CHALMERS

Scrum in Mechanical Product Development

Case Study of a Mechanical Product Development Team using Scrum

Master of Science Thesis
in the Master Degree Programme, Product Development

ÞÓRDÍS REYNISDÓTTIR

Department of Product and Production Development
Division of Product Development
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 2013

marel

MASTER'S THESIS

Scrum in Mechanical Product Development

Case Study of a Mechanical Product Development Team using Scrum

Master of Science Thesis in the Master Degree Programme, Product Development

ÞÓRDÍS REYNISDÓTTIR

SUPERVISOR & EXAMINER:
GÖRAN GUSTAFSSON

Department of Product and Production Development
Division of Product Development
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden, 2013

Scrum in Mechanical Product Development

Case Study of a Mechanical Product Development Team using Scrum

Supervisor & Examiner: Göran Gustafsson, Ph.D.

Chalmers University of Technology
Department of Product and Production Development
SE- 412 96 Gothenburg
Sweden
Telephone: +46(0)31-772 10 00

Gothenburg, Sweden, 2013

Abstract

The purpose of this study was to investigate if it would be possible for a mechanical product development team to use Scrum, an Agile Development framework.

The Agile philosophy and methods have revolutionized the software development industry in the last decade, and therefore it was of interest to see if this new way of working would be applicable in hardware development. The study focuses on if Scrum can be applied in mechanical product development, and if it needs any adaptations. Three case studies of Scrum or Agile methods used in mechanical and hardware design and development are presented. The primary case study follows a mechanical product development team through their 7-month experiment of using Scrum. The team in question co-operates with an embedded software development team in new product development. Two supplementary case studies are presented: The first on a single cross-functional hardware team using a Scrum like method, and the second, on an organization re-structuring. The methods used in order to form the case studies were observations, interviews and informal conversations.

The results indicate that it is possible for a mechanical development team to use the Scrum framework, with some minor adaptions of the framework. The team in the primary case made use of all main aspects of the framework, but the deviations were that the team was not cross-functional and a working product or product increment is not produced every iteration. The results also indicate that using the framework improved the team's work and progress, and the team decided to continue using the framework after the experiment. The conclusion is that Scrum can be used in a hardware environment, by mechanical development teams. Some adaptations might be needed, depending on each case, but this is also recommended in the literature to software development teams.

Key words: Product development, Scrum, Agile, Lean Product Development, Hardware, Mechanical

Acknowledgements

Firstly, I specially want to thank Mr. Ásgeir Ásgeirsson, the Innovation Community Director at Marel GRB, for believing in me and giving me the opportunity to work for Marel and getting the chance to do my Master Thesis project there. Secondly, I want to thank Ms. Rósa Björg Ólafsdóttir, Agile Center Manager at Marel GRB, for her full support, insightful discussions and mentorship throughout the project. I also want to thank her for the interest shown in my work and the input she provided. Thirdly, I want to thank IC-Fish, the development teams and its management, for giving me the chance to perform my study with them, and the time we have had working together throughout the experiment. Special thanks to the mechanical team for their patience and allowing me to observe and question them.

In addition to this, I want to thank those who gave their time and provided me with further case study material on the use of Agile and Scrum in hardware product development. Firstly, I thank Mr. Martin Labecker, Line Manager at SAAB EDS, Gothenburg Sweden, for giving me the chance to visit and see the way one of their hardware teams works. I want to thank Mr. Henrik Lindsjö for giving me an insight into the new organization at Andritz Hydro AB, Service and Rehab division in Sweden, and the interesting discussions we had through our correspondence. I also want to thank Mr. Mikael Lundgren, Agile and Lean Product Development Expert, for his correspondence, his further insight into the organizational restructuring at Andritz Hydro and insight into how Scrum is used in hardware and mechanical development in the industry.

I want to thank my family for their unconditional support throughout my studies and project work.

Last but not least, I want to thank my supervisor Dr. Göran Gustafsson, believing in the project and me and seeing the potential of taking it to a higher level. His enthusiasm, guidance and support, and inspired me during the project work. I am also very grateful to him for introducing me to Lean Product Development.

Table of Contents

List of Abbreviations

ACM – Agile Center Manager

IC – Industry Center

ICD – Innovation Community Director

LM – Line Manager

LPD – Lean Product Development

Marel GRB – Marel offices in Iceland; GRB stands for the town name Garðabær

PC – Product Center

PDW – Product Development Workshop

PM – Project Manager

PO – Product Owner

SM – Scrum Master

1 Introduction

Innovation and product development is the cornerstone to organizational growth in the market environment today. But for new product development the stakes are high, requirements increasing and there is a demand of delivering faster in order to beat the competition to the market.

New product development is a complex endeavour, which can often be difficult to handle and challenging to see in advance what the end result will be. There is great uncertainty and unexpected things happen along the way, which affects the scope and direction of a product development project. Therefore these projects can often be difficult to plan, and plans become obsolete soon after their creation.

New ways of managing development emerged within the software industry in the beginning of the 21st century, which revolutionized the way of developing software. Creators of alternative methods in software development came together and formed the Agile Manifesto in 2001 (Beck *et al.*, 2001a). Agile Software Development became a collective term for the new methods, which are all based on iterative and incremental development of self-organizing cross-functional teams, rapidly responding to change (Anon, 2013a). These methods have received good reviews from the industry and many large companies use them in their development today. One of the most widely used methods is Scrum, and is currently being used by companies such as Yahoo!, Microsoft, Google and more (Deemer *et al.*, 2010). Deemer et al. (2010) describe Scrum as "an iterative, incremental framework for projects and product or application development".

It seems that some perceive Scrum as a framework only applicable in software development. However when looking closely at the Scrum framework itself, its cornerstones have no direct link to software development. Therefore it is possible to conclude that Scrum can be used in any kind of product development. However, there is very little research literature available on the use of Agile methods in non-Software environments and case studies on the use of Scrum in these areas are scarce.

In some organizations hardware and embedded software teams work together towards developing a final product. It is of interest to investigate if mechanical teams within product development can also achieve the same benefits of using Scrum as in software development, and if this affects cooperation between mechanical and embedded software teams.

1.1 Purpose

The purpose of this study is to see if mechanical product development teams can use Scrum, with or without adaptations. The aspect of how the teams can synchronize and co-operate with other teams or stakeholders will also be studied.

The investigation was conducted within a product development team at Marel, a leading global provider of advanced equipment and systems for the food processing industry. A mechanical design team within the Icelandic offices of Marel started using Scrum in the middle of March 2013. This team works in close connection with an embedded software team that also uses Scrum. This thesis is a study of the results of this experiment that lasted from March to October 2013. Additional cases are also studied in order to get a wider comprihension of the application of Scrum in mechanical and hardware development.

1.2 Delimitations

The delimitations of the study are that only one main case is studied in detail, supplemented with two less detailed cases. The results might therefore not be universally applicable to other teams or organizations. The timeframe of the empirical study was 7 months, resulting in great amount of data and ideas arising, but not all aspects and perspectives could be accommodated within the scope of the master thesis study. It might also be possible that the author's close cooperation with the team

during the experiment might have an impact on the results. It should be noted that the author worked as a Scrum Master (defined in Ch. 4.2.3) for the team in the later stages of the experiment, and this could possibly affect the conclusions drawn in the study.

1.3 Outline of the Thesis

The outline of the thesis is as follows:

In Chapter 2, the *problem formulation* and research questions will be presented.

In Chapter 3, the *methods* used will be described. These concern the literature review and the empirical study. The scope of the literature review, including the research areas covered, is listed. The empirical study was performed using observations and interviews in order to form a case study.

In Chapter 4, the *theoretical framework* of the thesis is presented. The areas covered are: Agile Software Development, Scrum, and Agile in non-software development.

The *empirical study* will be presented in two separate chapters. In Chapter 5 the *case study at Marel* and its results are presented. The second part of the empirical study is presented in Chapter 6, where two cases from two different Swedish companies are presented. The first case is a result of an interview at SAAB EDS in Gothenburg, where a Scrum like method is used in hardware product development. What is interesting about this case is that there are several hardware expertise working together on one project, and methods from Scrum are used to co-ordinate this team of specialists. The second case is from Adritz Hydro AB Sweden, where one of the divisions has been reorganized, and the mechanical engineering teams work in a Scrum like fashion. This case is interesting as it shows a higher-level use of Agile methods, and retrospective work and planning is coordinated between functional and project teams.

In Chapter 7 the results will be *analysed and discussed* in relation to the theoretical framework on Scrum, and in Chapter 8 the *conclusions* will be presented, along with *recommendations* to Marel and others, along with suggestions of *further research*.

2 Problem formulation

In this chapter, the problem at hand will be formulated along with a short introduction of the organization where the main case study was conducted. This chapter concludes by posing the research questions of the study.

2.1 Introduction to Marel GRB

Marel is a leading global provider of advanced equipment and systems for the food processing industry. Marel offices in Iceland are referred to as Marel GRB; GRB stands for the town name Garðabær. Product development at Marel GRB mainly consists of mechanical design and embedded software teams. What is important to note is that together a mechanical development team and an embedded software team cooperate towards developing a single product. This makes the synchronization between the two functional teams vital.

2.2 The Problem Definition

Before this study was conducted, functional development teams within Marel GRB were working separately with different work systems. The embedded software teams use Scrum while most mechanical teams are using traditional project management methods, sometimes along with visual management. As noted above, to develop a single new product, a mechanical team and an embedded software team need to cooperate. This means that the two teams are highly depended on each other's progress in regards to the final result, the product. In general the sync between the work systems of the two types of functional teams was lacking.

Figure 2.1 visualizes the scope of the problem areas regarding organization of product development teams at Marel GRB. These were identified at the start of the thesis project definition and were divided into three main problem areas.

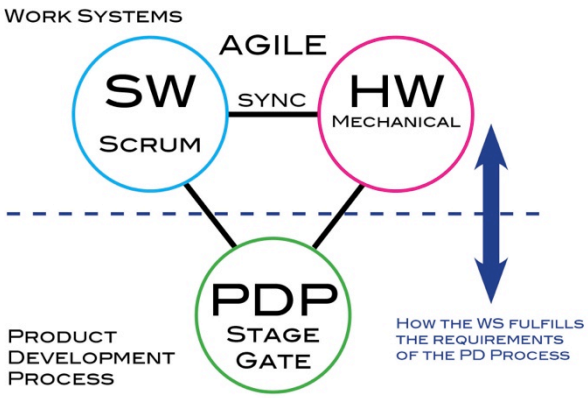


Figure 2.1 – Connection and interaction between different teams and between work systems and the PD process

Firstly, a possibility for improvements of work systems for the mechanical teams was identified when looking to the success of the Scrum software teams. In some cases mechanical teams did not have an actual work system in place. Secondly, the synchronization between work systems of mechanical and embedded software needed improvement. The communication between the two functional teams was often limited, and they were not aware of each other's status. The third issue concerned how to connect the work systems to the requirements of the product development process in place at Marel GRB.

The most pressing issue for the organization was to see if the way of working in mechanical development teams could be improved. After the overall problem scope had been defined, and internal decision was made at Marel GRB that one of the mechanical development teams should start using Scrum. The team in question was co-operating with an embedded software team that was also using Scrum. In light of this, the project scope was therefore narrowed down to focus on the first issue. That is, to see if there is a better way of working for mechanical development teams,

and in this case see if Scrum works for mechanical teams. The second issue, syncing embedded and mechanical work systems, was also studied to some extent. In order to narrow the focus of the scope it was decided that the third issue, connecting the work systems and product development process, would not be covered in this study.

2.3 Research Questions

With the focus of the study defined the primary research question (RQ) of this thesis is:

> *RQ: Can a mechanical product development team use Scrum?*

To support the primary research question the following secondary research questions were also posed:

> *RQa: Does the Scrum framework need adaptation for mechanical teams and do they need more complicated adaptation than software teams?*

> *RQb: What are there critical success factors when deploying Scrum in mechanical teams?*

In relation to the second issue in the problem formulation and for the context of this study, where a mechanical and an embedded software team develop a product together, the following secondary research question was also posed:

> *RQc: What effects does it have that two co-operating teams, an embedded software team and a mechanical team, are both using Scrum?*

The research questions are discussed and answered in Chapter 7.3.

3 Methodology

"A research method is simply a technique for collecting data." (Bryman & Bell, 2011, p.40)

In order to study the topic and answer the research questions posed, information and data needed to be collected. This chapter explains what methods were used to collect data. First the research approach and the overview of methods will be introduced and then each method will be described in more detail in the subsequent sections. In the last section of this chapter the validity of the study is discussed.

3.1 Research Approach

In this section, an overview will be given of the methods used in this study. These will be described in further detail in the following sections. In order to investigate and evaluate if mechanical product development teams can use Scrum, a literature review and an empirical study were conducted.

The topic is relatively novel and scarce evidence was found of previous research in published literature. Thus, there are very few case studies available to build on or compare with. According to Bryman and Bell (2007, p.28) qualitative research "can be construed as a research strategy that usually emphasizes words rather than quantification in the collection and analysis of data". Due to the lack of other cases for comparison, quantitative study was not plausible. In order to contribute to the current literature, which lacks examples of the use of Scrum in hardware development, a decision was made to use a qualitative approach to building case studies.

The research follows two parallel tracks, a literature review and an empirical study as illustrated in Figure 3.1.

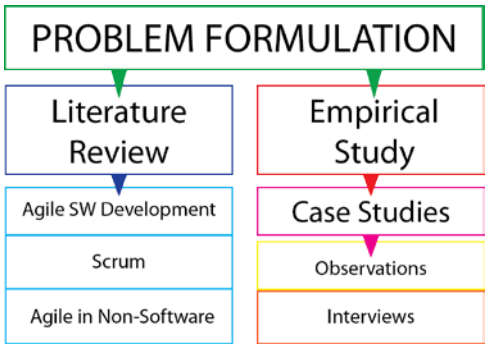


Figure 3.1 – The thesis research structure

The two tracks were conducted in parallel as the experiment in the primary case study experiment at Marel GRB lasted for the main part of the the thesis project duration. The purpose of the literature review was to gather a theoretical base of theory and literature on Agile Development and Scrum, as well as attempt to discover if other cases of Agile practices or Scrum being used in hardware or mechanical development, had been published. The empirical study is divided into two parts; the primary case study conducted at Marel GRB where an experiment of a mechanical development team using Scrum was followed; the second part consists of two smaller complementary case studies, providing insight in to Scrum like methods being used in mechanical and hardware design and development. The three case studies were conducted in order to provide further knowledge and insight to the current literature regarding Agile methods used in Hardware development. The qualitative data that the case studies are based on include observations, experiences, interviews and informal communication.

3.2 Literature Review

A literature review was carried out, based on the problem formulation and the research questions. The theoretical framework is based on literature on Agile Software Development, its history and

concepts; the Scrum framework; and evidence of Agile practices being used in hardware product development. At first, the intention was to bring other perspectives into the research by studying adjacent fields, such as Change Management and Lean Product Development, however in order to focus the content of the thesis these were excluded from the scope.

The main focus of the theoretical framework is literature on the Scrum framework. The most valuable sources used to describe the framework were the early books written on Scrum by its creators, and then the more recent guides and primers that summarize the method and add some elements that have been developed since the first publications were published. These sources were the following: the early publications on Scrum are the two books, *Agile Software development with Scrum* (Schwaber & Beedle, 2002) and *Agile Project Management with Scrum* (Schwaber, 2004). The more recent publications that were mainly used were *The Scrum Primer: A Lightweight Guide to the Theory and Practice of Scrum* (Deemer *et al.*, 2012) and *The Scrum Guide - The Definite Guide to Scrum: The Rules of The Game* (Schwaber & Sutherland, 2011) who's authors developed and sustained Scrum. Other sources were also valuable for the theoretical framework, but these formed the basis of analysis of the primary case study.

3.3 Observations

According to Merriam (2002), observations can range from the researcher being a complete observer to being an active participant. Hennink et al. (2011) refer to these types of observation as participant observation and non-participation observation. Participant observation is then categorized further into four levels of participation: passive, moderate, active and complete participation. According to Hennink et al. (2011, p.184) a researcher conducting a participant observation is required to: "spend a great deal of time in the study context"; "Develop close relationships with people they have not met before; and take detailed field notes." They also state that as a participant observer you therefor need to, among other things: build empathy and rapport with the community being studied; "learn to separate interpretation from observation". The observations conducted in this study varied from moderate to complete participation. Notes were made as data collection, during or as soon after an observation as possible.

3.4 Interviews

Interviews are often divided into three categories: highly structured interviews, or unstructured and semi structured interviews (Merriam, 2002). According to Hennink et al. (2011, p.109) an in-depth interview involves (directly quoted):

- Using a *semi-structured interview guide* to prompt the data collection;
- Establishing *rapport* (a trust relationship) between the interviewer and the interviewee;
- Asking questions in an *open, empathic* way;
- Motivating the interviewee to tell their story by *probing.*

Semi-structured interviews were used several times throughout this study, some of which might be categorized as in-depth interviews. During these interviews semi-structured interview guides were used; a rapport was established through prior communication; questions were asked in an open way and probing was used to elicit further information. Throughout the primary case study, informal conversations with team members and others involved were used to gather information, e.g. on how the team felt about the new work method, what the issues they identified and so on. Notes were made of interesting information gained, but no direct references to individuals within the team were made in the empirical results. All interviews were recorded and transcribed.

3.5 Case Studies

According to Merriam (2002, p.8), a case study is "an intensive description and analysis of a phenomenon or social unit such as an individual, group, institution, or community". What

characterizes a case study is not the topic of investigation, but the unit of analysis, which for a case study is a bounded system "selected because it was typical, experimental, or highly successful" (Merriam, 2002, p.8). The primary case at Marel GRB is of interest on the grounds of being experimental, as there was little indication in the literature of the use of Scrum in hardware development had been researched in detail previously, and even less with the focus on mechanical design and development teams.

According to Bryman and Bell (2007, p.62), "the basic case study entails the detailed and intensive analysis of a single case." They also state that a case can be a single organization, a single location, a single person or a single event. In this study three case studies are presented; the primary case study is about a team within an organization; the first supplementary case study is also team focused; and finally the third case study focuses on a whole organization.

3.5.1 Primary Case Study

The primary case study was performed with a multi-method approach, using a mixture of observations and interviews. The *observations* during the primary case study were participating observations varying from passive to complete participation. During the observations, informal conversations with team members and managers were used to elicit information. Furthermore, semi-structured *interviews* were held with team members, their manager and support staff.

The primary case study was performed in 3 phases as illustrated in Figure 3.2. In the first phase an on location observation was done of the mechanical teams Scrum start-up. The data collection consisted mainly of participating observations, passive to active participation, informal conversations, and making notes of these. The second phase consisted of several remote observations via the Internet using teleconference software, which were mainly passive observations. The third and final phase was conducted onsite, serving as the teams Scrum Master. During that phase the observations were complete participation observations, and informal conversations as well as semi-structured interviews were conducted.

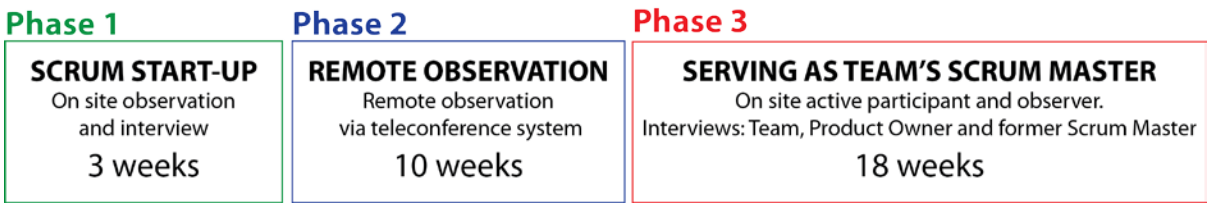


Figure 3.2 - The three phases of the case study

3.5.2 Supplementary Case Studies

In addition to the primary case study, two supplementary case studies were conducted. The first case study was conducted at SAAB EDS in Gothenburg, where interviews with management personnel provided information on their experience of using Scrum for mechanical teams. A passive observation was also made of one team meeting. The second supplementary case was made about the Service & Rehab (S&R) division of Andritz Hydro AB in Sweden. This was done through correspondence with the Product Manager of the division, as well as an external consultant that assisted the organization. The correspondence with the Product Manager was via email and telephone, and therefore no audio recording is available, but the case study was based on notes made during the correspondence and material provided by the Product Manager.

3.6 Validity

According to Bryman and Bell (2007, p.733) validity is concerned "with the integrity of the conclusions that are generated from a piece of research." There are some aspects that might affect the integrity of the conclusions of this study. They are presented in this section.

It should be noted that the author of the thesis worked closely with the team and formed somewhat personal relationships with the team members, as that of colleagues. During the third phase of the case study experiment she served as the teams Scrum Master and provided input into the teams work and how the methods were used, and therefore might have affected the results. But this also provided further insight of the subject at hand, and understanding of the teams work and the character of the team members.

In regards to interviews, two group interviews were held with the team of the primary case study. This might have affected the answers of each individual, but the benefits of the group discussions, spin off ideas as well of the holistic view of the teams thoughts and feelings was deemed more beneficial than what would result from individual interviews. The team was also quite united in opinion and team members seemed to trust each other, and therefore assumed the results were not greatly affected by this. It is still possible that due to the author's personal connection with the team, they will have spared some of their opinions. However, some of their opinions were quite direct and expressed freely and therefor possible that this has not affected the results.

What might also have affected the results was that the author was an employee working for the organization, and wanted the team to succeed. Therefor the author of the thesis was in some way partial.

The interviews conducted at SAAB EDS and the correspondence with Andritz Hydro AB Product Manager was only held with management personnel and not team members. It is therefore possible that the team member perspective is missing from those results.

External validity is concerned "with the question of whether the results of a study can be generalized beyond the specific research context in which it was conducted." (Bryman & Bell, 2007, p.727) It impossible to state that what worked well in the cases presented in the thesis will guaranteed work for other organizations. However, as the result of the thesis confirms that the Scrum framework works for the particular team in the primary case study, and this indicates that others might be able to use it as well.

4 Theoretical Framework

In this chapter the theoretical framework of the thesis will be presented. The topics, literature and cases presented were chosen to best support the primary case study of the thesis.

4.1 Agile Software Development

In the following sub sections Agile Software development will be presented; the history and background; high level principles and values; and some of the methods and practices that are regarded as Agile. Main focus will be put on the Scrum framework.

4.1.1 History and Background

In the middle of the 1990s, the common way of developing software was a heavyweight software development methodology, comprising of complete requirements documentation of design and architecture, followed by coding and then whole product was tested, based on a detailed test plan (Williams, 2012). This traditional and sequential life cycle approach is often called "the Waterfall" (Deemer *et al.*, 2010). It was the general notion that this was the only way of keeping projects on schedule and out of trouble, and if projects were facing problems, the belief was that all would be well if the methodology was strictly followed, but Williams (2012) states that the in reality it was far from being good.

Due to the rigid nature of the Waterfall approach, time estimates were often unreliable, as they did not accommodate for any unexpected events or changes (Deemer *et al.*, 2010). According to Poppendieck (2004) customers rarely or never used over 60% of system features and functions. This can be explained by the fact that requirements were based on the initial customer needs and were not changed throughout the process. Good ideas that came up along the way were not utilized and the process was so strict and set in stone, that these ideas were seen as a threat to the project instead of an opportunity to improve the product (Deemer *et al.*, 2010). This approach also put great emphasis on documentation. This resulted in overly detailed requirement documents that would not be read, and when they were, they often resulted in a misinterpretation of the information. The final result was products that did not reach their full potential and projects that finished late, due to unrealistic time estimates or changes in the project that could not be avoided (Deemer *et al.*, 2010).

In answer to this situation a group of software professionals and creators of alternative methods of developing software, formed the Manifesto for Agile Software Development in 2001 (Highsmith, 2001).

4.1.2 Manifesto for Agile Software development

In February 2001, seventeen people representing several alternative methods to software development, as well as others *"sympathetic to the need for an alternative to documentation driven, heavyweight software development processes"* (Highsmith, 2001) met up in a ski resort in Utah, USA. The result of this meeting was The Agile Manifesto. Although being independent thinkers and even competitors, with their different software methodologies, all seventeen agreed on the wording of the Agile Manifesto (Highsmith, 2001), which is as follows:

> *"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:*
>
> ***Individuals and interactions*** *over processes and tools*
>
> ***Working software*** *over comprehensive documentation*
>
> ***Customer collaboration*** *over contract negotiation*
>
> ***Responding to change*** *over following a plan*

*That is, while there is value in the items on the right, we value the items **on the left** more." (Beck et al., 2001a)*

The Twelve Principles where also formulated, complementing the manifesto and further explaining, "what it is to be Agile" (Agile Alliance, 2013). These are as follows (Beck *et al.*, 2001b), (directly quoted, with added emphasis):

1. Our highest priority is to satisfy the customer through **early and continuous delivery** of valuable software.
2. **Welcome changing requirements**, even late in development. Agile processes harness change for the customer's competitive advantage.
3. **Deliver working software frequently**, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. **Business people and developers must work together daily** throughout the project.
5. **Build projects around motivated individuals**. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is **face-to-face conversation**.
7. **Working software** is the primary **measure of progress**.
8. Agile processes promote **sustainable development**. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to **technical excellence** and good design enhances agility.
10. **Simplicity**--the art of **maximizing** the amount of **work not done**--is essential.
11. The best architectures, requirements, and designs emerge from **self-organizing teams**.
12. At regular intervals, **the team reflects** on how to become more effective, then tunes and **adjusts its behaviour accordingly**.

Williams (2012, p.72) conducted two surveys in 2010 "to weigh the community's view of the principles and use of associated practices." The results showed that the principles and practices are still valid, although Williams (2012) suggested some minor alterations of few of them. But in general the conclusions were:

"Even after almost a dozen years, they still deliver solid guidance for software development teams and their projects." (Williams, 2012, p.71)

4.1.3 The Waterfall Model

The Waterfall approach is a sequential life cycle that was traditionally the most common way of developing software (Deemer *et al.*, 2010). In 1970, Walter Royce wrote an article on a software development model (Royce, 1970) that is later cited and credited as the first formal description of the Waterfall model (Anon, 2013b). In this article he describes a sequential model that is common in software development, which can be see Figure 4.1. This model is what is commonly referred to as the Waterfall model.

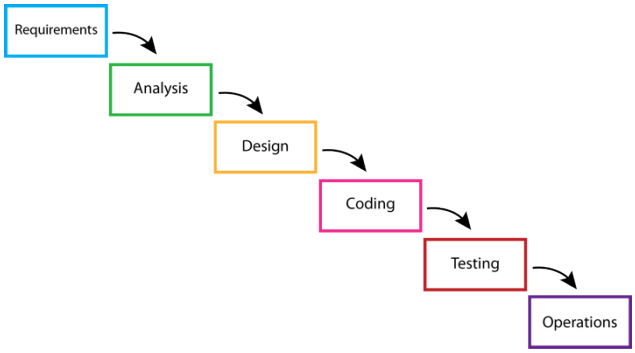


Figure 4.1 Common way of developing software in 1970 (adapted from Royce (1970))

Royce pointed out faults of this model and suggested improvements. He presented an ideal model where there would be iterations between the successive steps of development, but argued that the iteration loops tended to happen between stages further apart, than just successive steps, see Figure 4.2.

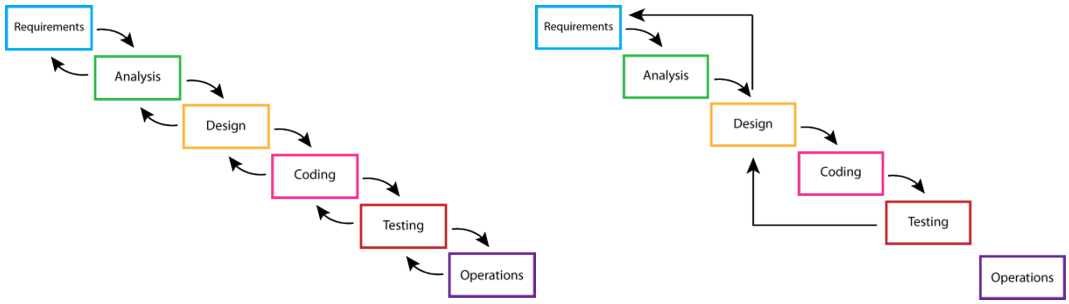


Figure 4.2 – Ideal model for software development (left) and the loopbacks that happen in reality (right) (adapted from Royce (1970))

He then proposes several steps and further improvements to the ideal model for software development.

A case study was conducted at Ericsson in Sweden, where issues with the waterfall model where investigated in the context of large-scale software development (Petersen, Wohlin & Baca, 2009). The study confirmed all of the issues reported in the literature, see Figure 4.3, and even discovered additional issues with the model. The issues regard e.g. high effort and cost in documentation, responding to change is very difficult, solving problems postponed, little opportunity for customer feedback etc.

ID	Issue
L01	*High effort and costs for writing and approving documents for each development phase.*
L02	*Extremely hard to respond to changes.*
L03	*When iterating a phase the iteration takes considerable effort for rework.*
L04	*When the system is put to use the customer discovers problems of early phases very late and system does not reflect current requirements.*
L05	*Problems of finished phases are left for later phases to solve.*
L06	*Management of a large scope of requirements that have to be baselined to continue with development.*
L07	*Big-bang integration and test of the whole system in the end of the project can lead to unexpected quality problems, high costs, and schedule overrun.*
L08	*Lack of opportunity for customer to provide feedback on the system.*
L09	*The waterfall model increases lead-time due to that large chunks of software artifacts have to be approved at each gate*

Figure 4.3 - Issues with the Waterfall model reported in literature and identified by Petersen et al. (2009)

The results of the study were that the most critical issues with the waterfall model were related to requirements and verification. Therefore it was concluded that in case of large-scale development the waterfall model is not suitable. Due to this Ericsson changed to an incremental and Agile development model in 2005.

4.1.4 Traditional vs. Agile

An understanding of the main difference between traditional and Agile Software Development can be accomplished by comparing the concepts of the Waterfall model and to the iterative nature of the Agile methods. Figure 4.4 illustrates the differences between the two concepts.

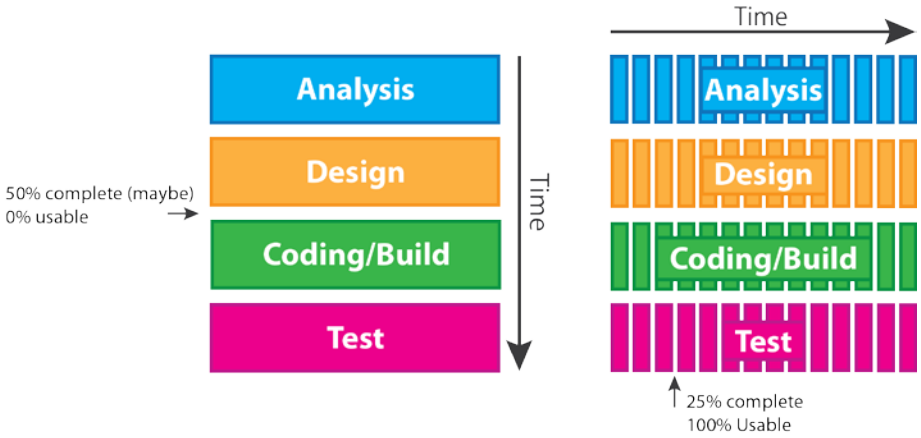


Figure 4.4 - Traditional vs. Agile Software development (adapted from Rasmusson & Pfalzer (2010) and Johnson (2011)

In Figure 4.4, the Waterfall model is presented on the left where the phases of the development process are carried sequentially over time. In the beginning, a software development project is carefully planned and each task needed to complete the software identified (Deemer et al., 2010). When the planning phase is fully completed, the design phase starts, and after fully designing the product, coding and building the software commences. As this is a sequential method, no testing of the software is performed until after the software product has been fully coded. This phase often reveals a great amount of errors that need to be fixed. This method does not respond well to change, as everything is set in stone from the beginning.

In comparison, Agile Software Development is based on an iterative life cycle (Deemer et al., 2010). This means that in one iteration cycle, an increment of the software is analysed, designed, built and tested. By the end of the iteration, the software increment is fully usable. As Figure 4.4 shows, these increments build up over time to form the final product. This method allows for quick and economic reaction to changes and errors that come up throughout the process. Schwaber and Sutherland (2011) note that incremental deliveries of a “finished” product ensure that there is always a potentially useable version of functioning product available.

As Figure 4.4 illustrates, the two processes differ by the usability of the product throughout the process. Halfway through the Waterfall process there is no usable code or product ready, while the Agile methods would have 50% of the final product finished and usable. This is related directly to when testing is performed in the process. It emphasizes the importance of testing continually throughout the process, as it allows for proactive measures when it comes to errors and software bugs.

4.1.5 Agile Planning

An Agile concept that is of importance is that of Inspection and Adaptation (Deemer *et al.*, 2012). This applies to the way Agile focuses on continuous improvements of teams, but this is also a fundamental concept to the planning and prioritization part of Agile methods.

In his book *Agile Estimating and Planning*, Cohn (2005) states that a plan is just a point-in-time guess and many things can affect the plan and make it invalid. These can for example be changes in personnel, technology, user requirements, the competition etc.

At the start of a new iteration a development team and its Product Owner (see definition in Ch. 4.2.2 and 4.2.3) adapt the next iteration to new knowledge that they have gained in the previous iteration. If the changes that have happened are likely to affect the plan, they adjust the plan. An example of knowledge gained could be that the team realizes that a certain type of task is more time consuming than they had planned for at first, or the Product Owner has gained some knowledge of change in customer requirements. Cohn (2005) stresses that this is not to say that Agile teams react to changing priorities in an ad-hoc way, priorities tend to be rather stable from one iteration to the next. However, the possibility of changing priorities between iterations that is a very powerful tool in maximizing the project's return on investment.

Cohn (2005) states the following as an important way of viewing projects and their deliverables:

> *"View a project as a rapidly and reliably generating a flow of useful new capabilities and new knowledge. The new capabilities are delivered in the product; the new knowledge is used to make the product the best that it can be." (Cohn, 2005, p.27)*

Cohn presents this view as opposed to projects being a process of sequentially executing project steps. He further states that this flow of capabilities and knowledge should be used as guidance through the project. Cohn describes two types of new knowledge: new **product** knowledge that further helps understanding what the product should be; and then new **project** knowledge, which is about the team, the risk involved, the technologies being used, etc. Cohn (2005) further states that this new knowledge is rarely accommodated for in project plans:

> *"We frequently fail to acknowledge and plan for this new knowledge. Failing to plan to acquire new knowledge leads to plans built on the assumption that we know everything necessary to create an accurate plan. In the world of software development, that is rarely, if ever, the case." (Cohn, 2005, p.27)*

Rather than focusing on making accurate plans when Cohn further notes that planning should be viewed as a practice of setting and revising goals in order to reach a long-term objective:

> *"When we acknowledge that the result [the product] is both somewhat unknown as well as unknowable in advance, planning becomes a process of setting and revising goals that lead to a longer-term objective." (Cohn, 2005, p.27)*

Different levels of planning

The Planning Onion is a concept that is used to describe Agile Planning. Cohn presents the Planning Onion visually as multiple layers or levels of planning, as illustrated in Figure 4.5.

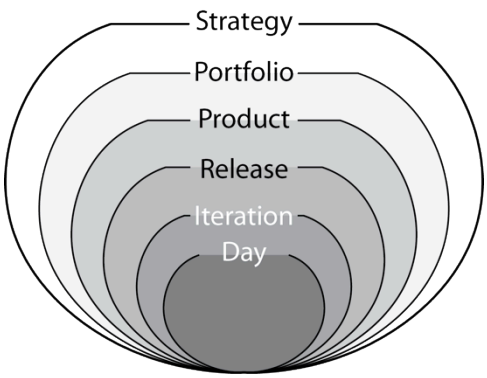


Figure 4.5 - The Planning Onion. The layers Agile teams do planning in at least the Release, iteration, and day levels (adapted from (Cohn, 2005)).

In Figure 4.5 there are six levels of planning listed. An Agile team is mainly concerned with the three most inner circles, or horizons as Cohn (2005) describes them, of the "Planning Onion". These are the Release, the Iteration and the current Day horizons.

In Release planning, the focus is on scope, schedule and resources, and this is normally done in the beginning of a project, but the Release plan is updated throughout the project Cohn (2005) states. Iteration planning is at the next level below, which is conducted at the start of a new iteration. According to Cohn (2005) the Product Owner, identifies work of high priority that the team is to work on in the next iteration based on the work that was accomplished in the previous iteration. The components of the Iteration planning are smaller than the ones of the Release planning. These can be called tasks, while the Release planning components are Stories or Themes. The lowest level is the Daily planning. Most Agile teams have some kind of a daily stand-up meeting, where work is coordinated and daily efforts synchronized. Cohn adds that:

> *"Although it may seem excessive to consider this planning in the formal sense, teams definitely make, assess, and revise their plans during these meetings. During their daily meetings, teams constrain the planning horizon to be no further away than the next day, when they will meet again. Because of this, they focus on the planning of tasks and on coordinating the individual activities that lead up to the completion of a task." (Cohn, 2005, p.29)*

The higher three levels, that normally do not concern the Agile teams, are Product, Portfolio and Strategic planning. In Product planning the Product Owner looks further ahead than the individual Release and plans for the evolution of the individual product. In Portfolio planning, products are selected in accordance to what "will best implement a vision established through an organization's Strategic planning." (Cohn, 2005, p.29)

According to Leffingwell (2010) the results of a Release planning session are used to update the product Roadmap. This Roadmap contains a series of planned Release dates; each Release has a theme, set of objectives, and a prioritized set of features (Leffingwell, 2010). Sometimes the term Roadmap is also used for the result of Product and/or Portfolio planning.

Estimating

Cohn (2005) states that estimates are shared in Agile teams; they are not made by a single individual on the team, or a single expert. As the whole team will do the work, all team members make the estimates.

Work items are often called a User Story or a Story, and the estimates used to express an overall story's size are often called Story Points (Cohn, 2005) or just points (Deemer et al., 2012). Cohn (2005) states that they are relative; the raw values do not matter, it is the relative values that matter. A story that has the size of two points should be twice as big as a story the size of one point. He further notes that there is not predefined formula for defining story sizes; a story-point estimate should be a mixture of the effort, complexity and risk etc. involved.

Cohn (2005) states that he finds that the best way for Agile teams to estimate is Planning Poker, introduced by Grenning (2002). Those that participate in Planning Poker are all members of a development team; Product Owner participates but does not estimate (Cohn, 2005). Each estimator has a deck of cards with a range of numbers, which is often: 0, 1, 2, 3, 5, 8, 13, 20, 40, and 100. The moderator of the planning meeting reads the description of each work item to be estimated, and the Product Owner answers questions that the team might have. When all questions have been answered, each estimator selects a card privately that represents his or her estimate. When everyone is ready and has chosen a card, everyone shows his or her cards at the same moment.

What can happen is that the individual estimates might differ significantly. Cohn (2005) states this is positive, as then the estimators with the lowest and highest estimates explain and a discussion can be held about the contents of the work item. After the discussion all estimators make a re-estimate and the process repeated. If the numbers have not converged by the first re-estimation, the process

is repeated with the goal of all estimators converging on one single estimate. Cohn (2005) gives an example of a second round of estimation where the estimates are 5, 5, 5 and 3, the moderator can ask the low estimator if he or she agrees with setting the estimate to 5 points, as the idea is to be reasonable but not to be absolutely precise.

4.1.1 Success Factors in Deployment and Sustaining of Agile Methods

Pikkarainen *et al.* (2012) presented a study on successful Agile deployment focusing on strengths and barriers. They stated the following in regards to management support and tailored process models:

> *"The analysis revealed the importance of management providing the necessary goals and support for agile development. It also indicated the significance of defining a tailored process model and giving developers the freedom to improve their own agile development process continuously during agile deployment." (Pikkarainen et al., 2012, p.675)*

They further gave four recommendations to managers the following that should be taken into account when planning an Agile deployment, which were (directly quoted):

> *"1) Make sure that the management is committed and gives continuous support for agile deployment*
>
> *2) Make sure that both team, management and all stakeholders has a clear vision, understanding and awareness of agile methods*
>
> *3) Give the needed freedom to the teams to tailor the agile methods to make [them] better fit to their specific needs*
>
> *4) Make the continuous tailoring of the agile based process model also in the organizational level" (Pikkarainen et al., 2012, pp.694–5)*

In regards to sustaining usage of Agile Methods, Senapathi and Srinivasan (2013) identified and presented nine critical factors that have impact: Management Support, Attitude, Motivation, Team Composition, Training, Agile Mind-set, Technical Competence and Expertise, Agile Engineering Practices, and Methodology Champion. They conclude that their findings indicate the following:

> *"Our findings also indicate that the right balance and combination of various factors with an emphasis on continuous improvement will be crucial for achieving true agile sustainability in organizations."*
> *(Senapathi & Srinivasan, 2013, p.119)*

4.1.2 Kanban

One of the methods that will be mentioned in this thesis and regarded as Agile or Lean software development is the Kanban method, developed by Anderson (2010), based on the concept of Kanban from Lean manufacturing.

The Kanban method has five core properties, which are (Anderson, 2010, p.15) (directly quoted):

1. Visualize Workflow
2. Limit Work-in-Process
3. Measure and Manage Flow
4. Make Process Policies Explicit
5. Use Models to Recognize Improvement Opportunities

One of the main elements of the method, Visualize Workflow, involves mapping the work process for a particular team. A Kanban board or wall is created for the team that reflects that work process. The phases of the workflow are represented in columns on the board. A second main element of

Kanban is that each column, or phase, must have a work in progress (WIP) limit, that is, a new task cannot be moved from one phase to another phase without another task being finished first. A visual example of the method is presented by Kniberg (2009), and can be seen in Figure 4.6.

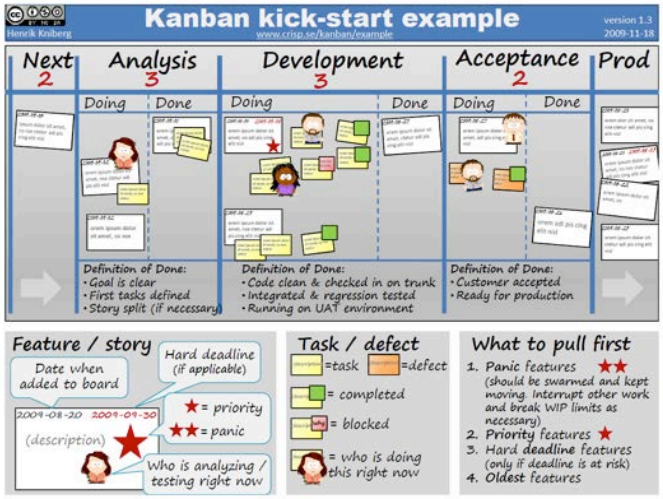


Figure 4.6 - An example of a Kanban wall and its elements, created by H. Kniberg (2009)

This method can be used in a work environment were a repetitive process is used for all tasks the team in question performs, such as an IT maintenance desk where all queries go through the same process. It can be questioned if the method is in applicable in Product Development, where work is not repetitive and uncertain.

4.2 Scrum

> *"Scrum is different. Work feels different. Management feels different. Under Scrum, work becomes straightforward, relevant, and productive." (Schwaber & Beedle, 2002, p.23)*

4.2.1 What is Scrum?

Schwaber and Sutherland (2011, p.3) describe Scrum as "a framework for developing and sustaining complex products." They further define the framework in the following way:

> *"Scrum (n): A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value." (Schwaber & Sutherland, 2011, p.3)*

The framework was developed by Ken Schwaber and Jeff Sutherland and has been used since the early 1990s for the management of complex product development (Schwaber & Sutherland, 2011). Schwaber and Sutherland (2011) note that:

> "Scrum is not a process or a technique for building products; rather, it is a framework within which you can employ various processes and techniques. Scrum makes clear the relative efficacy of your product management and development practices so that you can improve." *(Schwaber & Sutherland, 2011, p.3)*

The basis of Scrum is empiricism; empirical process control theory (Schwaber & Sutherland, 2011). What it implies is that knowledge is extracted from experience and decisions should be based on what is known:

> *"Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from* ***experience and making decisions based on what is known****. Scrum employs an iterative, incremental approach to optimize predictability and control risk." (Schwaber & Sutherland, 2011, p.4)*

Scrum comprises of Scrum Teams and their related roles, events, artefacts, and rules. Each of the framework's components has a special purpose and is essential to the success and usage of Scrum (Schwaber & Sutherland, 2011). In the following sections these will be explained in further detail, beginning with an overview of the Scrum framework, based on The Scrum Primer by Deemer et al. (2010). In the sections thereafter, different elements of the framework will be defined and explained in more detail, based on literature by several authors.

4.2.2 Overview of the Scrum framework

Scrum is an iterative and incremental framework, in contrast to the sequential Waterfall framework that is more conventional in software and product development. The Scrum framework allows for easier and less costly reaction to new information and unexpected change. The method consists of several events, roles and artefacts, which are illustrated in Figure 4.7. (Deemer *et al.*, 2010)

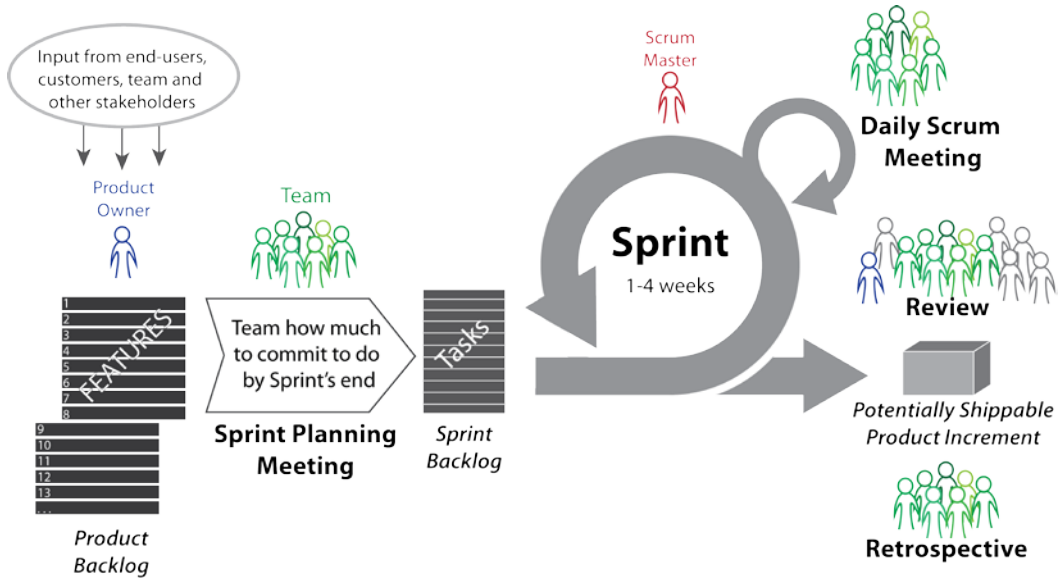


Figure 4.7 - The Scrum process (adapted from Deemer et al. (2010))

There are three roles in Scrum, the *Project Owner*, the *Scrum Master* and *the Team*. Together these three roles form the Scrum Team. The Product Owner is the person that is financially responsible of the product. That person prioritizes items according to the needs of the product and the stakeholders. The Scrum Master is a person with experience in Scrum and he or she supports the Product Owner and the Team in complying with the Scrum rules. The Scrum Master also protects the team from external interruption, in order for the team to be focused on their work. The Team is a group of approximately seven people that executes the prioritized tasks and develops the final product. (Deemer *et al.*, 2010)

The development work is performed in cycles called *Sprints*. Each Sprint can be from 1 to 4 weeks long, and are iterated without a pause. At the start of a Sprint the team commits to what work items they will complete by the end of the upcoming Sprint. The team chooses from a list of items, called the Product Backlog that has been prioritized by the Product Owner. This prioritization is based on input from all stakeholders of the product. When a Sprint has started, no changes can be made to the committed task list, nor can the Sprint be elongated. The Sprint will finish on a set date whether or not the Team has completed what it had committed to. (Deemer *et al.*, 2010)

The idea is that by the end of a Sprint a working product increment will be ready. That is, a part of the final product is done or finished. The definition of "done" is of high importance, as in the case of software development, the product has to be fully tested and integrated and ready for use to be considered "done". (Deemer *et al.*, 2010)

During a Sprint a 15-minute Daily Meetings are held. This is a way for the team members to communicate to each other what their status is. Each team member communicates the following; what they have done since the last meeting, what they are going to do now, and if there are any impediments in their way. This allows other members to offer their help, but further discussions about the details will continue after the Daily Meeting. All other technical or more detailed discussion can be taken up after the meeting. (Deemer *et al.*, 2010)

An important part of Scrum is "inspect and adapt". At the end of a Sprint, a Sprint Review is held that focuses on the product. One part of the Review is a demonstration of the finished product part to all stakeholders. However, the main purpose of the Review is an in depth conversation about the product between the Team members and the Product Owner, on the status as well as getting advice. Another meeting is held at the end of a Sprint, called Retrospective meeting. This meeting focuses on the inspecting and adapting the process and is facilitated by the Scrum Master. There the Team can identify how the process can be improved. (Deemer *et al.*, 2010)

In the following sections more detailed explanations will be given on the Scrum Roles, Events and Artefacts.

4.2.3 The Scrum Roles

There are three roles in Scrum: The Product Owner, the Scrum Master and the Development Team or team. Together these form the Scrum Team (Schwaber & Sutherland, 2011). Schwaber and Sutherland (2011, p.5) state that "Scrum Teams deliver products iteratively and incrementally, maximizing opportunities for feedback."

Schwaber and Sutherland (2011) also state that Scrum teams should be cross-functional. This means that all competencies needed to finish the work should be included in the team; they should not depend on others that are not a part of the team. Schwaber and Sutherland further (2011) note that Scrum Teams are self-organizing. In contrast to getting directions from others external to the team, self-organizing teams choose what the best way to accomplish their work is.

Product Owner

According Schwaber and Sutherland (2011) the Product Owner "is responsible for maximizing the value of the product and the work of the Development team" (Schwaber & Sutherland, 2011, p.5). He does this by managing the Product Backlog, according to Schwaber (2004). Schwaber and Sutherland (2011) state the Product Owner is the only person responsible for the Product Backlog management, which includes clearly expressing its items; prioritizing the items; ensuring the value of Development Team's work; making sure that the backlog is visible, transparent and clear to all, and that it indicates what should be worked on next; and make sure the Development Team understands the items on the backlog. Schwaber and Sutherland further note that the Product Owner can have the Development Team carry out the management tasks, but that the he or she is always accountable.

Schwaber (2004) states that the Product Owner represents all of the project's stakeholders. He further defines stakeholders as someone who is interested in the project outcome. This could be due to the stakeholder is funding the project, will use it or will be affected by it in some way.

Schwaber and Sutherland (2011) state that the Product Owner should be a single person, not a committee, and that his or her decision should be respected by the entire organization. They further state that only the Product Owner is allowed to tell the Development Team what requirements to work from, and the team is not allowed to act on what others might say.

Scrum Master

"The Scrum Master is responsible for the success of Scrum" (Schwaber & Beedle, 2002, p.31)

Schwaber (2004) states that Scrum Master is "responsible for the Scrum process, its correct implementation, and the maximization of its benefits" (Schwaber, 2004, p.142). Deemer *et al.* (2012) describe the Scrum Master as a teacher and coach that makes sure everyone understands Scrum principles and practices; everyone includes managers and Product Owners.

Schwaber and Sutherland (2011) describe the Scrum Master as a servant-leader for the Scrum Team, and his or her responsibility is to ensure that Scrum is understood and enacted. The Scrum Master does this by making sure that the Scrum Team complies with Scrum theory, practices and rules. Deemer *et al.* (2012) agree that the Scrum Master serves the team and press that he or she is not someone that manages the team members, nor is he or she a project manager, team lead or team representative. They further state that the same person cannot serve as both Scrum Master and Product Owner as the focus of the roles is different, and will lead to confusion, conflict and the person might end up micromanaging the team, which is the opposite of self-organized teams. Deemer et al. also note having a project manager in addition to practicing Scrum "indicates a fundamental misunderstanding of Scrum, and typically results in conflicting responsibilities, unclear authority, and sub-optimal results." (Deemer *et al.*, 2012, p.5)

The Scrum Master should also help those external to the team understand which of their interactions with the team are helpful and which are not, and help everyone change these interactions in order to maximize the Scrum Team's value creation (Schwaber & Sutherland, 2011). Managers that are external but connected to the team might need to change their way of managing, and Deemer *et al.* (2012) give an example of using Socratic questioning in order to help the team discover a problem solution, in contrast to deciding what the solution is and telling the team how to carry it out.

The main responsibility of a Scrum Master is to service the Development Team, Product Owner and the Organization by coaching and facilitating events, among other things (Schwaber & Sutherland, 2011). The Scrum Master services the Development Team by coaching them in self-organization and cross-functionality; removing impediments; teaching and leading the team towards high-value product creation (Schwaber & Sutherland, 2011). The Product Owner is served by coaching and teaching in regards to Product Backlog management (Schwaber & Sutherland, 2011). The Scrum Master services the Organization by leading and coaching Scrum adoption; planning implementations; work with other Scrum Masters to increase Scrum applications effectiveness; and "help employees and stakeholders understand and enact Scrum and empirical product development" (Schwaber & Sutherland, 2011, p.7).

Development Team

> *"A team commits to achieving a Sprint goal. The team is accorded full authority to whatever it decides is necessary to achieve the goal." (Schwaber & Beedle, 2002, p.35)*

According to Schwaber (2004, p.143) a team is "cross-functional group of people that is responsible for managing itself to develop software every Sprint." Schwaber and Sutherland (2011) define a Development Team as consisting "professionals who do the work of delivering a potentially releasable Increment of "Done" product at the end of each Sprint" (Schwaber & Sutherland, 2011, p.6). They press that no one tells the team how to do this, and that it is only the Team that creates an increment. They also note the Development Teams overall efficiency and effectiveness is optimized, by the organization structuring and empowering them to manage and organize their own work.

Schwaber and Sutherland (2011) list characteristics of Development teams as the following: Cross-functional; Self-organizing; there are no titles within a team; individual members can have specialized skills and focus areas, but the team is accountable as a whole; and teams do not contain sub-teams for specific domains, such as testing or business analysis.

Regarding team size, Schwaber and Beedle (2002) note that a team should have 7 members, give or take 2. Schwaber and Sutherland (2011) note that the team size should be "small enough to be nimble and large enough to complete significant work" (Schwaber & Sutherland, 2011, p.6). They further note that if a team has fewer than three members interaction decreases, which results in lower productivity gain, and small teams can encounter skill constraints, which might result in them being unable to deliver.

4.2.4 Scrum Artefacts and Definitions

The Scrum Artefacts that are presented in this section are the Backlogs, their items, burn-down charts and then the definition of done. Schwaber and Sutherland (2011) state Scrum artefacts represent value or work in different ways, which are useful for offering transparency and opportunities to inspect and adapt.

Backlog and Backlog items

> *"Product backlog is an evolving, prioritized queue of business and technical functionality that needs to be developed into a system" (Schwaber & Beedle, 2002, p.32)*

Schwaber (2004) defines two different backlogs: Product Backlog and its items and the Sprint Backlog and its items. The Product Backlog is defined as a "prioritized list of project requirements with estimated times to turn them into completed product functionality" (Schwaber, 2004, p.142). The time estimates are in days according to Schwaber (2004). He further notes that the higher the item is on the list, the higher the precision of the estimate is and that the list changes and evolves as the technology or business conditions change. The items on this list can be functional and non-functional requirements, as well as issues. The Sprint Backlog is defined as the list of tasks that the team is going to work on during a Sprint. A Sprint Backlog item is described as one of the tasks that have been defined by the team as required to turn the Product Backlog items the team has committed to, into system functionality. Schwaber and Sutherland (2011) state that the Product Owner is responsible the Product Backlog; that includes its content, prioritization and availability.

Regarding estimation Deemer *et al.* (2012) note that the Scrum framework does not define a special technique for expressing or prioritizing Product Backlog items, and neither does it define a technique or units for estimation. However they suggest some techniques that are commonly used. For effort estimation, what is common is to estimate in regard to relative size, by using "points" as a unit, sometimes called "story points". What is also a common technique is to track how many points are completed each Sprint, and averaging that number. The average is called "velocity". This information can be used to forecast Release dates, or how many features will be completed by a certain fixed date, given the average does not change (Deemer *et al.*, 2012).

Product Backlog Refinement is an activity done during a Sprint, but for splitting, analysing, re-estimating and re-prioritizing, items for future Sprints; not the current one (Deemer *et al.*, 2012).

Definition of "Done"

Schwaber and Sutherland (2011) state that a common understanding of when work is defined as complete must be shared and agreed by everyone involved, those that perform the work and those accepting it. In order for work to be "Done" it must conform to the agreed definition.

Burn-down

Schwaber (2004) describes a Burn-down graph as "the trend of work remaining a across time in a Sprint, a Release, or a product"(Schwaber, 2004, p.141). Data is based on the Sprint or Product Backlog; work remaining is tracked on the vertical axis and the time on the horizontal axis.

4.2.5 Scrum Events

Schwaber and Sutherland (2011, p.7) say that the events in Scrum "create regularity and to minimize the need for meetings not defined in Scrum." Following sections present these events.

Sprint

> *"The team works for a fixed period of time called a Sprint" (Schwaber & Beedle, 2002, p.50)*

One of the early published literature on Scrum by Schwaber (2004) defines a Sprint as a 30 calendar day time-box, where a team works on turning a Product Backlog that they have committed to "into an increment of a potentially shippable product functionality" (Schwaber, 2004, p.142). A time-box is "a period of time that cannot be exceeded and within which an event or meeting occurs" (Schwaber, 2004, p.143). A later definition of Sprint length by Deemer *et al.* (2012), states that a Sprint should not be longer than 4 weeks, and that the most common duration is 2 weeks. They further state that the Sprints continue one after another without a pause and are time-boxed, in the sense that the Sprints end whether or not the planned work has been completed and that Sprint length is never extended. Deemer *et al.* (2012) also note that normally the Scrum teams choose one Sprint length which they keep until they have improved and can start using shorter cycles.

Sprint Planning

> *"Customers, users, management, the Product Owner and the Scrum Team determine the next Sprint goal and functionality at the Sprint Planning meeting. The team then devises the individual tasks that must be performed to build the product increment." (Schwaber & Beedle, 2002, p.47)*

Schwaber (2004) defines a Sprint Planning meeting for one month Sprint as a 8 hour time-box, divided into two 4 hour parts. Schwaber and Sutherland (2011) note that it should be proportionately shorter for shorter Sprints. Schwaber and Sutherland (2011, p.9) describe the two parts of the meeting with the questions that they should answer:

- Part 1: "What will be done this Sprint?"
- Part 2: "How will the chosen work get done?"

Deemer *et al.* (2012) agrees with this definition and describes the planning meeting as one to prepare for the upcoming Sprint, and that it is often divided into two parts, the first one regarding the "What" and the second the "How".

Schwaber and Sutherland (2011) state that the whole Scrum Team creates the Sprint Plan collaboratively. Deemer *et al.* (2012) state that all roles should be present at the first part of the meeting, but the second half is optional for the Product Owner but it should be possible reach him or her for questions.

During the first part the Product Owner presents the items of highest priority on the Product Backlog (Schwaber, 2004). The Team and Product Owner work together to determine how much of the Backlog the team can turn into functionality in the next Sprint, and at the end of the first part the team commits to these Backlog items. During the second part of the meeting the team plans how to fulfil the commitment by detailing the work as a plan in the Sprint Backlog (Schwaber, 2004).

Schwaber and Sutherland (2011) state that by the end of a Sprint Planning meeting the Team should be able to explain to the Product Owner and Scrum Master how they are going to work to accomplish the Sprint Goal and create a product increment as a self-organized team.

Implementing technology and functionality satisfies a Sprint Goal, and it is something that the Development Team keeps in mind while they work. It can be a milestone of the higher-level product roadmap (Schwaber & Sutherland, 2011).

Sprint Review

> *"The Sprint Review meeting is a four-hour informational meeting. During this meeting, the team presents to management, customers, users, and the Product Owner the product increment that it has built during the Sprint." (Schwaber & Beedle, 2002, p.54)*

Schwaber (2004) defines a Sprint Review as a meeting, time-boxed to 4 hours, for a one month Sprint, which is held at the end of every Sprint where there the team demonstrates what was accomplished that Sprint to the Product Owner and other interested parties. Only completed product functionality can be demonstrated. Schwaber and Sutherland (2011) note that the length should be proportionately shorter for shorter Sprints.

Deemer *et al.* (2012, p.13) summarize the definition of a Review as "inspection and adaptation related to the product increment of functionality". They further note that this meeting is not just a "Demo", which is a common mislabelling, although it includes a presentation of the product increment. It should rather be based on the "inspect and adapt" element of Scrum, and should be a conversation between the Product Owner and the Team; for the Product Owner to learn how the Team and Product are getting on, and for the Team to learn what is new with the Product Owner and the market. Schwaber and Sutherland (2011) state that the meetings purpose is to inspect the product increment and make adaptations to the Product Backlog if need be. They further note that this is an informal meeting, and by presenting a product increment, the intention is to elicit feedback and encourage cooperation.

Regarding who attends the meeting Deemer *et al.* (2012) state that it should be the Team, Product Owner and Scrum Master. They note that the Product Owner should invite other stakeholders as appropriate. These could be customers, users, stakeholders, experts, executives or others that are interested. They further note that everyone that is present is at liberty to ask questions and give feedback. Schwaber and Sutherland (2011) just mention the Scrum Team and stakeholders in this regard. Schwaber and Beedle (2002) state that it is the Product Owner, management, customers and users that evaluate the product increment.

Sprint Retrospective

Schwaber (2004) describes the Sprint Retrospective meeting as a 4-hour time-boxed meeting (for a one month Sprint) that is facilitated by the Scrum Master. During the meeting, the team discusses the Sprint that has just passed and determines what could be changed which could make the upcoming Sprint more productive or enjoyable. Deemer *et al.* (2012, p.14) note that as the Review is to inspect and adapt the product, the Retrospective is to inspect and adapt the process and environment. They explain that it gives the team an opportunity to have discussions on what is working and what is not, and all agree on what changes they want to try. The facilitator of the meeting can be the Scrum Master, but they also suggest that it may be better to get a natural outsider, maybe another team's Scrum Master. They further note that there are many techniques available to conduct a Retrospective and that various techniques should be introduced over time. They also press that it is important not only to focus on problems, but also on positive things and strengths.

Schwaber and Sutherland (2011) state that the meeting should be time-boxed by three hours for a one-month Sprint, and proportionally less for shorter Sprints. The explain that the purpose of the meeting is to inspect the previous Sprint concerning people, relationships, process and tools; identify what went well and possible improvements; and finally to create an improvement plan in regards to the way the Scrum team work.

It seems that a description of a Retrospective meeting was not included in the first book on Scrum by Schwaber and Beedle (2002), but is included two years later in Schwaber's (2004) work. In later publications on Scrum it is included.

Daily Scrum Meeting

> *"Software development is a complex process that requires lots of communications. The Daily Scrum meeting is where the team comes to communicate." (Schwaber & Beedle, 2002, p.40)*

Schwaber (2004) describes the Daily Scrum meeting as a short meeting held every day for the team synchronize their progress and work, as well as report any hindrances to the Scrum Master to take care of. Schwaber and Sutherland (2011) state that it should be a time-box of 15 minutes, which Deemer *et al.* (2012) say can be 15 minutes or less. Schwaber and Sutherland (2011) note that it should be held at the same place at the same time every day. Schwaber and Sutherland (2011, p.10) state that this is where the work from the day before is inspected, and that is done by each team member explaining the following:

- What has been accomplished since the last meeting?
- What will be done before the next meeting?
- What obstacles are in the way?

Deemer *et al.* (2012) agrees with this and presses that this should be reported to the other team members of the Development Team; it is not a meeting to report to a manager.

4.2.6 Scrum Values

> *"Scrum is based on a set of fundamental values. These values are the bedrock on which Scrum's practices rest." (Schwaber & Beedle, 2002, p.147)*

The Scrum values are: Commitment, Focus, Openness, Respect and Courage (Schwaber & Beedle, 2002). These are the basis for all work performed in Scrum, and are a foundation for the process and principles. Scrum relies on these values and they are created through teamwork and continuous improvement (Scrum Alliance Inc, 2012). The Scrum Alliance (2012) describes the values as follows:

- **Commitment:** Having great control over one's own destiny, results in becoming more committed to success.
- **Focus:** Focusing only on few things at a time, good co-operation and production of excellent work. Valuable items are delivered sooner.
- **Openness:** As people work together, they practice expressing how they're doing, and what's in their way. What is learned is that it is good to express concerns, in order for them to be addressed.
- **Respect:** Sharing failures and successes, in working together, people start to respect each other, and to assist each other to become worthy of respect.
- **Courage:** Because people are not alone, they feel supported and have more resources at their disposal. This provides courage to take on greater challenges.

4.2.7 Scrum by the Book?

In regards to if the Scrum framework should strictly followed to every letter, Schiel (2012) notes that in his *Guide to Scrum Masters* he is not presenting rules or "the only way to do things". He notes that Scrum is neither a method nor a process, but a framework made up of principles and concepts, pressing that the latter is the most important. He further notes that a development effort's bottom line is customer satisfaction, and if the customer is satisfied with the product, then the development efforts are functioning:

> *"To the extent that agile development has helped you improve your software quality and improve your customer's satisfaction, you can rest assured that you have used agile "correctly.""" (Schiel, 2012, p.65)*

In some contrast to this Schwaber and Sutherland (2011) state that:

"Scrum's roles, artifacts, events, and rules are immutable and although implementing only parts of Scrum is possible, the result is not Scrum. Scrum exists only in its entirety and functions well as a container for other techniques, methodologies and practices." (Schwaber & Sutherland, 2011, p.15)

Schiel (2012) notes that Scrum provides improved performance, quality and productivity, but only to a certain extent. Scrum is not enough on its own, and it must be supplemented with strong engineering practices, infrastructure, behavioural practices and organizational self-discipline.

Agile practices that widely used but not prescribed in the Scrum framework

Some practices that are widely used by Scrum teams are not presented in the core Scrum Framework. Some of these are Agile practices; those relevant to this study are presented below.

User Stories

User stories are widely used within Agile Software Development. Use of or their definition is not described anywhere in the core literature on the Scrum framework. Despite this they are often used to represent Backlog items. Cohn (2004, p.4) defines a user story as:

"A user story describes functionality that will be valuable to either a user or purchaser of a system or software. User stories are composed of three aspects:

- *A written description of the story used for planning and as a reminder*
- *Conversations about the story that serve to flesh out the details of the story*
- *Test that convey and document details and that can be used to determine when a story is complete"*

Visual Management

Another thing that is not described in the core Scrum framework is visual management that is widely used by Scrum teams. Deemer *et al.* (2012) notes that many teams present their Sprint Backlog in the form of a wall sized task board, often called Scrum Board. He describes how the tasks, written on post-it notes, move across the board's columns during the Sprint. These columns are often labelled "To Do", "Work In Progress", and "Done". Deemer *et al.* (2012) also presents examples of a Product Backlog presented on a wall using visual management.

Planning Poker

Planning poker is not prescribed in the core literature on Scrum but is also a widely used technique for relative estimating of stories or work items, using points as a relative unit, which was explained in chapter 4.1.5.

4.3 Agile in non-Software & Hardware development

Although some might think that Agile methods only apply to software development, there are some that suggest otherwise. In a recent interview with a few of the creators of the Agile Manifesto (Jackson, 2012), they were asked what Agile's place is outside of software development. Ron Jefferies answers:

"The key ideas apply everywhere. These include, but are not limited to:

- *Putting people with need in direct contact with people who can fulfil those needs*
- *Populating projects with all the needed people and capabilities to get the job done*
- *Building work incrementally and checking results as you go*
- *Preparing for and influencing the future but not predicting it*
- *Making tasks concrete and quickly finishing them*
- *Giving people work to do and the knowledge to do it, not pushing them around like pawns on a chessboard*

- *Focusing on providing value frequently and rapidly not directly on cost*

Agile ideas are based on how people and organizations work best. There are specialized details we need to know regarding software, just as there are in any other domain. The principles, however, are broadly applicable." (Jackson, 2012, p.61)

In the same interview, Jim Highsmith stated that Agile practices and principles are relevant to any high uncertainty project:

"In any project that faces uncertainty, complexity, volatility and risk, there is a place for agile practices and principles." (Jackson, 2012, p.61).

Andrew Hunt further noted that Agile practices have little to do with software:

"At its heart, an agile approach has little to do with software; it's all about recognizing and applying feedback. The definition of "agile" that Venkat Subramanium and I proposed in 'Practices of an Agile Developer' states, "Agile development uses feedback to make constant adjustments in a highly collaborative environment". Notice there is nothing in there about software."(Jackson, 2012, p.62)

There are some, but few case studies that show that Agile or Scrum have been successfully used in other areas than software development. In the following sections literature and cases regarding use of Agile methods and practices in regards to hardware development environment swill be presented. These include a literature review, performed in regards to aircraft systems integration, and two cases of Agile methods in hardware development.

4.3.1 Agile methods applicable to aircraft systems integration

Carlson and Turner (2013) published a review of selected non-software Agile case studies, with the intentions of finding useful methods applicable and essential lessons useful to aircraft systems integration. They state that current aircraft systems integration is similar to the traditional waterfall model in software development. They emphasize the importance of systems integration and how the cost of change to architecture increases over the development time. The current integration is based on a sequential development methodology, which delays the integration until very late in the process, which results in time consuming rework, increased cost and over running schedule. The sequential methodology relies on large teams to deal with the technical complexity, which due to their size need more communication, as well as the complexity slows down the development speed (Carlson & Turner, 2013). They compare Belie's (1993) proposition of improvement of the aerospace industry, to Agile methods and find that they and Belie's proposal have similarities. Carlson and Turner identify these similarities as focus on team based development and concentration on feature development and rapid iteration (Carlson & Turner, 2013).

The case reviewed by Carlson and Turner (2013) that is of most interest in this thesis is the Johns Hopkins CubeSat case (Huang, Darrin & Knuth, 2012), however they also covered another hardware development case at Sonova. The case has the perspective of Requirements Engineering in regards to developing and manufacturing hearing aid solutions and equipment (Waldmann, 2011). Following are the key lessons that Carlson and Turner took away from the Johns Hopkins Multi-Mission Bus Demonstrator (MMBD) CubeSat case.

CubeSat is a program that allows universities to create small satellites that can be lunched as extra missions with other space lunches. The satellites have to be of standardized sizes and masses, which enable the extra mission to be inexpensive. This case will be discussed in further detail later. The key lessons Carlson and Turner found were the following:

- Empowered co-located teams enable design flexibility.
- Iteration pace is effectively increased by the use of incremental testing, which consequently reveals issues and problems that can then be dispatched.

- “A strong change process is essential.” (Carlson & Turner, 2013, p.471)
- “The flat organization and resulting flexibility with the emphasis on producing functioning articles played a crucial role in Johns Hopkins CubeSat success.” (Carlson & Turner, 2013, p.471)

The general lessons that Carlson and Turner (2013) took away from their case review, which are of interest in regards to this thesis, are the following:

- **Teams** should be diverse with knowledgeable members from all areas. This is needed for shared vision and accountability. Agile works best with small teams. Therefore, it might be best to make changes to the organizational structure, by flattening it.
- **Agile Coach** is important in order to educate teams on the process. Stakeholders also need to be educated about it.
- **Co-location** enabled one case study team to facilitate rapid design iterations, which resulted in flexibility and innovation.
- **Sprints length.** It is recommended to have shorter Sprints in order to receive more feedback from customers.
- **Process training** *“sounds like a worthwhile investment to change the culture, create appropriate expectations, and quickly implement effective methods.”(Carlson & Turner, 2013, p.473)*
- **Incremental testing** to *“keep the teams focused on meaningful work.” (Carlson & Turner, 2013, p.474)*
- **Prioritization of work** “to reduce wasted effort and rework.” (Carlson & Turner, 2013, p.474) Also to focus the work on the areas where most value can be created.

4.3.2 The Johns Hopkins CubeSat Case

This case shows how Agile Systems Engineering concepts were used in the Multi-Mission Bus Demonstrator (MMBD) project, designing and developing two small satellites at The Johns Hopkins University Applied Physics Lab (Huang, Darrin & Knuth, 2012).

As previously mentioned CubeSat (Heidt *et al.*, 2000) is a program that allows universities to create small satellites of standardized sizes and masses, and these small measurements enable “piggy back” lunches, or rideshares. This is done in order to split cost or use up extra available space on launch vehicles (Huang, Darrin & Knuth, 2012). The sponsor of the program was NASA (Huang, Darrin & Knuth, 2012). In the following sections a description or summary of how Agile methods were used is presented.

Agile Manifesto

Huang, Darrin & Knuth (2012) compare their case at Johns Hopkins to the Agile Manifesto. They expand the concept of the manifesto that is working software over detailed documentation working software is the priority, to “that working hardware takes precedence” (Huang, Darrin & Knuth, 2012, p.6). The four elements they identified as the key to the success of the project, and allowed for the process speed are summarized in the following sections.

Individuals and Interactions

They flattened the organization and gave a team of sub-system leads direct contact to NASA, the sponsor, which allowed for quick decisions. These sub-system leads where co-located which allowed for easy and accessible communication between subsystems, and quick decision-making through ad-hoc discussions. The Program manager also had authority to make any changes needed for the success of the system. They used a non-linear process development flow in order to meet cost and schedule.

The concept of “Build a little, test a little, and learn a lot” was extensively used in the project’s hardware development process. This means that testing is done in small incremental steps, instead of a number of design iterations, and many prototypes.

Emphasis on approaches toward a working system

Only one design review was funded in order to maximise the efficiency and value of design review and reviewers. This was called the Only Design Review (ODR). There CAD models or simulations were projected and manipulated in real-time to discuss the design and requirements were also discussed. These ad-hoc discussions were beneficial both to presenters and reviewers. Reviewers were selected in a way that they would add value. Informal peer reviews were also held throughout the project.

There was Minimal documentation and a short signature list for drawings. Plan documents were used as guides not as prescriptive documentation.

Manufacturing and procurement of critical and non-critical parts were prioritised. Procurement of non-critical parts was based on turn-around time and lowest cost. Critical parts in need of high precision and tolerance were made at the universities NASA certified manufacturing facilities.

Collaborative interface with sponsor

NASA, the sponsor, was an active collaborator as a result of having direct access to the sub-system leads. The sponsor participated in key reviews and provided feedback, frequently engaged in face-to-face meetings and participated in status meetings. Cost of idle time was avoided by questions being immediately clarified.

Responding to Change

Fluidic scheduling and tasking approaches were used in order to respond to change. A co-located interdisciplinary team was formed, having special interest in the schedule, cost, and changes to the scope.

Daily team reviews were held in front of a variation of a Scrum board. The Scrum board was used to track tasks and issues and allowed adjustments to the highest priority. It also tracked which team member was responsible for each task, when the task would be completed, as well as issues to be taken forward and decisions that should be made later. The board also made visible tasks that depended on other tasks and highest priority was given to bottleneck tasks. “Responding to change is emphasized, rather than extensive planning” (Huang, Darrin & Knuth, 2012, p.6)

It noted regarding the use of Scrum techniques that it allowed exploitation of variations and uncertainty. It was noted Scrum meetings need to be quick and focused and further ad-hoc side discussions can be taken up after. The board allowed part time members to see the status of tasks and who to talk to for further information. The board visually demonstrated at what speed each program element was going forward.

Recommendations

Following are directly quoted recommendations made by Huang, Darrin and Knuth (2012, p.8) that “should be used to promote agility to sustain new project development:

1. Use **small, empowered team** (deputies) with **direct link to the sponsor** keeping the sponsor engaged and well informed – embedding the sponsor into the team.
2. **Deputize the team** making each lead not only have the authority and responsibility for their subsystem but also for the interfaces and interactions with all subsystems. The **project manager** (sheriff) needs to be a **figure of authority** in the company, reporting to the highest possible level (above the matrix organization).
3. Leverage outside help, or **experts** (posse) on a **part time as needed** basis.
4. **Co-locate the technical leads**, systems engineer, quality/mission assurance manager, and program manager for at least some portion of the day, every day to review all tasks and issues, including cost and schedule.

5. Use **interactive design reviews** to help the program staff uncover issues with their concepts or designs. Select reviewers that can contribute and provide input, ideas, and insight.
6. **Tailor processes to the requirements** of the project; the use of the existing system/processes may be too much or too little. **Minimize the number of recorded documentation**, but important work must be configuration managed.
7. **Analyse and test as early as possible** to mitigate issues, but specifications and requirements will be worked throughout the development."

4.3.3 Wikispeed Modular Car

Another example of use of Agile practices in hardware development is the Wikispeed modular and road-safety-legal automobile, of which a functional prototype was built that travels 100 miles per gallon or 2.35 L/100km. Wikispeed is composed of self-organized teams that complete a new iteration of innovation during one-week Sprints (Denning, 2012). There were first 44 volunteers involved, in four different countries, but have now grown to 140 of part-time working volunteers. These self-organized teams use Scrum to re-evaluate each part of the car during each Sprint, as well as deciding on what highest priority features to work on next. They use Kanban picture boards for optimizing the Sprint work flow (Denning, 2012). Denning (2012) presents five key lessons that can be learned from the Wikispeed experience:

Customer Focus

Denning (2012, p.24) states Wikispeed's process "continuously searches for the best way to test and meet the customer's evolving needs." They achieve this by changing the role of the Product Owner to the person, which at that time has the clearest vision of the product and company. They also use test-driven development, where each week they test innovations using pre-defined test in order to best meet requirements and the Product Owners vision. They also achieve their customer focus by involving customers and asking them for feedback.

Self-Organizing Teams

Wikispeed uses self-organized collaborative teams that are distributed. Denning (2012) states that self-organization increases team velocity and morale. He further states that Wikispeed is single self-organized team of over hundred people, which exceeds the number prescribed by the Scrum framework. However, Denning (2012) states that the results from using a single massive team to innovate is that the communication is cleared and the team velocity is higher. They also use a form of the pair programming practice, to share knowledge. It is stated that this speeds innovation, is less costly, cuts time for non-productive training and reduces need for documentation (Denning, 2012).

Dynamic Linking

They work in short cycles and when they plan Sprints, they have a weekly stand-up call that lasts no more than an hour, which includes the actual call where everyone stating what they have done, and what they plan on doing, a demo video, and then off-line Sprint planning session and issue resolving (Denning, 2012).

The design is modular which makes it easy to make adaptations to the prototype. They accelerate responses to problems, and make use of contract first development, which involves thinking through interactions and interfaces between modules (Denning, 2012).

Values: Transparency and Continuous Improvement

Everyone can see the goals and the big picture at any given time, and anyone can make suggestions or raise flags. This is transparency. They also believe that there is always room for improvement, in order to add more value to the customers. That is continuous improvements (Denning, 2012).

Horizontal Communications

Communications are done peer-to-peer at Wikispeed, as opposed to being told what to do in traditional hierarchy. The Product Owner is not seen as a boss, and can be asked for further explanations of the goals and vision at that given time, which he or she can answer right away as they have a clear vision, as well as being available for others to ask for advice. All this communication is conversational.

5 Main Case Study: Marel GRB

The empirical study consists of two parts. The first part is the case study conducted at Marel GRB where Scrum is used by a mechanical product development and will be covered in this chapter. The second part is covered in chapter 6 where two other cases of Scrum or Agile methods being used in hardware development will be presented.

The Marel GRB case study is divided in to 3 chapters: firstly, the Scrum start-up, where the team was observed during their first weeks of using Scrum; second, the remote observation of the team going through its first Release; and final part is based on observations as the team's Scrum Master.

5.1 Introduction to Marel

Marel is a leading global provider of advanced equipment and systems for the food processing industry. Their product development mainly consists of mechanical design and embedded software teams. The study was carried out at Marel's offices in Iceland, referred to as Marel GRB.

Marel changed their global organizational structure in year-end 2010. The new organization focuses on the markets that Marel serves, which are the Meat, Poultry, Fish food processing industries, as well as further processing. The Organization was divided up into industry centers (ICs) and product centers (PCs). The ICs focus on entire industry processes, while the PCs focus on certain process steps. This is depicted in Figure 5.1.

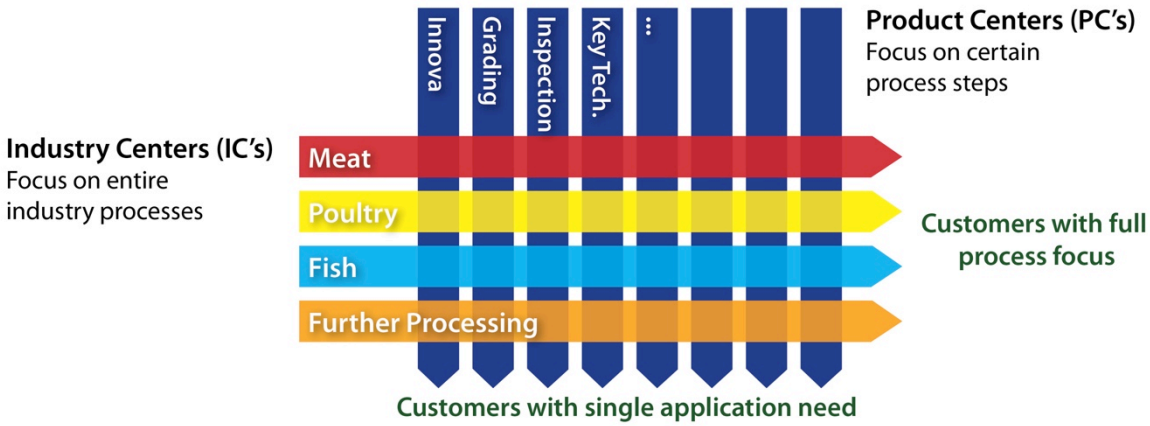


Figure 5.1 - Marel Organization Structure

The Marel offices in Iceland (Marel GRB) are one of many Marel Operating Units and include the three main ICs as well as four of the PCs, see Figure 5.2. For each operating unit there are basic departments such as HR and Finance. In addition to this, three communities support the ICs and PCs. These are the Innovation, Sales and Service communities.

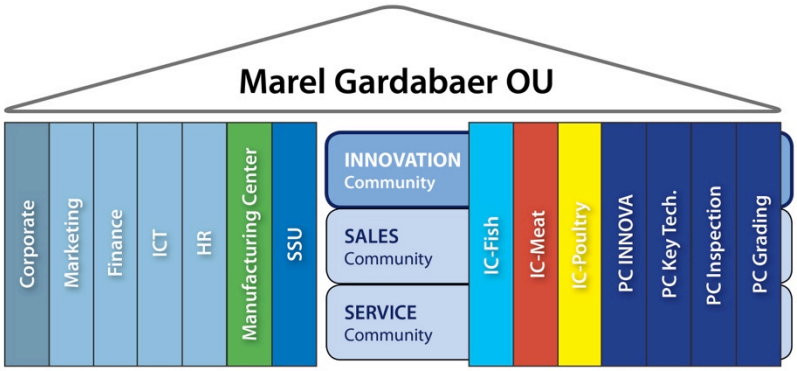


Figure 5.2 - Organization of Marel's Operating Unit in Iceland

Each IC and PC has their own product development team, which consists of a mechanical team and an embedded software team. The case studied at Marel GRB is one of the ICs mechanical development teams.

5.1.1 Agile and Scrum at Marel GRB

The first Scrum pilot team at Marel GRB was an application software team that started using Scrum in October 2009. Now there are eight software teams using Scrum, five embedded and three application software teams. In addition to this is the mechanical pilot team at that is the subject of this case study. The Agile Center is a part of the Innovation Community, which supports Agile and Scrum teams at Marel GRB.

Agile Start Up in 2009

During the Scrum start up in 2009, Marel GRB hired an Agile and Scrum consultant from the company Sprettur (en. Sprint). The start-up team and Agile Champion, now Agile Center manager, visited another company that had been using Scrum for some time. Speaking to people and managers that had first-hand experience of Scrum helped the team and it's Development Manager to see the value and possibilities in using the framework. The current Agile Center Manager at Marel GRB, became a full time Agile advocate at Marel GRB. She later noted that this had made quite a difference for the Agile start up, the culture change and its dispersion within the software teams at Marel GRB that there was a full time position dedicated to the Agile transition and supporting the teams.

The Agile Center

The Agile Center is a competence center within the Innovation Community at Marel GRB that helps the teams applying Agile, by training, coaching, facilitating meetings and serving as Scrum Masters. The Agile Center assist teams in their planning sessions, and supports them in working on their continuous improvement projects. They also support tools such as Jira and Confluence as super users. The Agile Center team consist of two specialists, one of whom is the Center's manager.

Scrum Teams Must-Should-Could (MoSCoW)

The Agile Center has defined what they think teams must do, should do and then could do in regards to using Scrum. The practices that teams **must** have are the following:

- One backlog
- One Product Owner
- One Scrum Master
- Sprints are 3 weeks or less
- Regular rhythm
- Sprint Planning
- Daily meetings
- Sprint Review
- Sprint Retrospective
- Use Relative Estimating
- Team is co-located
- The three Scrum roles as described in the Scrum framework are respected
- Product Owner and Scrum Master are **not** the same person

There are also critical practices that the teams **should** follow but can satisfy in other ways if strictly necessary. The ones that apply to this case are:

- Product Owner and Scrum Master are not part of the technical team
- Burn-down charts for Releases and Sprints are used
- Team has Definition of Done visible and use it
- Teams have working contract

Finally there are practices that the teams could have; they are desirable but not necessary. These are:

- Improvement backlog
- Impediment backlog
- Definition of Ready

General Scrum Start-Up and Education

According to the Agile Center Manager when starting up a Scrum team at Marel GRB, management often does not have full understanding of the training needed when starting teams and want to cut down the time spent on educating the teams. Therefor the initial education time has had to be shortened by half, from what is generally recommended and the timeframe for the initial training is shorter than the Agile Center Manager would like it to be. This means that there is less training in the Scrum framework and Agile ideas.

When starting up Scrum in a team, a session normally starts on Monday morning, and then by lunch on Friday the same week, the team has commenced. During the first day the theory is covered; by going through slideshows and doing exercises. After that they try to go through vision, roadmap and Release planning followed by Sprint planning and then the first Sprint commences. During this session they answer the questions "Why are you starting with this work system?", "What do you want to get out of changing the way you work?" and the answers are listed up. The teams' objectives are created from the answers, describing what results, improvements and benefits they expect to gain from the new work system.

This is a rough description of a Scrum start-up, which can then vary between teams based on each situation. There is a certain schedule and some basic things that are covered, but the start-ups differ between teams; depending on the project the team is working on and how far ahead they manage to plan, as the planning time frame varies.

The Agile Center Manager also noted that its preferred that team kick-off is held when starting to use Scrum. This has not been done for the mechanical pilot team, but is normally done and needs to be done with the team later on. This kick-off is about a three hour meeting with the objective of people getting to know each other better both personally and professionally. People share what is of interest to them in the current project and what they know and what they can contribute to the team. They also share some details about their personal life and describe what they think the perfect team member would be like. The results of this session, the values that emerge, are the input to the teams Work Agreement.

Experiences from Software Development Teams

In discussions with the head of the Innovation Community and an Agile Coach of the Agile Center the following experiences where elicited.

The Agile Coach noted that it is not good when the Product Owner (PO) is a part of the development team. This can result in the two roles being in conflict, which is not good as the PO is supposed to push the team, and this way the PO might start protecting and defending her/himself. The connection to the market might also get lost as the PO might be too technically focused, and the prioritization starts to get distorted and bias.

The Agile Coach further noted that a good PO sets the focus, and is able to say no to stakeholder requests, that do not align with the overall product focus. The PO prioritizes the work of the team and should transfer the market information to the team. She also said that a good PO should have knowledge about the market and the customers, but also the product and the technology. She noted that a person like this is often difficult to find.

The Innovation Community Director (ICD) noted that it is important that there is a technical leader within a team. He also stated that it is important that there is a PO outside the team that owns the product. The PO should make the critical decisions and prioritizes the features and work items for the team. He also noted that it is important that the Scrum Master (SM) is not a subordinate of the PO, and therefore not under pressure from the PO and can say his or her mind. The SM should also be a part of the team; that is, that the SM is friends with the development team. The Agile Center Manager added to this that all three roles should be seen as equals and they should all work closely and well together on friendly terms. The ICD illustrated the Scrum team setup with a drawing depicted in Figure 5.5.3.

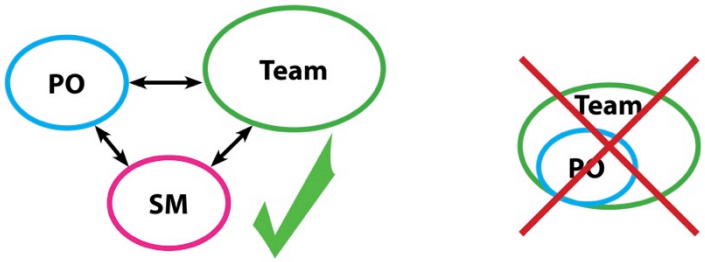


Figure 5.5.3 - Best set up of Scrum Roles. PO should not be one of the development team members.

5.2 The Case Study Introduction: Mechanical PD team using Scrum

The team studied in this case study is the development team of one of the industry centers (IC) at Marel GRB. The IC's development team is a composed of two teams, an embedded software team and a mechanical team. The case study will mainly focus on the mechanical team.

In early 2013, the IC's Product Development Manager (PDM) made a decision that the mechanical Team should start using Scrum. The previous summer the development team as a whole had tried to use visual management based on the Kanban method. This did not work well for the teams and they stopped and following that, the embedded software team started to use Scrum with good results. The Agile Center Manager noted that this time the team got more support and training than for the Kanban initiative. The PDM was their Product Owner (PO), and after seeing how well it worked for them, he wanted the same for the mechanical team. This was a top-down decision; a demand.

Before the Kanban experiment, as well as after it concluded, the mechanical team did not have a defined work system. There were often several projects ongoing, and sometimes there were just one or two individuals working on each one. They were even working on several projects at the same time, and these were often older projects where they had finished designing the product but were being asked for follow up help. The way it seemed to work was that the PDM talked to the mechanical engineers individually, and delegated tasks. There was often discrepancy in communication and lack of co-ordination and information sharing.

During the summer of 2012, a new project had commenced where the development team was a part of a higher-level research project in co-operation with a research organization abroad. A part of the large project was to develop a machine that included a combination of technologies, some which are similar to what Marel had developed previously and then others that were new to the company. This was a new product development project for the IC's development team. During the summer of 2012, there were several projects on-going parallel to this one, and only two out of five members of the mechanical team were working on it full time. The focus started to move more to the new product development project but when the team started using Scrum in late March 2013, the team was still partly working on other projects as well, although the new project had gained primary focus.

The Agile Center Manager conducted and directed the Scrum training, start-up, and implementation, and she served as the mechanical teams Scrum Master (SM). Sometime after the Scrum start-up, which at first had been a demand by the PDM, the SM decided that this should be an experiment and the team would get to decide whether or not to continue using the framework at the end of the experiment. This was done in order for the team to set their frustration and scepticism aside, give the framework a chance and try it out, as they knew they would get to decide on whether or not to continue. The period of the experiment was set to 7 months, and the decision was to be made at the end of October 2013. After three months of the Scrum experiment, the author of this thesis took over as the team's Scrum Master and stayed with the team until the end of the experiment.

Figure 5.4 illustrates the timeline of the Scrum experiment, which is divided up into Sprints and main events are notated. It can be used as a guide when reading the case study.

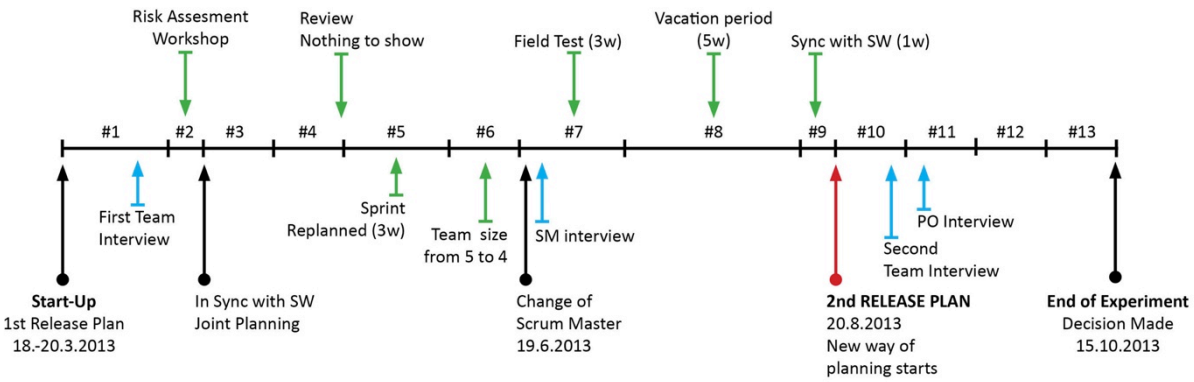


Figure 5.4 - Scrum Experiment Timeline

5.3 Part 1: The Scrum Start-up

The initiative stared on a Monday morning in March. First there was a two-day educational and planning session, where both the first Release and the first Sprint were planned. The first Sprint was observed and at the end of the Sprint the team was interviewed. The findings from the observations and interview will be presented in this section.

5.3.1 The First Training and Planning Session

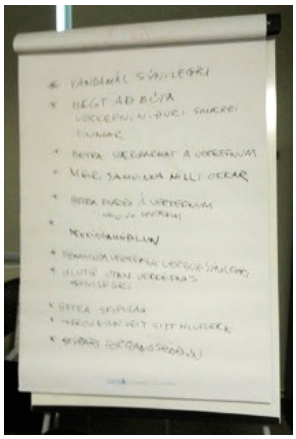
The following subsections present the two-day Scrum start up session for the mechanical team, held in the end of March 2013.

First Day

The Scrum start-up commenced on a Monday morning, the 18th of March, by the Agile Center Manager going through educational material with the team. The team had also attended an introduction session some weeks previous.

After the lecture the team with the help of the SM listed the objectives they wanted to reach by using Scrum. These were as follows:

- Make problems more visible
- Possible to divide project down to smaller units
- Better size estimation of projects/tasks
- More cooperation between us
- More flow of projects/tasks – overall and specific
- Knowledge distribution
- Progress of projects will become more visible
- Items external to project become more visible
- Improved organization
- Everyone knows their role
- Clearer priorities



The SM noted that these were classic examples of objectives when starting Scrum.

Next step was the Release planning. The team had broken the system they are designing down into four parts. For the central two parts of the system they had written down high-level work items, called Release Stories, for a prototype they were going to build. On the back of these story cards they wrote what should be included. These cards were prioritized and their size estimated in points using relative estimating by playing Planning Poker (see Ch. 4.1.5.)

The Release was planned to have seven Sprints each one the duration of 2 weeks, the first one slightly longer to accommodate for the Easter holidays. At the time of the start-up there were 5 individuals in the team. It was assumed that during a 2 weeks Sprint the team would be able to finish

25 points in total. The SM deducted one third as a buffer and the result was that the team of five should be able to finish 17 points per Sprint.

Second Day

The observation for this study started on the second day of the Start-up. At the start of the day, the team and SM reviewed the Release stories written the day before. The SM suggested that a large, 20-point Release story would be broken up into smaller stories. The story involved creating a prototype of a sub-system to be tested. Some of the team members were reluctant to the idea of braking stories up.

The general feeling when observing the team was that some of its members were sceptical about the new framework. This had also been the observation the previous summer during the Kanban experiment, but it felt like they were more positive about Scrum. During the previous summer, they had also been reluctant to work breakdown. But they agreed on breaking down the large 20-point Release story and started by breaking it into 3 parts and then estimated their size. The parts were:

1. Design the model in CAD (13p)
2. Make the design ready for construction (5p)
3. Observe and assist in construction of the prototype (3p)

The SM said that the first part was still too large and suggested that it would be broken down further. The team said that that was not possible, as the as the story involved a design of the machine and they said the work involved was iterative and there the different parts of the model would not design linearly, one after another. They further explained that this is because they need to go back and forth within the model, when one part is changed, it affects another part. One team member noted that if the story would be divided between two Sprints, one engineer would be able to finish a certain percentage of the whole story in the previous one. The conclusion was to divide the story into two, parts A and B. They both had the same description, with different size estimation.

The next step was to prioritize the Release stories (blue cards) and then distribute them onto the seven Sprints, see Figure 5.5. Priority and dependencies between stories were discussed. Work that needed to be done for other projects (green cards) was added to the timeline, and vacation was notated as well. Stories were marked with a pink post-it if external help or deliveries were needed.

Figure 5.5 – Release planning

There were four different external dependencies identified, which were:

- The embedded software team
- The Product Development Workshop (PD Workshop) that builds prototypes for all development teams at Marel GRB. This is often a bottleneck when the workshop is overloaded.
- PC Key-Tech (Operation System Product Center)
- Critical components that needed to be purchased. Here the main risk is long lead times.

At the end of the Release planning session, the plan was presented to the software team. They asked valid questions regarding information they needed to get from the mechanical team in order for them to provide the functionality the mechanical team needed to test the current mock-up.

After the Release plan had been finalized, the Sprint planning started. Then they took the Release stories (blue and green cards), and broke them further down into Sprint stories (white cards). Next step was to make tasks for the Sprint stories, which went well but the SM drove it by asking questions and writing the tasks down on post-it notes. Some stories were interlinked and shared tasks, which resulted in that some were put on one out of the two dependent stories. The SM noted that this is not recommended and that it is best that the stories are independent.

The tasks of the Sprint stories where discussed, what was included in the stories, and what should be tested. The team then estimated how many hours for each task of each story. The SM noted that this is only done when teams are starting to use Scrum, and later on, they become comfortable with only using points. This is only done to help the team to decide how much to include in the first Sprint. After making tasks for the stories and estimating the hours needed, the sizes of stories were also estimated in points using Planning Poker.

After the completing the planning, it was decided that daily stand-up meetings would be held at 9:45 every day. The SM briefly went through slides on teamwork. After this, the team took the cards to their work area, created a wall and put the stories and tasks in their respective places. See the creation of the wall in Figure 5.6 and what the finalized wall with the Sprint backlog and the Release plan, looked like at the end of it in Figure 5.7.

Figure 5.6 – The team sets up the Scrum wall

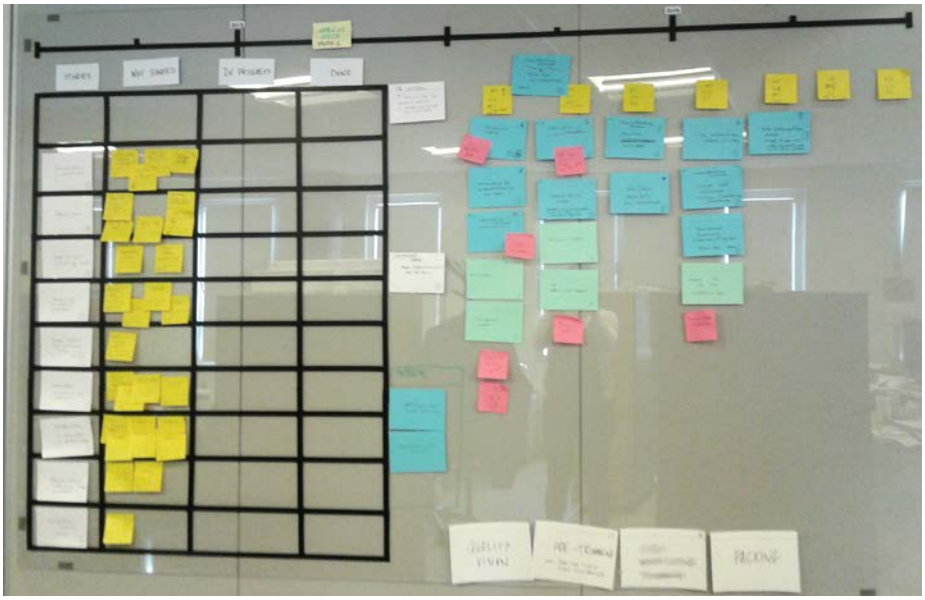


Figure 5.7 – The Scrum wall after the team's first planning session. Sprint plan on the left and Release plan on the right

Observations and Discussions

General observations during the planning meeting were the following:

- The SM did all the writing on cards, and sometimes she needed to ask and push the team for what to write down.
- There were technical discussions about the Release stories, some of which resulted in changes of description and size estimation.
- One team member noted regarding the plan "Well, we know it's going to change", noting that one of the stories was dependent on when they would get a slot in the PD Workshop. The SM noted that they should change the plan if they think it is unrealistic. However, the conclusion was that team members expressed that they felt this was the most realistic plan.
- During one of the breaks, the team was asked to rate how they felt about the Scrum initiative from 1 to 5; one indicating that they had no faith in it and five that they were very positive and certain it would make a difference. The rating was anonymous, and the results were: 4 – 4 – 4 – 3 – 2, with an average of 3,4.
- One of the IC's Operations Managers had attended some parts of the meeting during the first day and attended at some point during the second day as well. That person noted that she felt that the atmosphere had lightened up from the day before in the sense that the team had become more positive.
- One team member expressed his feelings regarding the Review meetings. He did not see the point in inviting everyone at the company to attend. He rather wanted to just invite stakeholders and those involved in the project. He noted that he would rather seek help from other departments when needed and use design reviews, prescribed in the Marel GRB PD process for that. It should be noted that all projects should include at least two design reviews, and this is not always the case.

5.3.2 The First Days of Scrum

This section presents the events observed and general observations during the visit at Marel during the main part of the first Sprint.

Daily Stand-Ups

The first daily stand-up meeting was held the morning after the planning day and was facilitated by the SM. The team stood around the wall in a half circle. One by one, holding a "Talking-token", a pen, the answered the three questions: "What did I do yesterday?"; "What am I going to do today?"; and "Are there any impediments?". Each of them moved a post-it note from the "Not started" column of the wall to the "In progress" column, and put their nametag on the task. The SM suggested that they would show initiative of what task to do, rather than ask their teammates, as they are not supposed to influence each other's choices of tasks, not at first at least. Figure 5.8 shows the wall after the first daily.

Figure 5.8 - The team's Scrum Wall after the first Daily Stand-up

As can be seen in Figure 5.8, two team members chose tasks from the lower part of the wall, and the SM noted that normally the stories were listed in priority, and if the stories priority they chose to work on was high, these stories should have been listed further up in the list. There were two reasons for the team members to choose these stories. The first one was that the stories were seen as a one-person design job for the duration of the Sprint. In order for these to be completed the one team member needed to start on it right away. It was noted that only one person can work on a CAD model at a time, and it is difficult for many people to be working on the same model. The second story ended up at the bottom of the list, as it was one of the stories that were not a part of the main project; a part of a "Green card" Release story (see figure 5.5). The task was long overdue and the deadline had passed and needed to be handed off. In retrospect, this task should have been placed at the top of the prioritized list, if they knew it was the highest priority.

Pink post-its were also discussed at the first daily meeting. The SM stated that they should be used for the following:

1. Unexpected task within a story, that needs to be performed for the story to be completed.
2. Unexpected external tasks, such as disturbances.

Following are the general observation from the daily meetings during the first Sprint:

- The SM told the team to try to take something from the stories listed higher up, as they have the highest priority. Otherwise, get suggestions from the team.
- The SM emphasized that they stand close together around the wall.
- Couple of days into the Sprint, the team initiated the meetings themselves and it was very efficient. When the SM arrived few minutes late, they had finished.
- The Product Owner is very high-level and was not always available and not fully dedicated to the development teams.

Observations and Informal Discussions

During the first site visit at Marel GRB, during the first Sprint, some interesting comments from informal discussions were gathered.

In a discussion with one of the junior engineers, the day after the planning meeting, he noted that he was quite positive towards the initiative. He noted that there were very good discussions during the planning meeting, on what needs to be done and what components needed to be ordered. He also noted that it had happened that one of the senior engineers in the team had said something to him, and then also talked to another team member, and then they realized they were both working on the same thing without knowing it. Later another team member noted that they now had much more of a plan, and that is what they needed, to have it black on white, but not in one of the senior engineers head.

During some discussions, the differences between work in mechanical development and software development were identified and compared. In relation to that, discussions came up regarded ordering of components and CAD modelling. It was noted that the mechanical team has to wait for components to arrive and prototypes to be constructed by the PD workshop. Often the components for prototypes are not ordered until the CAD design is completed. Some components cannot be decided upon until that late in the process, but it should be possible to order some in earlier. It was also noted that only one person could work on a CAD model at a time. It is difficult to divide the modelling and drawing work between many engineers, unless the interfaces between parts are very well defined.

The following was elicited regarding the team members' opinions on Scrum framework:

- One team member thought it was better to go all the way into using Scrum, in contrast to the Kanban experiment the previous summer. He said it would be better to try it, although it had originally been designed for software development, and then see what does not fit. Another one noted that during the Kanban experiment, there were not any rules and they were allowed to adapt it too much. There was also no one that pushed them.
- One noted that Scrum provides better workflow. He noted that there are more rules in Scrum, some of which might not fit mechanical development, but many prove useful.
- One team member noted that he was not sure if the all the work breakdown was necessary, and questioned if these things would not be done anyway.
- During a chat with one of the embedded software developers, it was noted that delays in the mechanical team, affect the software team but it does not stop them working, they just have to adapt to the situation. Nevertheless, they cannot test their software fully until the mechanical team is finished with their part of the prototype. This also indicated that the external dependencies and delays that affect the mechanical team also affect the embedded software team. In this context the SM noted that their work is interlinked and co-ordination is important, and further suggested that it might be best for the two teams to have the same Scrum Master.

5.3.3 End of the First Sprint - The Mechanical Team Interview

The team was relatively positive towards the change, some more than others were, but all of them had some concerns. The main concerns regarding Scrum were the following:

- Work breakdown being to detailed
- Differences between software and hardware development, and they therefore need to adapt the framework to their work and mechanical development
- They wanted to have flexibility, and felt it should be possible to take in new stories or stories from upcoming springs into a current planned Sprint, due to unforeseen events and waiting times

- The project situation was not as they are used to, they stated it was abnormal for five people to be working on the same project. What they are used to was working individually, or maximum two people per project

The benefits that the team could see with the Scrum framework were:

- Daily stand up
- Organize better
- Dedicated time for planning
- Rough planning ahead
- Getting an overview of the project and its status, knowing what is going on

The following subsections present the results of the interview in more detail.

Planning

One team member noted that he found the Release planning that they did when starting to use Scrum, where they deciding roughly what they are going to do for the approximately next three months, was an advantage. He added that a part of this was that there is a dedicated time for sitting down and discussing what they were going to do.

> *"I mean, the advantage is naturally, regardless of brake down and all that ... how it is actually done ... then just this time period that we've planned, however, deciding what circa we going to do the next, what 2 or 3 months, [...] ... it's like, I find that at least to be a certain advantage."*

Another member agreed with him, but questioned if those meetings need to be called Scrum, and another one added that this would just be open communication. The interviewer noted that maybe if a team has a framework like Scrum, then it pushes people to actually do it, and having those meetings. Some of them said that it could be a possibility, and one of them noted that with the framework they had documented the plan and made it visible. Another one added that the benefit with the project brake-down, is that one starts thinking ahead and that some problems might even come up that would not have arisen until later.

> *"Yes ... The advantage of this decomposition is that, one starts thinking about these kind of things ahead. And there might come ... even some problems that would not arise until [later]"*

A team member noted that he liked the Daily stand-up meetings, as long as they were not too long, and people did not go into too much technical detail or chat. He added that it should not be more than a minute per person. The others agreed with this and another one added that there is a risk of starting to talk about something that should not be discussed at this meeting. If the discussion is necessary, a separate meeting should be held. They also noted that the Daily stand-up would provide visibility for others, such as management, and that that is understandable that they might want that. One team member noted that the PO had requested a system like this in order to keep track of progress.

Framework Tailored to Software Development

One of their main concerns was that the framework would not be flexible enough. They felt that the framework would need to be adapted to their work and the nature of hardware development. One of the team members repeatedly noted that the Scrum framework was tailored to software development and there was a big difference between software and hardware development. One of his comments was:

> *" The disadvantage is maybe that this is tailored to software development. And we need to actually tailor it to us ... our needs. Rather than try to adapt ourselves to some software system. That naturally does not work. So we need to break some rules in the regulatory ... of this system."*

The main difference that they mentioned was that mechanical or hardware development dealt with physical prototypes, while software is digital. When a physical prototype brakes, it can take several weeks to get it fixed, while bugs in software can be edited and fixed digitally. In addition this, they noted that they are dependent on others when waiting on prototypes, as they are waiting for parts to be made or components to arrive from suppliers.

The team was asked if mechanical development was something like taking some steps forward, and then some backwards, and software was less like that. Some agreed that this could be a description of how mechanical development works. One also said that software probably has just as many problems, but it might be easier for them to fix. He added that it is possible that 3D modelling in hardware would be similar to software development, but when the prototype is made, the problems take longer to fix.

Dependence on Product Development Workshop and Suppliers

It became clear that the team had dependencies with internal and external suppliers, and often needed to wait for an uncertain amount of time for these suppliers to deliver. The Product Development Workshop (PDW) is one of these dependencies, which manufacture parts and construct prototypes for the development teams at Marel GRB. When ordering from external suppliers, the team is dependent on tem to deliver as well as the internal purchasing department to carry out the order.

As the PDW services all development teams at Marel GRB the lead time for prototypes to be built can vary depending on complexity as well as existing load on the workshop from other teams. Due the varying load on the PDW it can be difficult to estimate the waiting time. One team member indicated that it is often the one that "screams loudest" that is served first.

> *"It really varies a lot, with workloads of the workshop... and who screams the loudest..."*

It was also noted that until now they have not seen waiting times as a problem, as they are aware that these things take time.

Waiting Times and Flexibility

Waiting times are inevitable for the team, like one team member put it:

> *" Yes, there will be waiting times. It is not a question of if; it's just a question of when."*

The team had been challenged, during the Scrum start-up, if these waiting times could not be minimized or eliminated. One of the members noted that there would be waiting times and if people wanted to make them visible that is ok, but they should be aware of that there would always be waiting times. The waiting times that the team experience are firstly waiting for components and manufactured parts to be delivered and a prototype to be constructed; secondly the team needs to wait for new parts to be made if an unexpected error occurs during testing.

What the team was concerned about was that they felt the Scrum framework would not provide flexibility when these unexpected events occur.

> *"Now one feels a bit like being in a straitjacket."*

> *"I think perhaps, like now, if you say: "Ok, we just take this into [the Sprint]", no one knows how many problems we will encounter. Maybe we need to stop or skip something, because we need to construct something [else]. I envision that we need to pull in and out of other [Sprints] ... possibly, that there will more movement on the stuff."*

The flexibility they wanted to have was described in two ways. Firstly, they want to be able to have extra stories to work on, while they wait for the workshop or manufacturing to produce an extra part. Secondly, during construction and testing they might see that a part needs to be redesigned,

they would want to be able to stop working on other stories and start right away on redesigning in order to make new drawings for the PDW as soon as possible.

Next-in Stories

The software teams at Marel use so called “Next-in” stories. When planned work is finished before the end of a Sprint, “Next-in” stories are taken in. The team was asked if it would be useful for them to have “Next-in” stories ready, or if they would always have to create new stories when something unexpected happens. It was noted that “Next-in” stories could be an improvement; that these could be stories intended for the next Sprint and that they would not always need to create new stories if something happens.

> *“It would at least be at least an improvement to have Next-in”*
> *“It could be some [stories] that are just in the next Sprint”*

It was noted that if everything works out well, and they have finished what they committed to in the Sprint they could take in the “Next-in” stories. On the other hand, when problems arise they are not sure how many other stories are affected and need to be put on hold.

The interviewer noted that what the Scrum Master (SM) had been pointing out to them, was that one should commit to the plan that they are creating for each Sprint, and if there is something in the next Sprint that is so important to take into the current one, why was it not planned into that Sprint to begin with. The interviewer also noted that it is understandable that the situation can arise and that it might be possible to have flexibility and take in new stories, but those changes need to be justified. Later on in the interview one of the team members agreed with that.

> *“But I think it is a very good point that it must be, a bit justified.”*

Work Breakdown

In general, the team was concerned about the work breakdown done in Scrum. They thought the breakdown of stories and tasks was too detailed, from what they experienced at the first planning session. This was related to that they felt that there cannot and should not be too many people working on the same story. This concern might stem from the SM noting during the start-up that in Scrum anyone in the team should be able to work on any task. She also mentioned that others should be able to take over tasks that another team member has started.

One of the team members said that he felt the breakdown was too detailed, and did not think this was only the case for this project. He noted that they were maybe breaking down just to break down, and that he did not see how it would help them with the design, in other ways than chasing the system, and question if they should chase the system or if it should be helping them.

> *“[…] and maybe breakdown just to break it down. And I don’t really see that it will help us to design ... except maybe to chase the system. But it is a matter of should one chase the system or should the system to help us? At least [that is] what I think about this. I find this yes, there are things in this that are really quite good, and especially [to] know what's going on and stuff ... but this really detailed [breakdown,] I think maybe not ... "*

Another noted that they could not break work down into too small tasks because several people cannot be working on small details of one part. By the end of the interview, this same team member noted this concern again, but another one noted that they would just need to see how it will develop.

> *A: “Maybe, we do not need to go quite like that down here ... You know, 2 hours, 4 hours.”*
> *B: “Isn't that something that will just evolve now.”*

It could be seen in observations, both during the Kanban experiment the previous summer as well as during the Scrum start up, that the team had difficulties with task breakdown, and tended to want to

have one note for a relatively big project or task. There would just be one note with the name of that large chunk of work. It was suggested in informal discussions with the SM that this could be related to the fact that the team members are used to working individually, which is the old way of working at Marel GRB.

Work Distribution and Collaboration

When questioned if everyone could take over everyone else's tasks and continue working on them, one team member noted:

> *"I cannot see for example that I would run [and take over] the [*product part*] and ... because I have not been in that train of thought, then there would just go too much time in to that. You know, [it makes more sense] that those who have been doing that examine it ... I am quite sure that anyone can make this [*product part*]. It is just a question, you have put in a certain amount [of time], that then you have naturally thought about it a bit, so it would basically just be extra work to be letting someone else come into it, and go check how it is and like that, thinking about this, in order to be able to continue."*

Another team member noted that how it has traditionally been is to try to have at two engineers per project, because it is very good to be able to discuss the work. However, he further added that it does not make sense for all five of them to be working on the same thing, because then they would just get in each other's way. Other team members added to this:

> *"As I like that there is [...] that things are being studied together and that way ideas shared and stuff, then again I feel like ... when there are too many, then it just becomes more complicated."*

> *" If the group has become too large, there will be no consensus about the item. Each one just has their own opinion and ... and one does not know what to do."*

Further, on the subject of collaboration between team members, one of the team members noted that it was important to have someone that one could talk to about possible problems one is facing. This person would not have to do any work, but would provide support and help when needed. He referred to this person as "brother in suffering".

> *"You have to have a "Brother in suffering". But that's not to say that this **brother in suffering** is doing any drawing, [that way]. One wants to be able to seek someone [for support], and thus just root up [the problem] one is maybe just blocked, then you sometimes just need to get […] some discussion started. It usually gets resolved with a very short conversation. […] For me it is more to have them someone to seek to. (Others agree) ... You do it instinctively, one creates like some kind of relationships within the company that you just "Yeah ok, I'll ask this one" or asks that one or ask that one."*

Knowledge Distribution

Peer programming is an Agile software practice which was discussed with the team and they were not sure if this would fit them, and that they had not really worked that way. Some added that they might be doing it just by talking to each other in the office, as he thought that the main point is to learn from each other.

It was noted that one of the expectations of the new framework was knowledge distribution in regards to products. One team member noted that it would be better that more than one person worked on each machine, as knowledge would be more distributed and less risk of the knowledge of the particular machine being lost. He mentioned projects where this had happened and people left with valuable knowledge about certain projects were lost, and that this was very uncomfortable.

> *"No, I mean just like [*Product name*] and [*Product Name*]. There is no one you can bug if something comes up [in those machines]. That feels terribly uncomfortable."*

Abnormal Project Situation

In regards to whether Scrum works for a team like them or not, the team mentioned that they felt that the project situation at the time was very unusual. They were not used to having five engineers working on one machine, and as mentioned before they were used to work in pairs or individually.

Change in the Organizational Structure

In 2010, the organizational structure of Marel changed from being a matrix organization to a product and industry focused structure, and these changes took full effect in the beginning of 2011. Marel had also grown on a global scale through acquisitions and the largest one in 2008, where the company's turnover quadrupled. In the early days at Marel GRB there was just a single department of mechanical design engineers. Then as time went by, people started to become more specialized, different groups emerged for the different products types. What is also important to note is that in the old organization, there were mainly individuals not teams that worked on projects. This was also similar for the software engineers.

After the organizational change, were everything was divided up into PCs and ICs (see chapter 5.1) there is no special functional division for mechanical design engineers. There are just small groups of mechanical engineers for each IC and PC. In regards to there not being a functional community where engineers can seek each other's help and consultation in the new structure, one of the engineers raised his concerns:

> *"Well, I ... one has more worries today that this will be ... there are to be so much smaller groups. Maybe I do not feel it personally because I have been here long enough but ... but I could believe that, for those who are starting today ... the area [or group] they can seek to is rather small."*

An engineer relatively new to the company agreed to this and that this was his experience. One of them also mentioned that they know some engineers from other departments but they do not speak to them about technical problems. They regret not to have a functional division as well, as they are all dealing with the same problems. When asked if this was a danger, in the sense of people in different departments re-inventing the wheel, they agreed. It was also pressed that the ties to colleagues in other ICs and PCs are dwindling.

When asked if the Sprint Reviews could help with keeping contact with other mechanical engineers they did not really see that happening. This could be done at design reviews where people can come and look at and try a prototype and ask questions about it, down in the workshop. It should be noted that these design reviews are not held very often.

5.4 Part 2: Remote Observation

After the start up and main part of the first Sprint, the team was observed remotely for the next five Sprints. This includes the Review and Retrospective of the first Sprint.

5.4.1 Main Events

During this period the team continued using and implementing the Scrum framework. The main events that occurred where: the team synchronized their Sprint cycle time with the software team; a risk assessment workshop was held for the project; the team experienced an uncomfortable Sprint Review where they felt they had nothing to show; one Sprint was re-planned and one team member left the team. The timeline of events can be seen in Figure 5.9.

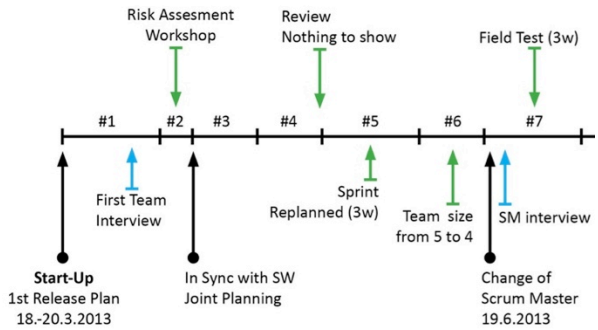


Figure 5.9 - Timeline for Part 2

After the second sprint, the mechanical team synchronized their Sprint cadence with the embedded software team and stared planning in the same room. Doing this enabled them to easily ask each other and the PO questions and present their plans to each other for review. The Sprint Reviews were also held jointly and in the start these were held in the workshop area where the prototype was presented.

Due to lack of clarity, of the project scope and uncertainties regarding technology to be used, the teams held a risk assessment workshop that helped them gain a common understanding, prioritize, and focus the future work.

The teams experienced a Review where the software did not work and they could not show the functionality they wanted to present. The mechanical team found this very uncomfortable, but it seemed the software team was not as affected by it. They noted that from this experience they had learned to include some activities in their plan that they had not done this time.

A Sprint was re-planned due to the request of the team to have a field test to test their concept. This resulted in their previous plan to be redundant and they re-planned the Sprint.

A team member was relocated elsewhere at the end of this period, decreasing the team size from 5 to 4. What marks the end of this period is when the author of this thesis took over the role of the team's Scrum Master.

5.4.2 General Observations

The following are observations noted during Retrospective or planning meetings, which were attended remotely via conference call systems.

Product Owner

It was generally difficult to get a hold of the PO, he had been absent and SM noted that he needed to be more involved. The teams felt they need more support from him, and that he attended daily meetings more frequently. It was also noted that it is not often that the teams were all together with the PO.

During the Planning meeting for the fifth Sprint, the PO attended and there were some very good discussions. The direction was not clear and his input was very important. The teams felt it was bad before this, when he was not there and questions came up and his input was needed.

Sprint Reviews

As previously mentioned, during the second Review meeting where a prototype was being shown, the software team's part did not work. The mechanical team was frustrated regarding the Reviews, both that there are so many people invited and also that they wanted to be able to skip them if they had not anything to show. It is possible that they felt uncomfortable inviting people to Reviews in that situation. In regards to Reviews in general, the team also did not feel they got any feedback, and thought it was awkward how few showed up compared to how many were invited. The software team were not as stressed about the Reviews, and noted that Reviews should not be postponed or

skipped, maybe just delay it by an hour while you try to fix things. The also suggested that the invite could say: “this was the goal - this is what is going to be reviewed”.

Risk Analysis

Some comments and observations were made regarding the risk analysis that was performed in the second Sprint. A comment was made by one of the senior mechanical engineers, which was essentially: *“Just attack the things, and solve the problem. Solve the problem, if it arises”.* However, in answer to this one of the junior engineers added that the senior engineers might know about risks that the junior engineers might not be aware of. A result of the risk analysis performed in the second Sprint the team wanted to have a meeting with the business people and customers, which was held later on.

Testing

Some team members did not see the point in documenting testing results. However, in answer to this, one team member gave an example from another department where people were testing the same things again, because the results are not documented.

Scrum Master

The SM needed to drive the Planning meetings and ask a lot of questions. She also wrote all notes and tickets.

5.4.3 Scrum Master Hand-Over

When the team had been using Scrum for three months and finished six Sprints, the author of this thesis took the role of the mechanical team’s Scrum Master, while the previous Scrum Master stepped back and served as a consultant for the team and the new Scrum Master. At that time, the previous Scrum Master (SM) was interviewed regarding the implementation and the status, as well as recommendations to others thinking of starting Scrum for mechanical design teams. The results of this interview are summarized in the following sections

Implementation

The SM stated that she thought the experiment and deployment of Scrum had gone well. She noted some things have come up and that they might not be following the framework by the book, but she thought it had proven effective for the project in question. She said she was content with having started this and that it had helped overall.

Team Status

At this time, one person had left the team to work elsewhere in the company. She said that generally she thought the team saw that Scrum was beneficial for them. She further noted that the team should get to decide at the end of October if they want to continue or not, as she does not want to force Scrum on teams. They had been generally positive towards the new system, but there had also been some negativity and the team had been sceptical about some parts of the framework. As the decision to continue was time boxed and the fact that they would get to make the decision might have helped them to give the new way of working a chance. The system is a great support for the Product Owner (PO) and that she thought the PO found it helped, and provided him overview. But she also noted that the framework might also put more pressure on the PO, because he needs to be there for the team and show up to the meetings, and that will make visible how busy he is.

Regarding communication between the mechanical and software team she noted that she thought that by starting to have planning sessions together in the same room had helped with synchronizing the teams. This had made it easier to see discrepancies, for example if the mechanical team allocated too few days for the software tests on a new prototype, the software team could comment on that during the planning session.

The team is dependent on others to construct prototypes and purchase critical components. Waiting times and delays have and can affect the team; that they deliver their work on time and their work efficiency. A fault the SM noted about the team was that she felt they had accepted delays and waiting times, and did not follow up with some of their dependencies, such as the procurement department and the PD Workshop. That is something she would like to see change, see them be more aggressive and better see the waste and cost of delays.

The SM felt that the group had been working a bit more as individuals than as a team during the past months. She noted that this could have been something that the SM could have emphasized more, that it is important to inform and involve others in the team. One example of this was that one team member had been sent to visit a customer for the PO, but the others were not informed about it.

The team had also been affected by some disturbances, e.g. when asked to help on older projects and products they were no longer working on. Another example was when the research team, which is a part of the overall development team, needed the teams help when they have used the product prototype for testing.

Hindrances

When asked about what the main hindrances that she had encountered in the implementation of Scrum have been she named the following:

- The team members were used to work as individuals and not as a team.
- The project was large and complicated.
- The Product Owner was very busy and was not always available.
- There were many external dependencies, more than in software teams. She also felt the team did not have enough overview over all these dependencies and synchronization was lacking.

Dependencies

The SM noted that the team was dependent on others, the most significant shown in Figure 5.10.

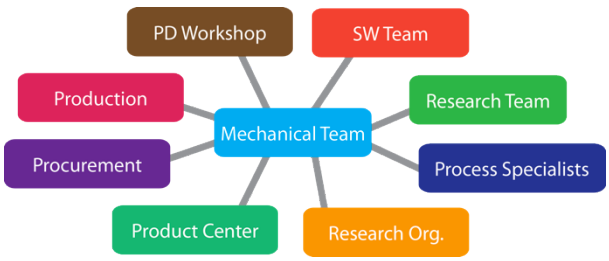


Figure 5.10 - Team dependencies

The closest dependencies are the software team that the mechanical team cooperate with in developing products. The full development team also includes a research team, which they were dependent on in some ways, and the research team depended on them for help during shared testing of prototypes. The team was working in co-operation with a research organization abroad, this might have been more of a dependency for the whole project team, but still could affect the team. As mentioned before the team is highly dependent on the in-house Product Development Workshop. Critical components for the new product needed to be procured and the team was dependent on the procurement team to negotiate and together find the most viable supplier. The team was working with technology that one of the Product Centers within Marel specializes in. Their experience and advice is crucial for the team. The team is also co-operating with Process Specialists within the Industry Center that is interested in the new product as a part of a bigger solution. Their input is important and their opinions can affect the product scope.

Enablers

Regarding what had helped the Scrum transition the SM mentioned the following:

- In-house knowledge and experience of Agile methods. She was the one to start using Agile at Marel and has experience of starting many software teams, both application and embedded teams. These teams want to continue using Scrum and felt their work had improved. This experience is very beneficial to the start-up of Scrum for the mechanical team.
- The team is co-located and sits together. They are also located very close to the software and research team.
- There was lack of direction and there was a need for change.
- The PO wanted this; he is the development director for this Industry Center. The Scrum transition has top-management support.

Framework Deviations

The SM said that the team was not going by the book, but noted that they were following the framework quite well. They have Planning sessions, Reviews, Retrospectives and Daily meetings, and the three roles, Team, Scrum Master and Product Owner. She stated that the deviations from the framework were:

- They did not use User Stories. The stories or work-items were not built on the user experience; they are technical. It was later noted that User Stories are not a part of the Scrum framework, rather an Agile best practice, but the team might not have been aware of that.
- They had changed the Sprint lengths a few times. This was decided before hand but not during a Sprint and was reasoned for. The SM noted that this was often due to some external constraints, when a prototype is ready for example. The Sprint lengths were also changed in order to synchronize with the software team and during holidays.

Co-operation with Software Team

The SM stated that it was possibly easier for the team to start using Scrum as they are co-operating with a software team that was already using the same framework. That both teams were using the same framework with the same cadence made it easier to co-ordinate the two teams. As the teams have joint Planning and Review meetings the SM stated that this gave the feeling of the teams developing a shared product and that it would give the PO better overview and allowed him to co-ordinate better, as he would know the exact status every other week. It also becomes clear sooner if one team is behind schedule and then the PO and the other team can take action if needed.

The Future

When asked what was needed in order for the implementation to be successful and that the team would decide to continue at the end of the experiment, the SM said that it would be important for the project to be successful and that the system would prove itself.

Regarding improvements that the team needed to reach for it to be a successful implementation, she noted that they needed to become bit more of a team and improve information flow within the team. The SM also noted that information flow between team and PO needed to improve.

One of the things the SM said she would like to see was the team functioning without the Scrum Master being there. Other teams that she has coached continue working by the framework although she might have been away. She presses that the Scrum Master role is vital but the team should be able to be self-sufficient when the Scrum Master is not there. The SM noted that the mechanical team still needed a Scrum Master to drive them.

Regarding self-sufficient teams, the SM noted that often all that was needed was one team member that saw the benefit and had the organizing skills and character to drive the system forward.

Regarding the future of Scrum in mechanical teams at Marel, the SM said she believed other teams might want to make use of it, and noted that some had shown interest in the results of the experiment. There is definitely potential for other teams to see the purpose in using Scrum. She also noted that she would recommend having the same Scrum Master for the two co-operating teams, the embedded software and mechanical teams.

Organizational Change Needed

Regarding what might need to change within the organization in order to make it easier for mechanical teams to use Scrum, the SM listed the following:

- Top management support is needed both on local and global level. Top management and employees need to see the business value in Agile practices and Scrum and the need for Scrum Masters and Agile Coaches.
- Need for deeper knowledge about Agile methods and how they can help the business succeed.
- Better training for Product Owners (POs) is needed.

The SM said that top-management support is crucial in order for Scrum initiatives to succeed. The SM noted that it is important that there is an understating that the company needs people that want to serve as Scrum Masters and take care of the teams. The SM felt that some might think of Scrum Masters as non-value adding overhead; yet another middle manager. However, the SM disagreed and said that there is business value in the work of Scrum Masers as they coach teams in order to deliver the right product faster, sooner and better. If that is not business value, the SM said she did not know what it was.

The SM stated that she felt that some POs lack deeper understanding of their role, which is mainly to prioritize the work of teams, based on business value. The SM thought that not all POs are doing this, and feels that the connection to the end customer is sometimes lacking. A method that is rarely used but the SM would like to see more of, is to compare business value of features to the effort needed to it out. The Agile Center intended to have advanced training for the POs in the near future.

Recommendations to Others

The SM noted that one cannot be too orthodox regarding the framework, but the SM has two rules that she never gives discount on for any Scrum team at Marel GRB:

1. The PO and SM can never be the same person.
2. There will always be Reviews at the end of Sprint, no mercy.

The SM stated regarding the Reviews, that if the team does not finish what they committed to, then they still have to stand up at the Review and say that they did not finish and they have nothing to show. She said that teams rarely do that twice in a row. This pushes them and the PO to plan better and be more realistic about what they commit to.

Scrum for Mechanical Teams in General

When asked if the SM thought that Scrum could generally work for mechanical teams, the SM noted that it might depend. She said it is not possible to say that it will not work for mechanical teams but it is difficult to say how much value will be added.

The SM said that in the Marel case there were definite improvements as there was no defined work system before the change initiative. For other organizations, it might depend on what system is previously in place. It is also not possible to say that Scrum is better than other tools for mechanical development, but the SM noted it is generally stated that Scrum is well suited in cases where the uncertainty is high, it is not exactly clear what the team is going to do and it is difficult to plan far ahead. This is because Scrum is an iterative process that allows for re-planning due to changes and re-evaluation of plans. However, the SM also noted that there would always have to be someone with vision and direction for the product.

The SM said that she could not imagine a project like the one at Marel to be broken down into detail and planned for the whole project period. A lot of effort would be needed and the plan would be redundant right way. I would be a lot of waste to plan a project like this one in detail like that. She agreed that that would be similar as the Waterfall process was used in software development before Agile.

When asked what benefit Scrum can provide to mechanical teams the SM mentioned the following:

- Way of planning in high uncertainty projects.
- Team of specialists with ownership of the product they are developing. System where teams develop to be self-sufficient with ownership of the product in development is much stronger than a system where a project manager delegates tasks.
- The framework provides the team and its management with fast feedback, and visibility and overview of the status of completed and unfinished work.
- Problems become known sooner. The problems will always be there but the system makes them visible, and they can be dealt with sooner.
- Visual management helps with identifying bottlenecks, for example, when tasks are often on hold due to the same reason.
- The Retrospective meetings provide a platform for continues improvements.

Adaptations to the Framework

When asked what adaptations mechanical teams might need to make in order to use the framework the SM listed the following items:

- Variable Sprint lengths might be needed in order to accommodate for e.g. field test.
- That User Stories are not used. It might not be possible to base definition of stories or work tasks might on the user experience.

She stated that otherwise, the teams should be able to use the other elements of the framework, such as Daily stand-ups, have a backlog, and do planning and tasking stories as well as use relative estimating. She noted that the mechanical team had been sceptic about relative estimating and Planning Poker but that it worked well for them and they were using it. The SM stated that relative estimating is very helpful and the benefit is that it is so quick, and afterwards it is possible to easily calculate the cost of each point.

What is Needed to Be Able to Implement Scrum

The first point the SM noted in regards to what is needed for a successful implementation of Scrum in mechanical teams is that the team cannot be against it; the team needs to want the change. Secondly, it is important to have someone with experience and knowledge of the framework and its background to implement the framework in the team. This could be the team's Scrum Master or an external consultant. It is also important that the Scrum Master understands the product and how the team works, as well as the teams connections and dependencies. It is also important that the organizational culture allows for trust between colleagues in teams; that the people are not afraid of more senior individuals in the team, that the culture is open where people can stand up and tell others about what they are doing and that they not afraid to make and admit to mistakes. Relaxed atmosphere in the team regarding these issues is important.

Furthermore, the SM noted that teams would need to change from working individually to working as a whole in one team. For people that prefer to work alone, this type of framework is not very comfortable. It is also important that the company does not have individual bonus systems, and there should be no special treatment for individuals. The team should be more important and awarded as a whole.

The SM noted that it is helpful that there is experience of Scrum in the software community of the company, where the intentions are to start Scrum for mechanical teams. But for companies that

have no experience of Scrum, the SM recommended to visit other companies with experience and allow the people that are to start using it, talk to people that are using the framework. Seeing it work elsewhere and getting first hand experiences from others helps build trust in the framework, both for managers and team members. In addition, if no one has experience of Agile or Scrum in the company, the SM recommended hiring consultants for the first implementations. They have experience and can build on previous cases and that makes it easier for people to trust them and give the framework and new methods a chance. The SM also noted that having a framework advocate within the company is important, and that it is seen as a full time job.

For the mechanical team at Marel it helped that the SM told them that this was an experiment and they would get to decide whether or not to continue after a certain amount of time. The SM recommended four to six months for the time box until the decision is made.

The SM also noted that it is vital that the teams, Product Owner and management, are educated and trained in the framework as well as the ideology behind it. It is also important that people that work with the team are familiar with the framework and the ideology. The SM noted that this is often a cultural change due to the new ideology.

The SM further noted that patience and perseverance is needed in an implementation like this. There will be difficulties, and the team might not agree with what the person who facilitates the implementation suggests. However, that is also a part of it all; no one should be telling the team what to do, that way the team will not mature.

Organizational structure

It was mentioned that many companies are matrix organizations, where functional lines provide resources to project teams. The SM noted that Scrum builds on permanent teams and project flowing to teams instead vice versa. Her opinion is that this is both more effective and easier to manage. This also forces the organisation to prioritize and compare projects. She suggested that the lines could have permanent teams, for example a software team, and when a project needs software resources, the Line Manager provides a team not individuals.

5.5 Part 3: Observation as a Scrum Master

During the last four months and eight Sprints, the author of this thesis served as a Scrum Master for the mechanical team.

5.5.1 Main Events

The main events during this period were that the team did a field test, vacation period and a Release planning session. At the end of the period, the team took a decision on whether to continue using Scrum or not. The timeline can be seen in Figure 5.11.

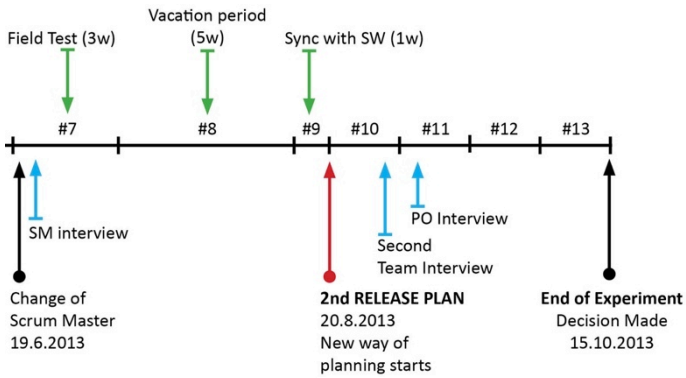


Figure 5.11 – Timeline for Part 3 of the case study

A field test was done on the prototype, where it was tested in production plant with actual raw material. The team got some helpful results, which proved some of their concepts to be good, and saw where improvement was needed.

After the field test had been performed, the main vacation period started and it was decided to have a long Sprint of 5 weeks. After the vacation Sprint was over a one-week Sprint was held in order to synchronize with the software team. After this, a Release planning session was held, to clarify the scope and create a rough plan towards having a final prototype built by the end of the year. A new way of doing Sprint plans emerged from the Sprint planning meeting after the Release planning session. The Release planning session and this new way of planning is presented in this chapter.

Formal interviews in regards to this thesis were conducted with the previous Scrum Master, the Development Team and the Product Owner within this period. When these interviews were held, is indicated on the timeline in Figure 5.11.

At the end of Sprint no. 13, the team made a decision on whether or not to continue using Scrum.

5.5.2 General Observations

Following are some general observations that were made during this period.

Daily Meetings

The team does not strictly follow answering the three questions of what they did yesterday, what they are going to do today and if there are any impediments, in that particular order. But the general feeling was that all this was covered during the meeting, but there is a possibility that some parts might be forgotten.

It was also observed that the team is quite independent during these meetings and drive it themselves. The Scrum Master stands on the side-line, and asks some questions in between.

Stories

The stories are not user oriented “User Stories” as has been mentioned, and are rather work tasks or work packages. However, the terminology for them has been Stories throughout the project.

Stories were written during the Sprint planning meeting, and had not been prepared beforehand. The team did not clearly define acceptance criteria or detailed requirements for the stories during planning, and they were also opposed to doing so. This was something the first Scrum Master had and was trying to get them to do, define what indicates that a story is completed. What they did instead was to write down the tasks that they intended to complete within the Sprint, in relation to that story. They do not have a definition of “Done”.

Planning

Midway through the period it was observed that the team was getting the hang of planning and was quite positive in regards to it, and seemed to find it useful. Some things that might be missing are: the lack of acceptance criteria and definition of “Done” as noted above; the Scrum Master wrote down all notes, in most cases.

There was no active burn-down chart used for Sprints, the statistics were only recorded into the Jira backlog system. After the Release Plan was established, the Scrum Master updated a burn down chart after each Sprint.

Co-ordination with the Software Team

During the Sprint when the field test was conducted, there was an incident of miscommunication or lack of communication. The mechanical team was getting the prototype ready to be transported out to a field test, and one of the mechanical team members had started unplugging the machine. Then one of the software team members called him and said that they needed to do some further testing before the machine would be transported, so he stopped. Then another mechanical team member, that was not aware of this, came down to the workshop and finished packing the machine, and it

had almost left the house before the two mechanical team members talked together about the software testing. After this incident, the Scrum Master went regularly between the two teams and checked the status, which might have improved and increased the communication. It seems that there had been some inconsistency between the plans of the two teams.

Scope Creep

During the whole project, there had been some uncertainties and clarity missing in regards to the scope, and the PO changed it several times. The team was frustrated with these turns in the project. It was later noted by the PO that it was not possible to define the scope clearly in the start of the project as there were so many different opinions about it, in-house and from different customers, and it was not obvious which opinion should be followed.

First Scrum Master – What Would She Have Done Differently

At some point in an informal conversation the Scrum Master that initiated the Scrum experiment, noted that there were some things that could possibly have been done differently in the implementation process: involving the team members more in the initiation; training the PO better, and activate him sooner; and more focus on training the team in work breakdown.

5.5.3 Release Planning & New Sprint Planning Method

At the beginning of Sprint no. 10, both teams had a full day Release planning session. In-house stakeholders were invited, and the IC's management emphasised the importance of the projects success. There was also time pressure put on the teams by management. The team got a final decision on what should be included in the next prototype form the PO. During the planning session, the team did a functional decomposition of the machine, and estimated the effort for each sub-component. These were then prioritized and very optimistic goals were set for finishing the design of the prototype, due to the time pressure. This meant that the team would have to overload each Sprint in order to reach the goal; they would need to exceed their average number of points per Sprint.

The day after the Release planning session, Sprint 10 was planned. This was a quite heated meeting, and the frustration was high in regards to the demand of detailed work breakdown. During the meeting a compromised way of planning was done, which the team turned out to be happy with.

Release Plan

The result of the Release panning session was the Release plan that can be seen in Figure 5.12. The blue cards are the "Release Stories" as the team called them, which in this case present different sub-systems or components of the machine.

Figure 5.12 - Release plan made at the start of Sprint no. 10 (image taken in a later Sprint).

The yellow post-its at the top represent two-week Sprints, and the Release stories (blue cards) were distributed over the Sprints according to their prioritization and size. What needs to be noted is that this might not be the most correct representation of the plan, as these stories will be worked on in parallel throughout several Sprints. However, it was decided to present the plan in this way of prioritization. This was because the wall would have been filled with cards if all these Release stories had been broken up into more cards and distributed correctly over the Sprints. This will be further explained in regards to Sprint planning in the next section. A Release burn-down graph seen on the right hand side of Figure 5.12 represents the progress over the whole Release, five Sprints in this case, and was updated after each Sprint has been completed.

Sprint Planning

The Planning meeting the day after the Release planning session was a tense meeting, as mentioned above. The team was frustrated with the demand to break down the Release stories further and divide them over the Sprints. In the spontaneous act of the Scrum Master trying to meet them halfway, the new way of planning emerged. The team turned out to be happy with this way of working, possibly as it did not pressure them to break the Release stories down in advance, just during the planning sessions. This way of planning is presented in the following section.

The New Sprint Planning Method

As mentioned above, most of the Release Stories (blue cards) of this Release Plan needed to be developed in parallel, and would therefore spread over several Sprints. What was done during the Sprint planning was to define tasks for each Release Story the team was going to work on during the upcoming Sprint, and a Sprint story was created around those tasks. In a way, parts of the Release stories are “dragged” into the Sprint from the Release plan. After this first Sprint Planning meeting after second Release planning session, the team continued to plan in the following way throughout the Release period.

During a Sprint planning meeting, a Sprint story was created for each of the Release Stories the team intended to work on during an upcoming Sprint. The Sprint stories are represented as white cards in Figure 5.13. The Sprint story had the same header as the Release Story it correlated to and then for each Sprint story, tasks that the team intended to perform in the Sprint were written on separate yellow post-it notes. This was called “tasking” stories.

Figure 5.13 - Blue circles: Stories going into this Sprint. Green square: Finished Release stories.

After the Sprint stories had been “tasked”, they were prioritized and then given size estimates, using planning poker and story points, see Figure 5.14.

Figure 5.14 - Prioritization and Planning Poker

After all stories had been given estimates, the sum of points was compared to the average velocity or capacity of the team. If the sum exceeded the average, the lowest priority stories should have been put aside as Next-In stories. This was not always the case as the team was working towards deadlines at the end of the Sprints, where they had to hand over drawings to production, and therefore committed to higher amount of points than their average capacity.

After the team had committed to the Sprint Plan, they formulated a goal for the Sprint, a "Sprint Goal", which should communicate the main objectives of the Sprint. Before the meeting was over the team presented their plan to the software team, also planning in the same location, and vice versa. Changes were made if needed, but the teams also asked each other questions before this point, regarding if they are dependent on each other for work to be done. This was incorporated in the plans of the teams.

When the Sprint is finished and the stories completed, the size of each Release story, in points, was reduced by the completed points, see Figure 5.15.

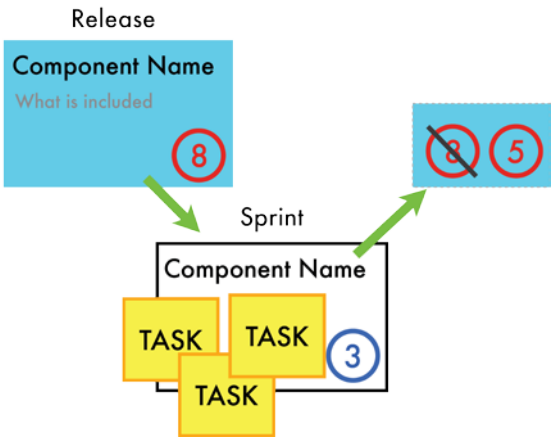


Figure 5.15 - How the point system works in the new way of planning.

Of course, this is not a flawless system, and one could not take for granted that the Release stories were correctly estimated to start with. This is why it is important that the team and Scrum Master ask, e.g.:

- Have we really finished this story, although we have finished all the points?
- Is there really 50% left of this story, or should it be re-estimated?

When a Release story had been completed, it was marked with a green post-it that said, "Done", which can be seen in Figure 5.13. What indicates that a Release Story is completed is that all the points have been finished and the team has reviewed that the work-item is actually completed.

Although this process has some flaws, what is important here is that the team was content with the way of working and that it solved some frustration that the team had. This way the team was at

least planning and defining what they were going to complete each Sprint. What was still lacking is more detailed acceptance criteria defining when stories, both Sprint and Release stories, are really completed. What was also missing was that the Release stories should be re-estimated, as it is likely that over time the team gets a better feeling, as well as concrete information, about the effort needed.

5.5.4 Summary of Team and Product Owners Final Results

The team was interviewed on the 3rd of September, having been working in the Scrum framework for 10 Sprints or almost six months. The team was asked to fill out a small survey before the interview. The Product Owner was given the same survey when he was interviewed on the 5th of September. The results are the following:

Advantages

The team was asked to name three or more advantages of the new work system. The answers were grouped into the following categories:

- **Daily Stand-up**: Three team members noted that the Daily meetings were an advantage, and noted that the meetings create discussions and the problems become clearer as they are discussed daily.
- **Functional breakdown**: One team member mentioned work breakdown as positive, and was most likely referring to the breakdown that was done at the Release Planning meeting, where the machines was broken down into sub-systems and components.
- **Overview and clarity**: Three team members mentioned that the work system gives a clearer overview of the project and its status.
- **Plan**: One team member mentioned that it was positive that there was some kind of a plan.
- **Communication and Co-operation**: Three team member noted positive elements in regards to communication and co-operation. It was mentioned that now the team is a whole, single entity and everyone has a chance to give feedback on tasks that others are working on. The work system supports knowledge transfer and creates a forum for discussions.
- **Focus**: One team member mentioned that the system has provided focus for the team, as other projects are dismissed to keep the team focused.

The Product Owner stated the main advantages to be:

- **Plan & Discussions:** The team is planning and discussing together.
- **Overview:** The work system provides overview for both the PO and management but also the team.
- **Co-operation:** The work system gets the mechanical designers “out of their corners”.

Disadvantages and Improvements Needed

The team noted the following disadvantages or improvements needed for the work system:

- **Sprint Reviews:** They noted that the Sprint Review is too extensive, and do not see the point in having them when there is little to review.
- **Time-consuming meetings**: Too much time goes into meetings, such as Retrospectives and planning.
- **Too much work breakdown**: All of them noted work breakdown to be a disadvantage. They found it to be unnecessary and that it possibly does not benefit their work. It is impossible to break some items down into different parts. One stated that because of the breakdown they know that the plans will not hold.

- **Scrum Masters too strict on the framework**: One team member noted that often "mountains are made out of molehills" and that the Scrum Master was acting as a Security guard of the framework.
- **Suppresses creativity:** One team member said the work system suppresses improvisation when designing, due to time pressure. It was not clear, if this was due to the time pressure from the Sprint duration or if it was due to management pressure regarding the launch of the product.
- **Changes due to new ideas:** One team member noted that scope of projects often changes due to new ideas. He could either have meant that therefore they cannot plan far ahead, as they get new ideas and the scope changes, or he was referring to changes made by the PO to the scope of the overall project.

The Product Owner noted that the work system had caused some turmoil within the team, that they were not fully convinced with it. He noted that this situation was preliminary, indicating that they might step out of their comfort zone and accept the work system later on.

One Change They Would Like to Make

When asked if they could choose one thing that they could change, their answer fell into the following categories:

- **Sprint Reviews:** One noted that he would like to change the Sprint Reviews.
- **Less work breakdown:** They want less detailed breakdown of stories, and have bigger chunks that they work on in parallel, i.e. take in stories from epics (Release stories).
- **Shorter meetings:** One team member noted that he wanted shorter planning meetings.

What the Product Owner wanted to change was to replace daily stand-up meetings on Wednesdays with Information or Synchronization meetings for the Product Owner and the two teams. He also noted he would like to have shorter Retrospective meetings.

Would They Like to Continue with the Work System

When asked if the team wanted to continue using the work system after the experiment was over, at the end of October, they all answered: **no**. When asked what needed to be changed in order to change their decision to **yes** their answers were:

- *"I think the system is too [strict] for my taste. A lot of time goes into meetings, and the Sprint Review would need to change. We don't need to invite half of the company every other week [to a meeting]"*
- *"Review meetings – they need to be reduced. Less demand for work breakdown, stories need to be worked on in parallel. Less red blinking lights [referring to strict Scrum Master]"*
- *"Less work breakdown"*
- *"Tailor the system to our needs, in order to help us, not to burden [us]. Because some things are very good but others [are not]"*

Despite their answers being negative, when regarding the results from the interview, presented in the following section, it seems they are willing to continue using the work method, or at least some aspects of it. They will continue if they can adapt it to themselves.

The Product Owner answered that he would like to continue using the work system. He noted some things that could not change about the work system, in order for him to want to continue, which were that the team should do planning and relative estimation (Planning Poker).

5.5.5 Team Interview Results

This section further describes the team's opinions and position towards the Scrum framework and the experiment so far. The interview was conducted late in Sprint no. 10, see Figure 5.11.

General opinions

The team was asked about what they felt were the advantages and disadvantages of the work system.

In regards to the advantages and the positive aspects, the team thought the system is good for keeping track of the project's progress. Synchronization of their work with the software team and product development workshop was seen as positive and stated to be good for the Product Owner in order to co-ordinate the teams plan to others. The system also helps the Product Owner to be more involved and know what is going on.

The team stated that the daily meetings make sure that everyone knows what everyone else is doing, and it is difficult to "get lost or hide in a corner" in this kind of system, everyone has to communicate. They all felt that the daily stand-up is very necessary. Everyone is able to give their opinion and give feedback or critique on things that they are not working on. This way all team members are involved in overall product.

They have not experienced it, but one team member noted that he could imagine that if a new person would start in the team the system is quite welcoming and that person would easily integrate. The new person would not be able to sit stuck somewhere alone for long, as he or she would need to attend the daily stand up and then the others could check how things are going.

The disadvantages that they saw were mainly concerned with the work breakdown, the demand to stick to the framework and the Review meeting. The first two issues will be presented in the following to sections and the teams opinions on the Review meeting is presented in the next sub-chapter, Scrum Events.

Demand of Too Detailed Work Breakdown

Work breakdown had also been a concern of the team at the start of the experiment. The team noted that the breakdown of work-packages and stories that had been demanded of them was too detailed, and if they were to do it that way it would be very time consuming, and then it might end up not to be used. One team member even suggested that it felt like the purpose of the breakdown was just to make more post-it notes and cards. These statements most likely relate to the fact that the there was a request from the previous Scrum Master for the team to break the Release stories, they had created by functional decomposition of the machine, further down and define them in more detail for each Sprint. This was not done, as the team was opposed to it.

It was also noted that some found it difficult to break the product and its components further down, when they did not have the final and full view of what the product will include and have not seen how it will work as a whole.

> *"I think the breakdown is way too detailed. And because it is so detailed, then of course it takes a lot of time to do. And then it's not certain […] it's very likely that we haven't decided on the breakdown, when we are breaking it down, so the odds that it will be like that [are not high]"*

Despite this, one of the team members noted that how it was done at the last Release planning session in August was good, because at that point they knew what should be included in the machine and what it should look like. However, he stated that the Release Plan should not be broken down any further and he noted that this level of detail might not always work, in other cases. Others agreed that they did not want the stories to be split up into more detail.

When asked, they agreed that the first Sprint Panning meeting after the Release Planning session in August had worked well and had worked out as planned for the on-going Sprint at the time. One noted that this way they could work in parallel on the different parts. One noted that everything beyond what we did at the Planning meeting would have been waste of time; it was just as it needed to be.

> *"Everything beyond what we have done, last time, would have been waste of time, I think. This was just like it needed to be"*

Further on one of the team members noted that he was not certain, but did not think other projects were any different when it comes to the work breakdown.

The Scrum Police

One of the team members noted that he gets very irritate when he feels that the team is being forced into a mould, and when something is done differently than what is prescribed "red blinking lights" arrive "like the police is after us". He also noted that if feels like they are being "told to do it this way, because this thing says you should do it this way". An example was given, where they were requested to break the Release plan further down:

> *"There is also one thing that I think is, has gotten on my nerves a bit about this system. It is being tried to force us into some mould. There are always some flashing red lights coming up, like the police is after us. That is one thing that really [gets on my nerves] …"*

When it was noted by the interviewer that the request for further breakdown might have been because there is a risk of not being able to estimate how much is left of the project if the plan is too rough, he said he suspected it rather to be just because the system says you should do it that way.

> *"Isn't it just because the system says so? […] That's what I suspect."*

When asked what would be one of the things they wanted to change, the same team member said "Cut the leash".

Scrum Events

Following section presents the teams thoughts and opinions of the Scrum events.

Sprint Review

There has been general frustration in regards to the Review meeting. This was not discussed in the first interview, as they had not held a Review by that time.

The first issue they had with the Sprint Review was that there were so many people invited, and also that very few showed up, at least few that had valuable feedback for the team. The team member with the most dominating opinion of the Review, stated the following:

> *"I just get "silly-chills" every time. […] Maybe I'm not quite correct, but … the coverage every two weeks, I think [too small] ... having to invite the whole organization to some Sprint Review every two weeks, I personally think is silly."*

Others agreed with this and one suggested that they would hold a purposeful presentation instead, that would be more interesting and everyone would benefit more from it. They also hinted that this might result in people actually showing up, which they do not feel people were currently doing.

One noted that he just found the Review to be a waste of time for everyone. Especially that people who have no input were showing up. He also questioned if this was not very costly for the company if all teams started to use Scrum, and if each team did not have enough to just be concerned with their own project, not having to follow other projects as well. Then the team noted that it would be interesting when the other projects and products are finished to attend a presentation about it, but not until then.

When asked how they would like to organize the Reviews two ideas came up. The first idea was to have a Review every other Sprint were they would have something more substantial to present about what they have done or learned. The second idea was to have a big Sprint Review when they think they have something interesting to show. In between these big Reviews, they would have smaller Reviews just with the Product Owner and the software team. This could be evaluated each Sprint, or they could try to set down milestones.

Example was taken of the upcoming Sprint Review later that day that they had nothing to show, and it was all very abstract or in a computer model. They had nothing to show anyone else than the Product Owner and the software team; however, they could be informed of the status.

They noted that at the next milestone, when they have finished the design of the machine, they could they could present the CAD model of the machine. One noted that he thought that the only Review meeting that he thought was justifiable, which was the one after the field test in June. The test had provided certain results, and they saw what worked and what did not, and could present those findings at the Review. But he did not see the point in having a Review were information on how a certain component should be, this way or that way, is presented.

It also became clear that they did not know who was invited to the Reviews, just that there were many people on the mailing list, and very few show up. The team member that was the most negative regarding the Reviews noted that he thought people did not take the Review invite seriously anymore, and thought they saw it as junk mail. Later, after the interview, the team sat down during a Retrospective meeting and reviewed the mailing list. It turned out to be mainly software teams, and few mechanical people. In the software community at the company, it is a general practice to send out invites to all embedded software teams, and then people decide if they will attend, depending on how interesting the content is for them. One embedded software engineer explained that she wanted to receive all invites, although most often she would not attend. She also noted that this was the general notion within the embedded software community at the company. The initial Scrum Master had created the mailing list, and later noted that she had asked the mechanical team whom to invite to the Reviews.

During that same Retrospective meeting, one of the team members noted that maybe they were not getting feedback because they were not asking for it. They might need to be more "aggressive" in showing what they have done and ask people what they think and what their opinion is about that. This Retrospective meeting was following a Review, where a CAD model had been shown of the whole machine structure.

Regarding the commitment to certain amount of work for a Sprint and presenting whether the team had finished or not finished what they committed to at the Review meeting, the team stated it did not put any pressure on them.

> *"If this is about putting pressure on if we finish or not, then this Sprint Review isn't working"*

They all agreed with this. They noted that many elements could affect them and unexpectedly come up which results in them not completing what they committed to.

In summary, the team thinks too many people are invited and too few show up that have valuable input. It seems they are uncomfortable with having little to show at the Review, if there are more people than the software team and PO invited. The frustration can be explained by that they do not feel they are benefitting from the Reviews and not getting valuable feedback. However, they might also be uncomfortable with showing unfinished models of the machine, or discuss the details of the technical problems they are dealing with at each time. They would much rather just show finished work, which is not the purpose of the Review meeting, as it should allow for usable feedback, inspection and adaptation.

Sprint Retrospective

It was noted by one team member that in the beginning, when they had started to use Scrum, the meetings had been useful, because they were used to introduce them to the Scrum framework and they were learning new things. However, lately they have lacked purpose, he stated.

When asked if the meetings would be more beneficial if the team would receive more education on the framework, he replied that would probably be the forum to discuss how they would use Scrum, but there was a question where they were headed. That is, if they will continue to follow the framework or not, by the end of October.

Others noted that they did not see the point in the meeting, with the purpose of improving their work, because the team communicates that much.

> *"I just think that the team is communicating that much [internally], that I think there is limited benefit from it, really."*

One noted that he felt that they were all keen on educating and improving themselves, and that they can always improve, so personally he did not need to sit down in a meeting to discuss it. He stated that he trusts his colleagues to inform him if there is something he could improve in his work:

> *"I have a feeling that we are all quite studious and always want to improve. I've always lived by that motto. One is never perfect, one can always do better. So, you know, I personally don't need to sit down, and discuss it, and I trust my work colleagues to point out to me if there is something in the way I work that could be improved."*

One noted that he has possibly misunderstood the Retrospective meetings, but another noted that the general notion is that there is very little that came out of this meeting, no results. During the Retrospective meetings, some but very few, improvement suggestions have come up. One team member noted that these things might have just come up during other meetings if they did not have Retrospective meetings.

In summary, the team does not see the purpose of the Retrospective and feel there is no outcome. This might mean that the Retrospective meetings might not be facilitated in the right way. This might also mean that they need to mature as a team to see how these meetings can benefit them and their work, and be open to come up with ideas to improve. The Retrospective might be one of the most challenging parts of the Scrum framework.

It should also be noted that a Scrum Master for an application software team within Marel GRB later pointed out that the team might not see the point in the meetings if there are no results or improvement projects that come out of the meetings. This is a valid point, and needs to be considered, both how to facilitate the meetings in order for them to give results, and how to encourage the team members to reflect on the way they work and come up with ideas how to improve in a systematic way. It was also noted that by the previous Scrum Master that the Retrospectives were not about personal improvements, which their comments point to; rather how the team wants to improve the way it works. She noted that it seems that further education is needed.

Sprint Planning

The team was very positive towards the planning meetings and all agreed that they were necessary, although it had been disputed what the results from the meeting should look like, in regards to work breakdown and task definition. They noted that they are not doing it by the book, but that they like the way it was being done at the time.

When asked if they saw benefit in playing Planning Poker and making relative estimates for the Sprint stories, two things stood out. Firstly, the estimates give an idea about the story size, and secondly, it becomes clear if there is a misunderstanding about what is included. This can be seen if people make very different estimates for a particular story, then these individuals can discuss the content and come to a shared conclusion, where all team members have the same understanding.

> *"Yes it gives … the outcome is giving us an idea about work size." – "It actually has, you cannot state otherwise."*

In summary, the team likes the Planning meetings, if they get to do them the way that they have been doing it since the last Release planning, in August. It is also beneficial to do the relative estimating as it reveals if there is not a common understanding of what is included in a particular work item.

Daily Stand-Up

They were very positive regarding the daily stand-up meeting, and noted that they are very beneficial and necessary.

Scrum Roles

This section includes the mechanical team's opinions and thoughts on the other two roles of the Scrum Framework: the Product Owner and the Scrum Master.

Product Owner

The main points made regarding the Product Owner (PO) were that his input and approval is necessary for the team, but noted that it had been lacking due to the PO's little attendance to daily meetings. The team also noted that the PO role was unclear; they did not fully understand its content and purpose. They also stated that due to the technical nature of their work items, stories, the team must create the backlog items themselves.

Attendance and Input

They noted that the Product Owner was more involved and more informed of what is going on, after they started to use Scrum. However, they also made a note of that he does not always attend, and that can be uncomfortable for them. They feel better if the Product Owner is aware of the status. They feel it is an indirect acceptance, if he attends the meeting and does not comment that he is satisfied. But if he is not there then they do not know if things are going or being done as he imagined. At one point, he almost never attended the daily meetings and that was very uncomfortable for them.

> *"[…] But one naturally feels better to know that he is aware of things. One feels that it is an indirect approval if he doesn't say any thing, that he is content. But if he doesn't show up then one doesn't know if it is going by his idea or [not]"*

> *"Well, he has started to attend better so, he is a bit more in the group but … For a time period, he nearly never attended the daily [stand-up] and that was very uncomfortable."*

The team noted that the PO does not have to attend every day, but at least when he is in the office or several times a week. It was noted that sometimes there are some discussions at the daily meeting that he should be involved in or would benefit from.

> *"Sometimes … some days, there are discussions there [at the meeting] that he would do well to hear."*

This had been observed earlier, when one team member noted that the PO had come in and started a discussion, most likely about technical issues, with the team. However, they had already had the same discussion and had come to a shared conclusion. The team member indicated that it would have been better if the PO had been there for the previous discussion.

Role Unclear

It was noted by a team member that he had not fully understood the Product Owner role, if he should be interfering or just monitor and observe. They indicated that they would rather that he would provide direction, market input and define the need, and then the team would take care of how to solve the problem or provide the technical solution.

But they agreed that it is probably very difficult to be the PO, that it is a tricky role, and his responsibilities are not clear. It was noted that they might not know how they wanted the PO role to be and one noted that he just sees it as a bit tricky role. Another one explained his understanding of the role in the following way:

"Like I've understood this, according to this Scrum stuff, the Product Owner owns the business … he is supposed to have the market, the business and what the business needs. And then we [the team] are supposed to figure out how we are going to fulfil this need."

Product Backlog and User Stories

When asked if the PO should work more on the backlog, create backlog items and prioritizing them, they answered that the POs work regarding the backlog should only be to make sure that they are working on the right things, but that he should not be solving them.

It was noted that it might not work in their case, for the PO to create the backlog items, and the reason for this is that the backlog is technical and not user oriented.

"Because we are in such technical work, and our backlog always becomes so technical, actually, that the Product Owner maybe hasn't understanding necessarily of these technical issues. […] He would need to be one of the specialists, you know, to be able to do it."

The same team member continued by comparing their work to how he imagined backlog work is done in software development, most likely referring to application software. He described how he could see a PO in that environment define features or stories around the customer. What he was actually describing were User Stories, and gave an example of an online bank user:

"I want a user of the online bank to be able to transfer money to [a family member] and be able to do that in a very simple and convenient way".

The user story example was compared to their case and their Product Owner. The result was that the PO can give direction, based on the market needs for the product, but the team should create the solutions, the PO should not interfere.

In regards to backlog work, the team concluded was that they should entirely do it; the team should create the backlog items. They also noted that in their work they are not able choose between features and have some of them in a first version of the product, and then add others later; in their case all features need to be included in the product from day one.

It should be noted, as mentioned previously, that User Stories are an Agile best practice, however they are not prescribed by the Scrum framework.

Unexpected Changes of Scope

There have been some unexpected changes to the product scope. These were often something that the Product Owner came up with and ended up changing the direction of the project. There was a recent example of this, which one of the team members mentioned:

*"When he comes with ideas about changing the machine, like when he added the [*new component* to the product]. […] When he comes with this, at what forum should he bring brutal changes like that? Is it … no I'm not sure."*

When asked if they found these "brutal changes" uncomfortable and if they thought the PO should have brought them up earlier, and maybe foresee them, they answered that it should have been brought up much sooner.

A: "Yes, much sooner."
B: "Yes, yes, not a question about it "
A: "Yes, needed to be brought up sooner at least, however it came up."

Scrum Master

In general, they were happy with the Scrum Master (SM) role, but there were also some concerns. They were not sure how they would succeed in keeping the wall up to date in the future, if they

were on their own, and if they would be able do all the administrative work such as writing all the notes and cards and booking meetings.

As mentioned earlier the negative point made about the Scrum Master was the "Scrum Police", when the framework rules were followed in too strict manner and was too much "by the book". They felt the Scrum Master needed to be more flexible.

One of the positive aspects of the Scrum Master role, they noted, was that he or she stands a certain guard, both of the team and the work of the team, by asking critical questions at the daily stand up. An example of this is to make sure that the team is working on the most important tasks, and that they are not working on other issues or projects, of less importance.

> *"But the benefit is that he has stood guard a bit, if there is something that is not acute or something like that, you know, or maybe dragged the Product Owner more in."*
>
> *"Yes and I also think, stood guard that [we] aren't spending time on [other projects/products], because that isn't acute. […] Yes, just out with it. So that, keeping focus ..."*
>
> *[Regarding monitoring the PO]"Yes. [That he isn't] putting us on something unnecessary. Keep us at it [The work they should be doing]."*

They also mentioned that the Scrum Master keeps them doing the Daily stand-ups, which they might skip otherwise. They do stand-up although the Scrum Master is not there, but if the Scrum Master would not be there, they said they would have skipped them more often. It was mentioned that it can be so easy to skip the stand-ups if they are very busy, but noted that it is still good to have them, despite being busy.

Further, it was noted that having a Scrum Master is important, and that the role is vital in facilitating the Planning meetings, controlling the meetings and keeping them on track.

> *A: "I think it is a must, just planning and all this, …"*
> *B: "There always needs to be someone that is maybe not technically involved in the project, that controls."*
> *C: "Yes, like planning, absolutely necessary. No question about it"*
> *B: "Yes that controls the meeting, actually. You know "Aren't we off-track now?", detects this, if we [going into] too much detail or …"*

Note: These results might be bias as the interviewer was the team's Scrum Master at the time, but the points that came up seemed to be honest. They might have left out some negative information.

Scrum Artefacts

Following sections present results in regards to the Backlog and Backlog Items, along with the team's visual management wall. This section also includes a discussion on commitment to Sprint plans.

Backlog and Backlog Items

Like mentioned previously, the stories or work items are made at the beginning of each Sprint and are technical, not user oriented. There is not a Product Backlog of features in that sense, but the Release Plan and the Release Stories have served as backlog. The team explained how they did not see how the Product Owner or anyone else could create a backlog of tasks for the team, which they would be able to use. One member also noted that if a detailed backlog would be created in advance for the team it would become obsolete.

> *"I think it's very peculiar if he [PO] can create it, when we that are working on this can't, really."*
>
> *"Because today the backlog, revolves around how we solve the issue in question. And actually we are planning how we are going to solve the issue in question, which is naturally a bit difficult*

to do in such product development work. To plan how we are going to solve something unknown in advance, that's a bit of a challenge, …"

The Wall

In general they are happy with the wall, and that they now have figured out how they are going to use it. Their view of the wall might also be influenced by the amount of detail the work is broken into and how the stories, which go onto the wall, are defined. When asked if they are satisfied with the wall and the detail as it was at that moment, they all agreed.

One benefit they noted was that when they have been away, for a few days, they could be updated on the status by looking at the wall. Two team members that had been at home ill the day before noted:

A: "Me, [when I arrived] this morning, I just walked in front of it and just, […] and just looked over it a bit, and noticed that we were going to plan the test for next Sprint and … […] I could envision a bit what happened yesterday." [A memo post-it note had been put on the board the day before.]
B: "Yes I also did that this morning; I started by checking the wall."

It was also noted that they could possibly develop it further and add new features to it, for example to inform others when people are out of the office. This way it would become some kind of an information board.

One concern regarded how they would maintain the wall, and the notes and cards that come with it, without a Scrum Master.

Burn-down Charts

They also discussed burn-down charts, and noted they did not feel they needed burn-down for each Sprint, but that it could be useful for the Release as a whole. This means the burn-down chart would be updated after each Sprint.

Commitment to Sprint plans

It was repeated that the pressure to finish what they did commit to during planning, does not affect them. When asked if they did not get a bit disappointed when they did not finish all the points they committed to, they all said no.

One elaborated that their goal was to finish the whole product, not just the Sprints, and that is the main goal. When asked if not finishing points for each Sprint did not affect finishing the whole product or the points in the Release plan they answered that it can vary a lot how much they finish each Sprint:

A: "Our goal is the 80 points [the sum of points in the Release plan]."
Interviewer: "But doesn't it affect, if you are going to finish 20 [points] but just finish 15, which must be 15 out of those 80?"
B: "[It can be] 30 next time."

When asked if there had not been a trend in how many points they finished each Sprint, then they answered that these points are very abstract and that product development revolves around getting good ideas. Sometimes they are just stuck and do not get any work done, but when they get a good idea, they get a lot more done.

"Yes, yes, but this is product development and this revolves around getting ideas and if you don't get ideas now, then you'll just have to hope you get good ideas tomorrow"

"Yes just, you can sit at work and you are just … there is nothing happening. There is no progress. You can just be thinking … can be talking to your colleagues and try to … and nothing

happens. But all of a sudden something happens and then just [indicating something going fast]"

They do not see the Sprint stories committed to, as very sacred. But they want to finish the whole, and argue that sometimes they are just blocked, do not get good ideas on how to solve issues, and then another day they get a good idea and work much faster.

Note: It should be noted that there has been a trend in how many points the team has completed each Sprint. The amount increased after they started detailed design, but there was not much difference in completed points in-between those Sprints.

It was later observed that one team member noted during a daily stand-up that he thought he was maybe working in too much detail on one work item, and was thinking about starting to work on something else, in order to finish the points for the current Sprint. The team then spoke about it and said that he might be right and that he had worked enough on the current part. This showed that there was some will to finish what they had committed to, although this way of thinking was not shared by all team members.

Reflection on Issues Foreseen in April 2013

In this section, the issues the teams brought up in their first interview are revisited.

Framework Tailored to Software Development

The team still thought Scrum is tailored to software development. The reasons they mention, is the user experience orientation that they said characterizes software development, and does not apply to them. It was also noted that they could write their stories that way, from the service personnel's perspective, but it would not make any difference for them; it would not help.

Another reason they stated was that in software development, one can start with few features and a simple product and then one can later Release new features and update the product. This is not possible for mechanic development, because each additional feature affects all the other features. The interfaces between them are not clean-cut.

As mentioned in the previous interview, another difference between software development and mechanical development is that there are waiting times for physical objects, in the latter case. It was also noted by the team that it is not possible for them to finish designing each component separately and sequentially.

Further, in a discussion on whether it would be possible to define clear interfaces between the different components, the team concluded that it might be more difficult for them than others, and further noted that might be possible in order to optimize the product, but that was not their objective as this is not a mass market product.

Note: It seems the team thinks user stories were a part of the Scrum framework, and therefore thought it was tailored to software development. This is not the case, as has been previously mentioned, and should be clarified.

Flexibility and Next-In Stories

In April, the team was concerned with flexibility of work items or stories for each Sprint, if something unexpected would occur. Now they noted that this was not a concern anymore as they are not defining and breaking the work down into as much detail as in the beginning. However, one of them mentioned that they were working on several projects during that time, and he thought they were not really working as a group at that time. They were still working some "old ghosts" of previous development projects.

One team member noted that they had been breaking the work down into so many cards or tickets, and sometimes one test solved many of these. When it was broken down into that detail, they needed Next-In stories. Now they could just continue working on more points of the corresponding Release story.

They were asked if this was easier now because they are in more detailed design, and there might be less uncertainty than before. One of them denied that there was less uncertainty now. Another noted the following regarding taking in extra points into a Sprint:

> *"But maybe because it was just in … it stretched over more Sprints, the item one was [working on], and instead of being able to continue, then it was so obvious if one was going over into another Sprint and working on it, when there was so much breakdown."*

Waiting times

In April there had been some discussion regarding waiting times. The Scrum Master at the time had challenged the team on why the waiting times were so long for purchased components and constructed items from the workshop. They agreed that there had been a long waiting time for components, but noted that these things just take time as they are dealing with physical components and production. They felt that there might not have been an understanding of this on the Scrum Master's part, and also noted that they felt the Scrum Master had pressured them to get express treatment but noted that if everyone would do that, things would take just as long.

Note: The first Scrum Master did not remember asking them for express treatment, and it is possible that they misunderstood what she meant. The intentions were to get the team to be critical, and not just accept waiting times without a justification. The issue regarding waiting times seems to be an organizational issue, as there is some extra waiting time added in the production management system that affects the team. Therefore, it might be out of the teams reach to change.

Product Development Workshop

In April, when the waiting time issues were being discussed, there was high load on the Product Development Workshop, and everyone was asking for express service. This resulted in the team's previous prototype to be delayed by a month. However, in beginning of September the same year, when the interview was conducted, there were no problems with the workshop; the situation was good. They agreed that it depends on how much pressure or load is on the workshop at a given time.

When they were told the PDW was starting to use Kanban, or visual management, they expressed little hope in that would solve the workshop issues, but when it was noted that this could make the issues more visible they agreed and one added:

> *"Yes possibly, could make it more visible. Then, possibly, our Product Owner could organize, his expectations, or put is expectations into that form.[…] Like that time [referring to the last prototype], there was [Product Center], they were [building] a system at the same time. And they just owned the product development workshop."*

Varied load on the PDW and one development team possibly occupying all it's resources, is another organizational issue that affects the team and their productivity and the team was not sure a Kanban wall would solve. One of the reasons for starting a visual management system at the workshop, was to transfer the responsibility of prioritization from the workshop foreman, that might just focus on the project that he gets the most "shouting" about, to the Industry and Product Center Development Managers. This way they can discuss which projects are the most pressing and negotiate the prioritization based on data, such as showcase deadlines, estimated margins, delay costs etc.

Work Distribution

There were some worries in the beginning regarding how they would divide the work between them selves. The first Scrum Master had also challenged the team at the time, saying that in Scrum teams, all team members should be able to do all tasks, there should not be specialists, and if some team members did not know how to do a certain task, they should pair with someone who did. They said that now when they did not have as detailed breakdown of stories, therefore it might not be as much of a problem now.

Other Issues

During the interview, two other subjects were discussed. Firstly, a discussion regarding the decision to start using Scrum, which was a top-down decision; secondly, acceptance criteria for stories; and thirdly, iterative work was shortly discussed.

Top-down decision

The decision to start using Scrum was a top-down decision made by the Industry Center's Product Development Manager, later the team's Product Owner. It was one of the concerns of the first Scrum Master that this might be a problem, as it is generally not recommended to force Scrum on teams. When asked if this had made a difference or had negative effect on them, it seemed that the team was not sure how to answer this at first, but they all agreed that they would never have initiated the use of Scrum themselves.

One noted that maybe one positive thing about the decision being top-down was that they tried, and although there were faults to the system, they could now envision how it could work for them. Another added that by seeing how it does not work for them, they have seen how it could work for them and that is positive. Third team member added that although they would take the decision to stop using Scrum, they would continue using some of it. Another agreed with this and said that there were some things that he felt did not fit them, but they had also realized that some things were worth keeping.

When further asked if it got on their nerves that they had been "made" to start using Scrum, a team member said that he had been positive in the start to try it out, and it had not irritated him. One member further added that the purpose is understandable and there is nothing abnormal about something like this being tried out.

Acceptance Criteria for Stories

Acceptance criteria was not discussed in the previous interview, but was an issue that came up throughout the experiment, in relation to definition of Release and Sprint stories.

When asked if it would help to have acceptance criteria for stories in order to know if they have finished something, which they do not do for the Sprint stories, they noted that they had done that for the Release Stories or at least have noted down what was included.

In regards to the size estimation of the Release stories, which were used as a basis for the Sprint stories, they noted that the point sizes were quite abstract, and it was difficult to say if they have finished 5 or 10 points out of 20 points, for a particular Release story.

Regarding acceptance criteria for the Sprint stories and knowing if what they had planned to do was finished, one team member noted that they might think they are finished with half of the work for a particular Release story, but later something comes up and changes need to be made. This results in work needing to be re-done and then they are maybe just finished with a quarter of the story.

When asked if they could then have acceptance criteria for the Release stories instead of Sprint stories, the team member answered no, and gave the same explanation regarding re-work needed when something unexpected happens. When he was asked if the final result would not be the same, for the Release story, whether or not something unexpected would, he agreed but repeated that they do not know when uncertainties happen and the previous work becomes obsolete.

Note: One assumes that it should be possible to define final acceptance criteria for a Release story. If something unexpected happens, and more work is needed than originally was planned for, the size estimation should be re-evaluated and increased in points, rather than lowering the percentage of the originally sized story.

Iterations

First, they did not see their work as iterations when asked about it. But when iteration was defined as something that is done roughly at first, and then finalized or designed in more and more detailed, they agreed that this was the way they work, and how mechanical design and development is.

> *"Yes, we are working a bit roughly here, "now this is good enough for the [*component*], ok then we just say that it is done for now", and then we come back to it and continue."*
>
> *A:"I think I iterate every day."*
> *B: "We are doing it, actually."*
> *C: "I work like that, I draw like that"*
> *D: "Yes"*
> *Interviewer: "First rough and then more detail over the whole?"*
> *D: "Yes"*
> *C: "Yes, actually."*

Note: This realization might be important in the future in order to help them see their way of working as iterative and that the work can be broken up into iterations. This could help with defining acceptance criteria for Sprint stories, as this way they could define what they want to have done by the end of the iteration, narrow the focus from the final project result to the result of the iteration.

Can Scrum be Used in Mechanical Product Development?

The team was asked if the Scrum framework works for Mechanical development, and if it needs any adaptations. The all agreed that adaptations are needed to the framework, but also noted that the adaptations they would need to do might not work for everyone else.

When asked if Scrum works for their particular case they replied that they think it can work, and that the way it was done during the previous planning meeting could work for them.

> *A: "I think that it could […]"*
> *B: "More than could, I'm sure that it would work, so …"*
> *C: "Somehow like we did it last planning, despite it [having been a heated meeting], then I think it makes sense a bit. When we know approximately what we are aiming for, to create, in these machines, know the components, so we can break the project down by components. But then there needs to be an understanding that these components need to be worked on in parallel, parallel to each other,[…]. And I think it's an alright system, that we can just drag into planning, you know, drag those points that we think can [be] […] worked on this [Sprint] from the big blue card [Release story]. So, "we are going to do this from this [Release story], this from that, this from that and this from that"."*

They also noted that it can be difficult to know exactly what the order of work will be, because it can quickly change, from one week to another or even just in one day.

Would They Stop Using Scrum Completely?

When asked if they wanted to turn back to not using the work system at all, skip everything, they answered that they would not. They would keep some aspects of it such as, the daily stand up, planning and the rough work breakdown for the Release plan.

Will They Continue When the Experiment is Over?

The team was asked if they want to continue using Scrum after the experiment ends in end of October. First answer was that they would not continue with the framework exactly as it was set up. However, when asked if they would continue with how the wall is today, with the new way of planning, the replied that they probably would.

> *A: "Yes"*

B: "Yes possibly, if there are is no red blinking security guard."
C: "I think we will most likely use part of it."
B " ... a lot of it."

They were further asked if they would like to keep the roles, the Scrum Master and Product Owner, they replied positively:

A: "Yes"
B: "Yes, I would like to keep that. But the Scrum Master just needs to be ready to … work with us to make the changes to the system, so that it will be useful for us."

In summary, they wanted to keep using the framework if they could make some changes. They also wanted to keep the roles of a Scrum Master and a Product Owner, on the condition that there would be understanding and cooperation in adapting the framework to their needs.

5.5.6 Product Owner Interview Results

This section describes the Product Owner's (PO) opinions and position towards the work system. It should be noted that the PO is also the Industry Center's (IC) Product Development Manager, and the Industry Center has three other locations globally where product development is done.

The Team

The PO said that the team was quite typical for mechanical design teams. The team is a good "cross-section", as they all different character types and have different backgrounds. They also have different experience levels.

Current Problems

The PO mentioned some problems regarding the team. Firstly, he did not think they were "on their toes" enough regarding risk factors. An example of this was components from suppliers that were critical to the success of the project and product. He felt the team just trusted that these products were being delivered with no problems, but they were not sure.

The PO also noted that the team did not show initiative in keeping a dialog with dependent parties, such as purchasing department or suppliers, to check if all was well or actions needed to be taken. The PO stated that he felt it was not the PO's job to step in and take care of these things, but the teams. Despite that, he had been taking care of some issues. He noted that there would need to be someone to question the team about these matters, and make sure risks are addressed and not forgotten about. But he also noted that it would be better that team as specialists, focus on what they are good at and there would be someone else to take care of the dependencies, but it was unclear who should do this, the PO or the SM or another individual.

Culture in Mechanical Engineering at Marel GRB

The Product Owner stated that there is a certain culture within mechanical design at Marel GRB. People have "owned" certain equipment or products, been specialist of a certain area and carried that with them through their carrier. That is where the team was coming from.

Planning was new to the team, before there were plans, but the milestones were very far apart, and they have just planned their own time by themselves. There were plans, but there was an unwritten rule, that delays were accepted. The PO stated that delays in product development projects are mainly due to the mechanical part of the project.

What also has characterised mechanical engineers at the company is that they generally work alone. There were discussions, between those that work together and those that work on the same projects, but there is no planning involved, the PO explained. He also noted that it happens way too often that engineers work individually on projects. That way there is no review and dialog. This

results in the product becoming homogenous from one person's point of view. How the problems are approach is not distributed in that way of working.

The old days

More than a decade ago, there was a sort of a start-up culture at the company. Every Friday the engineers would sit down together, and have a couple of beers together and relax. The PO noted that this created unity and a strong group. This has now changed, after the company moved to its current Iceland headquarters in 2002. The environment is different, open office spaces and many different departments, where the teams do not feel comfortable doing the same as before. The old culture is something the older employees miss, the PO stated.

General Opinions on the Framework

Following are the Product Owner's general opinions on the framework, its advantages and disadvantages.

Advantages

The PO stated that what he thought was the most valuable was that the team was planning and communicating internally. Secondly, he found the system provide overview for himself and the management, as well for the team itself. This overview was also a result of the daily stand-up meetings. During those meetings things, such are risk factors were being addressed, which he notes is a benefit; try to see where things could go wrong. He noted that this did not always happen unless there was someone asking the team about the risks, as mentioned earlier. He noted that a facilitator with a technical background would be needed to keep the team on their toes and ask questions.

Disadvantages

The PO stated that the new work system had caused turmoil and negativity within the team. The PO stated that all the disadvantages, were a result of the team being taken out of their comfort zone, and it would take some time to get them out of there. It would take couple of years and couple of projects, but the PO said they were on the right track.

He noted that the team felt there was more overhead, more administration and management, but this was not fully reasoned for. The PO did not agree and did not think that there would be any less overhead, calculated in man-hours, if traditional project management would be used.

One of the things that the team had been most frustrated about was the Sprint Review and he noted that this was due to it being out of their comfort zone. He personally thought the Reviews were very good.

Scrum Events

Following sections present the PO's thoughts and opinions of the Scrum events.

Review

The PO stated that he thought the Sprint Reviews were good, and there questions were asked, that otherwise would not be asked, at least not until too late.

> *"These Reviews are good. […] Questions are asked, that are completely valid, very good. They wouldn't come up if we didn't have this Review, except maybe [when it is] way too late."*

It was mentioned that the team felt uncomfortable with inviting so many people to the Reviews and sometimes felt it works as they were crying wolf. The PO replied that the team had decided in advance that the people that show up to the meeting have nothing valuable to contribute and the team would not benefit from it. He stated that when people do not see the value, they will work against it.

He also noted that this might be related to the mechanical designer culture, as they were not used to be asking each other questions and challenging each other, like this. He also noted that there is

difference between mechanical and software developers, where in software people might be more open to discuss different methods. He added regarding mechanical design being more personal, or personalized.

The PO also stated that the team needed to be met half way, but they would have to understand that it is done in order to change and improve the work, not to back down. The ideas he suggested regarded that the team could choose whom to invite to the Reviews, and they could maybe choose few people for each Review. He also agreed with the team that there were people attending the Reviews that did not contribute.

> *"But there are people […], I could point those out [who show up to the Review meetings] that are not contributing anything. And maybe, they just feel uncomfortable having so many people. Maybe we should just give them a chance to just invite some group, which they trust and they see could create some value. […] Then it's possible to, the mechanical team always needs to invite some, let's say four or five [people]. And then the [same] way around for the software team.*

The PO agreed that one of the benefits of the Review meeting is that the two teams are synchronized and updated on each other's status.

It should be noted that later on the team seemed ok with the Reviews, after looking through the invite list and some discussions. The Scrum Master was even allowed to add manufacturing and purchasing representatives to the list.

Retrospective

The PO noted that he could see how Retrospective meetings could be a challenge to the team, as it was outside their comfort zone. He added that first they might need to see value in the work system, start small and then later put the focus more on the Retrospectives. When asked he said he did not want to skip them. He also suggested that these Retrospectives might just need to take 10 minutes, by the coffee machine; booking a room for a meeting was unnecessary.

The PO noted that they had not analysed why some things that had gone a miss in the project, for example a critical component was not ordered by the purchasing department, due to some misunderstanding, and therefore arrived some weeks later than planned. He wondered what could be done to be proactive in situations like that, and noted that what went wrong in that particular case had not been analysed.

Planning

The PO stated that the team was planning and seeing the big picture. He stated that he thought the Release planning is very good to put down milestones, and without it they would just have been working like before, with an attitude of *"it will just finish when it is finished"*. Then there were maybe milestones every 6-12 months and then people would just start, and see how it would go. Regarding if re-engineering loops had been needed, he noted:

> *"Yes, but no one realized it until it was way too late to save anything, in order to pass the milestone."*

Due to this the way of planning in the new framework is definitely an improvement, also because everyone is involved, not just one person. He also noted that the discussions that arise during the meetings were very valuable. These often regarded technical concepts and how to solve problems. However, he questioned if these were too homogenous and if the team should invite external designers to come take part in them during the planning meeting.

New Way of Panning

The PO was asked what he thought of the new way of planning, which had started after the latest Release planning session; where parts of Release Stories were dragged into the Sprints. He answered

that if the team sees value in that method and they were getting points finished then it is fine. Although he noted that in the previous meeting, they had maybe not looked at where the highest risk lay. Nevertheless, he further noted that addressing the risk was a benefit of the method, but the team is not alert regarding risks. He said they think its ok to sit and wait to see if it would work out next week, or the week after that.

Daily Stand-Up

The PO said that the Daily Stand-up was almost the only forum where risk was addressed, if it did not happen when the team was working. He stated that they needed another short meeting where issues where addressed. He also suggested having optional invites for stakeholders, such as the workshop, purchasing and others. The purpose of these meetings would be to synchronize, inform and address critical issues. But he noted that something needed to be scrapped instead, adding more meetings would not be good for the moral and he suggested to skip daily stand-up on Wednesdays. He said they could just move the tickets the day after. He also noted that this was just an idea. This idea had not been presented to the team some weeks after the interview.

Scrum Roles

Following are the PO's thoughts and opinions on the roles of the Scrum Framework.

Scrum Master

The PO said that the Scrum Master (SM) had a big role during planning meetings, but when thinking about it he felt the role was backstage at the daily and Review meetings. When it was noted that the SM should be there to support the team, but not be in the foreground of the Daily and Review meetings, where the team is presenting to each other or to others. The SM should just observe, and if the team goes off track, the SM can step in. The PO said that this might then mean that the SM should be almost unnecessary at these meetings, and his understanding had been different. He said that his understanding was that the SM should jump in and tackle hindrances that come up. However, he also noted that sometimes the Scrum Master does not have the technical knowledge and then it is impossible for that person to tackle some problems, and that was understandable. He then posed the question of who should deal with these, and said that as the system as the current today the team should have solved those issues or hindrances, but that they had not been doing it. He noted that he had been stepping in and dealing with small issues. When it was noted that maybe the roles should be cleared for the team, he noted that maybe the PO is supposed to do this, but said that the problem is that he is very busy.

Product Owner

The PO noted that the role is very difficult. He was not sure if it was more difficult in this project or others, but thought that it might be, as there were so many different and diverse opinions on how to approach the product under development. Depending on who you talked to, customers or in-house people, the opinions were very different and often opposite, and it was not obvious which opinion to follow. The PO said that this was very bad for the project. This had resulted in the product scope changing and the direction taking some 90° turns, and this had made the team very uncomfortable. When asked if he thought it would have been possible to do things differently in the beginning, and avoid these changes in direction he answered that he did not think so. They needed to begin somewhere and because these opinions were so different, they would not get everyone to agree and get a clear full picture. He noted that he thought they were approaching this in the right way and what was being dealt with was the core of the higher-level system, which the product under development was a part of, independent of what was happening around it. Then they would see how they would grow around that core.

The PO noted that he the Product Owner role should ensure the business aspects of the product, but at the same time be aware of what is technically possible and not possible. He noted that if the PO would only focus on the business part, and would not understand the current days technical

hindrances, like things that would need great effort to develop, then the team could turn against the PO. He noted that it is a question of balance between business and technology, but they also need to challenge themselves technically.

Other issues

In this section, other issues that were discussed in regards to the framework are presented.

Project Owner's Presence and Communication

The PO stated that intended to be more available to the team during the autumn. When asked how this would affect the team, he said that it would mainly be for the morale, not that there was a bad morale within the team and teams, but there was an underlying unrest. He said this was due to that people felt that the project had been changing direction, there was time pressure and people were going outside their comfort zone. Therefore, he needed to be there for encouragement and support. He added that he would also help by taking the issues that no one else could answer, step in and help solve these. He noted that this might be something the PO is not supposed to do.

It was noted that the communication between the team and the PO had not always been the best. He hoped his increased presence in the coming months would mainly improve the communication, take care of the moral, and take care of the small issues, described earlier. He wanted to build good culture, by doing little things such as small celebrations after successful Sprints. He said the older employees miss the old days, where sitting down together and having a beer before going home on Fridays was a routine. That culture increased unity and solidarity within teams.

Influence on the Project Progress

When asked if the PO thought the framework had influenced the project's progress, he answered that it had had a positive effect. He noted that they were still learning and one of the things the team and he have struggled with was to set themselves goals e.g. Sprint goals. He said that it had been easier to set goals for the Release. He noted that the Sprint goals had not always been at the heart of the work during the Sprint, it was just a piece of paper, which people forgot, and the emphasis of work during the Sprint might not have reflected the goal.

He further noted that he would have approached the project and it is subject differently if he would do it over again. He said he would have looked at the product in a more functional way, and then would have tested each function independent of the others.

> *"I would look at it all much more functionally. That is, I think that that is the problem that they are dealing with. They just look at it as one machine, one chunk, and we should look at these functions. Then test each element individually, each function individually, independent of others, some things are dependent but … This is what we should have been using the time for the past […]."*

He noted that they had done this in some cases, but have not done it enough. He further noted that some things, risk factors, they had postponed too long to do anything about. This is why questioning and challenging the team during planning and daily meetings is important, he noted.

Top-Down Decision

The PO was asked if he thought it mattered that the decision to start using Scrum had been top-down. He answered that it did, but that it had been done with the purpose of being an experiment. He further added:

> *"It was top-down, but I think we have done it quite right. When we started with Kanban, which was not working for them at all, which was good, then they had something that was worse, that didn't work. Maybe just luck. They were motivated and positive at the start […] they gave it [Scrum] a chance. But then when they needed to step out of their comfort zone, then [the negativity started]."*

He further noted that they would never have started to use Scrum if it had not been a top down decision. They needed to hit some walls in order to get where they are today. He did not think the implementation could have been done in any other, better way.

Scrum in Mechanical Development – Does It Work?

When asked if he thought Scrum works for Mechanical development teams he replied:

> *"Yes it works, but it needs to be adapted."*

Adaptations of the framework

The PO stated that he thought they might need to have some kind of mix of methods, 80% Scrum and 20% project management. He further explained this that it is important to have someone asking questions and stepping in taking action if needed. This person should be able to summon stakeholders and the team and in order to deal with certain issues. An example of this would be to gather the team and purchasing in order to deal with issues or possible risk regarding critical components. He further said:

> *"There are many contacts, this is complex technology, this is new technology. The knowledge needs to be built. This is a great challenge, and it can't be expected that this [group] […] that is also supposed to design the machine, and are specialist in a certain field, that they are doing this. That just isn't fair. That's why I think … we can call it something different than project management …"*

He further noted that they could include this in the role of the PO, define the role wider than it is today. That is, he would take the technical issues, which he said is maybe something that is intended to be the SM's responsibility.

The PO stated that other adaptations would be to reduce the time for meetings and their weight, and then try to end up close to what the theory prescribes. He was most likely referring to a discussion regarding the Retrospective meetings, and that they will be kept on, but will be time-boxed and kept short. That way they would be there but still would not weigh very heavy.

Acceptance Criteria

Previously there had been some discussion on acceptance criteria for stories, and lack of a "Definition of Done" for the team. It should be noted that this is not a deviation from the framework, as it is not prescribed. Rather it is a best practice, like the User Stories.

The PO was asked if one of the adaptations would be not to have acceptance criteria. The PO said he still thought that at some point, the engineers do finish and put things away, and when they return to the task, it becomes a new story. Therefore, it might be a good for the team to think about when they are finished with a certain work item. He thought this was one of the consequences from the team designing the machine as one chunk, where they go from one thing to another when designing. He noted that it was not necessarily efficient to be working on everything within the machine at the same time.

Quit or Continue Using the Framework?

The PO stated that he would like to continue, in the way that had been discussed, including some kind of project manager that would be on top of everything, regarding risks, stakeholders and dependencies.

He also noted that this had to be looked at from the team's perspective. What the team was as faults, the PO said, was that Scrum takes up a lot of time. Therefor he suggested time-boxing the time that would be spent on the framework. In that context, he mentioned again to skip Wednesday daily stand-up for a co-ordination meeting. He agreed that high-level research project meetings could also be skipped. He suggested the Retrospective meetings to be shorter, and just held by the coffee machine. The Reviews could be a narrower group and let the team invite people they trust.

Note: Skipping Wednesday daily meetings and adding an extra co-ordination meeting had not happened, at the end of the experiment. The first Scrum Master noted that it might be better to investigate what and where the team sees waste in meetings.

5.5.7 Status at the end of the Study

In this section, the status of the team by the end of the experiment is presented, as observed by the the teams Scrum Master, along with some general observations.

They are a Team

From observations and comparison to how the group was before Scrum, it can be stated they moved from being a group of specialists working individually together, to a team that works together as whole. They showed a general resistance to change throughout the implementation, which can be assumed normal. But at the end of the experiment it seemed as they had reach a certain point where they saw the value in the work system, and did not want to go back to the way it was before, although they still had some opinions and were sceptic of some aspects of it.

There was an increase of the average amount of points the team was able to complete per Sprint. It was 16 but moved to 21 in the Sprints after the Release Plan. This can possibly be explained by them being interrupted less in regards to other projects, and those that were asked the most for assistance, consciously turned these away or put as little time in as the importance had been put on this particular project by the management. This change might also be explained by that after the second Release plan was made in August, the scope was clearer, there was less uncertainty, and the project moved from concept design to detailed design. There might be correlation between uncertainty and team velocity.

Lengthy Meetings

The team had expressed that they thought too much time was wasted on meetings, and therefore to their duration was keep to the minimum. By the end of the experiment the Reviews were time-boxed to 30 minutes, where each team was given 15 minutes, to present their work and receive feedback. The Retrospective meetings were also time-boxed to 30 minutes, and were sometimes finished in shorter amount of time.

Sprint Reviews

It felt the team had started to accept the Reviews, and stopped feeling as awkward about them. They started to show CAD models at the Reviews, and team members showed initiative in suggesting that they show a certain CAD model at the Review. This might be explained by where they were in the process, in the final stages of detailed design and had CAD models to present at the Review.

Earlier, they had been unhappy with the amount of people that had been invited to the Reviews, and it had been suggested that the list would be reduced to the bare minimum and people would be specially invited for each Review. The invitation mailing list was reviewed, and revealed that the main part of the long list were other embedded software teams. It is a common practice with in the embedded software community to send optional invites to all other teams. Therefore, it was decided not to change the list and even representatives from manufacturing and purchasing were added to the list, as stated earlier.

Sprint Retrospectives

Further improvements might be needed to encourage the team to reflect and come up with suggestions to improve the way they work. The Retrospective meetings for the mechanical team have been held separately, from the software team, but it is possible that there would be value in having a joint Retrospective now and again, maybe at the end of a Release or after larger milestones have been reached.

The Teams Final Decision

On the last day of Sprint no. 13, the team was asked if they would like to continue using the framework. They all agreed to continue.

This decision was taken on a meeting during before lunch and later that same day the team accomplished a big goal that they had set during the Release plan, which at the time seemed over realistic. The goal was to send all drawings of the machine to the product development workshop by the end of Sprint no. 13, except for some very few non-critical parts. Some days before the end of the Sprint, they were not sure they would accomplish the goal, and not even during that morning meeting, but later that day they did. The Release plan, with green post-its to indicate completed release stories, and burn down graph can be seen in figure 5.16

Figure 5.16 - Release Plan after sprint no. 13, when the goals the team set had been met

It should also be added that at a presentation of this thesis report, for the Marel teams management and team members, the Product Owner noted that he was certain that they would not have come this far without Scrum. That is, the team would not have accomplished the goal of sending all parts of the prototype to production by the set deadline if they had been working the old way.

One of the team members also noted that before, they had not put down as frequent milestones, as they had done when using Scrum. He noted that before they had not really got a feeling of how much work was left undone, until maybe shortly before a deadline. Scrum provided focus in this sense.

6 Supplementary Case Studies - Use of Agile Methods in Hardware Development

The second part of the empirical study contains experiences of using Agile, Scrum or Scrum-like methods in hardware product development from other companies than Marel GRB. The two cases presented in this chapter are from two Swedish companies: SAAB EDS in Gothenburg and Andritz Hydro AB.

6.1 SAAB EDS – Scrum-like Method in a Hardware Design Project

This case study presents how a multidisciplinary hardware team at SAAB EDS in Gothenburg, Sweden, used a Scum-like method in one of their design projects.

The people interviewed where two employees of SAAB EDS. The first one is one of the Line Managers within the mechanical line and a Project Manager for the particular hardware project. These are referred to here after as the Line Manager (LM) and the Project Manager (PM).

6.1.1 Organization and Project Introduction

SAAB EDS is a technical company within the defence industry. It is a matrix organization with several functional lines. The projects are allocated resources from the lines. The project in question was carried out by a cross-functional, with competences from the three different functional lines. These are power and cabling, mechanical and microwaves. There were no software competences involved in this project and the team did not co-operate with another team. There were only three people fully dedicated to the project, but around 12 people were involved in the project. Commonly, people work on more than one project at ones. The number of fully dedicated people changes throughout a project, and depends on what phase a project is in. Often there are more fully dedicated people in the beginning of a project and then the number decreases.

There were three functional Product Owners in the team, one of which also served as the Scrum Master. In addition to this, there was a Project Manager (PM), which was interviewed. The Line Manager (LM) noted that there would possibly be changes in organizing of teams like this one, in the next 6 months or year. The duration of this project was planned to be 22 months and at the time of the interview, May 2013, the project was 9 months in.

The development team worked on a physical prototype and some of the team members also verified the prototype. The product was built up of about 8 parts, and each part went through the phases of design, build of prototype and verification separately, they might not all have been in the same phase at the same time.

SAAB EDS Lean Transformation & Visual Management

SAAB EDS makes use of visual management and has gone through a Lean transformation, both in manufacturing and partially in product development. From what was observed, they might mainly have been using the visual planning and status boards as well as improvement boards, with PDCA cycles. All walls were covered in different plans, and some of them were quite extensive, covering many projects or departments. Therefore, employees are accustomed to the use of visual planning and boards. It was also noted that the Scrum had been used in software, but for longer than the Hardware teams had.

6.1.2 The Scrum-like Method

The project in question started to use what they called a “Scrum-ish” method at the start of the year 2013. Before that, they had a visual plan that had been broken into tasks for each person involved. The change that was made in the start of 2013 was that they started to use Sprints, each Sprint lasting three weeks. They had tried to use two weeks, but they felt that was to short.

No consultants were hired to assist in the implementation, and the team and its management just had a one-hour introduction presentation before starting to use the method. This included the Product Owners, one of which also served as Scrum Master.

Other hardware teams within the company had also started to use a Scrum-like method. At least one of these teams has received day or two of training that was provided by the line management. The LM seemed to think this particular team had also received this training but that was not so.

The PM thought the method worked quite well in the beginning, and there were no big hindrances or obstacles. This way of working was not fully new to the team, as they have worked with visual planning before in other projects. People were familiar with breaking down work and writing on notes. The PM thought the main difference between the Scrum like method and what they had used before was the Burn-Down chart.

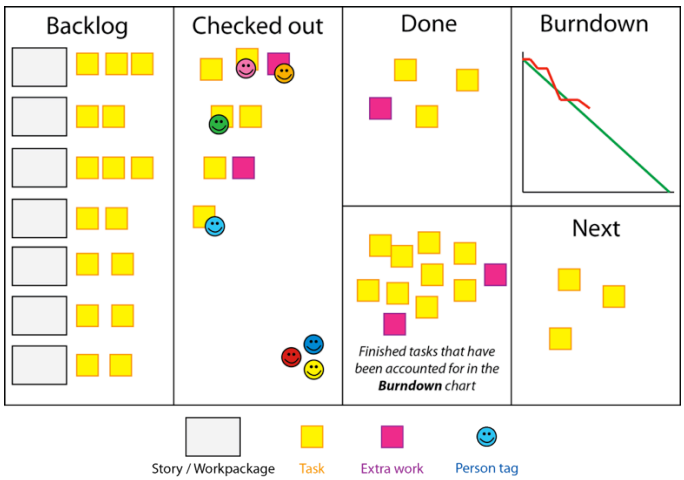


Figure 6.1 - SAAB EDS Hardware team's Scrum like visual board

The Scrum Wall

The team had a small project room where their Scrum board was on one wall. Figure 6.1 depicts the layout of the team's board, which is divided into four columns. The first three represent the status of different tasks and the last column contains the Burn-Down chart and a box for Next-In work items, which is referred to as "Next" in the figure. The first three columns are, the Sprint backlog, the checked out items and done, for work items that have been completed. In the Backlog column the Sprint stories or work packages are listed, and the tasks that adhere to each one of them. These tasks are then moved between the columns according to status. The done column was divided into two parts, and the tickets are moved from the upper to the lower section after the Scrum Master had accounted for them on the burn-down chart. The board was also equipped with little magnets with the team members faces on them. The magnets were used to identify which team member was doing what by putting them on the task notes.

Before the interview, the team's daily stand up meeting was held in their project room. The meeting was held in Swedish, therefore all spoken information was not quite clearly understood. What was observed during that meeting was that all members attending the meeting stood along the walls of the room, and one person coordinated the meeting. This person was one of the Product Owners that also served as a Scrum Master. He asked around what the status was for each task that was still in progress in the "Checked out" column. This person also stopped some discussions and suggested they would be continued after the meeting. These might have been getting too long, too technical or only applying to a part of the team.

6.1.3 Planning

The PM noted that they planned for 80% of the time available and 20% was allocated for extra work. The PM noted that this new way of working had improved the planning for the project. She noted

that before, only the PM made time estimates and then later tried to fix them. This way the whole group was united in the planning process and the result was a good plan.

Planning Before and After

The PM was used to do the planning on her own and that way she needed to know how long everything would take. The project in question included many different disciplines and components and it could be very hard to know exactly how long certain work items or phases would take. The new way of planning made it much easier, as the PM made a rough plan together with the Product Owners, and then it could be corrected afterwards for each Sprint.

Work Breakdown

The PM further noted that she had seen improvements in the work brake-down, that people were breaking the work further down by making more notes; one note for each work item. The LM added that these might be worth of 2 to 3 days of work per note instead of 10 days. The PM gave an example that a few Sprints earlier when there had been a note with an estimate of 10 days work. When the team was asked to break the note further down, they found it hard.

Prioritization Meetings

They had a planning meeting where the PM, Product Owners and one or two team member representatives from each of the different competence areas attended. There they decided on the work items for the next Sprint, sent the list out to everyone in the team and then the day after they had a planning session with the team. In this previous meeting the capacity of the Sprint was estimated indicating how much can be done in a Sprint. If during the second meeting, the team estimates more time for the work items that have been assigned to the Sprint than the capacity allows for, the remaining work items were put in the “Next” box on the visual wall. Hence, the way it worked was that the management team hoped to accomplish a certain amount of work during the next Sprint, and then the by making the estimates for what effort was included in that work, the team indicated if that was possible or not.

They had never finished all that had been planned in a Sprint yet, when the interview was conducted. The PM said that there was normally 10% left of the planned work. The units used for estimation were hours and days. The capacity was calculated for each Sprint, and therefore the number of days available differed between the Sprints.

Re-Evaluation of Priorities

When asked about adaptations that they had done to the Scrum framework, the topic of importance of stories came up. The PM was not fully sure how the Scrum framework works in this way, but had been taught that the priorities that were set for each story in the beginning should be kept throughout the project. This is not what they have done, as they have re-evaluated the importance of the stories in between the Sprints. The interviewer noted that that is more like Scrum, where priorities are re-evaluated and can be changed for the next Sprint.

When asked if anything had come up in a middle of a Sprint that changed the plan or needed extra work, the PM stated that they use pink notes for extra work. These pink notes were then used at the end of the Sprint to explain why other planned tasks or stories were not finished on time.

Furthermore, when asked if there had been any planned stories that became irrelevant during a Sprint because of unexpected events, the PM said that that had happened and the stories had been down prioritized. These had been taken up in the next Sprint and then automatically had high prioritization for that Sprint, as that story was running late.

A question was raised if there was ever a problem when these unexpected things happen that one would have to wait for external work or delivery, and might be idle in the meantime. The PM said that they actually had that problem at the time. Some parts had not been delivered on time for the team’s verification and therefore there were people waiting and did not have work to do in that

Sprint. What was done then, was to see if anything that should be in the next Sprint could be done during the current one. Otherwise the PM would contact the persons line manger to see if there is other work available for that person, for a couple of days. The PM noted that there is a risk when doing that, as sometimes people might not come back for two weeks or more.

Dependencies and Updates

Regarding deliveries to external parties, the delivery times were estimated using Gantt charts. The PM stated that with the new way of working, feedback about activities that would not be done on time were received much sooner than before.

The PM updated her boss and the in-house customer. When asked on how they felt about the updates and changes to the plan, she noted that these parties were informed earlier than was done before. She also noted that there is a buffer for some delay in a project, without the delivery date changing.

When asked if people were happier or felt more engaged by using this way of working she agreed. However, she also added that this method is also a way of making people more a part of the group. Because you have to talk about your note, you cannot stand in the corner and hide, but that was something people could do before.

6.1.4 Discussions on the Use of the Framework

Role clarity

When asked if they would develop into a more stricter Scrum framework, such as having a designated Scrum Master, which would not also be a Product Owner, the PM said that it might be better to have clearer roles, but noted that they did not have much training and thought it was working quite well for them.

No Reviews or Retrospective Meetings

The team had not been doing Review and Retrospective meetings at the end of Sprints. The reflection was at least not organized. The LM said that he found this was one of the most important parts, and encouraged the PM to make more use of it. The PM said that they might do a big reflection after the current Sprint as that would be the end of the design phase and the product would move on to verification.

Benefits of Scrum

When asked if they thought there were any benefits with using Scrum or a Scrum like method for teams, the answers mainly regarded focus and dedication. The LM noted that this specially applied when people are 100% dedicated to the project, and noted that there was still a problem of people being dispersed over several projects. This he said decreases the focus a bit, specially when there are many project leaders you have to answered to, that want you to do work for them, all at the same time. The LM further noted that he did not think that the planning part of the new method was much different from what was done before.

Future of Scrum for Hardware Teams

The LM said that what he saw in the future of the hardware teams, that these would be dedicated product teams. They would then follow the product through the whole process, as well as production. This had been a problem; products that go into production still need resources but they have been allocated elsewhere. These product teams could follow products in the way, that when a product becomes redundant they lunch a new one. He noted that they were not there yet, but he hoped to see it happen.

Previous Implementation of Visual Management

When asked, the PM did not think the team members had shown any opposition to starting to use the new method as they were used to visual planning before.

The previous implementation of visual planning had been the biggest step for people, where they had to put down in writing what they were doing. This was challenging as people might have felt that they were being watched. However, the PM had emphasized that this was only for them and she would not blame them if their estimations were wrong. The PM also noted that she thought it helps with feeling that one has accomplished something today, when it is visual like this.

The PM noted that it was hard to get people to write notes regarding their work, as they saw it as extra work. Then the PM thought that they realized the benefit in putting down in words what one was going to do. She further noted that this was important as a part of the company is in India, and they sometimes need to have work done there. When that happens a description of the work needs to be done and has to be sent to the company in India. The PM noted that this is the same as writing down notes about your own work, and that this was difficult for the engineers.

The PM noted that she had previously worked at another company that also implemented the use of visual planning. She noted that there was a fundamental difference between the two companies; the other company's strategy could be described as: "You have to work like this", while at SAAB EDS the strategy is "This is the way of working". She thought the latter one was more effective in order to get people involved and start using the new method.

Team Buy-In

When asked regarding team buy-in and commitment to the new work method, the PM stated that she thought that the team felt committed to it. When asked how the PM thought buy-in and commitment was elicited from people, she answered the reason was that they were planning together and they worked much more closely together. Everyone knew what everyone was doing, and if the team members had questions regarding the work items, it was possible to change their description to be clearer and more precise. In addition, if the management had too high goals they could discuss that and adjust the goals accordingly.

Is Scrum a Software Method?

The two interviewees were asked if they thought the method was a Software method, they both said no, and the LM added that he thought it was not a software method but a planning method.

6.1.5 Guidelines or Tips for Others

The interviewees were asked if they had guidelines or tips they would give others that were starting to use this kind of method. The PM noted a tip she would give would be to write detailed goals for the backlog items. The items have to be well defined, she possibly meant that there needs to be a definition of what needs to be done for the item to be finished.

The LM noted that it is important to find a good Scrum Master, as that person is going to make the team or system work. It should be made sure that this person knows Scrum and can teach the team and others how to use Scrum. He noted that if the person does not have the ability to communicate and work with people, he thought the meetings would turn out badly.

6.2 Andritz Hydro AB Sweden – Restructured Organization

"ANDRITZ HYDRO is a global supplier of electro-mechanical systems and services for hydropower plants and one of the leaders in the world market for hydraulic power generation".[1]

Andritz Hydro has five divisions, one of which is the Service & Rehab (S&R) division. That division is in charge of modernization and renewal, and provides solutions, products and services over the entire lifecycle of hydropower plants. The Swedish offices of Andritz Hydro AB Service and Rehab division have recently gone through changes in their organization and the way they work. This change was built on principles and practices from Scrum and Agile, and adapted to the special conditions and environment of the organization. The implementation started in January 2011, and in July 2013, the teams had gone through 21 Sprints, which includes longer summer Sprints. The work performed in this organization is based on contracted projects, not research and development. The divisions Product Manager provided the information and insights the case study is based on.

6.2.1 The Organization

The work that is performed at the S&R division is mechanical engineering. Each project is tailored to the customer, and therefore all products can be seen as unique. Their work could be categorized more as sales-design, than pure R&D product development. What characterizes the work of the Service and Rehab division is that projects are sold with fixed dates, and there are penalties for overrunning a certain dates. This means that if a project is late by 2 weeks, the whole profit of the project is burned up by penalties. This puts high pressure on planning and means that the organization cannot have any delays in their release schedules.

What is special about this case is that they do not run project teams, but instead have specialized teams for certain technical domains, built on the products topology. The product integration is performed by a small project team, which includes a Project Manager (PM), Systems Engineer (SE), in addition to a Site Manager during on-site construction. This way all technical teams are involved in almost all projects that go through the organization.

Their situation is that their operations in Sweden have expanded rapidly and due to this, it has been difficult to find experienced engineers in their domain. Therefore, it was necessary to find an alternative way of working. It was noted that they are not using Scrum by the book but are using concepts and ideas that fit their organization.

6.2.2 The Problem Situation

After the full merger of the original company (Waplans) into the Andritz Hydro group in 2008, the number of projects increased significantly. This is one of the main reasons for the need for organizational change and improvement of the way of working at the Swedish S&R division, hereafter called the organization. At this time, the organization was structured in way that there was a pool of engineers, which were then allocated to projects, and these projects were in charge of the engineers' time during the project. Projects can take 2-3 years due to long lead times, which for example is 1 year for casting. Due to this, there is low manning per project and therefore people were working on minimum 2-3 projects at a time.

The organization was having problems with managing increased market requirements, budget overshoots in larger projects, engineering loop backs in some projects, project managers being too involved in engineering management and there were concerns regarding the organization's planning. The strengths of the organization were that the working style was informal and non-hierarchical, the education level of staff was very high, and the age profile was young and therefore more likely to learn new things more easily. Future challenges identified were increased pressure to lower cost, and therefore keep engineering hours down and shorter lead times. The staff turnover was expected

[1] From the Andritz Hydro website: http://www.andritz.com/hydro.htm

to increase and administrative load would increase due to higher demand of internal and external requirements. Based on this the requirements for a new organizational structure were the following: shorter lead times without increasing engineering hours; small and large projects supported; rapid learning supported; it should be manageable; respect existing project process; and be responsible for downstream processes.

6.2.3 The New Organization Concept

In this section the new organization of the S&R division is presented. The Product Manager noted that this was just one approach that works for them, each company has to base their change efforts on their own context and the implementation has to be tailor made for the particular environment. A new organizational structure was created, see Figure 6.2.

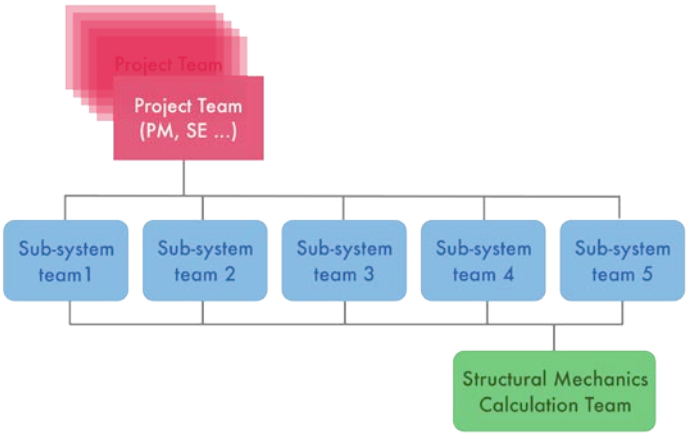


Figure 6.2 - The new organizational structure of the S&R division [2]

In this new structure, the engineering resources have been taken out of the ownership of the projects and put in to seven engineering teams. The teams mirror the physical topology of the product; that is, each team is in charge of one of the different subsystems or components of the product. In addition to this, a Structural Mechanics Calculation team services the engineering teams. Each project uses the technical support they need for their certain projects; in some cases only one of the teams might be involved, for other projects all of the teams can be involved. For each project there is a team with at least a Project Manager (PM) and a Systems Engineer (SE). The project teams are responsible for feeding the teams with the information they need, such as customer requirements, technical specification, and project specific details. The project teams are also responsible for prioritizing work packages or tasks within their projects. The engineering groups are responsible for their own work planning, which should in essence be like Scrum.

The Product Manager noted that not having project teams is a deviation from the Scrum framework, but noted that he and the consultant for the implementation had agreed that for this environment it was a good solution. Interfaces are not that complicated, they are reasonably manageable and taken care of by the Systems Engineers in the Project teams. The Project team unites the work of the different teams into one product.

One of the reasons the topological division of teams was chosen was that, before it might have taken 18 months until an engineer would repeat a task. Due to the young age profile of the organization repetition was need to gain experience. This way it is also easier to improve the process, as well as the team is responsible for the technology of each subsystem or component.

[2] Image adapted from: Henrik Lindsjö, Andritz Hydro S&R Sweden

The Product Manager noted that by changing the organization like this the project teams are purchasing results from the engineering teams instead of purchasing resources. This way the Project Teams prioritize the work they need from the engineering teams, and then the engineering teams are responsible for their own work.

The Three Planning Levels

The Product Manager explained that there were now three planning levels within Engineering: *Strategic*, *Tactical* and *Operational*. He also stated that different tools are needed for each level.

The **strategic** planning is a long-term plan that is not revised often. This level of planning was already in place. At this level, number of engineering resources needed is estimated, and it is made sure to hire the right people. The **tactical** planning is at a full project length level or 1-3 years, and is mainly concerned with engineering resources, such as roughly defining when certain types of resources are needed, critical deadlines and milestones. The **operational** planning is lowest level of planning and is concerned with the very detailed day-to-day planning. There are two time horizon for this planning level; detailed planning for the next month and reasonable detailed for the 2-3 months ahead. Operational planning is handled with the new way of working, the Scrum Framework, and is managed by the engineers themselves. At this level, it is too late to raise concerns in regards to resources.

Meeting Cadence

There are different types of meetings that have different cadence in the organization. The meetings, presented in Figure 6.3, fall in to the Operational and Tactical planning levels. Starting from the lowest level there are **Daily Group meetings**. The cadence of these meetings can differ between teams; some meet daily others 2 times a week.

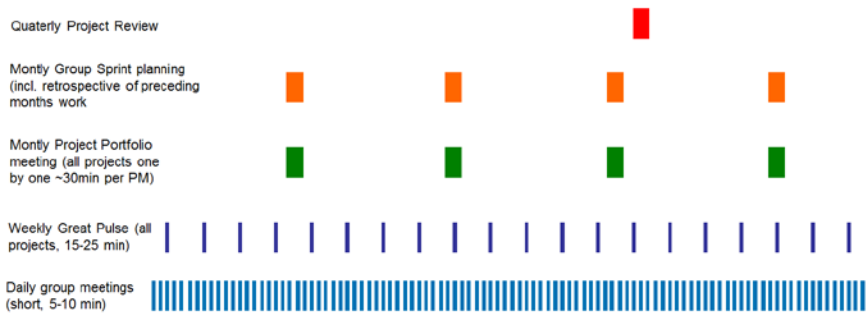


Figure 6.3 - The meeting cadence of the S&R division[3]

Pulse meetings are held weekly and cover deviations in the projects. The focus is on deviations from the plan, time schedule or quality. Cost is not covered at this meeting. **Steering committee meetings** are held monthly, where the project teams sit down with the steering committee and report on financial status. The **Sprint Planning meetings** are also held monthly and are interlinked with the portfolio meeting. The input in to the Sprint planning meeting are the requirements from, and prioritisation made in the **Portfolio meeting** with the steering committee. The Sprint Planning meeting will be described in more detail in the next section. The final meeting is a **Quarterly Review** where the focus is on finance, and is attended by a representative from Andritz headquarters.

The New Planning Method

The way the teams work is where Agile and Scrum are involved. All groups have one month long Sprints that are all coordinated. The first part of planning is done on high level where all engineering

[3] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

teams and system engineers meet. At first, all teams and systems engineers come together for a communal Retrospective meeting. Then each team is given half an hour to meet with facilitator and system engineers to receive their work packages. Systems engineers have work packages prepared for the next Sprint and have estimated their size together with the teams, prior to the meeting. During the half hour, prioritization is performed and decisions made on what will be included in the Sprint and what will not. This is also based on the directions given by the steering committee. Once this is done, each team takes their allocated work packages back to their workstation, where the team is co-located, and performs their own planning, distributing the packages internally. The teams have different planning models.

The Product Manager noted that it was important that the whole team attend the high level Sprint Planning meeting; it creates better understanding as well as buy-in within the team. The urgency is also felt in a different way than if just the team leader attends the large meeting and then has to come back to the team and repeat and report. This also allows the whole team to pose questions regarding the prioritized work allocated to them.

The Product Manager also noted that it is very important during the Sprint planning day to have built in self-improvement process, and this is independent of what model you use, Scrum or something else. The Retrospective meeting at the start of the planning day serves this purpose and the organization has a special continuous improvement board to keep track of improvement projects. Retrospective meetings handle deviations, which can be in the internal work processes of the teams, the whole departments or questions that have to be elevated to a higher level.

6.2.4 The Work Method in Action – Visual Management

In this section, the visual management tools that aid the different meetings of the S&R division, will be presented. The team boards or walls that are presented are all from the pilot team, the first team that started to use the Scrum-like method in their daily work.

Weekly Pulse Meeting

The weekly pulse meeting is held every Friday. There the project teams and the engineering teams meet with the line organization and the status is presented. The board used at this meeting can be seen in Figure 6.4.

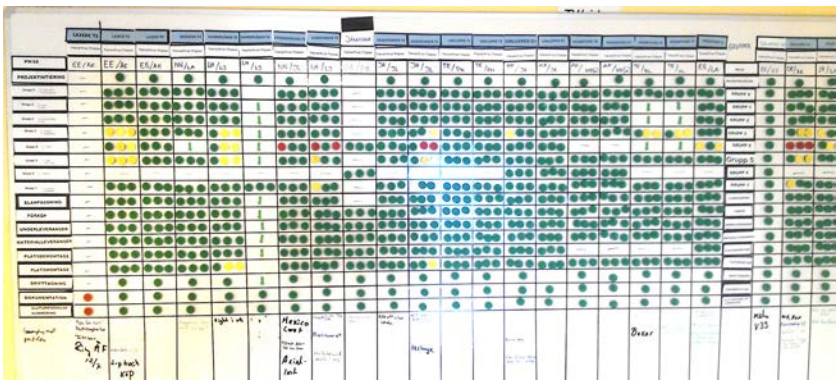


Figure 6.4 - Weekly Pulse Meeting board [4]

Each column represents a project; there can be around 20 projects on-going in parallel. The rows represent the different engineering teams and other departments that concern the projects. The value stream is from the top down. The three check points or dots in each box represent engineering, production and time schedule. If there is only one dot in the box, it means that there is just one activity. They have three different colours; Green means all is well; yellow means there is a problem

[4] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

but there is an action plan; Red means there is a problem and there is no solution to it and the people in concern cannot handle the situation on their own.

The Product Manager noted that this is a good tool and provides good overview for management of daily bases status, but a board like this should be set up depending on the way each organization works. Everyone is allowed to attend the weekly meeting, but the ones that need to be there are the project teams, Line Managers and team leaders.

Sprint Planning

During a teams allocated time at the high level Sprint planning meeting it receives the different work packages from the different projects. The SEs of the different projects put their requests on the team's board as can be seen in Figure 6.5. The figure shows a prioritization board of one component team. It shows how the different work packages are colour coded for the different projects. The packages are prioritized and the engineering hours needed for the requested work packages and the capacity of the team is compared.

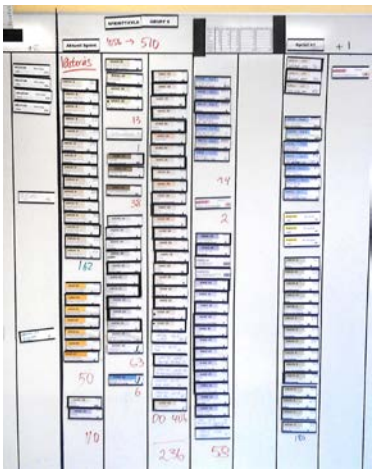


Figure 6.5 - Team prioritization board [5]

On the right hand side of the board, one can see the work packages separated from the others. These could not be taken into the Sprint due to the team capacity being lower than the engineering hours needed to finish all tasks. These will be first in line for the next Sprint, and might serve as next-in stories if all other work packages are finished before the Sprint is over.

The Product Manager noted that it is better to have a mental model of this system being a prioritization method, not a planning method. If people's mental model is that this is planning method, then they will show up to this meeting wanting to know about the future. However, if there is not enough resource capacity at that point in time, nothing can be done; that should be dealt with on the tactical level of planning, 3-6 months earlier. Therefor the Product Manager and others have started to "re-image" this work method as a prioritization tool.

Team Planning Board

Each team has their own day-to-day planning board, and they are all different. They have been given a certain amount of freedom when it comes to designing their boards and the way they work. An example of such a team board can be seen in Figure 6.6, which is the pilot team's board. The pilot team that was first to implement the new way of working. Their board is the most advanced of the teams.

[5] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

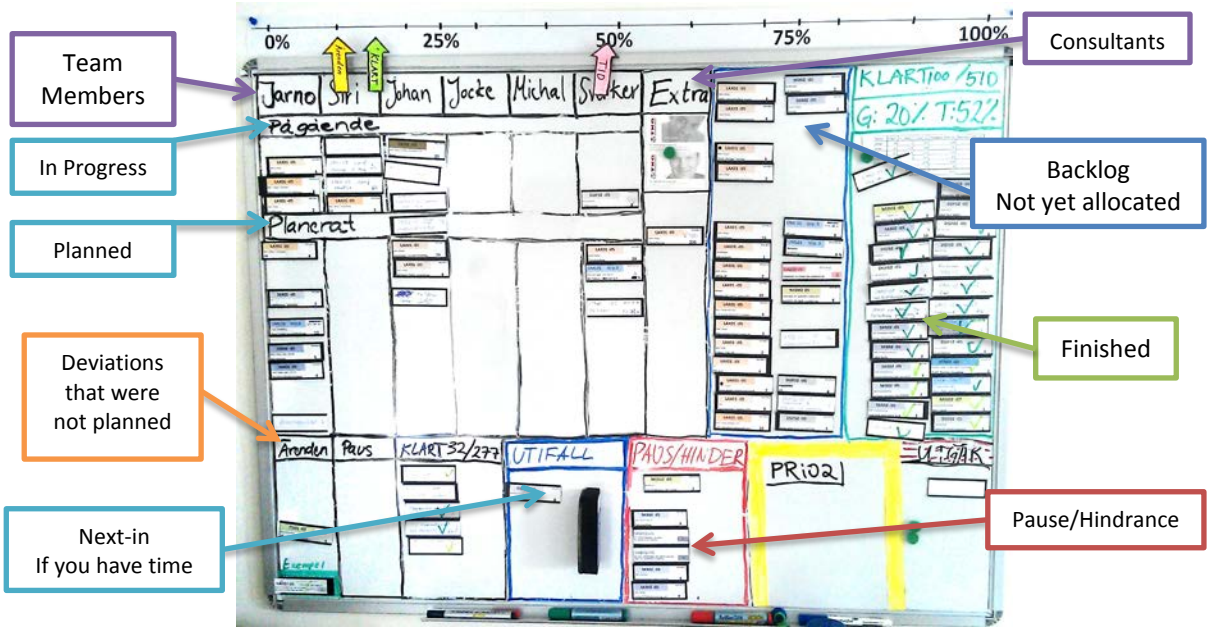


Figure 6.6 - Team planning board / Scrum wall [6]

On the top left of the board are columns for the different team members, and there are some work-packages allocated to them, that are under either the *In Progress* or *Planned* (but not started) status bars. To the right of those columns is the Backlog with items that have not yet been allocated to the team members. To the right of the backlog, in the green box, are the items that are finished. At the bottom of the board are some extra blocks. These regard deviations, extra work that was not planned in this Sprint and hindrances where task that cannot continue for reasons the team has no control over, and therefore paused and put in that box.

At the top of the board is a timeline with three arrows, see Figure 6.7. The scale is 0-100%, and the pink arrow represents how much time has elapsed of the current Sprint. The green arrow represents how much is finished of the planned work and the yellow arrow represents how much extra and unplanned work has been added to the Sprint. In the beginning of a Sprint, there is often a large distance between the pink and green arrow, but the distance decreases closer to the end of the Sprint. The timeline is updated during the daily meeting.

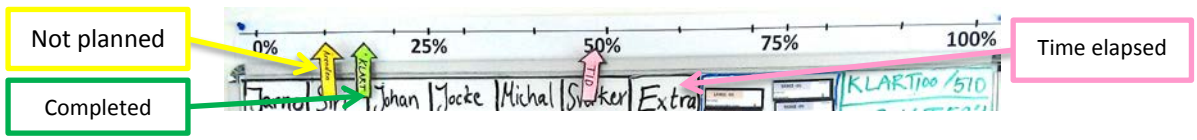


Figure 6.7 - Sprint Burn down timeline

The Product Manager noted that he walks to all the teams every Friday, and talks to each one for 15-30 minutes. This gives the teams an opportunity to ask questions.

Continuous Improvement Wall

It was noted in a previous section that the Product Manager found it essential to have a built in self-improvement process, in the work system. At the S&R division, they have a shared board between all the engineering teams and the system engineers for continuous improvements. The board can be seen in Figure 6.8.

[6] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

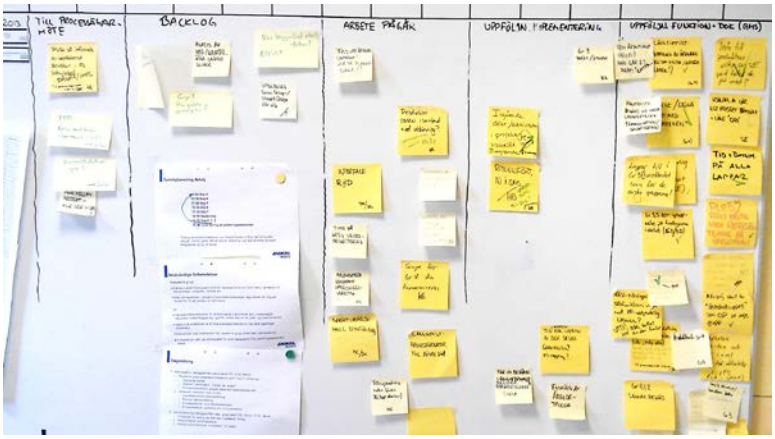


Figure 6.8 - Continuous Improvement Board[7]

The board is divided up into *Backlog*, *In Progress*, *Finished and implemented* and *Follow up*. The Follow-up section is very important, as it makes sure that the new improvements work as expected. An improvement note is not taken from the board until the change works, has been verified and documented and put into their ISO 9000 management system. The board is fed with notes that are created at the monthly Retrospective meetings. At the same meeting, they also go through the status of the notes posted at the previous Retrospective meeting.

The Product Manager noted that the process is very organic and that the people push for changes that really matter. If the change effort is important it happens, if it is not, nothing happens. That is ok, what matters is making all this visible.

6.2.5 Implementation Results

The implementation was done team by team and the pilot team was the most progressive team with most experience. They started in January 2011. All teams participated at the Retrospectives and could see how the pilot team was working, but they were not all forced to start working in Sprints. Then later on the other teams requested to start using the new method, after seeing the pilot team use it. ***The Product Manager noted*** that to a certain extend this was a top-down implementation, but it was not a problem as the pilot team felt they were not delivering and anything would be better than the way they were working before the implementation.

What Happened?

In order to see if the work method was making a difference, the Product Manager plotted the engineering hours consumed over time for each project that the pilot team worked on. A reference project, from before the implementation was used to compare with. Figure 6.9 presents the comparison of the first project the pilot team worked on after the implementation to the reference project. The y-axis represents accumulated design hours of a project, and the x-axis represents days passed in real-time. A curve represents the work done by the pilot team for a particular project; that is designing and building their component. Each curve has three steps:

1. Concept & Basic design - Few engineering resources, therefore a slanting curve
2. Detailed design - Many engineering resources, therefore a steep curve
3. Manufacturing - Few engineering resources, therefore a slanting curve

[7] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

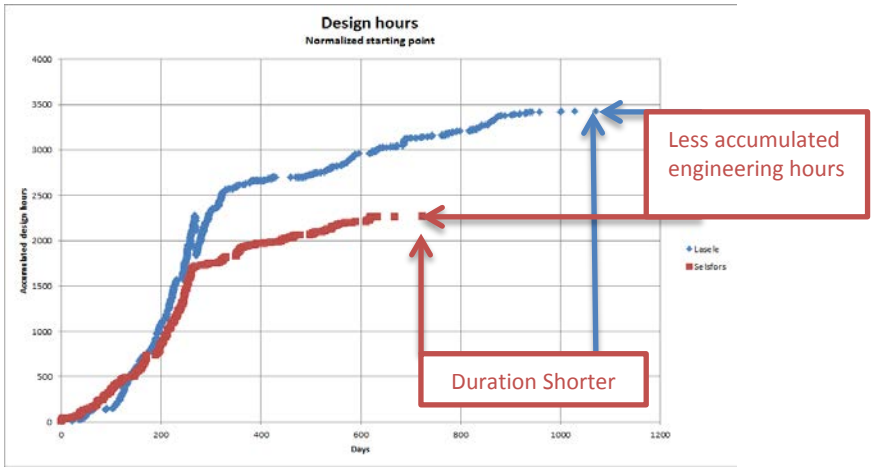


Figure 6.9 - The pilot teams first project after implementing the new process compared to a reference curve from before [8]

In The **Blue** curve represents the reference project and the **Red** curve represents the pilot team's first project after the new method implementation. As can be seen the team delivers sooner, using less engineering hours. For the following projects of the pilot team, an obvious improvement trend can be seen both in lead time and engineering hours spent, see Figure 6.10. The purple curve represents a project that was on going at the time.

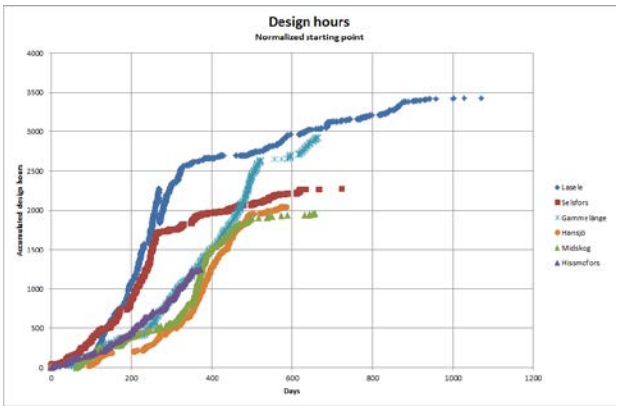


Figure 6.10 – Further improvements in the following projects, in the following order: Light blue, Orange, Green, Purple [9]

What the Product Manager wanted to note regarding these results was that several things should be considered:

- Experience level of the team was low. There was one team member with 8 years of experience within the company, but the other five had only been for 1-2 years with the company. Still they were able to deliver these results.
- There was enormous influx of new inexperienced employees.
- A new CAD system was implemented during this time.
- There were several new versions of component designs made during the time, resulting in low repeatability between projects.
- There was much better control over drawing quality, fewer loop-backs from production.
- There was much better control over the early phases in design.

[8] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden
[9] Image: Henrik Lindsjö, Andritz Hydro S&R Sweden

- Continuous improvement and self-improvement is built into the system.

As stated before, typically if a project is late by 2 weeks, the whole profit of the project is gone. It was noted that there is not as much budget overshoot now after the change of the organization and the way they work.

Three months after the main correspondence with the Product Manager, which this review of the organization is based on, he gave an update on the situation. He stated that the improvements seen previously were to a certain extent lost, as an increased trend of engineering hours spent can be seen. He stated that it is difficult to identify the reason for this, but he suggested two: Firstly, that the requirements from customers are increasing, secondly and more importantly that the organization is under resourced or overloaded. He stated that no system will work if there are insufficient resources available, and added that he thought the main constraint the new organization structure faces is the reluctance to have enough engineering resources available. That leads to long-term overload, people going between different projects and re-starting tasks too often, and this results in it being almost impossible to achieve good results.

The Product Managers reflections on the new method

The Product Manger noted that it is important that the concept is simple and rudimentary and that it is developed organically through the self-improvement process. That is, in the beginning the concept should be simple, and then the rest is developed by the teams. When they come up with good ideas, they should be amplified. Bad ideas management can dampen and try to get rid of those.

Continuous Improvements

The Product Manager noted that he thinks that the Retrospectives are the main part of the concept. He further noted that he is the Process Master and protects the process. Recently the team leaders were identified for the teams and there is a possibility that they might also take the role of process masters in the future. The team leaders that were just added before the summer of 2013 will have a role to some extent resembling that of a Scrum Master.

Monitoring Planned and Unplanned Hours in Parallel

The Product Manger noted that monitoring of planned and unplanned hours should be divided and managed in parallel. This idea came from their Agile and Lean Product Development expert, Mikael Lundgren. Planned hours should be allocated for and unplanned hours managed. The unplanned hours should be measured and what they include should be identified. An example the Product Manager gave was that a team complained about always getting questions from the workshop. When the team had to investigate and report the disturbance, it was identified that the team was to a significant level itself a part of creating the problems and disturbances. This could for example be due to them not being detailed enough when sending requests to the workshop, such as insufficient information on drawings etc.

Another example is the Structural Mechanics Calculation team, which works as a service team for the all the engineering teams. It is very important that there is a parallel system for them, as the other teams need them to be available. Therefor unplanned hours should be allocated for, and only 50% of their time should be planned. He further noted that in this kind of system, management cannot just continue to push work into teams. This puts pressure on management to make decisions and this is often where these systems fail.

Engineers are Happy

The Product Manager claimed that the engineers were very happy with this way of working. Now they get a management decisions every month and they know every day what to do, but before the change, it was a mess. In general, his impression is that they are happy.

The Product Manager later added that there is still frustration in regards to unplanned work. The system does not provide a functionality of how to handle it. He notes that the system makes it

obvious when amount of unplanned hours are increasing, but that it is very difficult to react to the change. This will be the issue that will be focused on next in the organization; to regain control over the prioritization and the unplanned work that is feed into the design teams.

Important to have External Consultation

The Product Manager noted that it was very important to have an experienced external resource in an implementation like this. It is important that this is someone trustworthy and external to the organization. The original idea was created and laid out by the Product Manager and then an external expert came in for 5 days and consulted on the system and its implementation. Although the expert was there for only 5 days the Product Manager noted that without him the implementation would have been much more difficult, even impossible. He could build on his experience, and predict what would happen if things were done in a certain way.

Limitations of the Method and Implementation

One of the limitations to the implementation that the Product Manager could identify was that they did not have team leaders from the beginning. But then again it was deliberate as it would have been impossible to choose the right one. They only appointed people to the post of team leader in early 2013, 6 months before the correspondence. Most of them were obvious for the post as they were very technologically strong.

Another limitation is that they do not have Sprint Reviews every month, just when they are needed. The Product Manager would like them to be every month where the teams could present visually what they have accomplished over the preceding Sprint.

Tips and Advice to Others

The main tip the Product Manager would give others intending to implement a new process like this was that the self-improvement and retrospective is the most important part and it should be ensured that it works. If that is done, one does not need to be so accurate as long as one mentors and coaches the process. One has to make sure one can handle whatever comes out of the process. One can also drive the Retrospective work. What one wants is to teach the people to self-improve.

He also noted that it is even better that the system or method is not even fully finalized in the beginning, because it allows the people to develop and make their mistakes. First, they need to start using it and then they can start using it to improve the product.

Is Scrum a Software Method?

The Product Manager did not necessarily see Scrum as a Software Development method. He noted that possibly that something is built every month might be more of a Software development practice, which is difficult in mechanical development. However, it might depend more on that the more uncertain the development, the more important the demos or Reviews are.

6.2.6 Comments from the Agile and Lean Product Development Expert

The Agile and Lean Product Development expert, Mikael Lundgren, made some notes in regards to the implementation and made some recommendations. He stated that in this case, it was not possible to use cross-functional teams; the engineers support each other and spreading them out into teams would have resulted in this support being lost. He stated that, although going against his experience of many years and being the biggest deviation from the Scrum Framework, making the teams functional was the key to the success of the new system.

In regards to adaptations to the Scrum framework that hardware teams might need to do, he noted that the natural cycles that occur in hardware development need to be respected, as these are governed by lead times for ordered parts. In regards to cases where hardware and software teams collaborate, he noted that time needs to be allocated for the software team to do tests on the hardware.

His main tip for those who intend to implement Scrum in mechanical teams, is not to use the Scrum terminology at all, rather talk about iterative development and how to structure plans and feedback loops. He noted that in another company where he implemented Scrum for mechanical engineers, where they felt this was a method tailored for software, what solved the issue was to allow the mechanical engineers work in shorter cycles than the software teams. They were also allowed to change the iteration length. They realized that the consultant was not using language specific to software and that calmed them.

He noted that in the case of Andritz Hydro, management of estimates is much better, and that they are managing stress on much better level. Co-operation has also changed, as before people bickered about issues between different departments, for example development and construction. Now everyone knows who to talk to in regards to different projects and components, due to the visual management boards being visible to all in their canteen. This way the information has stopped being people centred and has moved onto the visual boards.

He also stated that this was a top-down decision, was a success factor. He states that he thinks that Scrum and Lean initiatives will never succeed without top-management support. He does not take on assignments where top-management is not involved.

7 Analysis and Discussion

7.1 Comparison to the Scrum framework

In this section the Marel case will be compared to the literature on the Scrum framework presented in chapters 4.2 and 4.3.

7.1.1 What is Scrum and Why Scrum?

The work that the mechanical Marel team is doing is new product development, where there are many uncertainties and the product is complex, as it deals with technology that is complex and new to the team. There are also high hopes and pressure for the products success by the management. This fits well with Schwaber and Sutherland's (2011, p.3) description of Scrum as *"a framework for developing and sustaining complex products."*

In regards to using Agile practices in Hardware development, or in non-software projects, Jim Highsmith stated the following, previously presented in chapter 4.3:

> *"In any project that faces uncertainty, complexity, volatility and risk, there is a place for agile practices and principles." (Jackson, 2012, p.61).*

This statement aligns with the team and project in question, as there is a lot of uncertainty, risk and complexity in the particular product the team is working on.

Schwaber and Sutherland also state that Scrum is based on empiricism that "asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk." (Schwaber & Sutherland, 2011, p.4) As mentioned earlier, the project in question, as well as the general work of the team, is characterized by uncertainty and sometimes unpredictability, as they are dealing with many "unknowns". This is why the concept of empiricism, gaining knowledge through experience and base decision on that knowledge is applicable in this situation. It is possible that some technology and concepts will not work, and plans therefore need to be adapted. By working iteratively in Sprints, it has been possible to incorporate new knowledge acquired into the plans. This cadence of regular planning helps with making realistic plans for the upcoming Sprint, and helps with dealing with risk and unpredictable events that change the scope or direction of the project. Therefore, the iterative and incremental approach fits this situation quite well.

The team's work is in a sense iterative, as they start working in rough concept design of the machine and its subsystems, and then go into further detail later on. However, it is possible that the iterative and incremental approach could be utilized even further, with a more conscious division between the design iterations and more clearly defined product increments. As the different sub-systems are developed in parallel, the iterations would imply the product to be designed in more detail designed than in the previous iteration.

Situation Prior to Scrum

The Product Owner (PO) described the situation before the mechanical developers started to use Scrum. The main points were that too often people were working individually on developing products, then became specialists in that particular product and carried it with them through their carrier. A result of this is that no review or dialog occurs, which further results in a homogeneous product solution. The PO also noted that planning was new to the team. There had been plans but those only had few milestones that were far apart, and it was an unwritten rule that delays were accepted, and it was often the mechanical part that caused delays. He also noted that often the realization that a deadline would not be met, did not occur until it was too late, causing delays.

The Scrum framework has provided benefits concerning these issues: The group of engineers has become a team that works together as a whole, not as individuals. They are planning every 2 weeks, which means they are setting milestones shorter apart. The meetings of the framework, such as

Daily Stand-ups and the Review meetings, bring forth problems that might cause delays to the project, and can therefore be dealt with right away, in contrast to realizing that the problem can cause delays, when it is already too late.

7.1.2 Scrum Roles

The three roles of the Scrum framework, the Product Owner (PO), the Scrum Master (SM) and the Development Team, form the Scrum Team according to Schwaber and Sutherland (2011). They further state the Scrum Team should be cross-functional; implying that all competences needed to finish the work should be included in the team, and it should not depend on others that are not apart of the team. When comparing this statement to the mechanical team at Marel, the first result is that it is not cross-functional as the team only includes mechanical engineers. However, all the competences needed to design the machine, its mechanics and structure, are included in the team. On a product level, the two teams, mechanical and software, can design and develop the product as a whole, although that they still need technical assistance from other departments. They are also dependent on others for manufacturing and purchasing of components and construction of the product.

The team is not cross-functional as prescribed by the theory, but there are examples of non-cross-functional teams functioning well, for example at Andritz Hydro AB (see chapter 6.2). In regards to the whole development team, the co-operation between the mechanical and software team have improved and this can be seen as one of the benefits of using the framework for two co-operating teams. They are not fully cross-functional, but it has improved the communication and co-operation between the different functions: Mechanical, software, production and procurement.

Schwaber and Sutherland (2011) also state that Scrum Teams should be self-organizing; this means that they do not get directions from parties external to the team, but choose themselves what is the best way to accomplish their work. As a Scrum Team, including all roles of PO, SM and development team, the team is self-organizing. The development team needs assistance with their planning, but they decide themselves how to perform the technical aspects of their work. Sometimes the PO has influenced the technical solutions, and this has frustrated the team. In summary, in most sense the team is self-organizing.

Product Owner

It is stated by Schwaber and Sutherland (2011) that the Product Owner (PO) "is responsible for maximizing the value of the product and the work of the Development team" (Schwaber & Sutherland, 2011, p.5) and Schwaber (2004) states that the PO does this by managing the Product Backlog. The PO in the Marel case is and was responsible for both the product's value and both of the development teams' work, but did not directly manage a Product Backlog. In a sense, there was not an active physical Product Backlog; instead, the Release plans have served as the Product Backlog, both the first one that was more project oriented, and then the latest one that was more product and component focused. The way it has worked like is that the teams plan their Sprints, and the PO then approves that plan, or makes comments about it. Both the hardware and software team wrote all their Release and Sprint stories. The stories are quite technical, and are very solutions oriented. The mechanical team talked about that it would just be waste of time for the PO to prepare these.

On this note Schwaber and Sutherland (2011) state that the Product Owner is the only one responsible for the Product Backlog management, but that he can have the Development team carry out management tasks, such as expressing the backlog items, prioritize them, make sure they are clear to all, etc. The PO is still always accountable for this work.

As prescribed by Schwaber and Sutherland (2011) the PO is one person and not a committee. They further state that only the Product Owner is allowed to tell the Development team what requirements to work from, and the team is not allowed to act on what others might say. In regards to this, it was common that people came to the developers and asked them for help on projects

related to older designs they had made, and they did go and help. After the priorities were made clear, and that the current project should be focused on, the team started to try to deflect these queries, and from the progress they made, it can be assumed that this affected their work efficiency positively.

The PO noted that the team was not addressing the risks and wished they would be more aware and do more about it. However, another PO within the firm noted during an informal conversation, that this was the PO's job not the teams. This might indicate that the responsibilities are not fully clear to the PO and team, and might need to be clarified and defined, and it is possible that the PO needs to be more involved in addressing risk.

Schwaber (2004) states that the PO represents all of the project's stakeholders, those that have interest in the project outcome. For this project and product, these stakeholders are both external possible clients and in-house experts not directly connected to the project, but they have opinions on how the system should work. There have been many different opinions, many of which contradict each other, and as the PO stated in his interview, it was not clear which one to follow. Nevertheless, the PO has represented their views, has been the one in most contact with stakeholders, and has made changes to the product scope in line with their opinions, and this is in line with what is prescribed by the framework. However, what might not be in line with the framework is that when the PO has changed the scope it has been in regards to technical solutions, which should be the team's responsibility. This will be discussed further in the section on the development team.

In conclusion, the role of the PO in regards to addressing risk and involvement in prioritization could be clarified and defined further. It is also possible that the PO needs to be further educated on the Product Owner role and the framework as a whole.

Scrum Master

Schwaber and Beedle (2002) state that the success of Scrum is the responsibility of the Scrum Master. From this statement, one could assume that if Scrum is successful, the Scrum Master has done their job right. At least the team was content with the framework, and feel that it benefits them and that is an important measure of success.

Schwaber (2004) states that Scrum Master is "responsible for the Scrum process, its correct implementation, and the maximization of its benefits" (Schwaber, 2004, p.142). Deemer *et al.* (2012) describe the Scrum Master as a teacher and coach, which makes sure everyone understands Scrum principles and practices. The Scrum Master that took over the team in Sprint no. 7 is not a certified Scrum Master, and might need to deepen her knowledge of the principles and theory. It is also possible that the team could do with regular and further education and training in the framework.

Schwaber and Sutherland (2011) describe the Scrum Master as a servant-leader who's responsibility is to ensure that Scrum is understood and enacted, by making sure that the Scrum Team complies with Scrum theory, practices and rules. Servant-leader is a good description of the role, and that is what the Scrum Master has been in for the mechanical team. The Scrum Master is not a superior to the team, rather their friend in a way, but should drive them forward and guides them in using the system, by teaching and coaching.

Deemer *et al.* (2012) agree that the Scrum Master serves the team and stress that he or she is not someone that manages the team members, nor is he or she a project manager, team lead or team representative. The mechanical team's Scrum Master has not served any of these roles, and thus she complies with the framework.

Schwaber and Sutherland (2011) state that the Scrum Master should also help those external to the team understand which of their interactions with the team are helpful and which are not, and helps everyone change these interactions in order to maximize the Scrum Team's value creation. This is something that might not have been purposefully done, but could be a good way of improving the efficiency and effectiveness of the team.

In accordance to Schwaber and Sutherland (2011), one of the responsibilities of the Scrum Master has been to facilitate events. According to Schwaber and Sutherland (2011) the main responsibilities of a Scrum Master is to service the Development Team, Product Owner and the Organization, by coaching and facilitating events, among other things. The Scrum Masters at Marel have complied with this. They further state that the Scrum Master services the Development Team by coaching them in self-organization and cross functionality; removing impediments; teaching and leading the team towards high-value product creation. The Scrum Masters have maybe not purposefully coached cross-functionality, in other ways than noting that all team members should be able and allowed to take on any task in the Sprint plan, and this has seemed to work out. In regards to self-organization, it can be concluded that the Scrum Master has supported the team to self-organize their work, make plans, and to delegate task themselves within the team, during planning and daily meetings. The Scrum Master could have been more active in removing impediments, but might have assisted in identifying these and make sure someone takes on removing them.

Schwaber and Sutherland (2011) state that the Product Owner is served by coaching and teaching in regards to Product Backlog management. This was not done in concise way, and its possible that the Scrum Master could guide the Product Owner more and get him more involved in the backlog work.

Schwaber and Sutherland (2011) state that the Organization serviced by leading and coaching Scrum adoption; planning implementations; work with other Scrum Masters to increase Scrum applications effectiveness; and "help employees and stakeholders understand and enact Scrum and empirical product development" (Schwaber & Sutherland, 2011, p.7). This is exactly what the work of the Agile Center Manager and the mechanical team's first Scrum Master, is all about. In essence, her work concerns the empirical product development although that term has not been used.

Next steps are to have a shared Scrum Master for both the mechanical and embedded software team. The expectations are that this will provide better co-ordination and communication between the two teams, as well with other parties such as purchasing and manufacturing. In a way, risk will be handled better.

In conclusion, the team is happy with the current form of the Framework, therefore in some sense Scrum is successful, and the Scrum Master has done their job. Things to improve would be the following:

- Dealing with hindrances, could be taken over more by the SM, than before.
- The SM could coach the PO more in regards to Backlog management.

Development Team

Schwaber and Beedle (2002, p.35) state: "A team commits to achieving a Sprint goal. The team is accorded full authority to whatever it decides is necessary to achieve the goal." Early in the experiment the Sprint goal was often unclear and the team was not always aware of what it was, and did not work purposefully towards it, although they were working on what was included in the Sprint. But during the later Sprints of the experiment the goals became more defined and the team more decisive to reach them. This might be explained by the fact that in the later Sprints, the team was working towards physical deliverables to the workshop, but they will also have gained practice in setting goals. During the later Sprint there was also some overtime put in to accomplish these goals. However, due to the nature of development in the later stages of the experiment, which was detailed design and delivering drawings to production, the work might have been more tangible in contrast to the conceptual work before. Therefore, the goals might have been easier to form in the later stages.

Cross-functional teams: Both Schwaber (2004) and Schwaber and Sutherland (2011) state that the team should be cross-functional. As mentioned above, the Marel team cannot be defined as purely cross-functional, but within the team there are most of the competences needed to design the mechanical part of the machine, all though some guidance is sought from other departments. In the sense of all competences needed to design and build the whole product, they are not cross-

functional. This might be one of the differences between Scrum in application software and Scrum in hardware development, were embedded software teams and mechanical hardware teams develop products together.

Self-organizing teams: Schwaber and Sutherland (2011) states that Development teams should be self-organizing, and the mechanical team at Marel is, as they decide within the group who does what, and are not told by either the PO or the SM, who should perform what task. They are also self-organizing in a way that they create all the tasks, and list things that need to be done, although they might get assistance in writing them down, as well as in organizing their plans. They design the machine together as a whole, and it is not predefined who designs which part or performs what task. However, they can improve even further in becoming more self-organized, resulting in less need for assistance.

Working product increment: Schwaber (2004), and Schwaber and Sutherland (2011) both state that that the team should deliver something, an increment of "Done" product or part of software, each Sprint. Neither the mechanical nor the software team do this but they attempt to reach the goal they set for the Sprints. In the later Sprints of the experiment, what the teams delivered were drawings and bill of material (BOM) lists, which in a sense are increments or parts of the product. It could be possible for teams to deliver some kind of mock-up each Sprint, but this would need an organizational change where the teams would have the possibility to create quick mock-ups, or the workshop would be organized in a different way. In this sense, the Sprint could last longer to accommodate construction of mock-ups, but as there has been high uncertainty, and rapid changes and discoveries in this project, this would make the teams Sprint plans redundant.

Team characteristics: In regards to the characteristics Schwaber and Sutherland (2011) describe, the team fits many of them: they do not bare titles within the team; they have shared ownership and accountability of the whole product, although some have specialized skills best suited for different sub-systems. They further state that there should not be sub-teams for different domains; in a sense the software and mechanical team are sub-teams of the product team, but they are not dealing with the same problems and do not speak the same technical language. Therefore there is much more value in having the teams separate with good co-operation and communication between the teams, rather than having one big team with shared Sprint plans. In regards to sub-teams within the mechanical team, some of the engineers have taken over special parts of the machine, and they might not have seen it as convenient to rotate within the team. Despite this, other team members are always informed of the status of different subsystems, via the Daily meetings and general communication throughout the workday.

Told what but not how: Schwaber and Sutherland (2011) stress that no one tells the team how to perform their work, and that only the team creates an increment. That is, the PO can tell the team what to do, but not how to do it. This rule might have been breached with changes the PO made during the project, due to problems that have come up or new stakeholder information. Then the PO came up with technical solutions to solve these problems, which the team did not always agree with. It is possible that this could have been done differently, posing the problem to the team and asking for solutions, rather than pushing a technical solution on the team. Otherwise, the team might lose ownership of the solution and the product. This could be specified in the clarification of the responsibilities of the different roles. In general, the mechanical team has decided how to deliver the functionality required of the machine.

Team size: Regarding team size, Schwaber and Beedle (2002) note that a team should have seven members, give or take two. The team in question has only four members, which can be considered as a small team, the question is if it is too small. Schwaber and Sutherland (2011) note that the team size should be "small enough to be nimble and large enough to complete significant work" (Schwaber & Sutherland, 2011, p.6). This can be considered true for the mechanical team, as they perform significant work each Sprint, and are quite nimble, in a way it is easy for them to communicate and have overview of everyone's work and status, and can therefore more easily co-

ordinate and react to change. Schwaber and Sutherland (2011) further note that if a team has fewer than three members interaction decreases, which results in lower productivity gain, and small teams can encounter skill constraints, which might result in them being unable to deliver. This is still not the case for the mechanical team. They interact well, they have a good skill base and have been able to deliver. What Schwaber and Sutherland describe here, rather characterizes the way of working before Scrum.

In conclusion, As a mechanical team producing mechanical structures and functionality, the team members have a varied skill base and are able to deliver what is needed as a team. In that sense, they are cross-functional. However, on a product level, the team is not as the software and mechanical teams are separated, but work closely together, and they are dependent on other departments in regards to construction and purchasing. The team is small, but they are almost self-sufficient in regards to skills, self-organized and communicate well within the team.

At the end of Sprints, there are no deliveries of potentially shippable tested product increments or mock-ups. However, it is not worth lengthening the Sprints in order to accommodate for building mock-ups. There is more value in being able to react to change, and having realistic short-term plans, than always having a tested product increment ready at the end of a Sprint.

7.1.3 Scrum Artefacts

In this section, the analysis and discussion regarding the Scrum artefacts and visual Management will be presented. The artefacts include the Backlogs, Backlog items, Definition of Done and the Burn-down chart.

Backlog and Backlog Items

In regards to the Product Backlog, Schwaber and Beedle (2002, p.32) state that it is "an evolving, prioritized queue of business and technical functionality that needs to be developed into a system." Schwaber (2004, p.142) defines it as a "prioritized list of project requirements with estimated times to turn them into completed product functionality", which he further states should be in hours. He further states that the Product Backlog items can be functional and non-functional requirements, as well as issues. When comparing this to the Marel case, it can be stated that there is not really a physical Product Backlog available, apart from the Release plans that were made at the start of the experiment, and then the one created before Sprint no. 10. These were not expressed as requirements, rather as different projects and subsystems in the first Release and then in the later Release plan the Release stories were expressed as subsystems and components. In a way, the Release Plan serves as Product Backlog, but it has not been evolving or managed by the Product Owner.

In regards to backlog items and their estimates Deemer *et al.* (2012) state that the Scrum framework does not define a special technique for expressing or prioritizing Product Backlog items, and neither does it define a technique or units for estimation. At first the understanding of the mechanical team and author, that it was a deviation from the Scrum framework that the user stories were not written from the perspective of the user, but Deemer et al. contradict that view. This also means that if the team would stop using points and planning poker for estimation, this would not be a deviation from the framework. User stories and story points as described by Cohn (2004) and (2005), should be seen as techniques of expressing backlog items and estimating them, and can be very helpful. However, not using these techniques is not a deviation from the Scrum framework.

In regards to backlog item structure, it was noted by an Agile expert that he thought the focus was moving from the actual structure, as it is not that important. What he thought was important is the ownership of the backlog items, and further noted the following:

> *"As long they reside in the product Backlog the Product Owner is responsible for them, meaning that they will be expressed in user value. But as soon as the team takes ownership, they need to*

be expressed in acceptance criteria, quality criteria and analysis design. So they will evolve over time." [10]

In regards to ownership, the team has high ownership over their Sprint stories, as they create them together during Sprint planning. What might be lacking is a Product Backlog of which the Product Owner has ownership. This might be explained by the nature of the development, and that all features need to be included in the product for it to work as a whole and lower priority features cannot be added in later. The PO's ownership might be on a higher project level; the different products he intends the team to work on, but not necessarily, what lower level items should be worked on each Sprint. He possibly has ownership of higher-level features or subsystems that go into the Release plans.

In the Agile experts comment above, it is stated that the team should express backlog items in acceptance and quality criteria, as well as analysis design. Some of this might be related to software development, but the team could improve in writing acceptance and quality criteria for their stories. However, they should only do this in order to help them perform the work, but not as a documentation practice. This is something that up to this point, they have been reluctant to do. This might be one of the things that could be improved in the future.

Definition of "Done"

Schwaber and Sutherland (2011) state that a common understanding of when work is defined as complete, must be shared and agreed by everyone involved; those that perform the work and those accepting it. In order for work to be "Done" it must conform to the agreed definition. This has not been done for the team, and is sometimes unclear. It might be difficult to define a general definition for all tasks they complete, but is certainly something that could be done.

Burn-down

Schwaber (2004, p.141) describes a Burn-down graph as "the trend of work remaining a across time in a Sprint, a Release, or a product". Data is based on the Sprint or Product Backlog; work remaining is tracked on the vertical axis and the time on the horizontal axis. In the case of the mechanical team, after the latter Release plan was made they only have a Burn-down graph for the Release. They do not see the point in having a burn-down graph for each Sprint, but it is possible it could help and could be tried in the future.

Visual Management Scrum Board

Having a visual management board or wall is not prescribed by the framework but Deemer *et al.* (2012) notes that many teams present their Sprint Backlog in the form of a wall sized task board, often called Scrum Board. Having a wall like this has helped the team get overview of the project as well as the progress of the current Sprint. They noted that they used it to be updated to what has happened since they have been away, for example. They also use it to see what to start working on next.

7.1.4 Scrum Events

In this section the Scrum, events will be discussed, and how the team has complied with the framework described in the literature.

It was noted in informal conversation, that the team thought that Scrum and the Scrum meetings took too much time. It is understandable that the team thought so, as they were not used to meet up for Planning meetings and Reviews every other week. It cannot be proven fully, but it is quite possible that although these meetings felt time consuming they were very valuable. This is because a

[10] Interview with Mikael Lundgren on 28.8.2013

lot of discussions and conversations regarding the technical problems and solutions occur during these meetings, which might not have happened otherwise or might have come up later, in a less organized way. It is also possible that the length of meetings correlates with how much uncertainty there is in the project at the time. It could be observed that later on in the experiment when the team moved from concept design toward detailed design, the meetings, especially the planning and daily meetings became shorter.

Sprint

The mechanical team has been working in fixed periods of time as prescribed by Schwaber and Beedle (2002), or time-boxed to a certain amount of time as described by Schwaber (2004). The Sprints are most often 2 weeks and do not exceed 4 weeks, as recommended by Deemer *et al.* (2012). The Sprints lengths have been variable if there have been special circumstances and it is more practical for the team. These Sprint lengths have been changed in the cases of:

- Field test, where it is not practical to plan the next Sprint until they are over. This was done ones, in Sprint no. 7 (see Figure 5.4).
- When synchronizing with software teams Sprint cadence. This was done two times, Sprint no. 2 and 9.
- During vacation periods, this was done during the Easter break in Sprint no. 1, and during the summer vacation period Sprint no. 8.

Deemer *et al.* (2012) state that Sprints continue one after another without a pause and Sprints end whether or not the planned work has been completed and Sprint length is never extended. All these aspects fit the mechanical team, as the changes in Sprint lengths are decided beforehand, not during Sprints, and the Sprints are not extended after they commence. There was one exception to this in Sprint no. 5. Then conditions changed due to new information and the current Sprint plan was redundant and needed to be re-planned. There is nothing noted in regards to changing Sprint lengths in the core framework literature covered for this thesis, except Deemer *et al.* (2012) note that normally the Scrum teams choose one Sprint length which they keep until they have improved and can start using shorter cycles.

Sprint Planning

It seems that the planning meetings of the mechanical and software team, are somewhat different from the way the meetings are described in the theory. Customers, users or management do not attend the meetings and the PO and Team do not determine the next Sprints functionality and goal together as described by Schwaber and Beedle (2002) nor does the Product Owner present the highest priority backlog items as prescribed by Schwaber (2004). Most often, the team defines what they are going to do that Sprint by themselves, but the PO can comment and express his opinion if he is present, but he does not present what is next on the list. When the PO is available, he most often attends the start and end of the meeting.

The meeting is not divided into two half's, in regards to the "What" and "How", described by Schwaber (2004), Schwaber and Sutherland (2011) and Deemer *et al.* (2012). This is more interlinked at the mechanical team's later planning meetings, where they define what they are going to work on, and create tasks for each story, and then when they have estimated these stories, and accumulated the points, they evaluate if they will be able to take all the stories into the Sprint. They have not defined beforehand what functionality to work on and commit to that, rather they pick out the components they need to work on next, and create tasks for each component, that then are united in one Sprint story. So the What and How is intertwined as the team would not be able to define how much of the "What" stories they are going to include, until they have gone through the "How" part, creating the tasks for each story. It can also be concluded that they are going the other way around, and are not quite sure what exactly they are going to do for each part or component of the product, until they have gone through the tasking. Then they have defined what they want to do in

the Sprint, and can then compare it to their capacity, average amount of point, and see how much they can commit to. The commitment does not happen until they have gone through the whole process, and after that, they define the goal.

Deemer *et al.* (2012) state that all roles should be present at the first part of the meeting, but the second half is optional for the Product Owner, but it should be possible to reach him or her for questions. This might in some way be the case at Marel, as the PO is normally present at the start, and then approves the plan at the end of the meeting.

Schwaber and Sutherland (2011) state that the Sprint Goal can be a milestone of the higher level product roadmap, that is satisfied by implementing technology and functionality and that it is something the Development team keeps in mind while they work. In the early Sprints, the goals were it seems, not something the team always kept in mind, but later on when they had actual deliverables to other parties, such as the workshop, they have been more focused on the Sprint goals. In the later Sprints, the goals have also been linked to milestones in the product Release plan, which was to deliver drawings to the workshop in order to have a working prototype available later on in the plan. The team has incorporated short-term milestones in their goals, in order to be able to deliver at a larger milestone later on. An example of this, is when they started with sending all the parts that need trimming and milling, and take longer to go through the production system, in order for them to be ready at the same time as other parts later on.

In regards to the length of these meetings, they took very long in the beginning, but have now been narrowed down to two hours including breaks. According to Schwaber's (2004) definition of Sprint lengths, for a two week Sprint the meeting should be 4 hours total. It is possible that when the team starts working on new projects, where there is high uncertainty and the scope possibly unclear, that these meetings will take longer, as they will then accommodate more discussions.

Work Break-down:

Breaking down work, creating backlog items is closely related to the planning meetings of the mechanical team. During Release planning, the whole product is broken down in to sub-tasks of a development project, and in the latest Release planning session the product was broken down to the different sub-systems and components.

From the beginning, the team had problems with breaking down their work, and the team members were quite frustrated with needing to do so, as they did not see that it would help them. However, improvements can be seen from the team's status by the end of the experiment when compared to the first Release planning session. In the start of the project, a whole system of connected machines was to be developed. During the first Release planning session a prototype of equipment that was to be placed in front of the main prototype built in the later Release, was planned to be constructed. The team had created one large 20 point Release story for that prototype. When pushed to divide it up further they divided the story into 3 parts: Designing a CAD model (13 points); Make the design ready for construction (5 points); and observe and assist during construction of prototype (3 points). Later, when asked in retrospective about this prototype, they noted that this machine was much smaller with less functionality than the one main machine. After some discussion on if they could have divided the design into components like had been done for the main machine, they agreed and said that it could probably have been divided up into 4 sub-systems. This shows that some improvements have been reached in regards to work breakdown.

Sprint Review

The mechanical team has followed the framework by holding Reviews after each Sprint. However, the Reviews are one of the elements of the framework the team has been the most frustrated with.

Schwaber's (2004) states that the Sprint Review is held at the end of every Sprint where the team demonstrates what was accomplished that Sprint to the Product Owner and other interested parties. It is only allowed to demonstrate completed product functionality. Firstly, the rule about only presenting complete functionality has not been followed. It has been more of a status meeting,

where the two teams present what they did during the Sprint, their findings, results or hindrances they encountered. The mechanical team has often found this uncomfortable; they did not see the point in showing an unfinished CAD model or talk about minor decisions about technical aspects the team had decided on during the Sprint. One of the reasons for them not seeing the point in the Reviews, they stated was that they felt they would not really get any valuable feedback from the people that attend these meetings.

Those that attend the Review are the two teams, the PO, in-house stakeholders, dependent parties and sometimes others interested. The general rule at Marel is that the Reviews are open to everyone in the company. What the team complained about was that there were people showing up that did not give any feedback or input, and might not be able to as they were not directly involved in the project. They also felt there were excessively many people on the email inviting list, and that just a small portion of them attends. But what they later discovered was that on the list, created by the first Scrum Master, there were all embedded teams at Marel GRB. This is a common practice in the Marel GRB embedded community to invite all other teams to Reviews. The mechanical team had also not been keen on inviting people from the production management and purchasing, but later on accepted this, and it seems to have been valuable in regards to the production management cooperation. The frustration in regards to the Review was expressed by one team member from the first days of the experiment, and throughout it, and the others seemed to agree with him.

Comparing this to the theory Deemer *et al.* (2012) state the Team, Product Owner and Scrum Master should attend the Review. They note that the Product Owner should invite other stakeholders, as appropriate. These could be customers, users, stakeholders, experts, executives or others that are interested. However, this is not the case at Marel, it was a predefined list created by the Scrum Master in cooperation with the team. Schwaber and Sutherland (2011) just mention the Scrum Team and stakeholders should attend the Review, and Schwaber and Beedle (2002) state that it is the Product Owner, management, customers and users, evaluate the product increment. The Reviews of the mechanical teams have been not that much of an evaluation, more a time for presenting the status and receiving questions and some feedback.

Deemer *et al.* (2012, p.13) summarize the definition as “Inspection and adaptation related to the product increment of functionality”, and further note that it is a common misunderstanding that this meeting is just a “Demo” presentation. It should be a conversation between the Product Owner and the Team, firstly for the Product Owner to learn how the Team and product are getting on and secondly for the Team to learn what is new with the Product Owner and the market. This is not the case for the Marel teams, the PO is more on the side-line of these meetings, possibly because he is on track with what is going on with the team, by attending daily stand-ups. But it is possible that the team lacks the insight into the Market, which would happen by going by the definition of Deemer *et al.*

Schwaber and Sutherland (2011) also state that the meetings purpose is to inspect the product increment and adapt the Product Backlog if need be. The adaptation of the Product Backlog does not occur at this meeting, and it is not even presented there. Schwaber and Sutherland (2011) further note that this is an informal meeting, and by presenting a product increment the intention is to elicit feedback and encourage cooperation. This is in some sense achieved, but the team does not feel the feedback is that valuable. The PO noted in his interview, that it might be pre-decided by the team that they will not receive valuable feedback. However, it is possible that more feedback could be elicited by purposefully inviting the most knowledgeable individuals about the Sprints subject each time.

According to Schwaber’s (2004) definition, the Review should be two hours for a two week Sprint. For the main part of the experiment, or from Sprint no 1 to 9, the meeting lasted 1 hour, 30 minutes per team. This changed after the Release Plan was created but currently they just last 30 minutes, and each team gets 15 minutes to present and answer questions. The change was done in order to

manage the team's frustration about Scrum meetings being time-consuming and that they lack valuable feedback, and therefore important to keep the meetings concise.

In the later Sprints of the experiment, the team was able to show CAD models of the machine and its different functional mechanics. However, at the time, these were finished models and some parts had even been sent to the workshop for production. So there would not have been any room for adaptation. The team could be asked to show unfinished CAD, but the people at the Review would need to have the insight and be knowledgeable enough to be able to give valuable feedback on the unfinished work.

In conclusion:

- The purpose and structure of the Review could be reconsidered, in regards to who should attend and be invited, how the subject of the Sprint should be presented, and whether there should be a more in-depth conversation between the Product Owner and Team during these meetings, as described by Deemer *et al.* (2012).
- The invitation list could be reviewed, as there might be waste in inviting people, which do not give feedback. In contrast, these individuals could be learning something that can benefit them in their work. But all in all this could be reconsidered, and the possibility to send special personal invites to those the team would like to have attend each Sprint, as they think that person would bring valuable feedback.

Sprint Retrospective

According Schwaber's (2004) definition, the Retrospective should be a time-boxed meeting of two hours. The mechanical team's Retrospective meetings first lasted an hour, and their duration was shortened after Sprint no. 10, like the Review meetings, to 30 minutes. The Scrum Master facilitates the Retrospectives, which is in line with Schwaber's (2004) definition.

In regards to the purpose of the Retrospective meeting, Schwaber (2004) states that during the meeting the team discuss the Sprint that has just finished and determines what could be changed which could make the upcoming Sprint more productive or enjoyable. Deemer *et al.* (2012, p.14) note the Retrospective is to inspect and adapt the process and environment, as the Review inspects and adapts the product. They explain that it gives the team an opportunity to have discussions on what is working and what is not, and all agree on what changes they want to try. They further note that there are many techniques available to conduct a Retrospective and that various techniques should be introduced over time. They also press that it is important not only to focus on problems, but also on positive things and strengths. Schwaber and Sutherland (2011) add to this that the purpose includes identifying what went well and possible improvements and to create an improvement plan.

Retrospectives have been held from the start of the mechanical teams Scrum experiment, and different methods were used. The discussion focused both on what went well and on what could have been better. Some issues have been discussed during these meetings, and improvements suggestions have come up. Some have been executed others have not, but there has never been an improvement plan. The team noted that in the start, these meetings helped them understand the framework better, but later they felt the benefits had weakened, and did not see that the meetings were that valuable. This might be related to the new Scrum Master's lack of training in facilitating Retrospectives, but in the later Sprints, there have come up some valuable improvement suggestions that they have followed, such as rules for reviewing drawings before they are sent to the workshop. The facilitation of these meetings could be improved further; in order to harness the benefits it can provide the team.

Another Scrum Master at Marel GRB suggested that it could possibly be beneficial to have a big Retrospective with all those involved in the project. This Scrum Master also added that it might help the team see the purpose in the meeting if it yielded some results, like small improvement projects or tasks, which are taken into the Sprint plan.

In conclusion:

- Although the team has not fully seen the value in the meeting, they have supported discussions, and yielded some improvements of the process and the team's work.
- The facilitation of the meetings could be improved in order to get better results from the meeting, in form of defined improvement projects or tasks, possibly with an action plan. A special improvements board could even be used.

Daily Stand-Up Meeting

According to Schwaber and Beedle (2002) the Daily Scrum meeting is where the team communicates, and the meeting is described by Schwaber (2004) as a short meeting held every day for the team synchronize their progress and work, as well as report any hindrances to the Scrum Master to take care of. Schwaber and Sutherland (2011) say that it should be held at the same place at the same time every day. In accordance with this, the mechanical team meeting is held in front of their Scrum board every day at 9:45. They have shown initiative in starting the meetings themselves, but they stated that they sometimes forgot, and they would have forgotten more often if the Scrum Master were not there. It has been observed, and reported by the team members in interviews, that the meeting is very useful for them to synchronize their work. In regards to hindrances, the Scrum Master has not always taken care of them, but maybe made sure that it is decided that one of the team members contacts someone, purchasing for example, in regards to hindrances. The Product Owner has sometimes done this as well. It is possible that in the future, the SM can take on this role although it is important that the team does not loose connection to their dependency parties.

Schwaber and Sutherland (2011) state that the Daily meeting should be a time-box of 15 minutes and Deemer *et al.* (2012) say it can be 15 minutes or less. The mechanical teams Daily Stand-up meeting most often does not exceed the 15 minutes, and the team members are somewhat good at telling each other if the meeting is loosing track.

The team does not distinctly answer the three questions defined by Schwaber and Sutherland (2011, p.10): What they did yesterday; what they are going to do today; and if there are any obstacles in their way. Although they do not answer these precisely one after another, the content of what each one of them reports most often includes these aspects. If not, the Scrum Master asks them e.g. whether there are any hindrances.

Deemer *et al.* (2012) states that these three elements should be reported to the other team members and that it is not a meeting to report to a manager. They most often report to each other, although they might sometimes start directing their report to the Scrum Master and Product Owner during the meetings.

In conclusion, these meetings are beneficial to the team, and the framework is almost followed by the book.

7.1.5 Framework Adaptations

There are differences between software development and mechanical development, and the team, in both their first and second interview, stated that Scrum and Agile are tailored to Software development. Therefore, the team was convinced the framework would need to be adapted to mechanical development.

Jackson (2012) presents quotes by some of the creators of the Agile Manifesto that contradict this in some way. Jim Highsmith, one co-creator, stated that Agile can be used in any project that deals with risk and uncertainty, as presented in chapter 4.3:

> *"In any project that faces uncertainty, complexity, volatility and risk, there is a place for agile practices and principles." (Jackson, 2012, p.61).*

This fits very well to the mechanical team's project as it deals with high uncertainty, complexity and risk. Ron Jefferies stated that of course there are some specialized details that differ between domains, but that the Agile principles are widely applicable:

> *"The key ideas apply everywhere. […] Agile ideas are based on how people and organizations work best. There are specialized details we need to know regarding software, just as there are in any other domain. The principles, however, are broadly applicable." (Jackson, 2012, p.61)*

Some of the key ideas he mentions that fit well with mechanical development, are (Jackson, 2012, p.61):

- *"Building work incrementally and checking results as you go".* This can be very valuable, as in new development, such as in the Marel case, there is a lot of uncertainty and by checking the results or findings continually, the course of the project can be adjusted to these findings.
- *"Preparing for and influencing the future but not predicting it".* In a way, this is being pro-active, being aware that things can happen but that it is not possible to predict them precisely, which aligns with how the mechanical team describes their work. Something will happen, they just do not know when.
- *"Making tasks concrete and quickly finishing them".* The mechanical team has been able to do this in some sense, although the tasks could possibly be more concrete. However, this has helped them reach smaller milestones, rather than continuously working on one big chunk of work.
- *"Giving people work to do and the knowledge to do it, not pushing them around like pawns on a chessboard".* This also fits with the mechanical team, in the sense of project management. It is preferable to them to receive what project or task to work on from their PO and then they share knowledge internally, based on their have different level of experience, and delegate the work internally themselves, rather than a Project Manager or the PO doing that. This also increases communication and co-operation within the team.

Finally, Andrew Hunt further noted that Agile practices have little to do with software, and presents a definition of Agile development, which aligns with that:

> *"At its heart, an agile approach has little to do with software; it's all about recognizing and applying feedback. The definition of "agile" that Venkat Subramanium and I proposed in 'Practices of an Agile Developer' states, "Agile development uses feedback to make constant adjustments in a highly collaborative environment". Notice there is nothing in there about software." (Jackson, 2012, p.62)*

In conclusion, Agile practices should not be defined as being tailored to software development, and should be applicable to other domains, such as mechanical development.

Adaptations in Software Development

In regards to adapting the framework to the work and environment of the mechanical team, it is interesting to know if software teams do this as well. No detailed study was done of literature in regards to this, but a good indication that tailoring is important for software development was found. In regards to successful Agile deployment, Pikkarainen et al. (2012) state that their analysis *"indicated the significance of defining a tailored process model and giving developers the freedom to improve their own agile development process continuously during agile deployment."* (Pikkarainen et al., 2012, p.675)

It can therefore be concluded that software teams also need to tailor and adapt Agile methods and processes to their needs.

Difference between Software and Hardware

In order to understand how Agile practices and the Scrum framework might need to be adapted differently to mechanical development, the difference between software development and mechanical development was studied. It should be noted that there is also a difference between application and embedded software development, as the embedded software is dependent on hardware to run and test on. Testing can be done virtually and updating can be performed via the Internet, which is not possible for mechanical development. Ultimately the software needs to be tested on the hardware, which is the main difference between application and embedded software, along with that the embedded software might not be as user-oriented as application software. What the mechanical team referred to as software, can be assumed to be mainly user-oriented application software development.

First of all the main difference between software and mechanical or hardware development is that the latter deals with physical objects. There are therefore two main aspects that affect mechanical development more than software development, additional **waiting times** and that **testing** can be performed less frequently. The waiting time the team experiences is firstly waiting for parts to be manufactured in-house; secondly waiting for purchased components to arrive; and thirdly for the PD workshop to construct the prototype from the parts and components. Testing is interlinked with these waiting times, as it cannot be performed until a physical prototype has been constructed.

This difference is what might be the main reason for the mechanical team not being able to deliver a **working product increment** or potentially shippable product (Deemer *et al.*, 2010) at the end of a Sprint. It is unlikely that a potentially shippable product can be created each Sprint in mechanical development, but in some cases, prototypes might be produced that is more of a working product increment. This might be possible in mechanical or hardware development where there is access to create quick mock-ups, rapid prototyping and testing. However, the current work environment at Marel GRB, and possibly the nature of the project, do not allow this to happen at the end of every Sprint. Instead, it could be stated that a working prototype could be produced by the end of each Release cycle, which seems more plausible. The result of a Sprint in mechanical development, can be reaching goals that move the team closer to be able to produce the working prototype by the end of the Release. This could be producing CAD models, or gaining further knowledge in regards to uncertainties.

In regards to designing physical objects, the mechanical team expressed that the work needed to be performed in **parallel**, that is, different components need to be developed in parallel. But as observed, and stated in the latter interview with the team, their work is iterative, where they design the whole product to a certain extent and then stop and go another round and design all components or sub-systems in more detail. Therefore, in a sense they are working iteratively and should be able to have at least a rough CAD model of a whole product ready by the end of iteration.

Another characteristic of the Marel GRB case, and its project, are all the **dependencies** the team has, see Figure 5.10. This is assumed not to be as common in software development projects. This is not in line with what is stated by Schwaber and Sutherland (2011) that all competences needed to finish the work should be included in the team. However, combining all these competences in one big team would not be practical, as many of these parties serve other teams as well, such as the PD workshop and the purchasing department. The software and research teams focus on different elements than the mechanical team, their terminology is different and it does not seem practical to combine these under one big team.

In conclusion, there are two main deviations from the prescribed framework. Firstly, that the mechanical team does not produce a working product increment every Sprint, therefore testing is not done each Sprint. Secondly, that the team is not cross-functional as not all competences needed to design and manufacture the product are included in the team.

7.1.6 Success of the Implementation

To answer the question whether the Scrum implementation for the mechanical team was successful, an analysis needs to be done of whether the team was able to confirm with and benefit from the Scrum framework. Schwaber and Sutherland (2011) state that:

> *"Scrum's roles, artifacts, events, and rules are immutable and although implementing only parts of Scrum is possible, the result is not Scrum. Scrum exists only in its entirety and functions well as a container for other techniques, methodologies and practices." (Schwaber & Sutherland, 2011, p.15)*

The team includes all the roles as prescribed, as well as the events. The teams at SAAB and Andritz for example skipped the Review meeting, and the SAAB team did not use Retrospectives. The artefacts that are missing are the definition of done and possibly an active Product Backlog, but the team uses a Burn-Down chart for the Release Plan. They have a Sprint Backlog, and their Release Plan in a way serves as a Product Backlog. However, the Product Backlog might be one of the artefacts lacking. All in all most of the framework is included, so it is assumed that the framework the mechanical team uses can still be called Scrum.

In line with Schwaber and Sutherland's (2011) statement, Schiel (2012) notes that despite Scrum providing improved performance, quality and productivity, it only does so to a certain extent and Scrum is not enough on its own. It must be supplemented with strong engineering practices, infrastructure, behavioural practices and organizational self-discipline.

In the case of the mechanical team, this is important, as Scrum will only help frame and structure the work, but there will always have to be strong engineering practice, self-discipline etc. There might be an opportunity to improve the engineering practices, as for example the field test performed during the experiment, were not very formal or structured.

Schiel (2012) notes that a development effort's bottom line is customer satisfaction, and if the customer is satisfied with the product, then the Agile development efforts are working. In the case of the mechanical team it is not possible to know right away if this is the case, as they do not collaborating closely with a customer, and therefore not possible to evaluate the customer satisfaction until the next field test or when the product has been launched.

7.1.1 Success Factors in Deployment and Sustaining of Agile Methods

As presented earlier Pikkarainen *et al* (2012) state that it is important, in deployment of Agile methods, to define a tailored process model and that teams should get the freedom to improve Agile methods to their needs. They also conclude that their analysis revealed *"the importance of management providing the necessary goals and support for agile development"* (Pikkarainen et al., 2012, p.675). They further recommend that management should be committed and provide continuous support; make sure that team, management and all stakeholders should be knowledgeable about Agile methods; the tailoring of the Agile based process model should also be in the organizational level.

The framework has been tailored to the team in some way, but it is possible that the team could be encouraged to suggest further changes to fit their needs. This might surface in the future, as they become more comfortable with the notion of continuous improvement, and confident in the use of the framework.

Senapathi and Srinivasan (2013, p.119) identified nine critical factors that impact sustaining usage of Agile Methods and concluded that their findings indicate that "the right balance and combination of various factors with an emphasis on continuous improvement will be crucial for achieving true agile sustainability in organizations." Following is the comparison of the nine factors to the mechanical team:

Firstly, the factor of *technical competence and expertise* fit the team well, as these are very competent mechanical engineers with high experience level within the team. The *team composition*

factor can be assumed good for the team as they all work well together and there are no obvious social problems present. In regards to *motivation*, they are relatively motivated, but that might be a factor to improve. The team received good *training* at the start, but it is possible that they need more purposeful continues education on the framework and Agile principles. In regards to *Methodology Champion*, the Agile Center serves as one for the organization, and the Scrum Master can be seen as the team's champion, but it is not possible to point out a methodology champion from within the team. *Management support* has been good on the department level, although the management could be better educated in the framework. According to the interview with the previous Scrum Master, organizational management support is lacking. The factors that the team lacks the most are *Agile Mind-set*, *Agile Engineering Practices* and *Attitude*. It cannot be said that they have an Agile Mind-set and Agile Engineering Practices. Their attitude is also often negative in regards to the framework, and possibly not solutions oriented, but their attitude has improved throughout the experiment.

Finally, the emphasis on continuous improvement Senapathi and Srinivasan (2013) mention, is still also lacking, and is one of the things that need to be incorporated in the use of the framework.

Was the Implementation Successful?

In regards to the Scrum implementations success the team decided to continue using the framework, which must indicate that it was successful, when it comes to team buy-in. What might have helped them to take that decision is that the goals of the Release plan were reached. The team might also have started to accept or feel better about the framework after they could change it a bit, when the demand for work breakdown was lowered.

In order to know if the implementation was successful, it should be compared to what the situation was like before, and what has improved. Firstly, in the later Sprints of the experiment the team started reaching goals, and finally delivered what was planned on time in accordance to the very ambitious Release Plan. Compared to the delays and lack of milestones the Product Owner had described about the old way of working, this can be seen as a great improvement and success. Another improvement has been the co-ordination and communication with the Software team, which one of the Software engineers stated had improved a lot, and gave examples of previous miscommunication. This can be very important to the progress of a project, and can be assumed another improvement and success. It should also be noted that communication and co-operation has also improved within the team, and they have stopped working as individuals in a group and now work as one team.

What might have affected the success is that this is a small team, only 8 people including the software team, and this makes communication much easier in contrast to larger teams working on one product. It might also have helped that there was trust and camaraderie was built between the mechanical team and their current Scrum Master, which might have helped them overcome their frustrations and resistance to change, and helped them give the framework a change.

It is concluded that the implementation was successful, and this indicates that the team will be able to sustain usage of the framework, if some improvements are carried out.

Recommendations in Retrospect

In retrospect, some elements might be important in an implementation like this. Possibly one should start with implementing the basic framework, Scrum roles, events and main artefacts, and then later on add more best practices when the team has become familiar with the basics. It is still important not to skip the basics of the framework, but examples of what might be improved later on would be in what detail stories are written or acceptance criteria, definition of done, etc. As long as the team is planning, breaking their work down into manageable parts, and following the Sprint cadence, to begin with that is a good start. Then when the teams have developed even further they can start going into further detail on how to define backlog items or stories, possibly to define testing as well.

7.1.2 Next Steps

The next step for the mechanical team is to start using continuous improvements more, which is highly dependent on the facilitation of the Scrum Master. They should also try to define better what the acceptance criteria or definition of done in some way for their stories. This should only be done in order to help them perform the work, but not as a documentation practice.

The roles should be clarified in more detail, defining the responsibilities for each one. This can be done in collectively between the team, Scrum Master and Product Owner, and have all parties accepted the definition, ensuring everyone has the same understanding.

After the experiment, the two teams will start having the same Scrum Master, which will probably increase the collaboration even further. The Scrum Master could take on more of the hindrances the teams face, making sure that no miscommunication happens between the teams or third party dependencies.

7.2 Comparison to other Cases

In this chapter a short comparison analysis of the SAAB EDS and Andritz Hydro cases and the Marel case will be presented. Two other cases of the use of Agile or Scrum-like methods in hardware development at other organizations are presented in the thesis in chapter 4.3, along with a literature review by Carlson and Turner (2013). These cases, the Johns Hopkins case and Wikispeed case, prove in some way that it is possible to use Agile methods in hardware development. They present good recommendations that organizations intending on implementing Agile practices could benefit from. However, as the organizational structure of these cases, a university lab and a volunteer team of more than 100 people, are not closely linked with the main case of this study they will not be analysed or compared with in this thesis. The intention was to present them, firstly to show that using Agile in hardware development has been done before and secondly to provide future reference to others, that might benefit from their recommendations.

In addition, as the general lessons presented by Carlson and Turner (2013) based on their literature review proved to have meaningful input in regards to the Marel case, a short analysis will be presented in the following section, prior to the comparison analysis of the SAAB EDS and Andritz Hydro cases.

7.2.1 Review of Agile Case Studies for Applicability to Aircraft Systems Integration

Carlson and Turner (2013) published a review of selected non-software Agile case studies, with the intentions of finding useful methods applicable and essential lessons useful to aircraft systems integration. The general lessons identified by that Carlson and Turner (2013), which are relevant to this study, are compared to the Marel case in this section. These lessons concern teams, Agile coach, co-location, Sprint length, process training and prioritization of work.

Teams: Carlson and Turner (2013) state that teams should be diverse with knowledgeable members from all areas. This is needed for shared vision and accountability. They further state that Agile works best with small teams; therefore, it might be best to make changes to the organizational structure, by flattening it. The mechanical team at Marel GRB is composed of knowledgeable members, and in a sense, they have diverse experiences within the mechanical sector, despite not being a cross-functional team. The team is small, only four members and the organizational structure of the department is relatively flat, with few hierarchy levels.

Agile Coach: Carlson and Turner (2013) state that it is important to educate teams on the process. Stakeholders also need to be educated about it. The Marel GRB team received relatively extensive education, but the stakeholders and management might need to have been educated more. It was also very important for the implementation that the team had an Agile Coach from the start and throughout the experiment.

Co-location: Carlson and Turner (2013) state that co-location enables teams to facilitate rapid design iterations, resulting in flexibility and innovation. The mechanical team is co-located and there is great amount of communication between all team members. This allows them to consult each other and get rapid feedback. These would be design iterations, although they might not be performing physical prototype tests frequently.

Sprints length: Carlson and Turner (2013) recommended having shorter Sprints in order to receive more feedback from customers. The mechanical team has short Sprints, and this allows them to adjust their plans to new information, but they do not interact frequently with potential customers.

Process training: Carlson and Turner (2013, p.473) state that process training *"sounds like a worthwhile investment to change the culture, create appropriate expectations, and quickly implement effective methods".* In regards to changing the culture at Marel GRB, this has mainly happened locally with the teams that apply Agile practices, but as stated by the Agile Center Manager, understanding on an organizational level is lacking.

Incremental testing: Carlson and Turner (2013, p.474) state that incremental testing should be practiced to *"keep the teams focused on meaningful work".* This is a very valid point, something that the Marel team has not done, but could be a possible improvement of the way of working in future projects.

Prioritization of work: Carlson and Turner (2013, p.474) state that prioritization of work is important in order *"to reduce wasted effort and rework".* They further note that this also focuses the work on the areas where most value can be created. Prioritization of the mechanical team's work both helped them in planning and getting an overview of the work needed and deadlines ahead. They also improved in following the priority throughout the experiment. In regards to focus on maximising value creation, that was not the main focus of the mechanical team's prioritization, rather risk and the time a certain work item would take. In a way that is value for the project, to work on the more risky aspects and start on those items that take the longest, in order to meet deadlines. In that sense, the prioritization focused on value in some way.

7.2.2 SAAB EDS

The main characteristics of the SAAB EDS case will be compared to those of the Marel case in this section. These are the following:

The team: The SAAB EDS team was created around a project, with resources from different functions, and only 3 out of 12 are fully dedicated. It was a cross-functional team but they would all be relocated after this project. The future vision of the Line Manager was to have dedicated product teams. Fixed product teams, which take on projects has proven successful at Marel, this could be seen in the case of the mechanical team. This should still be possible despite the SAAB EDS teams being cross-functional.

Team management: There were three functional Product Owners, one of which was also the Scrum Master. It seemed that the team reported more to the Scrum Master than to the other team members. This opposes one of the basic rules Marel has about Agile teams, as the Scrum Master and Product Owner are two conflicting roles and Deemer *et al.* (2012) also state that the same person cannot serve as both Scrum Master and Product Owner as the roles' focus is different. They state that it will lead to confusion, conflict, and the person might end up micromanaging the team, which is the opposite of self-organized teams.

Education and training: The team and its management only received a one-hour introduction to Scrum. They recognize that this is not Scrum by the book and call it a "Scrum-ish" method. It was noted that other hardware teams were getting education that is more extensive. Compared to the training the mechanical team got at Marel and that the team still needed guidance throughout the experiment, the SAAB EDS team lacked training and coaching. What the SAAB team also seemed to lack was a consultant or someone knowledgeable to guide them through the implementation, which was performed by the mechanical team's first Scrum Master at Marel. In addition to the Product

Owners and Scrum Master in the SAAB EDS case, there was also a Project Manager for the project, the one interviewed. Deemer et al. (2012, p.5) note having a project manager in addition to practicing Scrum *"indicates a fundamental misunderstanding of Scrum, and typically results in conflicting responsibilities, unclear authority, and sub-optimal results."*

Deviations from the framework: The SAAB EDS team used the Sprint and planning aspect of the framework, visual management and Sprint burn-down, and kept track of non-planned work, but did not have Reviews or Retrospectives. They might be missing some discussions and feedback by not presenting what they have accomplished at the end of each Sprint. The Retrospectives could also provide them a platform to improve their work, and this seems to be a common practice within the organization, and should therefore not need to be a big step to add.

Planning: In regards to planning, they seem to have very active Product Backlog management by the PM and POs, through their prioritization meetings which is very good. The planning is also performed in co-operation with those that actually do the work, which provides good feedback and results in plans that are more realistic, this is very good.

Work Breakdown: It seemed that the engineers had had problems with work breakdown, mainly when the Lean Visual Management implementation. It was also noted that they had some problems with breaking work further down in the new planning method, e.g. breaking a 10-day task down into 2-3 day tasks. This is in line with one of the main challenges and frustrations of the Marel mechanical team. They also noted that the main changes from what they were used to with visual management were the Sprints and the burn-down chart. Marel could learn from SAAB EDS's use of visual management walls and continuous improvement boards, which is more extensive than just Scrum boards.

Benefits: The main benefits identified by the Project Manager and Line Manager, where focus, dedication and that everyone are engaged in the planning process. They stated that they did not think Scrum was a software method, rather a planning method. They recommended to those starting out to find a good Scrum Master and define the goals of work items.

This is the way of working: What was also very interesting was that in regards to the Lean transformation at SAAB EDS, the feeling was that "This is the way of working" as opposed to "You have to work like this", which is much more inspiring and provides a better work environment.

7.2.3 Andritz Hydro AB

In this section the case of the organizational change and application of Agile and Scrum practices at the Service and Rehab division of Andritz Hydro AB, will be discussed and compared to the Marel case.

What is similar between the Andritz Hydro AB case and the Marel case is that these are all mechanical engineers, but what is different is that Andritz does contracted work tailored to the customer, but the Marel team is in New Product Development. The Andritz Hydro case is also of a much larger scale, as the whole organization was transformed and reorganized.

The New Organization Structure is very interesting and that the project teams now just purchase results from the teams instead of resources from the functional line. This seems to result in more focused and efficient work. People are not distributed over many different project teams and working on many different functional aspects at the same time, rather the teams stay the same and focus on mastering one functional aspect, by repeating the same tasks more often.

Non Cross-functional teams: The teams are not cross-functional, in contrast to what is described in the literature (Schwaber & Sutherland, 2011), but the method still works well for the organization. A factor in this might be that the interfaces between components are not so complex and are manageable for the project teams. It also provides better knowledge distribution and employee development as mentioned above. This case supports the indication that non cross-functional teams can work in Scrum.

High-Level Planning: Their high-level planning and the continuous improvement is also interesting as it involves the whole organization, and makes everyone involved and informed, top to bottom.

Method Champion: The Product Manager is the Method Champion, which seems to be one of the main success factors of this new organization. It supports that top-down decisions and top-management support is vital in change initiatives like this one. Marel could take some example in this, the method champions for the whole of Marel GRB and the one within the department (IC) the mechanical team is a part of, not at top-level.

Top-down decision: The consultant stated that the top-down decision was a success factor. There were worries at Marel that the method had been forced on the mechanical team, but they would not have started using the methods by themselves. Therefor this case supports the top-down decision made.

Deviations from the framework: One of the things they have not done in regards to the Scrum framework is Sprint Reviews. This is something the Product Manager was interested in incorporating in the future. They did not have Scrum Masters, but team leaders were appointed in early summer 2013 and serve as team method champions, possibly correlating to the role of a Scrum Master. As mentioned above, another deviation is that these are not cross-functional teams, but it works out well for them.

Prioritization method: Another thing the Product Manager noted was that he wanted people to see this as a prioritization method, rather than a planning method. This was due to resources needed at the operational level should be accounted for before the planning of Sprints takes place. There, prioritization of work should be the focus.

Pressure on Management: It was noted that in this new way of working, management could not continue to pressure teams with more work and the new way of prioritizing and estimating team capacity, pressures management to make decisions. This is what the engineers liked, that there is a management decision made every month and they know exactly what to do. This is most likely the reason for stress being much better managed after the change, as the engineers did not feel the same pressure and work overload as before.

Self-organized teams: It is proposed by the literature that Scrum teams should be self-organized (Schwaber & Sutherland, 2011). It seems that teams were relatively self-organized after they receive the prioritized list of work-packages from management. They did their own internal planning or work distribution and they are also encouraged to develop their own visual management boards.

Continuous improvement: The Product Manager stressed that the continuous improvement aspect is the most important part when implementing a new process like this, and it should be made sure that this works. This is something that Marel could improve in and could take up the practices presented by Andritz Hydro. The Product Manager further stated that it is important that it is simple and rudimentary and that it is developed organically through the self-improvement process. Start of simple and then developed by the teams themselves. It could be questioned if the Marel implementation could have been simpler, and if the team could have developed it more themselves and that way the team would have gained more buy-in sooner. What might be the difference is that the characteristics of the Andritz Hydro pilot team, which were up for trying and developing the new way of working, compared to the Marel mechanical team that was positive but sceptic and needed guidance.

Unplanned work is difficult to handle: Although the new system made unplanned hours visible, it did not provide functionality to handle these. This is one of the existing problems and cause for irritation. The unplanned work seems to be external work subjected on the teams. What happened at Marel was that the scope was changed. Problems with unplanned work for the mechanical team was more tasks or aspects the team forgot to add during the planning meetings or unexpected issues that occur during the Sprint and need to be handled. How to manage this can also be confusing and no apparent solution is used for the mechanical team other than adding pink post-its

to the wall for the extra work performed. It could be possible to closely manage these as is done at Andritz Hydro, to see if there is a trend, as well as to try to improve the planning process.

Mechanical Engineers: It was noted by the Agile expert, that assisted in Anditz Hydro's change, that in another company he had consulted in, what had settled the mechanical engineers frustrations was that they were allowed shorter cycles than the software teams, and they could change the iteration length depending on what phase they were in. The latter is similar to the Marel case, where they vary the iteration length when needed.

Implementation results: Improvements in lead-time and engineering hours spent on a project, indicated that the new organization was a successful change effort and has provided high value to the organization, especially as there are very high deadline penalties in that industry. It was also stated that they had not had as much budget overshoot as before. Unfortunately, later it was noted that the improvements gained with the new method had been lost, mainly due to the organization being under resourced.

Lack of resources affects the success of the new organization and work method: as the Product Manager noted later, the improvements the new organization had gained earlier had been lost to certain extent. He concluded that this was likely due to increased customer requirements but mainly because the organization was understaffed and overloaded. He made a good point that no system will work if there are insufficient resources. This was not a problem in the case of the mechanical team at Marel, but can be seen as a potential risk on an organizational level.

7.3 Answers to Research Questions

The research questions posed for the thesis will be answered in this section based on the case study results and the previous analysis and discussion. The primary research question of the study was the following:

RQ: Can a mechanical product development team use Scrum?

Three secondary questions were posed to supplement the main research question. These were the following:

RQ_a: Does the Scrum framework need adaptation for mechanical teams and then do they need more complicated adaptation than software teams?

RQ_b: What are there critical success factors when deploying Scrum in mechanical teams?

RQ_c: What effects does it have that two co-operating teams, an embedded software team and a mechanical team, are both using Scrum?

These will be answered in the following sections.

7.3.1 Can a mechanical product development team use Scrum?

RQ: Can a mechanical product development team use Scrum?

Based on this study the straightforward answer is: **yes, it is possible**. Some adaptations might need to be made to the rules, and the definitions of the responsibilities of the roles depending on the developments environment, but the basic framework, its roles, events and main artefacts should be applicable to mechanical development and to other non-it development.

In regards to whether Scrum is a Software development practice or not, there is nothing in the literature that explicitly indicates that it is. The element of the framework that seems to be easier in software development to follow is to be able to produce a working product increment each Sprint.

Whether Scrum is better than other frameworks or systems for mechanical development, might be difficult to say, however it was at least better for the mechanical team at Marel GRB, which had no defined work system prior to Scrum. The study indicates that Scrum can be very helpful in cases

where there is high uncertainty, risks, and when things are constantly changing. Therefore, it might be especially beneficial in new product development, without sating that it will not be useful in projects that are more straightforward. It should be noted that Scrum is most likely not applicable in environments where it is not possible to plan work at all[11], e.g. where teams take care of maintenance requests, then Kanban is better suited.

7.3.2 Adaptations to the framework

RQ_a: Does the Scrum framework need adaptation for mechanical teams and then do they need more complicated adaptation than software teams?

As noted previously, Pikkarainen et al. (2012) state that it is important to tailor Agile process models to the needs of each organization, and that the development team should have the freedom to continuous improve their own Agile development process throughout the Agile deployment. This supports that software teams also need to adapt Agile frameworks to their situation. It also indicates that letting teams improve their process supports team buy-in.

What characterizes mechanical development is that the development deals with physical objects that cannot be changed. This needs to be taken into account and the work process respected. Therefore, some rules of the framework might need to be broken but otherwise the basic elements of the framework, its roles and events, should be applicable to mechanical development.

The adaptions that need to be made can vary between cases, based on organizational structure, nature of the development etc., but the bottom line answer to this research question is yes, there are some adaptations needed or deviations that will occur. However, it should be noted that based on the Marel case, none of the frameworks main elements were completely missing.

The adaptations made at Marel were the following:

- The development team is not cross-functional in the sense that it includes all skills needed to design and build the final product.
- The team can vary Sprint lengths, in order to accommodate special events. This is only done if practical and properly justified.
- There is not a Product Backlog that is filled and organized by the Product Owner. The reams Release plan serves as a Product backlog for the team.
- The Development Team creates a Release plan with guidance from the Product Owner, which then prioritizes the plan and approves it.
- The Reviews are not a dialog between the Product Owner and Development Team, rather a presentation of status to other stakeholders or interested parties, with the intention of receiving feedback.

7.3.3 Possible Success factors

RQ_b: What are there critical success factors when deploying Scrum in mechanical teams?

When intending to deploy the Scrum framework as prescribed by the literature and guides provided on the framework, there could be many factors that affect the deployments success. It is not possible to state which these factors are in general, for Scrum deployment in mechanical teams, based on this qualitative study as it does not provide any concrete quantitative data. However, there are some indications presented in the case studies of this thesis. The following factors indicate to affect successful deployment of Scrum by the teams in those cases:

[11] Interview with Mikael Lundgren on 28.8.2013

- Experienced Agile Coach that implements the framework, and that there is a Scrum Master or a Method Champion that supports the team and keeps them on track, whether that is the same person that is in charge of the first days of implementation or not.
- Top-management support and understanding is also important, as it is difficult to keep the framework going without it.
- Related to the previous point, it seems it is important for the initiation of starting to use Scrum is a top-down decision.
- It is important to get team buy-in. If the team is frustrated with or does not see that the framework helps them or adds value to their work, then the framework will most likely not be sustained. Therefore the team should be involved in the development of the framework or work system, and they should be encouraged to adapt and improve the framework to their needs.
- It is important that the Product Owner is actively involved and provides vision and direction.

When looking to Marel GRB in particular, the success factors and enablers identified were:

- There was lack of direction and there was a need for change.
- In-house knowledge and experience of Agile methods was very beneficial to the start-up of Scrum for the mechanical team.
- The team is co-located and sits together. They are also located very close to the software and research team, they co-operate with.
- The PO wanted this; he is the Development Manager for this Industry Center. The Scrum transition has top-management support, although some understanding of the framework might be lacking.
- The team made a connection with the Scrum Master, on a friendly note and saw that the SM was ready to help them adapt the framework to their needs, while at the same time providing discipline. This balance seems important.

In this context, it should be right to mention the main challenges and hindrances faced during the Scrum transition of the mechanical team at Marel GRB were:

- One of the bigger challenges for the team was work breakdown.
- Neither Andritz Hydro organization nor the SAAB EDS team held Reviews. The Marel team showed that it is possible, but still needs further improvement.
- The team members were used to work as individuals and not as a team.
- The project was large and complicated.
- The Product Owner was very busy and has not always been available.
- There are many external dependencies, more than in software development, and the team lacked overview of these.

7.3.4 Effects of co-operating teams using Scrum

RQ_c: What effects does it have that two co-operating teams, an embedded software team and a mechanical team, are both using Scrum?

The Marel case study revealed that when the embedded software team and the mechanical team were both using Scrum, it had a positive effect on their co-ordination and co-operation, with increased communicaition. Shared planning meetings and joint Sprint Reviews supported this. Sharing these events also supports the feeling of shared ownership of the final product. The need for these improvements could be clearly seen in regards to testing of physical prototypes. The two teams deal with different problems and challenges although the final product or prototype is their shared goal. During testing, the two teams were interested in different aspects, and each other plans,

needs and objectives in testing were not apparent between the teams. However, as they depend on each other's work for their part to function, the co-ordination is vital. The Scrum framework provided a platform for this co-ordination, and improved their communication and cooperation.

8 Conclusions and Recommendations

This study's purpose was to test the hypothesis that it is possible to use Scrum in mechanical product development teams. A case study was conducted at Marel offices in Iceland, where a mechanical product development team, developing a new product, started using Scrum. It was set out to be an experiment, and that the team would get to decide whether or not to continue to use the framework. The experiment lasted for 7 months and by the end of it the team decided to continue using the framework. This was not an obvious choice as the team had shown some frustration with the framework throughout the experiment. The deployment of Scrum for the particular case can be seen as a success, as the team has followed the framework in most sense, despite not fulfilling some criteria and breaking some of its rules. The Scrum Team consists of the three prescribed roles, holds all prescribed events, and has most of the frameworks artefacts.

Two other cases were studied, SAAB EDS and Andritz Hydro in Sweden. The first one describes a cross-functional hardware team, using a "Scrum-ish" method for planning and co-ordination. The latter case describes how a complete organisational structure was transformed, using some ideas and concepts from Agile and Scrum in mechanical development.

The main conclusion of the study is that Scrum can work for mechanical teams. It is not possible to say whether or not it is better than other ways of working for mechanical teams, however it has proven helpful and showed progress in the case studies presented. The study also shows that Scrum can work for non cross-functional teams, despite cross-functional teams being recommended in the literature.

This study shows that using the Scrum framework for mechanical development teams can provide the following benefits:

- Increased co-operation and co-ordination.
- Increased communication within team and with external dependent parties.
- Better overview of the project or development in questions, to all involved; from junior engineers to Product Owners and top management.
- Prioritizing work, results in a clearer focus and less time spent on redundant work.
- The framework fosters distribution of technical knowledge within the team and distribution of information on the project to everyone involved, especially critical dependency parties, e.g. through Sprint Reviews.
- More frequent feedback, both in regards to technical solutions as well as the project plan, as the progress is measured bi-weekly.

What needs to be considered when applying Scrum in mechanical or hardware environments is that the physical nature of the final product should be taken into account and the natural process of developing physical objects respected. Due to this, some adaptations need to be made to the framework, prescribed in the literature, in order to accommodate this. The adaptations can differ between organizations, depending on their environment, the projects involved etc. However, the adaptations or deviations from the prescribed framework, made at Marel are the following:

- The team was not cross-functional in the sense that all competences needed to design and construct the final product were not all a part of the team.
- The team may not be able to produce a working product increment each Sprint, although in some special cases it can be possible. Instead, the focus can be to reach a certain goal e.g.

designing a part to a certain extent, or have a part of the product ready for production by the end of a Sprint.

- The Release Plan served as a Product backlog and was not directly managed by the Product Owner. The Development team was more active in creating the backlog items, and this was mainly done during Sprint Planning meetings.

The main lessons learned from the case studies are the following:

- The Marel team has benefitted from the use of Scrum. They set milestones and have been able to reach them on set time. The whole team gets better overview of the project and product, and knows the status of each team member's work. It has helped involve the busy Product Owner in the work and planning.
- It is a planning and prioritization method. At Andritz Hydro AB, the objective is to rebrand their way of working as a prioritization method, as resource allocation needs to be handled at a higher level.
- Team buy-in and ownership of the work method is vital. If the team members do not see the benefits, it will not be sustained and the main purpose of using the framework, helping the team in its work, is lost.
- It is important that leaders or Product Owners are actively involved and give direction. This is even more important in a new product development project like at Marel.
- An Agile coach with experience is important for the start-up, and if the experience is not available within the organization, external assistance should be sought.
- A Scrum Master is vital in the deployment of the framework, as he or she provides support and discipline to the team and helps them stay on track. The Scrum Master needs to have the ability to coach, have good knowledge of the framework and be able to gain the development teams trust.
- As expressed in the SAAB EDS case, it is important that the message given by the organization in regards to work methods or way of working, is in essence "This is the way of working" as opposed to "You have to work like this", which is much more inspiring and provides a better work environment.

8.1 Recommendations

In this sections recommendations will be presented firstly to Marel and secondly to the industry and academy in general.

8.1.1 To Marel

The recommendations to Marel regard further improvements needed on a team and organizational level. These are the following:

- Continuous improvement efforts can be more purposeful. It has shown some results but can be utilized even further.
- Continuous education on the framework might be needed for the team. Parts of the framework that seem to need further improvement are the Sprint Reviews and Retrospectives. This could be solved by training and further education for the team.
- The roles and their responsibilities have been somewhat unclear. It is suggested that the development team, Product Owner and Scrum Master create their own tailored Framework Definition. This would include the responsibilities of each role, and the objectives and rules of the different events etc. This should be agreed upon by all parties, and provide a clearer understanding of the framework, in the way the mechanical team uses it. This would also be beneficial for the embedded software team.

- Keep the structure of the Scrum framework, but continue improving and adapting to the needs of the teams. These changes need to be justified and later these can be added to the team's Framework Definition.
- The organization is becoming more aware of Scrum and Agile. However, understanding of the essence and benefits of Scrum and other Agile methods, is still lacking. Educating the organization and top-management in particular, might increase the harnessing of benefits, and also make those that the Scrum teams interact with, more understanding of the way they work, and what role they play in that context.
- As has been suggested by the Marel GRB Agile Center, further education for Product Owners will be beneficial. They have to understand the framework, as misunderstandings can affect the work of the team.
- The Product Owner suggested looking at products from a functional perspective and testing each individual function, for future projects. This would go well along with Lean PD practices. It is therefore recommended that Marel and the team in question look at the possibilities of using practices such as concurrent engineering, knowledge gaps and trade-off curves.
- In the future, it would be interesting to see if it will be possible for mechanical teams to be able to test mock-ups more frequently.

8.1.2 To the Industry in General

Recommendations to organizations thinking of deploying Scrum in their mechanical or hardware development teams:

- Top-management support is important for the success of sustaining this way of working.
- Educate teams and their managers and/or Product Owners. It is important to know the ideology, theory and philosophy behind the practices.
- An experienced Scrum Master or Agile coach is needed to lead the implementation and deployment. That individual should not only know the framework but the principles and values behind it. Experienced coach builds trust in new teams, as the experience encourages them to see that the change effort can work for them.
- It is important not to go blindly by the book, be critical and see the value. Adaptations are often needed, and they can improve teams' work even further and increase team buy-in. However, do not cut corners unless it is well reasoned for. Keep the basic elements of the framework and then adapt to your needs.
- Adapt the process and framework for your organization, define how you apply the framework and make everyone aware of the definition.
- There will likely be some reluctance to change. Be prepared for this, be patient, and work towards helping the team see the benefits and how the framework can help them.
- The study indicated that many have difficulties in regards to work breakdown, when not accustomed to it. However, it is possible to improve in work breakdown, although work should only be broken into a reasonable detail that helps the team.
- Sprint Reviews and Retrospectives can be challenging for the teams, but can provide a lot of value in feedback and continuous improvement.

8.2 Further research

In this chapter, further research ideas will be presented, some of which had to be excluded from this thesis.

Firstly, it would be interesting to see further case studies of Scrum applied in mechanical and hardware development.

It would be interesting to study more thoroughly what adaptations software development teams have made to the Scrum framework and if these are applicable in hardware development. In addition to that, it would be of interest to compare successful software teams further and their success factors to hardware development.

It was suggested by one of the software team's Product Owners at Marel, that the frustration and issues the mechanical team faced, where things that the software teams faced when they were starting to use Scrum, several years ago. It would be of interest to compare this further, to see the similarities in frustrations, hindrances and challenges that software and hardware teams meet during start up and deployment of Scrum. This might allow mechanical teams to steepen their learning curve.

It would be very interesting to study this topic from a change management point of view, looking at double-loop learning and other theories. It could be possible that if certain change management principles are applied in implementation of Scrum and its further deployment, it could help with overcoming the resistance to change, helping teams successfully applying the framework and a steeper learning curve could be achieved.

It was the intention to include a review of Lean Product Development (LPD) in the study, as it is closely related to Scrum, specially Scrum in Hardware development. It was even stated by the previously mentioned Agile and LPD expert referred to earlier in the report, that he felt Scrum was a LPD practice[12]. Due to lack of time and that the scope needed to be narrowed, this was excluded from the thesis. It would be interesting to study Scrum in hardware development in relation to LPD academically, as well as practically to see if the Marel development teams could utilize some of the practices provided by Lean Product Development literature. It would also be interesting for the organization on a higher level. Practices and concepts that were of most interest are Knowledge Gaps; LAMBDA for continuous improvement; Set-based design and Trade-off curves; higher level visual management; and the capture and reuse of knowledge through LPD knowledge management practices. The practice of identifying and closing knowledge gaps could replace risk management; the focus would be on learning and not eliminating risk. This could be very useful in the way the Product Owner described how he would have developed the current product by dividing it up to different functions. Then the different functionality knowledge gaps could be closed.

[12] Interview with Mikael Lundgren on 28.8.2013

References

Agile Alliance (2013) *The Agile Manifesto*. Available from: http://www.agilealliance.org/the-alliance/the-agile-manifesto/ [Accessed 29 July 2013].

Anon (2013a) Agile software development. *Wikipedia, the free encyclopedia*. [online]. Available from: http://en.wikipedia.org/w/index.php?title=Agile_software_development&oldid=578783113 [Accessed 26 October 2013].

Anon (2013b) Waterfall model. *Wikipedia, the free encyclopedia*. [online]. Available from: http://en.wikipedia.org/w/index.php?title=Waterfall_model&oldid=578123644 [Accessed 27 October 2013].

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., *et al.* (2001a) *Manifesto for Agile Software Development*. Available from: http://agilemanifesto.org/ [Accessed 19 April 2013].

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., *et al.* (2001b) *Principles behind the Agile Manifesto*. Available from: http://agilemanifesto.org/principles.html [Accessed 29 July 2013].

Belie, R.G. (1993) *Three enabling technologies for integrated product development*. In: [online]. January 1993 Reno, NV. Available from: http://arc.aiaa.org/doi/pdf/10.2514/6.1993-923 [Accessed 31 July 2013].

Bryman, A. & Bell, E. (2011) *Business research methods*. 3rd ed. Cambridge ; New York, NY: Oxford University Press.

Bryman, A. & Bell, E. (2007) *Business research methods*. [online]. Oxford university press. [Accessed 27 October 2013].

Carlson, R. & Turner, R. (2013) Review of Agile Case Studies for Applicability to Aircraft Systems Integration. *Procedia Computer Science*. 16pp. 469–474. [Accessed 16 May 2013].

Cohn, M. (2005) *Agile estimating and planning*. Pearson Education.

Cohn, M. (2004) *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional.

Deemer, P., Benefield, G., Larman, C. & Vodde, B. (2010) *The Scrum Primer 1.2*. Available from: http://goodagile.com/scrumprimer/scrumprimer.pdf [Accessed 21 February 2013].

Deemer, P., Benefield, G., Larman, C. & Vodde, B. (2012) *The Scrum Primer 2.0*. Available from: http://assets.scrumfoundation.com/downloads/1/scrumprimer20.pdf?1352449266 [Accessed 25 July 2013].

Denning, S. (2012) How Agile can transform manufacturing: the case of Wikispeed. *Strategy & Leadership*. 40 (6), pp. 22–28. [Accessed 2 September 2013].

Grenning, J. (2002) Planning poker or how to avoid analysis paralysis while Release planning [online]. *Hawthorn Woods: Renaissance Software Consulting*. 3. [Accessed 11 October 2013].

Heidt, H., Puig-Suari, J., Moore, A., Nakasuka, S. & Twiggs, R. (2000) *CubeSat: A new generation of picosatellite for education and industry low-cost space experimentation* [online]. [Accessed 8 August 2013].

Hennink, M.M. (2011) *Qualitative research methods*. London ; Thousand Oaks, Calif: SAGE.

Highsmith, J. (2001) *History: The Agile Manifesto*. Available from: http://agilemanifesto.org/history.html [Accessed 21 February 2013].

Huang, P.M., Darrin, A.G. & Knuth, A.A. (2012) Agile hardware and software system engineering for innovation. In: *Aerospace Conference, 2012 IEEE*. [online]. 2012 pp. pp. 1–10. Available from: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6187425 [Accessed 25 July 2013].

Jackson, M.B. (2012) Agile: A Decade In. *PM Network*. 26 (4) p.pp. 58–62. [Accessed 30 July 2013].

Johnson, N. (2011) Agile hardware development – nonsense or necessity? *EE Times*. [online]. Available from: http://www.eetimes.com/design/eda-design/4229357/Agile-hardware-development---nonsense-or-necessity- [Accessed 17 May 2013].

Leffingwell, D. (2010) *Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*. Addison-Wesley Professional.

Merriam, S.B. (2002) Introduction to qualitative research. *Qualitative research in practice: Examples for discussion and analysis*. pp. 3–17. [Accessed 25 April 2013].

Petersen, K., Wohlin, C. & Baca, D. (2009) The waterfall model in large-scale development. In: *Product-Focused Software Process Improvement*. [online]. Springer. pp. pp. 386–400. Available from: http://link.springer.com/chapter/10.1007/978-3-642-02152-7_29 [Accessed 29 July 2013].

Pikkarainen, M., Salo, O., Kuusela, R. & Abrahamsson, P. (2012) Strengths and barriers behind the successful agile deployment—insights from the three software intensive companies in Finland. *Empirical Software Engineering*. 17 (6), pp. 675–702. [Accessed 12 August 2013].

Poppendieck, T. (2004) *The Agile Customers Toolkit*. [online]. Available from: http://www.rallydev.com/documents/Rally_Agile_Customers_Toolkit.pdf [Accessed 21 February 2013].

Rasmusson, J. & Pfalzer, S.D. (2010) *The Agile Samurai: How Agile Masters Deliver Great Software*. Pragmatic Bookshelf.

Royce, W.W. (1970) *Managing the development of large software systems: Concepts and Techniques*. In: [online]. August 1970 Los Alamitos: Orginally published by TRW. pp. pp. 1–9. Available from: http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf [Accessed 26 July 2013].

Schiel, J. (2012) *The ScrumMaster Study Guide*. CRC Press.

Schwaber, K. (2004) *Agile Project Management with Scrum*. United States of America: Microsoft Press.

Schwaber, K. & Beedle, M. (2002) *Agile software development with Scrum*. [online]. Prentice Hall Upper Saddle River. [Accessed 7 October 2013].

Schwaber, K. & Sutherland, J. (2011) *The Scrum Guide – The Definitive Guide to Scrum: The Rules of the Game*. [online]. Available from: http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum_Guide.pdf#zoom=100 [Accessed 25 July 2013].

Scrum Alliance Inc (2012) *Scrum a description v2012.12.13*. Available from: http://agileatlas.org/images/uploads/corescrum.pdf [Accessed 25 July 2013].

Senapathi, M. & Srinivasan, A. (2013) Sustained Agile Usage: A Systematic Literature Review. In: *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*. [online]. 2013 pp. pp. 119–124. Available from: http://dl.acm.org/citation.cfm?id=2461016 [Accessed 17 May 2013].

Waldmann, B. (2011) There's never enough time: Doing requirements under resource constraints, and what requirements engineering can learn from agile development. In: *Requirements Engineering Conference (RE), 2011 19th IEEE International*. 2011 pp. pp. 301–305. doi:10.1109/RE.2011.6051626.

Williams, L. (2012) What agile teams think of agile principles. *Communications of the ACM*. 55 (4), pp. 71–76. [Accessed 29 July 2013].