria for all five tools. The table uses different kinds of ratings:

- A ✔ or a ✗ to indicate whether a certain aspect is available or not.
- N/A if an aspect is not applicable.
- ++ (very good), + (good, o (neutral), - (bad) & -- (very bad) to indicate the level of a certain aspect.
- A number to indicate the amount of an aspect.

A green colored cell means that a tool is better than the remaining four tools in terms of that very aspect. If two tools share the same mark, the cells is not colored at all.

Criterion	JIRA	ScrumWorks Pro	TFS / Urban Turtle	Rally	VersionOne
Usability					
Dashboard	✔	✔	✔	✔	✔
Number of Dashboard information sources	4	1	2	8	5
Tasks show the user / avatar / description	✗/✔/✔	✔/✗/✔	✔/✗/✔	✔/✔/✔	✔/✗/✔
Drag & Drop	✔	✔	✔	✔	✔
Keyboard Shortcuts	✔	✗	✗	✗	✗
Context-based help	✗	✗	✗	✗	✗
Creating tasks	✔	✔	✔	✗	✔
Intuitive use	+	o	+	+	-

Criterion	JIRA	ScrumWorks Pro	TFS / Urban Turtle	Rally	VersionOne
Resources					
Minimum resolution	XGA	XGA	< XGA	XGA	SXGA
Browser requirements	+	+	++	-	o
Quality					
Phone support	-	+	N/A	o	o
Response time	+	+	+	+	+
Integration					
Number of products in vendors portfolio	13	4	3	1	1
API	✔	✔	✔	✔	✔
Connectors for third party products	✔	✔	✔	✔	✔
Number of available plugins	400+	N/A	N/A	89	N/A
Functionality					
Full screen mode	✗	✗	✗	✔	✔
Columns customizable	✔	✔	✗	✗	✗
Font size	-	--	-	+	--
Story estimation (Storypoints/Time)	✔/✔	✗/✔	✔/✗	✔/✗	✗/✔
Task estimation (Storypoints/Time)	✗/✔	✗/✔	✗/✔	✗/✔	✗/✔
Motivation					
Comments	✔	✔	✔	✔	✔
Task dots	✗	✗	✗	✗	✗
Automatic page refresh	✗	✔	✗	✗	✗
Acoustical / visual trigger	✗/✗	✗/✗	✗/✗	✗/✗	✗/✗
Costs					
Hosted	++	+	N/A	-	o
Download	++	+	o	N/A	-

Table 17: Scrum Tools Summary

Counting the amount of green cells produces the following outcome:

	JIRA	ScrumWorks Pro	TFS / Urban Turtle	Rally	VersionOne
Green cells	6	2	2	3	0

Table 18: Scrum Tools Results

However, this outcome does not mean that JIRA is the best choice for all companies that are interested in using a Scrum software tool. Instead, each tool has certain aspects in which it outperforms the others:

- JIRA is the cheapest option and offers a lot of functions, but is not yet as mature as the other tools.
- ScrumWorks Pro uses an outdated approach by splitting its tool in two client applications, but leads quality-wise.
- Microsoft Team Foundation Server with Urban Turtle is frugal when it comes to resources and offers a great integration (however Microsoft-only), but scores quite low when it comes to usability.
- Rally has the greatest maturity and a lot of well-thought features. It is the only tool that uses a reasonable font size for its task board. However, it is also the most expensive solution for hosted tools.
- VersionOne also offers a lot in terms of functionality, but is not as user-friendly as Rally. The information dashboard is a unique feature but comes at a cost, as VersionOne is the most expensive solution for downloadable tools.

Although all five tools are capable of representing a Scrum task board electronically, this tool evaluation backs the paper vs. software evaluation in section 3.3, namely that all software-based task boards score low when it comes to accessibility and motivation. Not all tool vendors have grasped these aspects: only two out of five vendors have implemented a full screen mode and only one vendor uses a big font size (although this font size is still too small to support good readability). In combination with big screens that lack any way of inputting data, software-based task boards cannot compete with their paper-based pendants as they are reduced to information panels that are ignored after a short time. Acoustic or visual triggers could create task board awareness; unfortunately none of the five vendors has implemented anything at all that comes even remotely close. On top of that, even important basic functionality like automatically refreshing the screen when changes are made is not implemented in any of the tools. Instead, only ScrumWorks Pro at least refreshes the task board every 30 seconds.

However, the evaluated tools also back up the higher score of software-based task boards when it comes to archiving. All five tools automatically archive sprint data, three of the tools allow checking back on previous sprint data when viewing the task board, and the remaining two have a dedicated function to do so. The evaluation has also shown that tool vendors make extensive use of connecting their product with software from other vendors to support company workflows. All vendors have made sure that the products within their portfolio are able to exchange data with each other; each tool features an API as well as various connectors, some of the tools also feature plugins or apps that allow external developers to create additional functionality to improve these tools. By doing so, tool vendors try to compensate missing features that are currently not available within software-based task boards.

4.5 Wrap Up

The purpose of this chapter was to reach the second part of the first half of the research goal by giving an answer to research question 4.

In section 4.1, criteria for selecting Scrum software tools that form the basis for this evaluation were defined, since a great amount of Scrum tools exist – too much to evaluate them all within the scope of this master thesis project.

Section 4.2 reused some criteria from the software versus paper analysis in chapter 3, next to defining new criteria that were needed to perform a dedicated software versus software evaluation.

The actual evaluation of Scrum tools that are able to represent a task board electronically was performed in section 4.3.

Finally, section 4.4 concluded the results and showed that the evaluation backed up the analysis findings in chapter 3 next to the fact that the representation of a software-based task board is in some aspects limited and cannot outperform its paper-based pendant.

By answering research question 4, the first half of the research goal, namely the task board analysis has been completed. The following chapter will solve the second half of the research goal by defining company guidelines.

5 Company Characteristics & Guidelines

The prerequisites for providing an answer to research question 5 (see section 1.7.1), namely to define company characteristics and to create company guidelines for selecting the appropriate task board variant have been fulfilled by indicating what a Scrum task board exactly is about (prerequisite one, chapter 2) and by performing the first part of the research goal, the task board analysis (prerequisite two, chapter 3 and 4).

Now that these prerequisites are set, this chapter will give an answer to research question five (defining Scrum task board-related company characteristics and guidelines). By doing so, the second half of the research goal (company guidelines) will be reached.

Before these characteristics can be identified, again certain prerequisites have to be fulfilled.

The first prerequisite is to identify usage scenarios of Scrum task boards. Knowing how task boards are used is helpful to derive company characteristics. This is done in section 5.1 by consulting internal experts next to using the outcomes of the task board-related literature review.

The second prerequisite for defining company characteristics is to visit companies to research how task boards are used within a real life setting. The outcomes of this field research are described in section 5.2.

As soon as both prerequisites are handled in section 5.1 and section 5.2, section 5.3 will identify company characteristics by using the outcomes of the task board-related literature review next to internal and external expert opinions. These characteristics are then validated by using the outcomes of the field research performed in section 5.2.

By using the relevant company characteristics identified in section 5.3 and by using the outcomes of the first part of the research goal, namely the task board analysis (chapter 2, 3 and 4), the actual company guidelines are created and expressed by using a decision graph in section 5.4. This section will make good use of the findings of this master thesis project to validate these guidelines.

5.1 Scenarios

The first step towards the direction of company characteristics is to identify the possible usage scenarios of a task board, as indicated above. The scenarios described in this section help to perform the field research by creating categories in which the evaluated companies can be put, see the next section.

A task board can have different forms based on its usage context. The original form of the task board was based on plain paper cards for the stories and Post-Its for the tasks – which were put on a large sheet of paper, on a white board or on the wall (Cockburn, p. 77-78, 2001). This method was suitable since in the past teams were mostly collocated.

However, things have changed rapidly and today it is not unusual that teams are spread over different locations (Herbsleb & Moitra, 2001). As a result, the companies have to think about how every team member can get access to the task board. Software-based task boards are one option, but also other options exist which are subject of this section.

Table 19 gives a brief overview about the key characteristics of each task board usage scenario.

	Paper	Paper & Audio / Photo / Video	Software	Software with big screens	Paper & Software in Coexistence
Task board type	Paper	Paper	Software	Software	Paper & Software
Target team type	Collocated	Distributed	Distributed	Distributed	Distributed
Task board data synchronization	N/A	Manual	Automatic	Automatic	Automatic & Manual

Table 19: Task board usage scenarios

The scenarios are set up with the support of internal experts; they are also partly supported by literature, if available.

The following sections will describe the five identified usage scenarios in a detailed manner.

5.1.1 Paper

A recent survey showed that 26% of the companies still favor paper – which is more than any single tool has accomplished up until now (Azizyan, Magarian, & Kajko-Mattson, 2011). This is not without reason, as Scrum relies on face-to-face communications. Paper is not the most popular tool because it was the first tool available – instead, paper has some unique features that are hard to top by any other medium.

First, it is cheap. One is able to start introducing a task board by simply using a sheet of paper and some Post-Its, thus entry costs are low. Secondly, a paper-based task board is big; boards are usually several meters high and wide. As a result, text readability is good, even several meters away – of course that depends on the readability of the person's handwriting. The task board is very important to Scrum as it is one of the most used artifacts, due to the fact that the team has to interact with the task board in order to update the status of the tasks they are working on. Visibility and visual feedback is thus very important (Berczuk, 2007).

However, some benefits come at a cost. It is easy to stick a paper-based task board onto the wall, but as soon as it is attached to that wall, the task board cannot be moved. Some teams like to take their task board with them to the sprint planning 1 & sprint planning 2 – a meeting that is typically in some kind of meeting room. To avoid this issue, some teams pin their paper task board onto a bulletin board with wheels, which they can easily move to different locations. This mobility however comes at a cost, because the dimensions of those bulletin boards are limited which diminishes the benefit of good readability from a certain distance.

5.1.2 Paper & Audio/Photo/Video

The major issue with paper has already been described in section 2.6.1, namely that it is not suitable for distributed teams, e.g. teams that are located at different sites. Although a paper-based task board does not allow automatic synchronization of the tasks, manually updating the task boards is a valid option that is sometimes used by companies that want to use paper as a medium for distributed teams.

In this scenario, a normal paper-based task board is available at each location and all the task boards should show the same information regarding the status of all tasks. The actual synchronization has then to be performed by hand, for example by the ScrumMaster or, if a ScrumMaster is not available at the location, by a chosen representative. As a medium, the different locations can use a plain teleconferencing system, a digital camera or a webcam, since audio-conferencing systems have sufficient voice quality and are easy to setup (Berczuk, 2007).

Usually, the team-representatives of each location will have a telephone conference at the end of the day in which they will tell each other which tasks have been updated, so that the counterpart can update the tasks on the local task board accordingly and vice versa. Alternatively, each representative

could also take a photo of the task board and mail it to others in order to decrease the overhead of a telephone conference. Another option would be to track all the task boards constantly by using a webcam. However, this option is often not available due to slow Internet connections or due to company policies, as will be described later on.

5.1.3 Software

By using dedicated Scrum software tools, companies are able to swap their paper-based task board with its software-based pendant. For distributed teams, using a software-based task board is quite common as software enables synchronization between different locations within a few seconds. Same as with paper, software-based task boards also have some unique features.

In the first place, software allows the archival of Scrum-related data (Perry, 2008). When using a paper-based task board, at the end of each sprint, all the task and story cards are thrown away – although, they are stored in boxes sometimes. Either way, it is impossible to reconstruct what has happened in past sprints. Software allows looking into the past; the sprint data are saved and everyone interested in this data is able to see at a glance what has happened to each story / task at any given moment in time.

Secondly, software enables all kinds of reporting capabilities ranging from charts over statistics to anything, which could make sense for a company. Interested managers can track the status of their teams, but also the teams themselves benefit by getting a detailed and permanent updated view of their performance.

Thirdly, software makes time registration easier by tracking how much time a team member has needed to complete a task. Of course, this does not always work as expected as it can happen that a task takes more than one man-day to complete, resulting into inaccurate data. Therefore, most of the available Scrum software asks the team to estimate the time it will take to complete a task before the team start working on them.

5.1.4 Software with Big Screens

The disadvantage of using a software-based task board is a decrease in communication and discussion amongst team members (Paasivaara, Durasiewicz & Lassenius 2008, Perry 2008) as well as a decrease in accessibility (Perry, 2008). The decrease in communication and discussion possibilities lay in the nature of software: it is only visible on monitors. Therefore, instead of looking at a big task board, each team member will use his own machine to update his tasks instead of walking to the board, where he might run into other team members, which in turn could result into an increase in communication. Thus, by using a software-based task board, inevitably team members will get more isolated from each other. By a decrease in accessibility is meant that the progress of the team is not visible anymore to every person that is walking by the office. Once again, the team gets more isolated.

To counter this, big screens (often in the form of high-definition TV's) are introduced inside the team rooms. Instead of having a large sheet of paper sticking on the wall, a large monitor will inform everyone about the team's status. However, current Scrum software is not capable of providing a suitable full screen mode.

As a result, the font sizes are still far too small to enable reading text from more than two meters away, as has been shown in chapter 4. Also, big screens usually do not have any input possibility. If a team member wants to change the status of a task, he has to do it by using his own machine – a decrease in communication and discussion can thus not be avoided, as updating the status of the tasks does not take place at a central point.

5.1.5 Paper & Software in Coexistence

Instead of having to choose between paper and software, a scenario exists where both mediums are used simultaneously but in a somewhat different context (Perry, 2008).

It is possible to use a paper-based task board as primary source and a software-based task board as a backup (Berczuk, 2007). The idea is to combine the strengths of the paper-based task board with the ones of a software-based task board. For this scenario, the paper-based task board will be used the

same way as it would be used in the paper-only scenario, with the exception that every action on the paper will be transferred once a day into the software-based task board. By doing so, one can archive the sprints and create detailed reporting schemes without having to deal with issues a paper-only or a software-only solution would come with.

However, this scenario creates unwanted overhead for the person who has to administrate both systems. Errors are likely to occur more often as only one person is using the software-based task board instead of the whole team. Also, some of the benefits of a software-based task board like being able to comment on tasks or actions are unused; the functionality of such a task board is limited.

5.2 Company settings

Now that the usage scenarios of Scrum task boards are known, these scenarios can be used to categorize the companies that are subject of the field research described in this section. Next to categorizing the companies, this section will show how these scenarios are used at companies.

A total of four different companies were visited. The companies were randomly drawn from the client pool of bor!sgloger consulting. There was no reasoning behind choosing those four companies but that it were the companies that allowed a visit to perform the field research.

The companies were visited with the intention to research which type of task board they were using and how satisfied they were with their choice. This section will describe the company settings; the companies cannot be named for obvious confidentiality reasons. Unless mentioned otherwise, all teams are using Scrum.

For each of the four companies, some background information regarding the company is given, the Scrum teams are described and the task boards of the teams are evaluated.

An overview of the companies is given in section 5.2.5.

5.2.1 Company 1

Background

Although Company 1 is with around 600 employees a large company, only a minor part of the company is actual busy with research & development – the major departments of the company are manufacturing and maintaining the product-portfolio. Company 1 is located in Germany, but has multiple sites on all continents.

One year ago, Company 1 started a Scrum pilot project with one of their software teams to investigate whether Scrum makes sense and offers the proposed advantages. The pilot project period has recently been finished with results better than expected and Company 1 is currently eager to introduce Scrum to their other project teams.

Teams

The pilot project team consisted of nine project members – four of them (including the ScrumMaster and the Product Owner) were located in Germany, three of them were located in Vietnam and the remaining two team members were located in India. The team members in Germany supported the three distance issues (temporal, geographical & cultural) described in 3.2.10. The ScrumMaster stated that it was quite hard to find a point in time that was acceptable for all three team fractions: if the team members in Germany started working, the team members in Vietnam had already have their lunch. Next, both the sites in India and Vietnam did not allow the usage of webcams – for a quite banal reason: both sites did not just host the Scrum team of Company 1; instead hundreds of other people were working there as well – the result of outsourcing. The supervisors of these sites feared that if one team started to put up webcams all other teams should want them as well, which in turn would result into higher costs. The cultural distance issue was also encountered: the team members located in Germany stated that their colleagues in Vietnam and India were somewhat withdrawn, they rarely spoke when having a audioconference and they did not take the lead in a discussion. Still, they were hard workers and they delivered high quality software – as soon as they understood the requirements well. Since the project team itself was divided amongst three different countries, the team was classified as a distributed one.

Task board

Due to the fact that the team was located at three different sites, it was using a Scrum software tool (Urban Turtle, see section 4.3.3). Overall, the feedback regarding Urban Turtle tended to be a bit more negative than positive – however, some of the arguments could not be reproduced while evaluating Urban Turtle for this master thesis project; a possible reason could be that Company 1 was using an older version.

Yet, most of the arguments the team has stated correspond with the findings mentioned in section 4.3.3:

- The ability to build the code from out of the browser is seen as a major advantage, however the team does not use Microsoft Visual Studio and is thus not able to do so.
- The task board is shown on multiple pages due to the space-consuming layout.
- The blue header bar consumes too much unnecessary vertical space.
- The interface provides the team with enough information.
- The task board design matches the one of a paper-based task board quite well.

The last point shows that the team is not using a paper-based task board at each different location, as described in 5.1.5. Instead, the team is following the software-only approach as described in section 5.1.3. When asking why the team was not using a paper-based task board next to Urban Turtle, it argued that it did not like the limitations of paper and the resulting overhead of maintaining two task boards.

5.2.2 Company 2

Background

Company 2 is a software-only company. It has around 300 employees.

Like for Company 1, Company 2 features several departments next to the research & development department. Although located in Germany, the company has two different sites, where each site is responsible for a different set of products. Next to two sites A&B in Germany, Company 2 has a third site C in Romania.

Teams

Currently, the company has eight Scrum teams; six of them are located at site A, the remaining two are located at site B. The team size varies from a minimum of four people up to a maximum of six people.

Task board

Interestingly, the company uses both approaches: some teams from site A are supported by those from site C, thus using the distributed approach whereas site B is autonomous, thus collocated. The distributed teams (teams that have their team members located at site A and site B) are using an open-source Scrum software tool to work together. Yet, after using it for several months, the teams are not satisfied with the functionality of the tool and the company has chosen a replacement, namely Atlassian's JIRA. Reasons for switching the tool were:

- The current tool does not support archiving of sprint data.
- Adding tasks consumes too much time.
- The current tool does not offer a dashboard.
- Low usability of the current tool.

As one can see, the issues are mostly usability-related. This underlines once more the importance of the usability criterion (see 4.2.1). The distributed teams were supported by a big HDTV screen, which was merely used for the daily scrum standup meeting, as it did not offer any input possibility. Besides that meeting, the screen was not used at all – the font size was even smaller than the ones of the tested tools – therefore, readability was very low.

The autonomous two teams at site B do not use any Scrum software tool at all; instead, one of the teams uses a mixture of Scrum and another project management framework; the other team uses Scrum solely supported by paper. Although the first team is somewhat out of scope because of not using plain Scrum, its task board is comparable to a paper-based Scrum task board.

When asking the teams of all three sites, the distributed teams state that they were not satisfied with their task board (based on their current Scrum tool which has not been replaced when visiting the company) in contrast to the collocated team. The only problem with the latter one is that the task board with a size of 2 meters * 1,5 meters was too small – an issue that is currently worked on.

5.2.3 Company 3

Background

Company 3 has around 1500 employees.

Again, the research and development department where the Scrum teams are located is manageable; most of the employees are busy with supporting and maintaining the application.

Company 3 is located at one site only, there are no other (foreign) sites – thus teams are collocated. Therefore, it is no wonder that the team is using a paper-based task board. Yet, the scenario is not paper-only; instead, paper and software are in a peaceful coexistence as indicated by the scenario described in 5.1.5. At this company, there are currently three teams that are doing Scrum and a fourth team that just has started.

Teams

All teams at Company 3 have at least five members; the maximum is currently seven team members.

Task board

The teams are using a paper-based task board that is as long as the wall (around 4 meters) and 1,5 meter tall. Releases, sprints and stories are planned in Atlassian's JIRA. When a sprint starts, the stories are written on cards and put on the wall as well. When asking whether the team was satisfied with its setup the answer was an unanimous yes – the team was very pleased with its paper-based task board and the product owner had a powerful tool that supports him with writing and prioritizing the stories.

5.2.4 Company 4

Background

Company 4 has around 4000 employees.

Team

Company 4 has a nearly identical setting compared to Company 3 except that it has one more team and that its team size is averagely one employee less. Thus, Company 3 has five teams and with about five members per team.

Task board

Again, a paper-based task board is supported by a Scrum tool, which also happens to be Atlassian's JIRA. In contrast to Company 3, the task boards of Company 4 are pinned on bulletin boards that are portable – the team is thus able to take the task board to a meeting room, which is ideal when doing Sprint Planning 2 (see 2.4.3). However, the task boards suffer from space limitation – the portable bulletin boards are approximately 1,5meters * 1,5meters big. However, the teams have stated that they managed to get all tasks on the board till now.

Like for Company 3, the team was pleased with its task board. The Product Owners were mainly using the Scrum tool to organize & plan their upcoming stories, which they printed yet again on cards to show them to the teams.

5.2.5 Overview

Now that the four companies have been described, Table 20 summarizes the outcome of this field research.

	Company 1	Company 2	Company 3	Company 4
Number of employees	600	300	1500	4000
Number of Scrum teams	1	8	4	5
Scrum team size	9	4 - 6	5 - 7	4 - 6
Scrum tool	Urban Turtle	Ice Scrum	JIRA	JIRA
Task board usage scenario	Software	Software with big screens Paper	Paper & Software in Coexistence	Paper & Software in Coexistence

Table 20: Field research summary

This field research has shown several interesting aspects:

- The number of employees that use Scrum to do their work varies from 1% (Company 4) to 14% (Company 2) of the total number of employees. The relatively small amount of employees that use Scrum is explained by the fact that software engineering is only a small department at all four companies. Departments that are significantly larger are distribution and production in all four companies.
- The size of the Scrum teams is around six people (with the exception of Company 1). This matches the Scrum team size suggested by literature (Sutherland, 2003, Schwaber & Beedle, p. 36-37, 2002) quite well.
- Four of the five usage scenarios described in section 5.1 were experienced. Based on the small number of evaluated companies, this was a welcome coincidence.
- Besides two teams at Company 2, all Scrum teams are using Scrum software tools, although in a different manner. It is arguable that the overhead introduced by administrating multiple Scrum teams somehow forces the usage of Scrum software. This will be covered in section 5.4.
- However, the inverse of the above-mentioned aspect is not true as is shown by Company 1: Although this company has just one single team, it uses a software-based task board. The reason for this choice seems to match the distance criterion described in section 3.2.10, namely that software-based task boards offer synchronization, which is an advantage for distributed teams like the one of Company 1.

5.3 Company Characteristics

Now that it is known that five task board usage scenarios exist (section 5.1) and how task boards are used at different types of companies (section 5.2), characteristics that identify which company properties are of relevance have to be defined. With the help of company characteristics, the guidelines for helping companies to make a choice in terms of a task board type can be created (as will be shown in the upcoming section 5.4).

To find those characteristics, once again the literature review has been used and experts have been consulted. Within this section, at first potential candidates for company characteristics have to be identified. This is done by reviewing agile tool surveys in section 5.3.1. Additionally, an internal company expert has also been consulted.

Secondly, the potential candidates identified in section 5.3.1 have to be analyzed to determine the company characteristics that influence the company's choice of a task board type. This is done in section 5.3.2.

Finally, the applicable company characteristics will be summarized in section 5.3.3. These characteristics will figure as input for the company guidelines in the upcoming section 5.4.

5.3.1 Potential Company Characteristics

One of the outcomes of the literature review (1.8.1) were four agile tool surveys, when companies were asked regarding their tooling choice next to provide details about their company (Azizyan, Magarian, Kajko-Mattson 2011, Behrens 2006, VersionOne 2009).

The surveys are the same as the ones that were used to identify and define criteria for evaluating the Scrum software tools in section 4.2. The limitations that were identified in that section partly also hold true for this section:

- Potentially biased surveys that were conducted by tool vendors or consulting companies (Dubakov & Stevens 2008, VersionOne 2009).
- No specific focus on Scrum task boards (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009).
- Surveyed persons were mostly managers and not Scrum team-members, however Scrum team-members are the ones that have to work with the task board (Azizyan, Magarian, Kajko-Mattson 2011, Dubakov & Stevens 2008, Behrens 2006, VersionOne 2009).

In addition to those limitations, those four tool surveys served a different purpose than identifying company characteristics to make a Scrum task board related choice:

- The first survey (Behrens, 2006) is the one that offered the highest amount of potential company characteristics. The host of this survey was a consulting company, which sets the focus on potential company characteristics like “Number of sites” or “Number of teams” next to the obligatory questions regarding which project management tools were used within the surveyed companies.
- The second survey (Dubakov & Stevens 2008) collected usage data of companies that requested a trial evolution of their Scrum tool. The survey merely asked which project management tools were used within those companies and was therefore negated since it did not contribute in identifying potential company characteristics.
- The third one (VersionOne 2009) surveyed the participating companies regarding agile project management tools, which is not surprising as a tool vendor hosted this survey. However, also potential company characteristics were identified.
- The fourth survey ((Azizyan, Magarian & Kajko-Mattson, 2011) is, as pointed out by its authors, an objective survey that does not suffer potential bias like the previous ones. This survey has put its focus on tool usage and needs of software companies as well as which tools meet the industrial needs. Due to its focus, the survey offered rather little potential for identifying company characteristics as shown by Table 21.

Although none of the surveys thus focused on company characteristics for Scrum task boards, five potential company characteristics were extracted. Consulting an internal company expert backed some of these characteristics. Also, with the help of the internal company expert, a sixth potential company characteristic was identified.

Despite the fact that no fully applicable survey was found, the following table lists the potential company characteristics according to the literature review and expert opinions.

Characteristic	Azizyan, Magarian, Kajko-Mattson	Behrens	VersionOne	Internal Expert Opinion
Agile method	✔	✔	✔	✔
Number of sites		✔	✔	✔
Number of teams		✔		✔
Company size		✔	✔	
Tool preference				✔
Team type	✔			

Table 21: Company characteristics found in literature

Note that most of these six potential company characteristics are quantitative. In combination with the paper versus software task board analysis in chapter 3, which offers more quality-related aspects, the company guidelines in the upcoming section 5.4 will be created.

The remainder of this section will discuss the six identified potential company characteristics.

Agile method

All surveys as well as the internal company expert support the agile method characteristic. It is a very general question that asks the participating companies to state the agile method they are using, e.g. Scrum, eXtreme Programming, Kanban, Feature Driven Development, Crystal, etc. - only to name a few. The most used agile methods are:

1. Scrum
2. eXtreme Programming
3. Kanban

A more recent study (Komus 2012) shows that Scrum is still the most used agile method, although the positions of Kanban and eXtreme Programming have switched.

Number of sites

This characteristic is self-explanatory; the companies were asked how many sites they run at different locations. If a company states that it has more than one site, it is very likely that teams are distributed over multiple sites although in theory the teams could also be collocated at each site. As a result, this characteristic is nearly the same as the "Team type" characteristic; see below.

Number of teams

This characteristic is only mentioned by one of the three surveys – the companies were asked to state the number of teams that are following an agile method within the company. The reason for this characteristic was that a company with less than ten teams that follow such an agile method has not implemented this method company-wide. Yet, the survey (Behrens, 2006) lacks proof for this assumption.

Company size

The company size characteristic simply states the number of employees of the whole company. The reason for that characteristic is that companies that have less than 100 employees are more likely to adopt Agile Methods (Behrens, 2006).

Tool preference

The "Tool preference" characteristic is used to identify the personalities that are forming the development team. Different types of employees exist that might favor a paper-based task board over a software-based task board and vice versa.

Team type

By "Team type" is meant whether a team is collocated (all team members are at one location) or distributed (team members are spread over multiple locations). One survey (Azizyan, Magarian & Kajko-Mattson, 2011) sub-divides the collocated teams into small and large collocated teams. However, the survey lacks stating a number that is used to decide whether a collocated team is a small or a large one. As described above, the "Team type" characteristic is roughly the same as the "number of sites" characteristic with the exception that theoretically it is possible that only collocated teams exist at each site, thus the "number of sites" characteristic is not equal to distributed teams per se.

5.3.2 Analyzing the Company Characteristics

In the previous section, potential candidates for company characteristics were identified. The purpose of this section is to try to map these characteristics to the outcome of the field research (section 5.2) in order to investigate which of these potential candidates can be used for creating company characteristics that in turn can be used to create the company guidelines. This analysis is done with the help of consulting external and internal company experts. Also literature is used to explain certain company characteristics.

Table 22 maps the results of the field research in section 5.2 to the potential company characteristics identified in section 5.3.1. Note that:

- Company 2 has been split into two sub-parts, as they are different from each other. Site A & C have distributed teams which are spread amongst both locations, whereas site B is more independent and has collocated teams (see 5.2.2). Therefore, for this analysis, Company 2 can be treated as two independent companies.
- The task board usage scenarios described in section 5.1 have been added to this table as they show the choice that was made by each company.

Characteristic	Company 1	Company 2 (site A & C)	Company 2 (site B)	Company 3	Company 4
Agile method	Scrum	Scrum	Scrum	Scrum	Scrum
Number of sites	3	2	1	1	1
Number of teams	1	6	2	4	5
Company size	600	300		1500	4000
Tool preference	Software	Software	Paper	Paper	Paper
Team type	Distributed	Distributed	Collocated	Collocated	Collocated
Scenario	Software	Software with big screens	Paper	Paper & Software in Coexistence	Paper & Software in Coexistence

Table 22: Mapping company settings to characteristics

The remainder of this section will analyze which of the six characteristics serve as a company characteristic and therefore qualify themselves as input for the company guidelines and which of them are not applicable.

Agile method

All supervised companies are using the Scrum method (except one team of Company 2's site B – although this team also uses a task board), which is obvious as the task board this master thesis project is evaluating is a Scrum artifact. Thus, the agile method itself is not a characteristic – instead it should be seen as a requirement, because companies that are not using Scrum or companies that do not want to use Scrum are not the ones that will use these company guidelines.

Number of sites

The "number of sites" characteristic can be directly used to support the decision for a suited task board type. Companies that are running multiple sites have mostly distributed teams whereas companies that are running only one site have mostly collocated teams.

However, this is not universally applicable: the field research in section 5.2 has shown that a company that has multiple sites does not automatically use the software-only approach.

Therefore, it is questionable whether this characteristic is applicable. Yet, if a company has multiple sites, in general it can be assumed that the teams are spread amongst these sites; at least if:

- The sites develop software and / or hardware.
- The sites are using Scrum.
- It is necessary to divide the teams between those sites.

Number of teams

At first sight, no correlation between the "number of teams" characteristic and the choice for a certain task board type seems to exist. However, experts have stated that the more teams a company has, the more likely it is that the choice will tend towards a Scrum tool in terms of planning and reporting. Although these aspects are not related to a certain task board type, they can influence this choice.

Company size

A relationship between company size and task board type could not be found. First, none of the visited companies has less than 100 employees, thus the reason for that characteristic is out of scope (Behrens, 2006). Second, neither it can be stated e.g. that companies that have less than 1000 employees are using a paper-based task board nor can be stated that companies that have more than 1000 employees are using a software-based task board and vice versa.

Tool preference

Experts have stated that there are different types of teams that favor a paper-based task board over a software-based one and vice versa. For example, the team members of Company 1 stated, that in their opinion a paper-based task board is childish and does not look professional. The team members of Company 3 stated exactly the opposite: they liked to work with a paper-based task board as it continuously reminds them what they have to do next to feeling a kind of relief when a task was put into the "Done" column. Although the team-members are the ones that have to work with a task board, the ScrumMaster has to negotiate with the management in such a way that both parties can accept a particular task board choice; e.g. an example could be that the team states that it wants to work with a paper-based task board although they are a distributed team – it is then the job of the ScrumMaster to propose solutions, like the ones that are given in section 5.1.

Team type

As indicated by literature (Perry, 2008), a distributed team is a team that is dispersed amongst multiple locations and a collocated team is a team that is located at one single location. In Chapter 3 it has been shown that distributed teams are likely to use a software-based task board whereas collocated teams are more likely to use a paper-based task board.

However, like the "Number of sites" characteristic, one is not able to link a certain team type automatically to one of the task board types: a distributed team could work with a paper-based task board and a collocated team could work with its software-based pendant.

Yet again, it is thus questionable whether this characteristic is applicable. However, based on the assumption that a distributed team normally uses a software-based task board and a collocated team a paper-based one, this characteristic is applicable.

5.3.3 Applicable Company Characteristics

In the previous sections, we started by fulfilling the prerequisites for setting up applicable company characteristics by identifying five task board usage scenarios (section 5.1) and by performing field research at four different companies to evaluate their task board usage (section 5.2).

These prerequisites were followed by defining company characteristics in section 5.3. At first, possible candidates for company characteristics were identified by using agile tool surveys and by the help of an internal company expert in 5.3.1. In section 5.3.2 these candidates were analyzed.

The purpose of this section is to summarize the outcome of the analysis that was done in the previous section by showing, which characteristics are applicable and which are not. By doing so, a starting position for creating the company guidelines, which can be found in the following section, is created.

Table 23 summarizes the characteristics and indicates whether they are applicable. All characteristics with the exception of the "Tool preference", which was introduced by the internal company expert, were found using the literature review.

Characteristic	Applicability	Analysis result
Agile method	✗	Not applicable, since all evaluated companies use Scrum. Using Scrum is a prerequisite; otherwise companies would have no interest in the upcoming company guidelines.
Number of sites	(✔)	Applicable, but based on the assumption that multiple sites indicate that the teams are distributed whereas a single site indicates that the team is collocated.
Number of teams	✔	Applicable, as the amount of administrative overhead is growing with each additional team.
Company size	✗	Not applicable, as there is no relation between the size of a company and the choice of a task board type.
Tool preference	✔	Applicable, as employees have to work with the task board frequently. Therefore it is important that it is accepted.
Team type	(✔)	Applicable, but based on the assumption that a distributed team uses a software-based task board and a collocated team uses a paper-based one.

Table 23: Applicable company characteristics

This table shows that there are three cases:

- Company characteristics, which are not applicable.
- Company characteristics, which are applicable but are based on an assumption.
- Company characteristics, which are applicable.

As a result, the "Number of teams" characteristic as well as the "Tool preference" characteristic are applicable. The same is true for the "Number of sites" characteristic and the "Team type" characteristic if both assumptions are taken into account. The "Agile method" characteristic and the "Company size" characteristic are not applicable.

However, the four characteristics that are applicable are in turn based on an assumption, namely that the companies that were subject of the field research are representative. Based on the small number of evaluated companies, this is difficult to say. Yet, although the choice for these four companies was not based on any criteria but only on the fact that they were the companies that were open for the evaluation, these companies represented four of the five identified task usage scenarios (section 5.1). It is therefore reasonable that although by coincidence, those four companies represent a rather complete snapshot of the available possibilities for creating company guidelines in the upcoming section.

Since four company characteristics (two of them based on assumptions) have been identified and validated, these four characteristics will serve as starting point for the company guidelines.

5.4 Company Guidelines

Now that the applicable company characteristics have been defined, the actual company guidelines can be created.

In section 5.4.1, a decision graph will represent these guidelines in a graphical manner. Further on, the section explains the branches and provides a validation for this model.

Since the decision graph also advises to use an intermediary solution next to paper- & and software-only solutions, section 5.4.2 will explain this coexistence scenario thoroughly.

5.4.1 Decision graph

As described previously, the applicable company characteristics are used as starting points within the following decision graph:

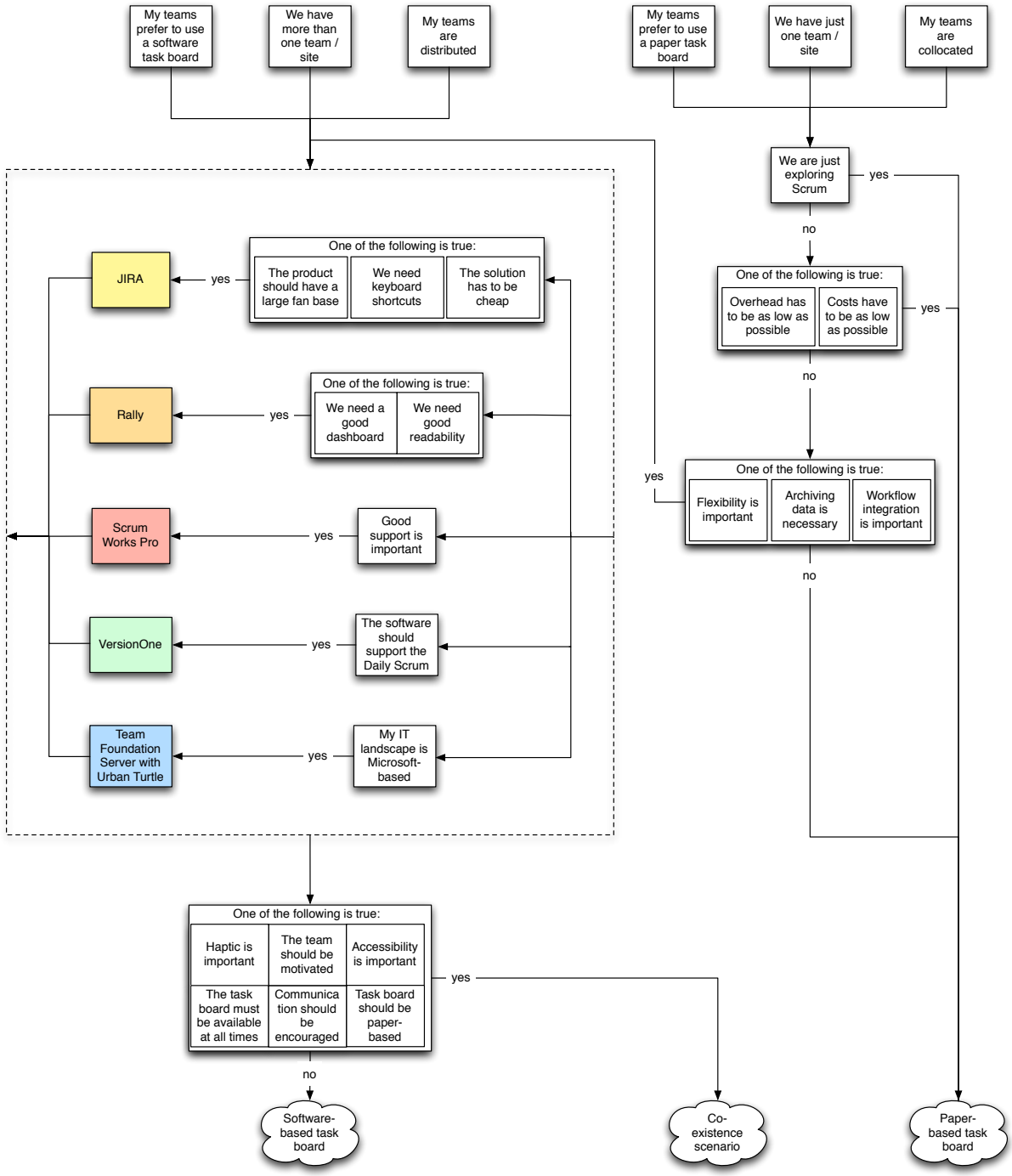

Figure 16: Company guidelines in form of a decision graph

The rationale behind this decision graph is to give companies a tool, which helps them to make the right decision instead of offering all available combinations of possibilities, no matter whether they make sense or not. This section will explain the decision graph in a detailed manner by validating each of its sub parts.

The decision graph combines the results from chapter 3, 4 and 5. It has six starting points that form groups of three. As mentioned above, the starting points are the applicable company characteristics. The ones with an assumption were also included and for this section the assumptions are treated as being correct for the cases in which they are false are mostly of theoretical nature only (see section 5.3.3).

Each of the resulting two branches contains thus three items, which are the following:

- Branch 1
    - My teams prefer to use a software task board
    - We have more than one team / site
    - My teams are distributed
- Branch 2
    - My teams prefer to use a paper task board
    - We have just one team / site
    - My teams are collocated

Two of the starting points (one per branch) may sound confusing: "My teams prefer to use a software / paper task board" – why should the decision graph use a starting point that decides which task board variant should be chosen before even the first decision of the graph is presented? The rationale behind this choice is twofold: first, these two starting points result from the "Tool preference" characteristic, which was identified as applicable characteristic in section 5.3.3. Second, both starting points do not automatically result into choosing the corresponding task board variant; e.g. if a company uses the "My teams prefer to use a software task board" starting point, then the decision graph still offers the possibility to choose the coexistence scenario (see the remainder of this section) eventually. The same is true for the "My teams prefer to use a paper task board" starting point: companies can be led to both task board variants (software- or paper-only) next to the before-mentioned coexistence scenario. Thus, both starting points do not bias the decision of companies – quite the opposite is true: companies who think that they already have made a choice are confronted with a tool that might make them to reconsider their decision.

Branch 1

Branch 1 leads to a software-based task board that optionally is combined with a paper-based one whereas branch 2 leads to its paper-based pendant with the option to decide to use a software-based task board as well. Literature supports these starting points: Scrum tools are more often used if the team is distributed amongst multiple locations and less often used if the whole team is at a single location (Azizyan, Magarian & Kajko-Mattson, 2011). The more teams a company has, the larger the company is and the more likely it is that the company uses software tools to administrate the Scrum process (Behrens, 2006). Finally, Scrum tools can help teams if they are distributed (Berczuk, 2007).

One of the results of the analysis in chapter 3 was that a software-based task board has more disadvantages than advantages. Yet, software-based task boards are the primary choice for companies that have distributed teams that are geographically dispersed amongst multiple locations. Therefore, the first branch redirects to choosing a software-based task board. Of course, companies that can identify themselves with branch 1 could also ignore this decision graph and choose for a paper-based only solution – however, this is not recommended and as a result not modeled within the decision graph. Yet, this does not mean that those companies have to work with a software-only solution, see below.

The next step for branch 1 is to choose one of five available Scrum software tools that can represent a task board. The five Scrum software tools are the ones that were evaluated in chapter 4. For each of the five tools, the abilities for which they outperform their competitors are listed. Choosing one of these five tools is a sub-process and therefore a dashed rectangle has been drawn around this process to indicate that lower level decisions have to be made. Tools that have multiple abilities in which they

perform better than their competitors have also multiple aspects within the decision graph. For example: where ScrumWorks Pro should be chosen if good support is more important than anything else, Rally should be chosen if good readability is necessary ***or*** if a good dashboard is needed.

If companies are not able to make a decision, for example because they like the features of multiple tools, it is recommended for these companies to read the detailed analysis of the tools in chapter 4 to get more insight.

After a tool choice is made, the decision graph offers a final decision for branch 1 by presenting six criteria that the companies have to evaluate if any of the statements are true. These six statements are the criteria based on which a paper-based task board would be a better choice than a software-based one. In case that at least one of these six statements is true, the suggested choice for these companies is to use the coexistence scenario that has been described in section 5.1.5. The idea behind this coexistence scenario is to let the teams use a paper-based task board in combination with a software-based one by transferring once a day the progress that has been made from the paper-based task board to the software-based one. By doing so, the advantages of both task board variants are combined minus two criteria (overhead & costs), which are not valid if one chooses this scenario, since the paper only variant is cheaper and requires less overhead. Companies for which none of the six statements are true should choose the software-only variant.

Branch 2

Branch 2 represents companies that can identify themselves with one of the three starting points, namely that their teams want to work with a paper-based task board, that they have just one Scrum team or are located at just one site or that their teams are collocated.

The first decision that is offered by the decision graph is linked to the second and third starting point: companies have to decide whether they are just starting with Scrum and want to explore it and see if it is suitable for them or if they are already using it on a regular basis. The rationale behind this decision is to advise a paper-only task board for companies that want to explore Scrum. This variant offers all advantages of paper-based task boards next to being the cheapest solution with the lowest amount of overhead, which is not the case if the before-mentioned coexistence scenario is chosen.

As a next step, companies that are already using Scrum have to evaluate whether overhead and costs should be as low as possible. Again, the reason for this decision is the same that was described above: if one or both statements are true, yet again the advice is to use the paper-only task board.

If costs and overhead do not have to be as low as possible, the next decision that has to be made is whether the advantages of a software-based task board (flexibility, archiving, integration – distance has been left out since this branch of the decision graph is about collocated teams) are of importance. In case they are, the companies have to follow the process of choosing a matching Scrum software tool as has been described previously for branch 1. By doing so, it is also possible that these companies can choose to use a software-only task board, if all six statements are false. If one of the six statements is true, the decision graph advises to use the coexistence scenario, which will be explained more thoroughly in the upcoming section.

Final words

To sum up, the decision graph gives the following recommendations: Companies that just have started with Scrum and do not want to invest lots of money should start with using the paper-only task board approach. A paper-based task board has many advantages as compared to its software-based pendant. This was shown in chapter 3. Companies that evaluated Scrum successfully and want to implement it are likely to work with a Scrum software tool sooner or later as it is arguable that the costs coming up by administrating multiple Scrum teams somehow force to use a Scrum software tool. As described previously, these tools also feature task boards. Yet, the software-only task board approach has more disadvantages than paper-based ones (see section 3.3). If such companies can accept those disadvantages, their teams can do their daily routines by using a software-based task board. All those companies that cannot accept the disadvantages that are introduced by solely using a software-based task board should consider the before-mentioned coexistence approach.

5.4.2 Coexistence Scenario explained

Now that the decision graph has been presented and explained, it has been shown that companies do not have to choose between a paper-based task board and a software-based task board only. In fact, the coexistence scenario, which recommends combining both task board variants, is presented as a third option. Briefly explained in 5.1.5, this section elaborates the coexistence scenario approach.

Introduction

The analysis in chapter 3 and its outcome described in section 3.3 has shown that both task board types have unique aspects. The analysis of Scrum tools in chapter 4 supports these findings: software-based task boards share a lot of disadvantages that paper-based task boards do not have. Whatever decision a company makes, it will come at a cost: if the company chooses to use a paper-based task board in favor of a software-based task board, the company has to accept limitations in certain areas, e.g. integration, archiving or perhaps coming into problems when it comes to distributed teams. If the company chooses for a software-based task board it has to accept limitations in other areas, e.g. accessibility, costs or motivation. Thus, no matter which choice a company makes, it has to accept certain disadvantages.

Fortunately, also an intermediate solution is added to the decision graph (Figure 16), which provides a solution to these problems. In section 5.1.5, the coexistence scenario has been described which proposes to use a paper-based task board that is backed up by a software-based one. The paper-based task board is used by the team members for doing their daily work and a dedicated person, for example the ScrumMaster, updates the changes occurring on the paper-based task board within the software-based task board accordingly. By following this approach, distributed teams are still able to benefit from the advantages that come with a paper-based task board without struggling with its disadvantages, which are neutralized by using the software-based task board to synchronize the paper-based ones at multiple locations.

Works for both team types

However, applying this scenario does not just work for distributed teams, it is also suitable for collocated teams, as they also benefit from archiving sprint data and integration with other tools. As a result, a recent study has shown that 31% of the surveyed collocated teams are already using paper and software simultaneously (Azizyan, Magarian & Kajko-Mattson, 2011). This survey has also shown that more than 90% of the surveyed persons indicated that ease-of-use is the most important aspect when using a Scrum tool, which supports the recommendation of this coexistence scenario as ease-of-use is far better when using a paper-based task board as shown in section 3.2.9. Yet again, this recommendation is also backed by another recommended practice (Perry, 2008).

However, one has to keep in mind that the task board is only a small part of Scrum and one of the few where paper outperforms software in terms of amount of advantages compared to the amount of disadvantages. Nowadays, Scrum tools support the complete Scrum process and there are many areas where paper does not stand a chance against software, for example in terms of product backlog planning, tracking or reporting. Already more than half a decade ago, a survey indicated that 40% of the mentioned companies did state that they were using a Scrum tool (Behrens, 2006). As a result, most of the companies are dependent on Scrum software tools and arguments like the higher cost of software do not count anymore, as the companies have to use a Scrum tool anyway. Field research (section 5.2) supports this argument: large companies with multiple teams unanimously use software, although not in its pure form as described in section 5.1.3. Still, it makes sense to embed a paper-based task board into the process, due to its many advantages as described in section 3.3.1.

Disadvantages of the coexistence scenario

Using the coexistence scenario does not mean that there are no disadvantages at all; two of them remain: costs and overhead. As both types of task boards need to be bought one can argue that this solution is even more expensive than solely using software-based task boards, however as indicated in section 3.2.6 the costs of a paper-based task board are insignificant. In contrast, the costs of a software-based task board are quite the opposite: licenses are expensive (see chapter 4) and IT hardware has to be bought and maintained. Yet, one can argue that most of the companies that do

Scrum use a Scrum software tool anyway due to reasons that were described previously. The increase of overhead is also significant, as at least someone has to learn how to use the software-based task board. It is up to the team to decide whether a dedicated person should update the software-based task board or whether every team member should do it on his or her own. If the latter is true, the company has to buy more licenses and team members have less time to do their actual work although it is questionable whether updating the software-based task board consumes that much time. However, no matter who is keeping both types of task boards in sync, special attention is required when changes have to be transferred. Under ideal circumstances, all persons in charge for the synchronization at each site will have a daily meeting, e.g. by using a telephone conference setup to inform each other about the progress and the changes that have to be made.

Summary

This coexistence recommendation of both task board types follows the best-of-bread approach by combining the strengths of both variants. At this moment, it offers the best aspects from both sides without introducing major drawbacks. Of course there might be certain scenarios where a paper- or software-only approach makes more sense than a combined one. Still, companies should be aware of the advantages and disadvantages they have to accept when choosing the former or the latter.

5.5 Wrap Up

In this chapter, five different usage scenarios were identified in section 5.1 by using the findings of the literature review next to consulting internal company experts.

Section 5.2 listed the companies that were subject of the field research and showed which task board variant are used by the individual companies and how satisfied they are with it.

In section 5.3, the literature review as well as expert opinions were used to define company characteristics that serve as starting points for the company guidelines.

Finally, in section 5.4 the characteristics found next to the findings in chapter 3, 4 and 5 were used to create a decision graph which embeds the company guidelines for choosing a task board variant. This last section also showed that the coexistence scenario described in section 5.4.2 is an interesting intermediary solution, which combines many of the strengths of both paper-based and software-based task boards.

By creating the company guidelines, the second half of the research goal was reached.

6 Conclusions

Now that the research goal has been reached and all research questions have been answered, this chapter is used to present the results, to point out limitations as well as options for further research and to reflect on the research goal.

6.1 Results

This master thesis project has theoretical as well as practical results which will be discussed in this section.

Theoretical results

First of all, the Scrum task board artifact has been analyzed thoroughly. Its paper-based and software-based task board variants have been evaluated extensively. Criteria have been identified and defined to perform this analysis. A total amount of twelve criteria was used to perform this analysis and each of these criteria had in its turn a set of properties. In general, paper-based task boards score better than software-based ones: the paper-based variant outperforms the software-based task board in terms of eight criteria (accessibility, motivation, haptic quality, costs, availability, overhead and communication). As a result, the software-based task board performs better in terms of the remaining four criteria (flexibility, integration, archiving and distance). However, this result does not implicate that a paper-based task board is better than a software-based one due to the fact that, depending on the type of the company, some criteria are more important than others. For example: for a company that has geographically dispersed teams, the outcome of the distance criterion might be far more important than the outcome of the overhead criterion. The results of the analysis have been described in a more detailed manner in section 3.3.

Second, criteria have been defined again to evaluate Scrum software tools. These criteria are based on the foregoing criteria of the software versus paper analysis. However, due to the different scope (software versus software), some criteria were not applicable and therefore were stripped. For example, the haptic quality criterion is not relevant for this software analysis since the setup for using these Scrum software tools is essentially the same (namely a computer with a screen, a keyboard and a mouse). Yet, new criteria were identified and added; for example the quality criterion, which is software-specific by evaluating the response time as well as the availability of the vendor's service. Like for the paper versus software analysis that has been described above, again for each criterion a set of properties was defined, for example the cost criterion has two properties (the expenses needed for a tool that can be downloaded and the expenses needed for a tool that is hosted by the vendor). By using these criteria with their corresponding properties, the Scrum software evaluation was performed, see the practical results below for more details.

Finally, different task board usage scenarios were identified, listed and summarized. Next to the two common scenarios, namely using a paper-based task board and using a software-based task board, another three possible usage scenarios were added. The first of these remaining three was the combination of paper with audio, video or photo – changes that occur on the task board are synchronized with other sites by using a audio or video telephone conference or by taking pictures with a digital camera. The second additional scenario that was identified is to use a big TV screen, which is used to display the software-based task board. The purpose of this scenario is to enhance accessibility; however the readability of the stories and tasks is low, next to the fact that normal TV screens lack any support of inserting data. Thus, the accessibility of this scenario cannot be compared with the accessibility of a paper-based task board. Finally, the last scenario identified is to combine a paper-based task board with a software-based one. The idea is to use the paper-based task board for the daily activity at each site and to transfer the changes that occur on the paper-based task board to the software-based one to synchronize the sites.

Practical results

First, five Scrum software tools have been thoroughly examined: JIRA (Atlassian), ScrumWorks Pro (CollabNet), Urban Turtle (Urban Turtle), Rally (Rally Software) and VersionOne (VersionOne), see the upcoming section 6.2 for the rationale of choosing between these tools. To perform this evaluation, the above-mentioned criteria were used. The results support the findings of the paper versus software

analysis, namely that the task boards of the evaluated Scrum software tools were quite weak in terms of e.g. accessibility, yet quite strong when it comes to distributed teams that are located at multiple sites. Each of the five tools has aspects in which they outperform the remaining four vendors next to aspects that could be improved. JIRA offers a lot of functionality and is the cheapest tool, however it lacks maturity. ScrumWorks Pro leads quality-wise but uses an outdated approach by splitting the tool in a web and a desktop client. Urban Turtle offers great integration possibilities for companies that have a Microsoft-based IT landscape but has its disadvantages in terms of usability. Rally offers the best task board and is very mature, but is one of the more expensive tools. Finally, VersionOne also offers a lot functionality-wise, however, it is quite complex and also is one of the more expensive tools. Companies that have used or want to use the company guidelines and need more information regarding tool specifics can use the findings in chapter 4 to do so.

Second, four companies were subject of the field research. These companies were evaluated in terms of the task board they are using and how satisfied they are with their current solution. The results have shown that four of the five task board scenarios (all but the one that combines paper with audio, video or photo) were used at these companies. Therefore it is assumed that the field research represents a rather complete scope. The field research has also shown that only a small number of employees are actually using Scrum, compared to the total number of employees within the companies. The reason for this is that the departments that develop software are rather small, compared to departments like production or distribution. Another outcome was that four of the five companies that were subject of the field research are using a Scrum software tool; one company uses the software-only approach, another company uses a software-based task board in combination with a big TV screen and two companies combine both task board variants. Only one company is using the paper-only approach. As a result, it is arguable that companies that have multiple teams are in need for a Scrum software tool to administrate the overhead.

Finally, all findings of this master thesis project have led to the definition of a decision graph (see section 5.4.1). This graph can be given to any company in form of a handout that serves the need of a quick, but precise tool to make a decision between either task board variants or an intermediary solution that combines paper with software. The graph draws on all of the previously mentioned results: it uses the paper versus software criteria, the software versus software criteria, a selection of the usage scenarios and the key aspects of all five Scrum software tools.

6.2 Limitations & Further Research

This master thesis project has also some limitations and possibilities for further research, which will be discussed within this section.

Limitations

First, it cannot be claimed that the lists of criteria for the software versus paper analysis and the Scrum software tools evaluation are complete. However, multiple sources (e.g. literature review and expert opinions) were used to identify and evaluate the criteria. Besides, the criteria have their roots within different fields: economics (costs), psychology (motivation, haptic quality) or technology (integration, availability, archiving), only to name a few examples. Therefore, both lists are quite comprehensive if one considers the limited time and manpower behind this master thesis project.

Second, due to the small sample size of companies, it cannot be claimed that the results of the field research are complete. However, due to the fact that four of the five usage scenarios presented in section 5.1 were experienced, it does indicate a trend and supports the other findings of this master thesis project.

Third, the same is true in terms of the above-mentioned task board scenarios described in section 5.1; it cannot be guaranteed that this list of scenarios is complete. Yet again, using multiple sources (literature review and expert opinions) helped to identify the existing task board scenarios. Since the results of the field research support the identified task board scenarios and the other way round, it is arguable that this limitation and the previous one are insignificant.

Fourth, due to constraints of this master thesis project, only five Scrum software tools were subject of the software evaluation in chapter 4. As a result, the decision graph only offers a choice between those

five tools for companies that need to select a software-based task board. However, the rationale behind choosing these five tools was based on a recent study (West & Hammond 2010), which identified the top ten vendors of agile software tools. This list was then reduced to five tools by applying selection criteria (the vendors have to offer a free trial and the tool has to be browser-based to avoid time consuming setups). Thus the five Scrum software tools that were evaluated are the most important ones that are currently available and therefore it is again arguable that this limitation is also negligible. Yet, adding more tools will make the decision graph more complete as is written below.

Finally, although the companies that were subject of the field study offered a rather complete spectrum, all of these companies are located in Germany. The same is true for the company experts, who are also from Germany. It is therefore unknown whether the findings of this master thesis project are applicable in other countries as well. However, there are a few arguments that show that the results of this master thesis project are not just national applicable. First, all Scrum software tools are made by vendors that are located in foreign countries, most of them in the US. It is unlikely that these companies would build Scrum software tools that target the German market only, especially since most of their websites are in English – one could argue that if these vendors target the German market, they would also build German websites. Second, two of the companies that were subject of the field research had foreign sites for which these findings are also applicable. Finally, the principle of a Scrum task board is defined within the Scrum framework, which is used globally. Therefore, it is unlikely that these findings are not applicable in foreign countries.

Further Research

Based on the described limitations, a few suggestions for further research can be made.

First, in order to improve the paper versus software analysis as well as the Scrum software tools evaluation in an even more thorough manner, some of the criteria could be extended, e.g. the integration criterion of the Scrum software tool evaluation (see section 4.2.4) could also include a quality property. But also other criterions, like the haptic quality criterion could form the basis for a dedicated study in the field of psychology.

Second, next to the criteria that were identified, it is possible to extent this list, for example because criteria might exist that are currently not known or do not exist yet due to limited technology.

Third, in terms of the field research it would be beneficial to include more companies to make the company-related findings more objective. It would be interesting to evaluate non-German companies to investigate whether the findings of this master thesis project are also applicable outside of Germany, although it is assumed that they are, as shown before.

Fourth, the decision graph would be even more complete if more Scrum software tools could be evaluated and added. However, the current version with its five tools is a good starting point for companies, since these five tools represent the most important ones that are currently available, as has been explained in the limitations section above.

Finally, consulting different company experts could reveal that there might also exist other task board scenarios next to the five that were identified in section 5.1, although it has been shown that the field research supported these task board scenarios and vice versa. One can therefore assume that the list of scenarios is quite complete.

6.3 Reflection

Now that the results, the limitations and possibilities for further research have been described, the purpose of this section is to reflect on the research goal and the corresponding research questions.

As explained in section 1.6, the research goal was twofold and the first half of the research goal was split on its turn into two parts:

Research goal

- Task board analysis
    - Evaluation paper versus software
    - Evaluation Scrum software tools
- Company guidelines

To reach the research goal, five research questions were defined in section 1.7. The remainder of this section will reflect on each of these five questions.

The first three research questions were used to reach the "Evaluation paper versus software" part of the research goal, the fourth research question was used to reach the "Evaluation Scrum software tools" part and therefore to complete the first half of the research goal. The second half was reached by answering the fifth and final research question.

1. What is the role of a task board in Scrum?

Research question one was used to give an introduction to Scrum and its task board. To answer this question, literature in the form of Scrum-related books was used. To make the reader familiar with the task board, another literature study was used that reviewed task board related articles.

It is arguable that this Scrum introduction (given in chapter 2) could have been moved to the appendix and / or that it could have been less detailed. However, it was important to make sure that the reader would get sufficient background information to grasp the Scrum framework so that he would be able to place this master thesis project into the right context.

Although the Scrum framework exists for quite some years now, changes are still applied (since it is an agile framework). However, as the task board is a very important Scrum artifact it is highly unlikely that it will be removed some day.

2. Which criteria can be used to compare a paper- and a software-based task board?

The second research question was answered by identifying and defining criteria that can be used to compare a paper-based task board with a software-based one. Again, the task board related literature review was used, but also experts were consulted who provided valuable input for this master thesis project. Identifying criteria was difficult when the master thesis project was started: although the literature review supports many of the criteria, it did not explicitly list them. Instead, the experts helped a lot to define criteria. They stated which aspects are important and therefore should be covered within the analysis. As soon as the criteria were defined, it was much easier to use the literature to support these criteria. In the end, the list contained 12 criteria that were used as input for the third research question.

Answering this research question had shown that the help of experts was essential for this research, since the results of the task board-related literature review were rather poor. On top of that it was interesting to see that internal experts had other opinions than external experts. By consulting both types of experts, the list of criteria became more thorough.

However, as described in the previous section, some criteria could not be described as extensive as possible due to several limitations. Still, the criteria themselves are solid since they have their roots in multiple fields (e.g. psychology, economics or technology) as has been described in the previous section. It is therefore reasonable to assume that they do not have a date of expiry.

3. What are the advantages and disadvantages of paper-based and software-based task boards?

The purpose of the third research question was to use the list of criteria that was made by answering the second research question in order to analyze the paper-based and the software-based task board

variants. Again, the analysis was performed by consulting experts and by using the task board related literature study. The result of the analysis was that the paper-based task board won eight of the twelve criteria; the software-based task board won the remaining four criteria.

Yet, based on the characteristics of a company, the criteria have different weights, e.g. some criteria that are more important for one company can be less important for another one and vice versa. Therefore, the table, which presents the scoring of both variants, can be misleading. However, the decision graph presented in the previous chapter made the weights of the criteria more transparent.

Finally, in contrast to the criteria definitions, the outcomes of the analysis are only temporary valid since most of them are technology-specific. For example, in terms of the motivation criterion, software-based task boards have a huge, yet currently unused potential: they could play a sound if a change occurs or flash a light. They could be customized in such a way that they display additional information that is requested by the team, etc. The same is true for the accessibility criterion: if large screens (several meters wide and high) get more affordable next to an input device which provides the user with similar haptic feedback like moving the Post-Its, then the software-based task board might win most of the criteria that are currently linked to its paper-based variant. However, it will take some time until such technology will come in the market in an affordable manner.

4. Which software-based task board tools exist and how well do they perform?

Research question four was answered by evaluating a set of Scrum software tools that are able to represent the task board electronically. Out of the list of the top ten vendors of agile software, five tools were selected using selection criteria that were based on the constraints of this master thesis project next to findings of the literature review.

As described in section 6.2, the criteria were as thorough as possible, however, further research could extend certain criteria. This time, defining criteria was a lot easier compared to defining the criteria for the paper versus software analysis, since some of them could be reused and experts were directly consulted. Again, the literature review was used to support the criteria that were found by consulting both types of experts. Similar to the second research question, the defined criteria and their corresponding properties could not be analyzed for each tool completely, again due to limitations in terms of time and manpower. Yet, the analysis is sufficient enough to give companies an impression regarding the major advantages and disadvantages of these tools. However, in terms of usability, the analysis would benefit from adding a laboratory research where subjects are asked to work with the Scrum software tools in order to get more quantitative results. Right now, this has been done by means of the field research; still, only two of the five Scrum software tools that were evaluated were used at these companies.

The results of this Scrum software tools evaluation are even faster outdated than the results of the paper versus software analysis described above – as a matter of fact, Atlassian already released two new versions of JIRA after this evaluation was finished. Thus, the release cycles of Scrum tool vendors are quite short. If the tool vendors would focus a bit more on their task boards, for example by implementing acoustical or visual triggers, results might change in a short period of time – in favor of the tools.

5. Which Scrum task board-related company characteristics and guidelines can be defined?

The last research question was answered so as to reach the second half of the research goal by creating company guidelines that can be used by companies that have to make a Scrum task board choice. However, before creating these guidelines, certain prerequisites had to be fulfilled. Experts were consulted regarding task board usage scenarios: five scenarios were identified and backed up by literature, if possible. The next step was to perform field research; four companies were visited and their task board usage was evaluated. Four of the five task board scenarios were experienced during the field research; therefore it is assumed that these companies represent a rather complete set.

The findings of the field research were used as input to identify company characteristics. The purpose of these characteristics was to determine company-related characteristics that would influence companies in the making of a task board choice. Consulting experts and using the literature review identified six possible candidates. The analysis of these characteristics showed that two of them were

applicable, two of them under certain assumptions and that the remaining two were not applicable. Finally, for creating the company guidelines, the applicable company characteristics were used as starting point.

The guidelines were visualized in form of a decision graph, which serves as simple, yet effective tool for companies that have to make a task board related decision. However, multiple iterations were necessary to create this decision graph. At first, it was a rather simple figure, which did not use the findings from the previous chapters and therefore was rather poor. The second iteration drew on all findings of this master thesis project and was therefore more complex, yet much more complete and effective. The third and final iteration was used to tweak the graph and to remove some semantic errors.

As stated above, the decision graph was explained and validated by using the findings of chapter 3, 4 and 5 of this master thesis. As a result, it also suffers from having a date of expiry, since the results of the paper versus software analysis and the Scrum software tools evaluation will expire sooner or later. However, it will take some time before software outperforms paper in terms of all twelve criteria. This is also partly true for the Scrum software tools: although their release cycle is rather short as was shown by the JIRA example above, the key aspects in which each tool outperforms their competitors are not likely to disappear; at least as long as vendors do not radically change them.

The main difficulty of creating the company guidelines was to find a starting point. This was done by means of identifying company characteristics. However, due to the low amount of potential candidates for these characteristics it is questionable whether this approach was the most efficient way to find appropriate starting points for the company guidelines. Still, the four applicable company characteristics that were found provided reasonable starting points.

6.4 Final words

Performing this master thesis project has shown that newer technology does not necessary have to be better than its predecessor. Although Scrum software tool vendors try to claim that using paper is an outdated approach and does not offer any advantages, this statement should be relativized; therefore the following quote summarizes the results of this master thesis quite well:

"As simple as the whiteboard is, it makes W[ork] I[n] P[progress] continuously visible, it enforces WIP constraints, it creates synchronized daily interaction, and it promotes interactive problem solving. Furthermore, teams evolve methods of using whiteboards continuously, and they have high ownership in their solution. In theory, all this can be replicated by a computer system. In practice, I have not yet seen an automated system [that] replicates the simple elegance and flexibility of a manual system." (Reinertsen, p. 205, 2009)

References

Agile Manifesto, 2001, http://www.agilemanifesto.org/ accessed at 01.10.2012

Amabile, T., Kramer, S., The Progress Principle: Using Small Wins to Ignite Joy, Engagement, and Creativity at Work, 2011, Harvard Business Review Press, ISBN 9781422198575

Azizyan, G., Magarian, M.K., Kajko-Mattson, M., Survey of Agile Tool Usage and Needs, 2011, AGILE Conference p. 29-38

Behrens, P., Agile Project Management (APM) Tooling Survey Results, 2006, Trail Ridge Consulting, LLC

Berczuk, S., Back to Basics: The Role of Agile Principles in Success with an Distributed Scrum Team, 2007, AGILE Conference, p. 382-388

Cockburn, A., Agile Software Development: Software Through People, 2001, Addison-Wesley, ISBN 9780201699692

Cohn, M., Agile Estimating and Planning, 2005, Prentice Hall International, ISBN 9780131479418

Dubakov, M., Stevens, P., Agile tools. The good, the bad and the ugly, 2008, TargetProcess Inc.

Gloger, B., Scrum – Produkte zuverlässig und schnell entwickeln, 2011, Carl Hanser Verlag 3rd edition, ISBN 9783446425248

Grenning, J., Planning Poker or How to Avoid Analysis Paralysis While Release Planning, 2002, http://renaissancesoftware.net/files/articles/PlanningPoker-v1.1.pdf accessed at 01.10.2012

Herbsleb, J., Moitra, D., Global Software Development, 2001, IEEE Software Volume 18, Issue 2, p. 16-20

Hossain, E., Bannerman, P. L., Ross Jeffery, D., Scrum Practices in Global Software Development: A Research Framework, 2011, PROFES'11 Proceedings of the 12th international conference on Product-focused software process improvement, p. 88-102

Jimenez, M., Piattini, M., Vizcaino, A., Challenges and improvements in distributed software development: A systematic review, 2009, Advances in Software Engineering, Article ID 710971, p. 1-14

Kan, S.H., Metrics and Models in Software Quality Engineering (2nd edition), 2002, Addison-Wesley Longman, ISBN 0201729156

Komus, A., Status Quo Agile Verbreitung und Nutzen agiler Methoden, Juli 2012, Studie des BPM-Labors der Hochschule Koblenz, Version 1.11

Paasivaara,M., Durasiewicz, S., Lassenius, C., Using Scrum in a Globally Distributed Project: A Case Study, 2008, Software Process: Improvement and Practice Volume 13, Issue 6, p. 527–544

Parnin, C., Görg, C., Rugaber, S., TaskBoard: Tracking Pertinent Rask Artifacts and Plans, 2009, Program Comprehension, 2009, ICPC '09. IEEE 17th International Conference, p. 317

Perry, T., Drifting Toward Invisibility: The Transition to the Electronic Task Board, 2008, Agile 2008 Conference, IEEE AGILE '08. Conference, p. 496-500

Reinertsen, D. G., The Principles of Product Development Flow – Second Generation Lean Product Development, 2009, Celeritas Publishing, ISBN 9781935501001

Robles-De-La-Torre, G., The Importance of the Sense of Touch in Virtual and Real Environments, 2006, IEEE Multimedia 13(3), Special issue on Haptic User Interfaces for Multimedia Systems, p. 24-30

Royce, W., Managing the Development of Large Software Systems, 1970, Proceedings of IEEE WESCON 26 , p. 1-9

Sarkan, H. M., Ahmad, T. P. S., Bakar, A. A., Using JIRA and Redmine in Requirement Development for Agile Methodology, 2011, 5th Malaysian Conference in Software Engineering, IEEE

Scharff, C., Guiding global software development projects using Scrum and Agile with quality assurance, 2011, Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference, p. 274-283

Schwaber, K., Beedle, M., Agile software development with Scrum, 2002, Prentice Hall, ISBN 0130676349

Sutherland, J., SCRUM: Keep Team Size Under 7!, 2003, http://scrum.jeffsutherland.com/2003/02/scrum-keep-team-size-under-7.html accessed at 01.10.2012

Sutherland, J., Schwaber, K., Business object design and implementation: OOPSLA '95 workshop proceedings, 1995, The University of Michigan. p. 118

Sutherland, J., Schwaber, K., Software in 30 Days: How Agile Managers Beat the Odds, Delight Their Customers, And Leave Competitors In the Dust, 2012, Wiley 1st edition, ISBN 9781118206669

Takeuchi, H., Nonaka, I., The New Product Development Game, 1986, Harvard Business Review

Uy, E., Rosendahl, R., Migrating from SharePoint to a Better Scrum Tool, 2008, AGILE Conference, p. 506-512

VersionOne Inc., State of Agile Development Survey 2009, http://pm.versionone.com/StateOfAgileSurvey.html accessed at 01.10.2012

West, D., Grant, T., Agile Development: Mainstream Adoption Has Changed Agility, 2010, Forrester Research Inc.

West, D., Hammond, J.S., The Forrester Wave: Agile Development Management Tools, Q2 2010, 2010, Forrester Research Inc.

Wirdemann, R., Scrum mit User Stories, 2011, Carl Hanser Verlag, ISBN 9783446426603