CHALMERS
UNIVERSITY OF TECHNOLOGY

Scaling Agile Development

A case study at Saab business area Surveillance

Master's thesis in the Quality and Operations Management Programme

Jonas Bodén
Camilla Johansson

Department of Technology Management and Economics
Division of Innovation and R&D Management
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2016
Report Number: E2016:114

REPORT NO. E2016:114

Scaling Agile Development
A Case Study at Saab Business Area Surveillance

JONAS BODÈN
CAMILLA JOHANSSON

Supervisor Chalmers
Lars Trygg

Supervisor Company
Thomas Ridderstråle
Karin Thorvaldsson

Department of Technology Management and Economics

CHALMERS UNIVERSITY OF TECHNOLOGY

Göteborg, Sweden 2016

Scaling Agile Development
A case study at Saab Business Area Surveillance
JONAS BODÉN, CAMILLA JOHANSSON

Technical report no E2016:114

Department of Technology Management and Economics
Division of Innovation and R&D Management
Chalmers University of Technology
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Cover:
A range of products from Saab's different business areas

Reproservice
Göteborg, Sweden 2016

Abstract

During the last decades companies have experienced a more challenging nature due to a globalized market with tougher competition, fragmented, demanding markets and diverse and rapidly changing technology. This has created new problems for R&D departments in the form of increased product complexity, shorter product life cycles and more product variation all of which creates a higher level of uncertainty (Sommer, Dukovska-Popovska & Steger-Jensen, 2014). This led to the need for new product development methods which combine speed and flexibility and one of them is agile development. Agile development originates from lean and is mainly used in software industry but there are examples of leading companies that have successfully integrated agile development techniques within all their development processes. There are however few examples of enterprises who has managed to become completely agile (Cooper, 2014).

This thesis investigates the possibility of scaling agile development in companies which integrates both hardware and software. Scaling meaning implementing and managing agile development to all teams and from developers up to program management. The thesis also investigates the benefits and drawbacks with this introduction and how the transition can be performed. An inductive approach was used when collecting data and analysing the results. Since the study aimed to combine theory and observation to reach a deeper understanding in the area of scaling agile development, this was seen as the most appropriate approach. During the thesis 33 interviews were conducted at the investigated department at Saab, in addition three expert interviews were conducted with consultants from leading agile companies and finally 11 interviews with four different case companies.

The investigation showed that it is possible to scale agile development in companies which integrates both hardware and software. And the study showed that several agile methods can be used in this type of development such as Scrum and Kanban. The studied companies showed to have improved both productivity and quality since they made the transition. One of the drawbacks from scaling agile that has been uncovered is that the transition is a long journey which requires a large investment. Several frameworks for scaling and managing agile development has been researched and all of them show potential for companies that integrates both hardware and software. However, it is crucial to choose a framework which is aligned with the organisation’s culture and not implement a framework because it worked in another organisation.

Keywords: Agile, Scaling Agile Development, Scrum, SAFe, Agile Culture, Kanban

Acknowledgements

The conclusion of this thesis means that five years of studies at Chalmers are over for us. Agile development is a hot topic and that has become clearer throughout this study. The people we have met are interested in our work and has been eager to share their experiences. This means that the study has been both fun and educational for us as authors and we could not be happier with the way people we have met has treated us.

We would like to thank our supervisor Associate Professor Lars Trygg at Chalmers for his support during this thesis. A special thanks Thomas Ridderstråle, Karin Thorvaldsson and Tomas Berling for providing us with advice and support. Additionally we would like to express our gratitude to all the people within the baseline program who have shared time and knowledge with us during this study. Lastly we would like to thank the case companies and experts who provided us with useful knowledge and their experiences.

Best Regards
Jonas & Camilla

List of Abbreviations

ART	Agile Release Train
C2	Command and Control
I&V	Integration and Verification
ILS&SS	Integrated Logistics Support & System Safety
IP	Innovation and Planning
LeSS	Large Scale Scrum
MS	Mission system
PI	Program Increment
PO	Product Owner
PPM	Project Portfolio Management
SAFe	Scaled Agile Framework
SM	Scrum Master
WIP	Work In Progress
XP	Extreme Programming

Contents

1.Introduction

Saab is a global company developing a range of world leading products and solutions from military defence to civil security, with the vision to keep society and people safe (Saab Group, 2016).

1.1 Background

During the last decades companies have experienced a more challenging nature due to a globalized market with tougher competition, fragmented, demanding markets and diverse and rapidly changing technology. This has created new problems for R&D departments in the form of increased product complexity, shorter product life cycles and more product variation (Sommer, Dukovska-Popovska & Steger-Jensen, 2014). One of the main problems that have occurred from the new challenges is the possibility of market demands changing during the product development lead-time, creating the need for changes in the product design. According to Cohen (2010) the cost of changes increases exponentially during the development process and changes can also cause uncertainty within the company. However, if these changes are not made the product will not meet the market demand and might have limited success, therefore the company needs to find a way to make these changes This has created the need for new product development methods which combine speed and flexibility, some of these methods are Lean Product Development, Design for Six Sigma and Agile Product Development. Agile development has derived from lean, which focus on creating a flow and identifying bottlenecks. Both Design for Six Sigma and Lean Product Development has been adopted in large manufacturing companies, Agile Product Development however has mainly found success within software companies. There are examples of leading companies that have successfully integrated agile development techniques within their development process and experienced good results, but few have adopted a completely agile method (Cooper, 2014).

Agile product development is based on the agile manifesto that was written in 2001 and was originally meant for developing software. Agile development is based on four value statements:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan (Hunt 2006)

Different methods have been developed in order to work according to the values. One of these methods is Scrum. Scrum is a way to continuously deliver working software by breaking down the development work into packages called stories and these stories are planned into sprints. After each sprint the product is tested and verified. This provides the development team with quick feedback both from early testing and also the possibility of early customer input (Schwaber & Sutherland, 2016).

Another commonly used agile method is Kanban. Kanban is based on five core elements:

- Visualize the workflow
- Limit work-in-progress (WIP)
- Manage flow
- Make policies explicit
- Implement feedback loops

In practice this usually means that a Kanban board is used to show the tasks that are going to be performed. A maximum number of tasks is set for the development team and the product owner is responsible for putting up new tasks. In opposite to Scrum there are no fixed sprints with deliveries, which make Kanban more flexible (Al-Baik & Miller, 2015).
In order to meet the requirements of more speed and flexibility agile product development has garnered more interest in recent years. However, there are critics to this method, Tathagat (2015) is calling the idea of following an agile process and then thinking that agility is achieved a farce. He argues that agility is about supporting the mindset that prepares the employees to adapt in an evolving environment and continuously find more effective ways to solve problems. This shows that there is still a debate within the research community regarding how agile methods should be used. There are success stories from smaller companies and projects but it’s hard to find examples that focus on how this can be adapted to larger manufacturing organisations.

In 2014 a department at Saab decided to reorganise their organisation. Prior to 2014 the development had been conducted in customer projects using a stage-gate model, this lead to that each project came up with customised solutions to similar problems which caused unnecessary work. The new organisation introduced a baseline program that would handle all the customer projects and develop a base product from which customer adaptations would be built, see Figure 1. The product that the baseline program develops is a new generation of an existing radar solution (Saab Group, 2016). The product structure of the radar can be considered to be complex since it is integrating both hardware and software.

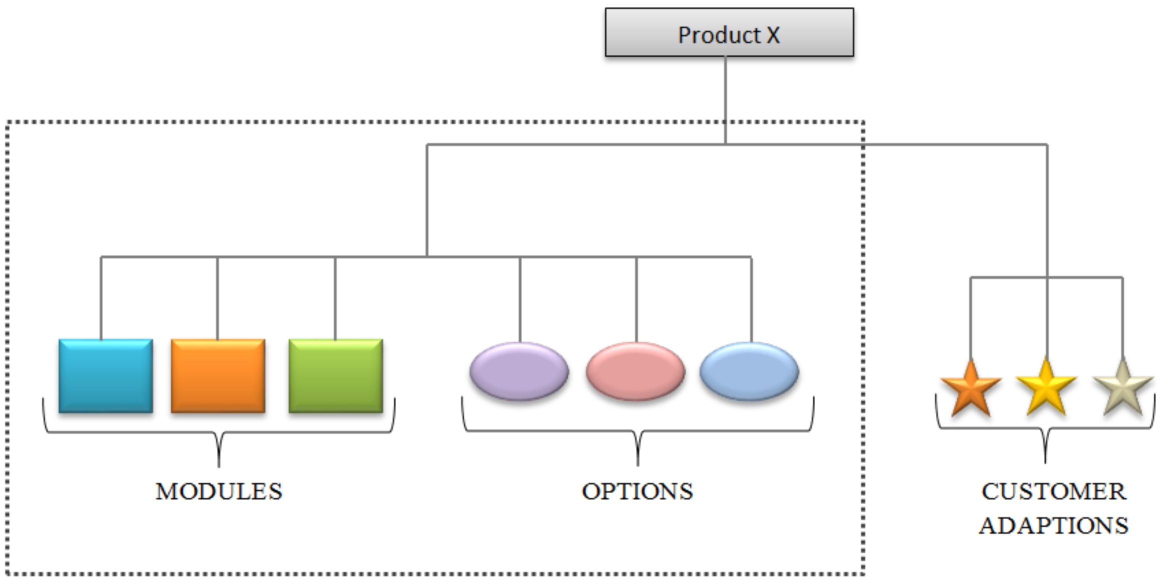


Figure 1: From base product to customer adaptions

The baseline program currently consists of five subprograms and one project. The subprograms will work continuously with the development of the product while the project is limited in a time-frame. The five subprograms are: Mission System (MS), Sensor Development, Command and Control (C2), Integration and Verification (I&V) and Integrated Logistic Support & System Safety (ILS&SS), see Figure 2.

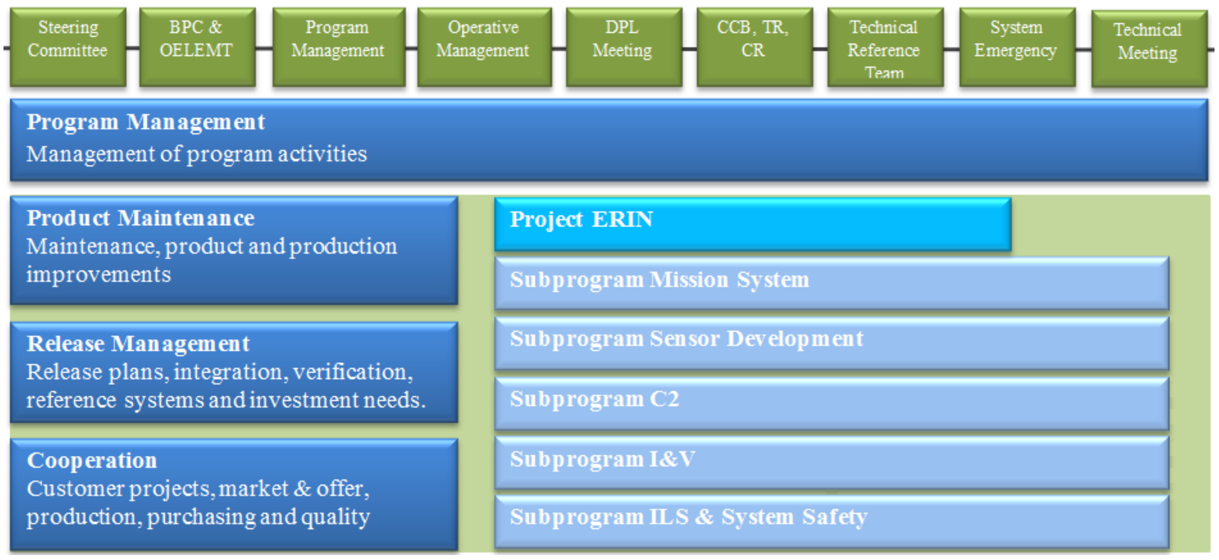

Steering Committee
BPC & OELEMT
Program Management
Operative Management
DPL Meeting
CCB, TR, CR
Technical Reference Team
System Emergency
Technical Meeting
Program Management
Management of program activities
Product Maintenance
Maintenance, product and production improvements
Release Management
Release plans, integration, verification, reference systems and investment needs.
Cooperation
Customer projects, market & offer, production, purchasing and quality
Project ERIN
Subprogram Mission System
Subprogram Sensor Development
Subprogram C2
Subprogram I&V
Subprogram ILS & System Safety

Figure 2: The baseline program

From its inception, the baseline program has been supposed to perform the development according to the agile development principles. This has however been implemented differently within the different subprograms. Sensor development and C2, who mainly develops software, works in Scrum teams and have adopted the agile ways of working. However, the other subprograms are not as far along when it comes to agile development and this leads to problems. One example is that the integration and verification is not handled continuously, which would be needed to gain quick feedback. Instead the system rig is closed for updates and new deliveries during two to three weeks when tests are performed do to time consuming installations. Another problem is that the synchronization between the subprograms is lacking due to the different ways of working and this also causes troubles in the verification of the product.

Since this discrepancy in ways of working exists the program management now wants to further scale the agile development within the baseline program. This means that they want all subprograms to work with similar methods and tool (horizontal scaling) and that the management, from operative management to program management, should support the agile methods (vertical scaling).

1.2 Purpose

The purpose of this thesis is to investigate the possibility to create a structure that allows the subprograms in the baseline program at Saab to work according to agile development procedures. If the agile methods are considered suitable an investigation will follow on how agile approaches can be scaled vertically and horizontally, to all subprograms and from developers up to the program management. If the agile methods are not applicable for all subprograms an investigation on how to enable agile methods in the appropriate subprograms will be conducted. To justify agile development an analysis of the potential benefits of using these principles will be conducted. The investigation will also include what factors are important to consider when scaling and implementing agile development in an organisation. The aim is to find key factors that are crucial for a successful scaling of agile development.

1.3 Problem Analysis and Research Questions

Today, the most used development frameworks in companies are the Waterfall and Stage-Gate models. However, one of the main problems that occur within these processes is that they are not very flexible. The aforementioned models have been developed in many variations in order to fit specific company's needs; there already exists models that follow a more agile system within certain stages (Cooper, 2009, 2014). There also exist models for scaling agile development; the SAFe-Model is one of the models that is used by larger companies today but mainly in software development (Scaled Agile, 2016). Therefore, to deliver an analysis of agile development to Saab we first need to understand the different agile methods that are available and which of them that are applicable in this case. That leads us to our first research question:

● What different Agile Product Development methods exist and how can they be applied for development programs which integrates hardware and software, such as at Saab's baseline program?

Saab is currently working with agile development methods within the sensor development subprogram, however the other subprograms have not yet incorporated this way of working to the same extent. This means that the possible benefits of agile development are not maximized, since the subprograms are not synchronized, causing delays in the validation and testing process. This leads to the second research question:

● What are potential benefits and drawbacks from scaling the agile development to the different subprograms and from developers up to program management?

Scaling the agile development could provide a solution to the problems and get every project within the baseline program to undergo the same process. Most research focus either on how to scale agile development in software companies, where the method was developed (Fowler & Highsmith 2001), or centre around the transition to hardware development in smaller projects. Therefore, a study must be performed to find the key factors for this type of implementation. This leads to the final research question:

● What are the important factors to consider for Saab when scaling Agile Product Development and how can they make the transition?

1.4 Delimitations

The delimitations of this study are that it focuses on large companies, defined as a company with more than 250 employees. The reason for this delimitation is that the study aims to investigate the problems that occur when such a flexible method as agile development is used in a company with a rigid structure and documentation routine. The analysis and recommendations in the thesis is also delimited to one department at Saab and the other departments will only be used as case studies. As stated in the introduction Project ERIN is part of the baseline program but will come to an end in spring 2017. Since this thesis will result in recommendations for Saab and is to be delivered in the beginning of 2017 Project ERIN will be delimited from the investigation.

2. Methodology

One of the most important factors to consider when conducting research is how to design the method to be able to answer the research questions. The choices made for the method of this thesis are presented below.

2.1 Research strategy

There are two common strategies for conducting a research strategy, quantitative and qualitative research (Bryman & Bell, 2015). A quantitative research strategy involves numerical data for example questionnaires or historical data analysis. One of the advantages of a quantitative strategy is that historical data can be used which reduces the data collection. It also provides fast interpretation through statistical analysis. However, the drawbacks are that it is not flexible and that it is not an effective way to understand processes. A qualitative research strategy instead focuses on non-numerical data, for example interviews. The strengths of a qualitative research strategy are that it provides a possibility for in-depth understanding of people's opinions and to adjust the data collection based on the findings. It is also more applicable when generating new theories. The drawbacks are that the data collection requires more time and resources and the analysis might be more difficult to perform (Easterby-Smith et al., 2015).

This thesis was performed mainly by using a qualitative strategy since there is existing knowledge in the software industry but a lack of knowledge regarding how it can be adopted into other areas and a generation of theory is necessary. Therefore, interviews were chosen to be the main method of data collection instead of numerical data.

2.2 Research Approach

There are three main research approaches, inductive, abductive and deductive. A deductive approach aims to confirm a theory by observing reality and reaching a specific conclusion. This is mainly used with quantitative research. With an inductive approach researchers try to combine theory with real world observations to draw general conclusions. An abductive approach tries to find the best prediction to certain observed events and is used when researchers are investigating areas where there are uncertainties (Bryman & Bell, 2015).

During this thesis, an inductive approach was used when collecting data and analysing the results. Since the study aimed to combine theory and observation to reach a deeper understanding in the area of scaling agile development this was seen as the most appropriate approach. The inductive approach also allowed the thesis to be exploratory and let the findings guide the direction of the continued study.

2.3 Research Process

The first research question, *What different Agile Product Development methods exist and how can they be applied for development programs which integrates hardware and software, such as at Saab's baseline program?* was initially investigated by a literature study to get a deep understanding about the area and use as a background for the case studies. Once the various methods were investigated the case studies were used to understand how they could be applied in reality. Expert interviews were conducted with consultants who work according to agile development principles every day. The interview with the consultants provided insight about how agile development is operationalized within the software industry and what their view of adapting it to other industries is. The interviews were semi-structured since this is preferable when working with qualitative data since the respondent's point of view and reflection is of interest (Bryman & Bell, 2015).

The second research question, *What are potential benefits and drawbacks from scaling the agile development to the different subprograms, from developers up to program management?* was also initiated by a literature study to investigate previous attempts at adopting agile development into other fields. After the literature study cases were chosen to use for a comparative case study. The case study was performed to investigate the result of previous attempts to scale agile development by companies in a similar situation to Saabs. The comparison was performed by deeply investigating two organisations within different industries and in different stages in their agile maturity. The interviews performed at Saab were used to identify how scaling agile development could benefit their department by identifying their current problem areas.

The third research question, *What are the important factors to consider for Saab when scaling Agile Product Development and how can they make the transition?* was investigated by analysing the result from the previous research questions and comparing that to how Saab's organisation is working today. Interviews and observations was carried out at Saab throughout the entire project to get knowledge of how they are operating today and what views they have about agile development. One survey was also conducted to determine the organisational culture which was considered to be one important factor for implementing agile development. Expert interviews were performed with people who had been involved in earlier agile transitions to get insight about success factors and obstacles.

2.4 Data collection Primary Data

The primary data in this research consists of interviews and case studies, additionally the research also included a survey and two workshops, all of which are described below.

2.4.1 Interviews

The main data collection in this research has consisted of qualitative interviews, there are two main types of qualitative interviews, unstructured and semi-structured. The main method used in this thesis is the semi-structured. Qualitative interviews are more flexible and seek information and answers from the recipient's point of view. The interview provides room for the recipient to go off tangent, and it is often encouraged that they do, since this provides insight to the researcher about what the recipient considers to be important and not (Easterby-Smith et al., 2015). This leads to deeper knowledge and understanding about the area investigated. However, a few unstructured interviews were performed initially to get insight about the area investigated.

The unstructured interview is similar to a normal conversation, where only a topic or initial question is provided the recipient. The semi-structured interview is also very flexible, but the researcher has prepared a set of questions, often called an interview guide. The interview guide is self-explanatory; it is a set of questions which guide the recipient through the interview. It provides room to go off tangent and get answers from the recipient's point of view, but the questions guide the recipient so the interview is focused around the topic investigated (Bryman, 2012). The semi-structured interviews were chosen since this research attempted to identify how the department at Saab organise their work. The semi-structured approach also helped to identify problem areas in the organisation since all interviewees were given the opportunity to express their experience and opinion of how they are organised. To initiate the data collection, an open discussion regarding the thesis topic and its issues were held with each subprogram separately. The discussions provided fast input from all personnel but also established engagement in the thesis from all, not only the selected interviewees. The interviews in this investigation are one of the main primary data collections. It was used to build a current state of how Saab operationalize and organise their development. The interviews also focused on the obstacles and issues interviewees have in the current way of working. The interviews were conducted with personnel from all subprograms at Saab, in total 33 interviews were held. The interviews were performed with both engineers and management from each subprogram, the guideline used for engineers is presented in Appendix A, and the guideline for managers is presented in Appendix B. The researcher also conducted three expert interviews to get better insight and support when conducting this research. All experts are experienced agile consultants working with agile implementations in leading companies. The guideline used in the interviews are presented in Appendix C.

The interviews in this research have been conducted with two interviewers, which are encouraged by Bryman & Bell (2015). Having two interviewers is beneficial since, after opening the interview, one interviewer can focus on conducting the interview while the other is focused on taking extensive notes. The passive interviewer can also focus on observing body language and expressions used in the interview and are always allowed to intervene at any time if noticing the interview going off topic (Bryman & Bell, 2015). The use of multiple interviewers also provides better opportunities for discussion during the interviews, which often leads to a better understanding of the recipient's opinions. A drawback from using multiple interviewers is that the cost regarding time is greater. Since two interviewers were used the interviews in this research were not recorded, this since the transcribing the recordings are time consuming and it can be uncomfortable for the interviewee. The transcribing was performed immediately after the interviews, using the extensive notes taken during the interviews. This was since not having a recording it is more beneficial to perform transcription before performing new interviews so no information is lost or mixed up with other interviews.

2.4.2 Sampling

The interviews were conducted one subprogram at a time, starting at I&V which is the final receiver in the baseline program. This to find issues and obstacles at the end of the process to have the knowledge of these as the interviews are conducted throughout to the subprograms working in the initial phases of development. The sampling was conducted using the snowball approach. Snowball sampling is when the researchers initiate the interviews with a few selected recipients, where the recipients then can be used to recommend additional personnel to interview (Bryman, 2012). Snowball sampling was chosen since a lot of personnel operating in the baseline program are spread out in many various tasks, and it could be difficult to find a sample which could represent the operational structure of the entire baseline program. For the initial interviews in each subprogram, the recipients were recommended by the program management, where the recipients where personnel with good experience and overall view of the subprogram. From these interviews there were frequently recommendations of additional personnel the recipients felt would contribute to the investigation. The interviews were performed with a mix of management, technical experts and engineers. The number of interviews in each subprogram varied depending on the subprograms size and how much the assignments could vary within each subprogram. Interviews were held continuously until the researchers found they had a good sample and the data necessary to analyse each subprogram. *Table 1* shows the number of interviews conducted at Saab and with case companies.

Subprogram	**Number of Recipients**	**Case studies**	**Number of recipients**
Mission System	11	Saab Department X	2
I&V	8	Saab Department Y	4
Sensor	9	Ericsson	3
ERIN	1	Volvo Group	2
C2	3	Experts	3
ILS & SS	3		

Table 1: Interviews conducted for this investigation

2.4.3 Case studies

The researchers studied several companies which had performed similar attempts as this thesis investigates. The company studies were conducted by interviews with one or several people responsible for the change process. A similar approach as described above was used during the interviews used at the company visits. The interviews were semi-structured with an interview guideline prepared which enabled analysis and evaluation among several companies. The guideline used during company visits is presented in Appendix D. After each company visit a concluding analysis were provided the participant to ensure the researchers had understood them correctly. After all company cases were performed an analysis of their implementation approach was established. This to evaluate their implementation and transition and what could be concluded from this in this research.

2.4.4. Survey

One survey was conducted to categorize the organisation according to Schneider’s (1994) culture model. The survey was a self-completion web-based survey, this method was chosen because it is cost effective way to gather the view of many respondents. The disadvantages with this method are that response rates are low and that certain groups can be overrepresented (Easterby-smith et. al., 2015). The respondents consisted of three engineers from each subprogram and all line managers, in total there were 16 respondents. The survey questions weren’t developed by the researchers, instead Schneider’s template question was used. The questions were all closed questions with four alternatives, one alternative for each type of organisation. This made the data analysis simple when determining the organisations culture. The questionnaire used for the survey is presented in Appendix E.

2.4.5 Workshop

Two workshops were conducted at Saab to increase commitment to scale agile development. One workshop was focused on team structure and formation while the other workshop was centred on frameworks for scaling agile development. The first workshop was conducted with representatives from both management and the engineers. The aim for the workshop were to spread knowledge about what defines effective teams and to simulate what a new team formation could be if chosen to reorganise. The purpose of the simulation was to initiate the idea that Saab has the possibility to reorganise and introduce a motivation for the potential benefits. The workshop resulted in several new ideas of how teams could be organised but needs additional planning and preparations if the decision is made to reorganise. The second workshop was focused on frameworks for scaling agile development and was conducted with the program management at the department. The goal with the workshop was to share knowledge about ways to scale agile development and how it could be adopted to fit Saab organisation. The purpose of the workshop was to build commitment to scale agile development, since it is a difficult and resource demanding transition. The workshop was mainly an educational opportunity, not including many exercises, informing management that delegating responsibilities to teams does not mean that management lose all control.

Both workshops had similar preparations, following a guideline for how to conduct successful workshops by Dr. Richard Tiberius and Dr Ivan Silver (2001). The initial preparations were to specify the objective and goals for each of the workshops. Once the objective was set the structure was formed and evaluated several times, using feedback from experienced agile consultants which conduct similar workshops in their work. These consultants are the same people which the expert interviews were performed with. A short description and schedule were provided all attendees before the workshops, so everyone would be prepared and know why they were chosen for the workshop. All possible material, such as post-its, flipcharts and hand-outs were thoroughly chosen so everyone could visualise their ideas and be provided with information and inspiration during the workshops, mainly for the workshop about team structure. After the workshops, a summary of everything was established and provided to the attendees.

2.5 Data Collection, Secondary Data

The findings presented in the theoretical framework provided a basis upon which the analysis could be performed. It has also been used both to further support findings from the interviews. The literature study was conducted in two steps, first to understand the area of scaling agile development and in the second stage to analyse the data. Bryman and Bell (2015) defines a literature review as a way to learn from previous research in the area by summarizing an existing body of research. This increases the understanding of the research topic and facilitates a faster research. There are two different types of literature reviews, systematic and traditional. The systematic literature review aims to draw conclusions regarding a specific research area by using a methodological approach. The traditional approach aims to provide a broader overview of the research field and does not need to follow a methodological approach.

The literature review in this study was carried out according to a traditional approach. To further understand and explain the findings from the primary data, secondary data was collected from books, journals and websites. The information was collected via search engines such as Summon and Google Scholar with key search terms such as, scaling agile development, agile development, The waterfall model and Scrum.

2. 6 Data Analysis

When analysing the interviews, grounded theory was used. Grounded theory is the most commonly used framework when analysing qualitative data (Bryman, 2012). The findings from the interviews were then coded into concepts that were used to get a deeper understanding in upcoming literature review and interviews. The interviews at Saab were not recorded but instead annotated and immediately summarised to find the key points of the interviews. These key points from the interviews can be seen as concepts within grounded theory and when more data was collected these key points were organised into categories. These categories then made it easier to find what further data collection was needed in form of interviews and literature reviews.

When the data was collected the grounded analysis framework suggested by Easterby-Smith et. al. (2015) was used. It consists of seven steps to reach a theory, see Figure 3.

Familiarisation

During the familiarisation phase the researchers check the data to understand what has been collected. During this stage, it is important to consider the focus of the study, the main themes in the data and from what point of view it is being expressed. Interviewee to interviewer relationships should also be considered. During our study the transcripts and summaries of the interviews were studied to find important passages and themes in the data. The different views of the departments and the reason behind the differences were discussed.

Reflection
Easterby-Smith et.al. (2015) states that this stages focuses on making sense of the data. This should not be done by putting the data in a conceptual framework but instead by comparing it to the existing knowledge of the subject. In this study, it was performed through revisiting the literature review to find the different views of scaling agile and putting the findings into the perspective of the agile principles. The focus was to find theories and frameworks that were supported or challenged.

Open Coding
To create links within overwhelming and messy data coding can be used. A code is a short phrase that summarises the meaning of a passage or statement. This code can then be visualized to create links between the different codes. Since this study was focused on solving problems at the case company the coding was centred on identifying existing problems. Important factors for successful implementation of agile methodologies were also identified in the benchmark and expert interviews.

Conceptualisation
During the conceptualisation stage the researchers tries to identify patterns among the codes. The codes are compared and assigned to different categories, these categories are then evaluated to find themes and concepts. During this stage in this study the codes were analysed into different problem areas. These problem areas were then further categorised into themes that were

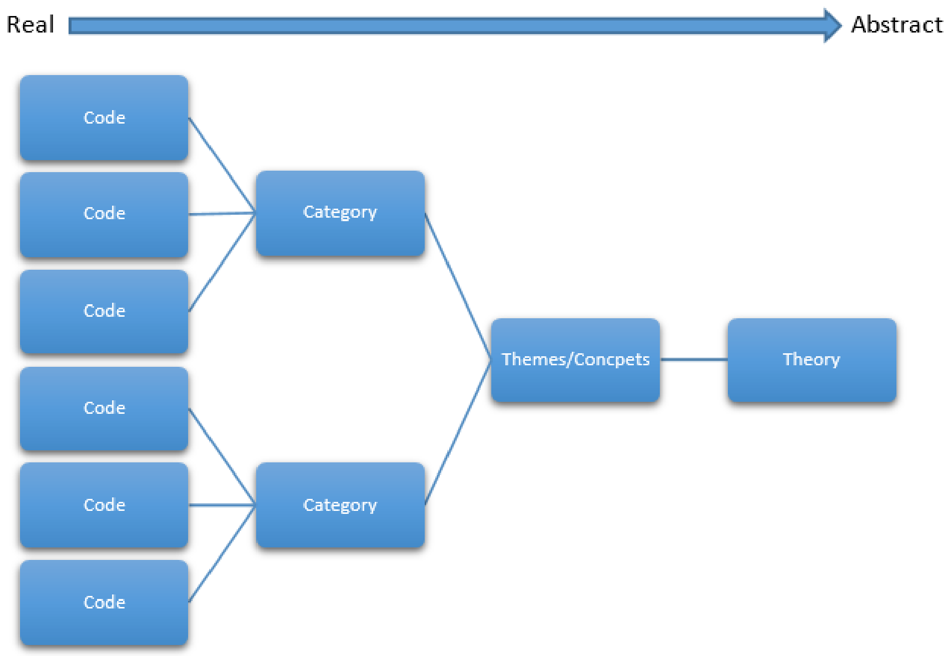


Figure 3: Overview of the analysis process

Focused Re-coding
After the first codes and categories had been determined, the most important ones are used in the second coding step. Here the number of codes and categories are fewer to provide a more focused analysis. The process is iterative and might require the researchers to go back to original transcripts to find differences. The first coding step aims to develop a framework and the second round provides more in-depth analysis of the data. The re-coding in this study consisted of finding differences in opinions regarding the themes and categories and analyse these to find the real problems.

Linking
During this step patterns between concepts should becoming clearer. These patterns consist of how the different concepts and categories relate to one another and these can be integrated to a theory. This theory can thereafter be reviewed by both experts and respondents. This study generated several theories that were presented to respondents to receive feedback. The theories were also presented to the tutors at both Chalmers and Saab for further input.

Re-evaluation
After receiving feedback the researchers can feel that further investigation is needed, some important factors can have been omitted or misunderstood. In this thesis early findings were presented to two of the departments at Saab, thereafter respondents could give feedback. This reinforced certain views while other focus areas were found to be missing.

2.7 Research Quality
Two of the main principles for evaluating research quality in a quantitative research study is the criterions of trustworthiness and authenticity (Bryman & Bell, 2013). Trustworthiness consists of four criteria's: credibility, transferability, dependability and confirmability.

Credibility
The credibility criteria evaluates if the research is credible and acceptable to others (Bryman & Bell, 2013). During this research credibility has been ensured by working with respondent validation and triangulation. Respondent validation has been established during interviews by summarising findings and content of interview in a document which then was provided to the respondent. This gave the respondent the opportunity to ensure the researchers had understood them correctly and also give them the opportunity to withdraw comments or opinions. After all interviews had been performed with each subprogram a short presentation were held so all had the opportunity to give feedback and ask questions. When the analysis of all interviews was established it was also presented to the respondents. Triangulation were used to ensure credibility for the whole thesis, by using multiple sources of data. The data used in the research was interviews, workshops, literature study and internal documents such as policy and program descriptions.

Transferability
Transferability is to ensure the results from the research can contribute to other surroundings other than the organisation investigated (Bryman, 2012). This has been ensured by not only

focusing on Saab but also benchmarking other companies. The research has also increased the transferability by performing general analysis and solutions based on type of organisation instead of specific recommendations to Saab.

Dependability
To ensure the research is dependable the researchers has adopted an auditing approach; meaning that record are kept from all various phases of the research (Bryman & Bell, 2015). This thesis has ensured high dependability by clear description of problem formulation, chosen methods and how sampling was decided upon. Though the summarising of interviews has been decided to be anonymous and not shared with others, the questions used and the overall empirical data is presented and described to enable external auditing if desired.

Confirmability
The criteria of confirmability concerns if the researchers have acted in good faith during the research and not let their personal values or knowledge affect the research (Bryman & Bell, 2015). This can be difficult in qualitative research since it can be hard to remain objective. For this thesis the researcher's previous knowledge about the investigated area was rather limited which has increased their objectivity during the research.

Bryman and Bell (2015) recommends including the criteria of authenticity for determining the quality of the research. Authenticity is a set of five criteria to consider the political aspects of the research. The criteria are: fairness, ontological authenticity, educative authenticity, catalytic authenticity and tactical authenticity.
Fairness concerns if the research fairly represents the members of the social setting investigated. Since the interviews in this research has been performed with a large amount of personnel working in various subprograms and the distribution has been from program management down to developers it can be concluded that the results fairly represent the entire department. The ontological authenticity concerns if the research has helped the investigated social setting to better understand their environment. The researchers have enabled the ontological authenticity by having multiple presentations and shared their analysis with them. The educative authenticity have been established by also presenting the analysis of all the different subprograms at Saab, this provides them a better understanding the surrounding subprograms and not only be aware of their own structure and issues. The catalytic authenticity concerns if the research has engaged the personnel in motivation to change. By presenting the findings from interviews this has shown to motivate and engage the personnel to transition to something new. By realising problems and opportunities from other subprograms the personnel have provided several suggestions for future improvements. The tactical authenticity regards if the research has motivated the people studied to actually take action and change. This has been established in this thesis by conducting multiple workshops where simulations of improvements have been carried out. This leads to increasing the personnel's engagement and it is easier to motivate them to take action if the solution comes from within, a bottom-up solution.

2.8 Quality, Secondary Data

Saunders et. al. (2009) states that survey organization and scientific journals can be considered trustworthy since their reputation depends on the quality of what they publish. Since the researchers within the area has not yet published much regarding the area of scaling agile development other sources was needed. Therefore, consultancy reports and frameworks used by consultants has been studied in the literature review. The most reputable consultancy organisations and most commonly used frameworks was studied to gain knowledge of how the practitioners handle these problems today. However, it is important to note that secondary data have been collected for a different purpose and therefore must be evaluated before application (Bryman & Bell, 2015). To ensure reliability and validity triangulation has been used (Neuman, 2011), in this study this is defined as that two or more sources agree.

2.9 Ethics

When performing research such as this, ethical issues will be highly important. Bryman and Bell (2015) defined four main areas which need to be considered, these are: Harm to participants, lack of informed consent, invasion of privacy and deception. When the interviews were conducted; all four areas were taken in consideration. To ensure no ethical issues would occur the following approach was used in the interview.

At the beginning of each interview the researchers explained the purpose of the interview, informed them that they would remain anonymous and informed them what the data would be used for afterwards. All interviewees have been anonymous in the report and the analysis has been established as a summary for each subprogram. This enabled the researchers to write the analysis general so it would not be possible for others to conclude specific individual opinions or statements. After each interview a summary with the main conclusions were provided to the interviewees afterwards for them to have the possibility to take back or comment on statements. The participants were also informed that if they at any time should feel uncomfortable with either a question or an observation they had the right to not answer or participate. After all interviews were conducted within a subprogram and the analysis was established it was provided all interviewees before it was shared with any supervisors or stakeholders. The analysis was provided to the interviewees to ensure them that the analysis is general and their individual opinions and statements cannot be found in the material.

3. Theoretical Framework

This chapter presents the information found in the literature study. It describes the agile methodologies, frameworks for scaling agile development and additionally organisational culture.

3.1 How Agile evolved

The software development started off in the 1970s with methods based on the waterfall model. This meant that testing was not performed until the very end of the project and potential problems that occurred in the early stages were not caught before then. These problems could require major redesigns of the product and therefore be costly. Because of that more iterative approaches began to evolve with incremental and evolutionary development features (Cobb 2015). These iterative and incremental development (IID) methods begin to get a hold of the software community in the late 1970s and early 80s leading to several new methods being developed in the coming years. These methods include Joint Application Design (1986), Rapid System Development (1987), Rapid Application Development (1991), Scrum (1995). Another important development during these years was the Rational Unified Process (RUP) which became widely used. However, the method that garnered the most interest was Extreme Programming (XP) and this lead to the unprecedented success of agile methods in the early 2000s (Larman & Basili, 2003).

As can be seen many methodologies were developed in the 80s and 90s which lay ground for the agile revolution. Since there were so many different opinions a cohesive agile approach was needed.

3.2 The Waterfall Model

The waterfall model, illustrated in Figure 3, was developed in the 1970s to provide a structured way to perform software development. The method builds on successive phases, but as each step is performed and the design gets more detailed, there is an iteration of previous and succeeding steps but not with the steps that is further away in the waterfall. The testing phase at the end of the cycle is very critical, if the requirements made in the earlier steps are not met here significant redesign might be necessary and this leads to a longer lead time and higher costs (Royce 1970).

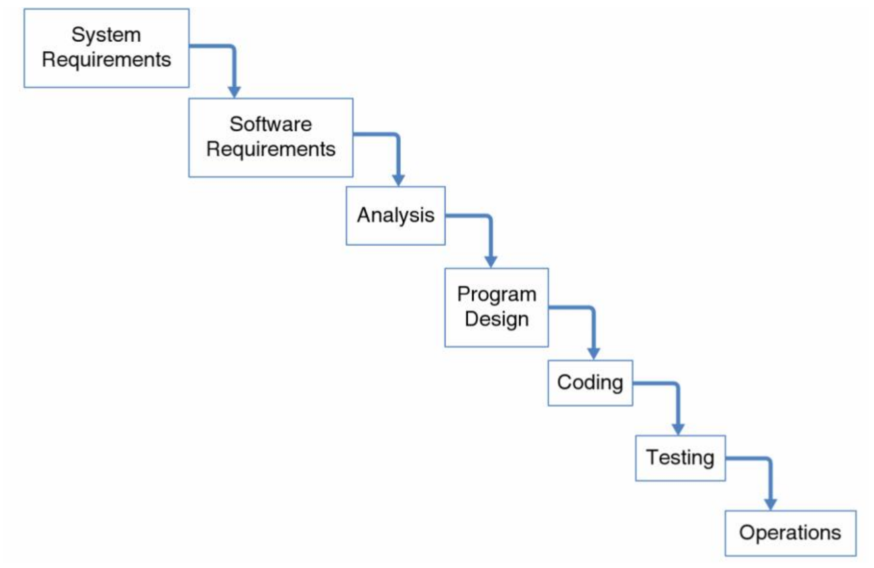


Figure 4: The Waterfall-model (Royce, 1970)

Royce (1970) suggests five modifications to minimize the risks of the waterfall-model. First, he suggests that a preliminary Program Design step should be put before the analysis. This assures that analyst and program designers will start communicating earlier and increase their knowledge of the system. This allows for a more iterative approach for the system design and should give a better input to the coders. Royce (1970) also proposes the documentation of the design is important, he claims that the value of the project design is within the documentation and that the communication and hand-offs will be insufficient without it. The third improvement is to do the project twice, i.e. make a pilot project. This allows for early testing and to find trouble spots in the design and this leads to a higher quality in the delivered product. Royce (1970) also states that testing is the most important part of the project and therefore it must be planned early. Since testing does not occur until the end in a waterfall project it must be thorough and go through all functions in the product. The last improvement is to involve the customer in early stages in order to get feedback and make sure that the product fulfils the customer requirements.

3.3 Agile Manifesto

In 2001 leaders of the agile movement in software development met to try to find common ground around their theories and methods. Since they proclaimed less standardization and modelling they wanted to find guidelines for how to work in their new agile way. This resulted in the agile manifesto which is derived from four value statements (Fowler & Highsmith 2001):

Individuals and interactions over processes and tools: Shows that although processes, methods and tools can be helpful in a development project, it is the people involved who have the largest impact on success. Therefore, assigning the right people to the task and encouraging communication is of outmost importance (Hunt 2006).

Working software over comprehensive documentation: According to Hunt (2006) sometimes more hours are spent on documenting your work than on producing the actual product, the software. The focus in agile development is to produce the product on time and according to specification, not on the documentation of the project.

Customer collaboration over contract negotiation: When meeting the customer the time should be spent on involving them in the project and trying to create value for them. Today however a lot of time is spent on contract negotiations where requirements are set instead of letting the specifications come from working together with the customer (Hunt 2006).

Responding to change over following a plan: Agile development does not focus on pre-set requirements or plans, instead the ability to change is important. This does not mean that a plan does not exist; rather it evolves with customer feedback and changes in demand (Hunt 2006).

With these value statements as a base, twelve principles are derived, Figure 5. These principles can be used to understand agile software development and the guidelines used within this methodology. It can also be used as a checklist to see if your project is developing according to the agile methodology (Hunt 2006).

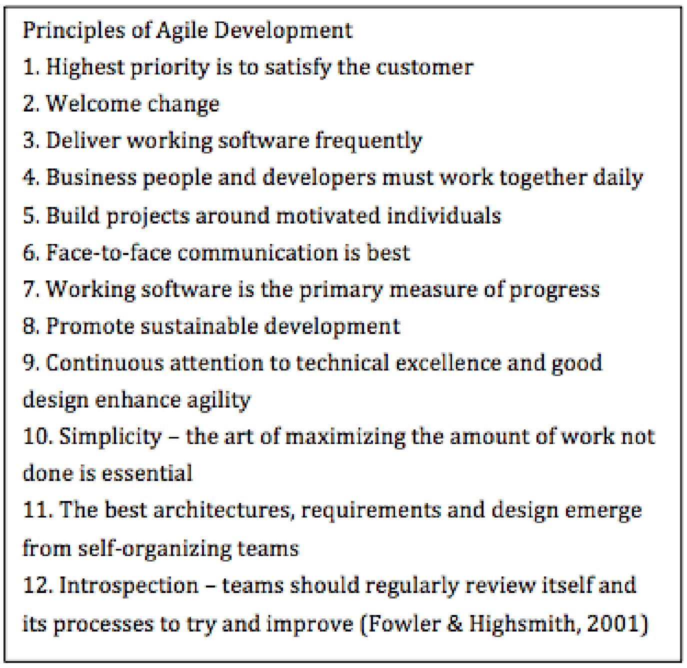

Principles of Agile Development
1. Highest priority is to satisfy the customer
2. Welcome change
3. Deliver working software frequently
4. Business people and developers must work together daily
5. Build projects around motivated individuals
6. Face-to-face communication is best
7. Working software is the primary measure of progress
8. Promote sustainable development
9. Continuous attention to technical excellence and good design enhance agility
10. Simplicity – the art of maximizing the amount of work not done is essential
11. The best architectures, requirements and design emerge from self-organizing teams
12. Introspection – teams should regularly review itself and its processes to try and improve (Fowler & Highsmith, 2001)

Figure 5: The principle of Agile Development (Fowler & Highsmith, 2001)

In general, these ideas strive to meet the demand of the rapidly changing and uncertain business world that has emerged in recent years. The method aims to move the processes from a heavyweight to a lightweight process by trying to reduce unnecessary documentation and focus on value-adding activities. It also strives to be adaptive in its process instead of predictive, meaning that instead of trying to predict what the customer will want when the product is developed adaptations are made during the process. The focus in agile development is on the people, not the processes, since it is they who create the product (Hunt, 2006).

3.4 Agile Methods

As we will further discuss there are several different methods in use today, however they all build on a common theme. They are trying to create a working solution for the user and at the same time being able to adapt to changing customer requirements. Compared to more traditional methods, who also try to create a working solution for the user, the emphasis in agile methods is the ability to change during the process. This leads to a difference in emphasis during these types of projects. In an agile project the time and resources are fixed but the functionality delivered at the end of the project is flexible. In a traditional development project the specification are set up in the beginning and therefore the functionality is fixed while time and resources are flexible. This means that a traditional project is dependent on planning and timelines to make sure that the functionality is delivered. Instead agile projects are managed through a backlog with epics, features or user stories to be developed. This backlog is then prioritised with the most business critical stories to be developed first. (Hunt, 2006)

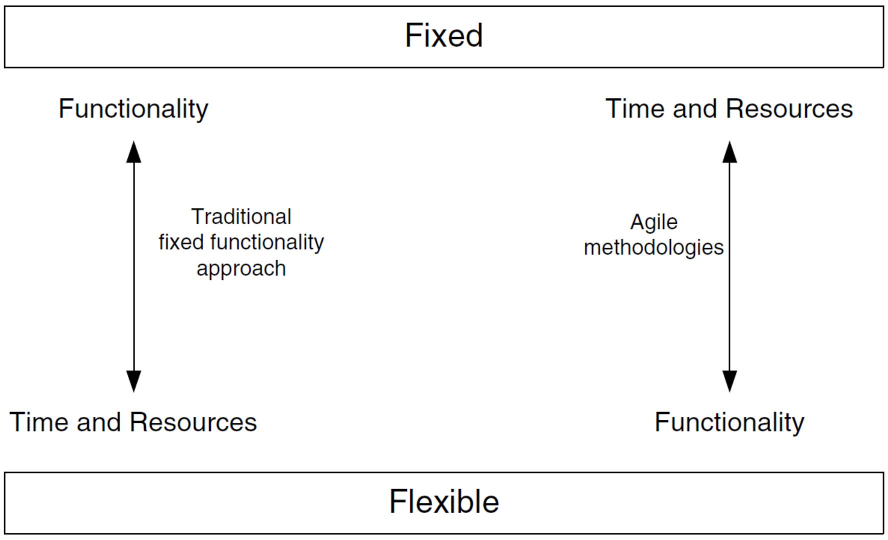


Figure 6: Illustration of the difference between agile and traditional methods (Hunt, 2006)

Epics, features and user stories.

Common for many agile frameworks and methodologies is the way of establishing assignments to teams from the product roadmap. Most agile methodologies use the terms of *epics, features* and *user stories* to explain tasks and assignments (Coehlo & Baso, 2012). As illustrated in *Figure X*, tasks are broken down from epics into several features which then is additionally broken down to smaller user stories which are developed by the teams. A user story is a small task that will be developed and delivered in the form of functionality. The user stories are small enough to be developed by a single team during a short time period but large enough that it will bring the user value and it is usually described in a short sentence. An example of how a user story is described is: I [as a user] want [a function] that gives me [value]. An epic is a very large user story, which cannot be developed by a single team over a short period of time. An epic usually expands over several releases and is broken down to several features. A feature is a set of user stories which can be related to each other and together forms a package of functionality (Target Process, 2014). Figure 7 illustrates how a user story is broken down form a large epic.

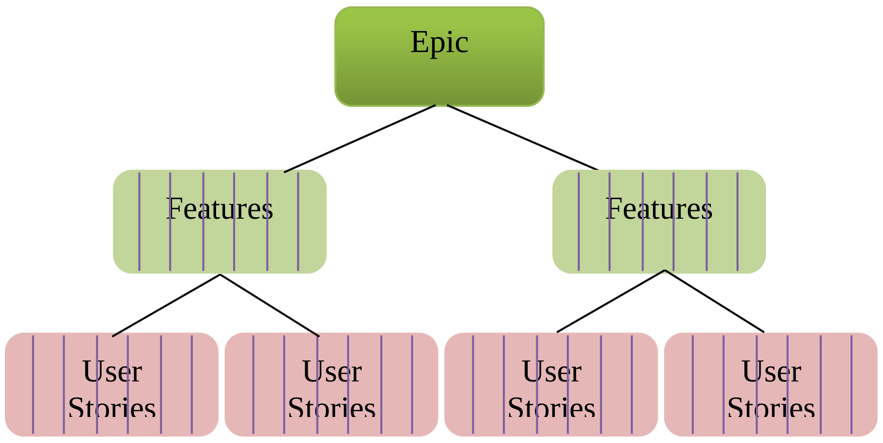


Figure 7: How a user story is broken down from an epic

Many agile teams attempt to make an estimation of the user stories size, to enable the planning and ensure the user stories will be completed on time. There are several ways to estimate user stories but one of the most common is working with story points. Story points is an estimation on the size of a user story where development complexity, overall efforts and risks are taken in consideration. The benefits of working with story points rather than man-hours is if for example the agile teams evolves and become more efficient, a re-estimation is not necessary since it is not based on man-hours. The only thing that is changed if a team becomes more efficient is the velocity in which it develops the user stories (Coehlo & Baso, 2012).

3.4.1 Extreme programming

Agile development is based on the work of empowered teams. Therefore, the work within the teams needs guidelines, Extreme Programming (XP) is one of the most used practices software development (Beck, 2000). The teams work in short iterations of usually one to two weeks, using 10 practices to ensure high quality code, see Figure 8.

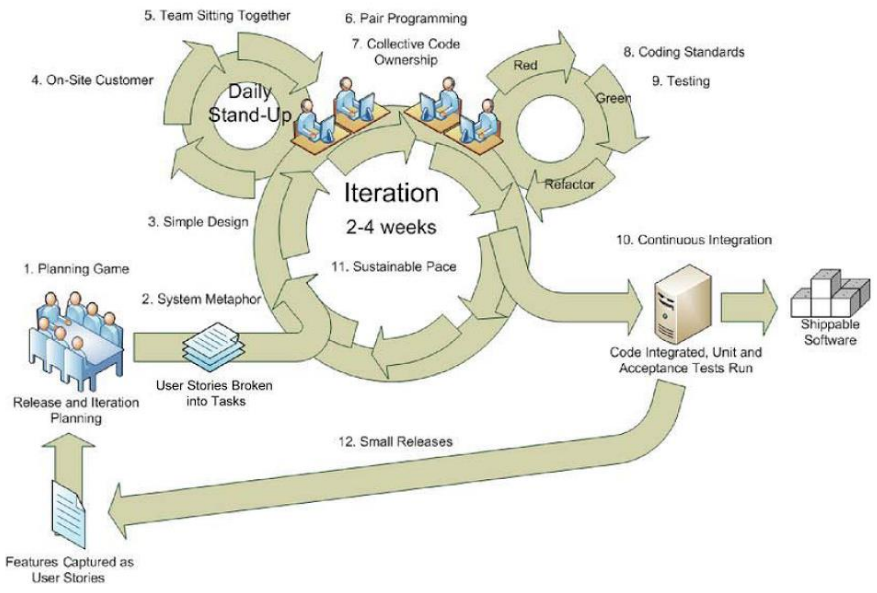


Figure 8: The practices of XP (Blankenship et. al. 2011)

Practices

Planning Game: This practice defines the features of the project as user stories and the release points for these stories are planned. When the backlog is planned, development task is broken down from the user stories selected for the iteration (Beck, 1999).

Small Releases: XP is trying to reach customer satisfaction and achieving business value by delivering quality software. To achieve this, releases must be made often anywhere from daily to monthly depending on the project (Beck, 2000).

System Metaphor: To communicate well with the customer a universal language is necessary in order to explain complex systems in an understandable way (Beck, 1999).

Simple Design: The code should be tested continuously and contain the fewest classes and method and no duplicate code. This can be summarized as "*Say everything once and only once* (Beck, 1999)"

On-Site Customer: It is important to have close customer collaboration and the ideal situation is to have a customer on-site to answer question regarding how the system will be used.

Team Sitting Together: It is important to be sitting together to efficiently receive help and feedback from the colleagues. This also increases the group dynamic and sense of camaraderie (Blankenship et.al. 2011).

Pair Programming: Pair programming is a practice that sets developers in pairs for them to work on a task together. They will use one computer there one of them will write code and the other one will look ahead to the next feature and assist with design decisions. This enables real time review of the code and provides a higher level of quality than inspection only. It’s less likely that two developers will overlook the same mistake (Blankenship et.al. 2011).

Collective Code Ownership: The code is open for all developers, meaning that they can fix problem or add functionalities in any part of the code. This is enabled through sitting together and by pair programming; this creates the sense of a collective ownership of the code (Blankenship et.al. 2011).

Coding Standards: To facilitate collective ownership best practices are used to ensure that all code is created in a consistent manner and to help keep the design simple. If everyone follows the same guideline the code will be easier to understand and make it easier to fix other developer’s codes (Blankenship et.al. 2011).

Testing: High quality code is one of the foundations in XP; this leads to testing being performed throughout the process. The first step in testing is to identify the acceptance criteria for the use stories, unit tests are then written to ensure that these criteria are fulfilled. Finally, the user acceptance testing is performed and is preferably automated as much as possible, these acceptance criterions are also derived from the story’s acceptance criteria (Blankenship et.al. 2011).

Continuous Integration: Testing parts of code is not enough to guarantee quality; the code also needs to be integrated to make sure that it works together in the system. Integration is performed continuously and early in the development process. This is done through an integration server where automatic tests are performed to check that the build is not broken. The results from these tests can be visually notified to the developers, managers and customers. This provides a daily tracking of progress and also ensures a quick feedback loop to enable the developers to quickly fix the bugs. The sooner a bug is fixed, the cheaper it is to fix it (Blankenship et.al. 2011).

Sustainable Pace: The short release cycles in XP leads to those key activities such as requirements gathering, design, development, testing and deployment all happen on a continuous basis. This provides the possibility to include the problems that are found into the next iteration. Because of the quick feedback the developers can work at a sustainable pace. More traditional development methodologies save testing and feedback until the end of the development cycle, which creates a higher pace during these periods (Blankenship et.al. 2011).

3.4.2 Scrum

Scrum is one of the most used frameworks in agile development and has been very successful in the software industry. Scrum was developed by two software developers, Jeff Sutherland and Ken Schwaber, in 1995 and in 2010 they wrote the Scrum Guide. The Scrum Guide is a 16 page “playbook” containing all the activities and rules of Scrum (Schwaber & Sutherland, 2016). The information in this following chapter about Scrum is based on the Scrum Guide by Schwaber & Sutherland (2016), unless stated otherwise. Scrum is not a tool or technique; it is a framework in which you can use several tools and techniques. Scrum is an iterative framework where the development is performed in short sprints instead of the traditional waterfall development. A sprint is between 1-4 weeks in which the product is planned, developed and tested in each sprint. Figure 9 illustrates the difference between traditional waterfall development and Scrum. This iterative development process provides fast feedback-loops for the developers which makes it easier to find bugs or errors early in the development

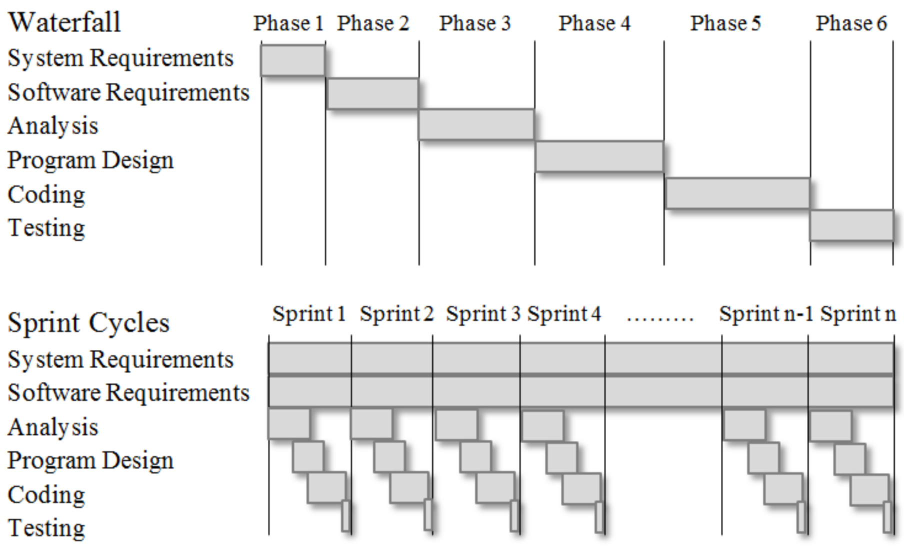


Figure 9: Scrum sprints compared to traditional waterfall development

The framework consists of Scrum teams, where each member has a specific role, and then there are several events and rules that must be followed to successfully play the game. A Scrum team consists of five to nine self-managing team members working as Developers. The teams are self-organized and cross-functional to the extent that every team should have the competence necessary to accomplish their tasks without external help. Working self-organized means that they are provided only the tasks, not directed how to perform the tasks. There are two other roles apart from the developers within the Scrum team, the Product Owner (PO) and the Scrum Master (SM). The PO is the link between the Scrum team and the key stakeholder during the projects. The PO is also responsible to bring new features or user stories to the Scrum team. The SM helps the team to self-organize and has no authority over the developers; the SM is also responsible to remove impediments/obstacles that occur during the sprint.

The Product Backlog

The product backlog is a list containing all features and requirements needed in the product; it also includes all bugs and technical work that needs to be performed. The list is constantly prioritised from most important at the top to least important in the end of the list. A product backlog is never complete, as long as the product exist, the backlog exists and is continuously updated to ensure the product stays competitive in the market. The product owner is responsible to update and prioritise the backlog and communicate the items on the list to the Scrum teams so everyone has a clear understanding on what will be performed. The product owner in collaboration with the Scrum team also frequently conduct *Backlog Refinement* which is an activity where they add new details to the product backlog and/or makes new estimations or re-prioritise the features. The Scrum team is responsible for scheduling backlog refinement and the activity only consumes up to 10% of the team's total work capacity.

The Sprint backlog
The sprint backlog consists of a set of items from the product backlog which are selected for the coming sprint. It is up to the Scrum team to decide and commit to how much work can be included in the sprint backlog. This is done through analysing the past sprint and how much the team managed to produce then. The features or user stories chosen are then described and broken down to smaller tasks by the Scrum team with support from the PO.

The Product Owner, PO
The PO is always only one person and is responsible for managing the product backlog and communicating it to all Scrum teams, essentially deciding what will be developed during the sprint. The PO is the link between the Scrum teams and stakeholder, ensuring the product backlog is prioritised according to the product's needs and that everyone clearly understand all items on the backlog and knows what will be included in the coming sprints.

The Scrum Master, SM
The SM is responsible for the Scrum practices, that the development team understand the rules, practices and rules. The SM often also has a role as one of the developers, working to reach the sprint goal. The SM has no authority over the developers, he acts more as a coach for the team, ensuring they have what they need to reach sprint goal and that no impediments has occurred. The SM is responsible for the speed of the team.

The Development Team
The development team consists of optimally 5-9 team members who all have the title of developer. Different members might have various skills and focus areas but it is still the team as a group that is accountable for reaching the sprint goal. The team is self-organising which means that no one, not even the SM, can tell them how to make the sprint backlog into a potentially releasable increment.

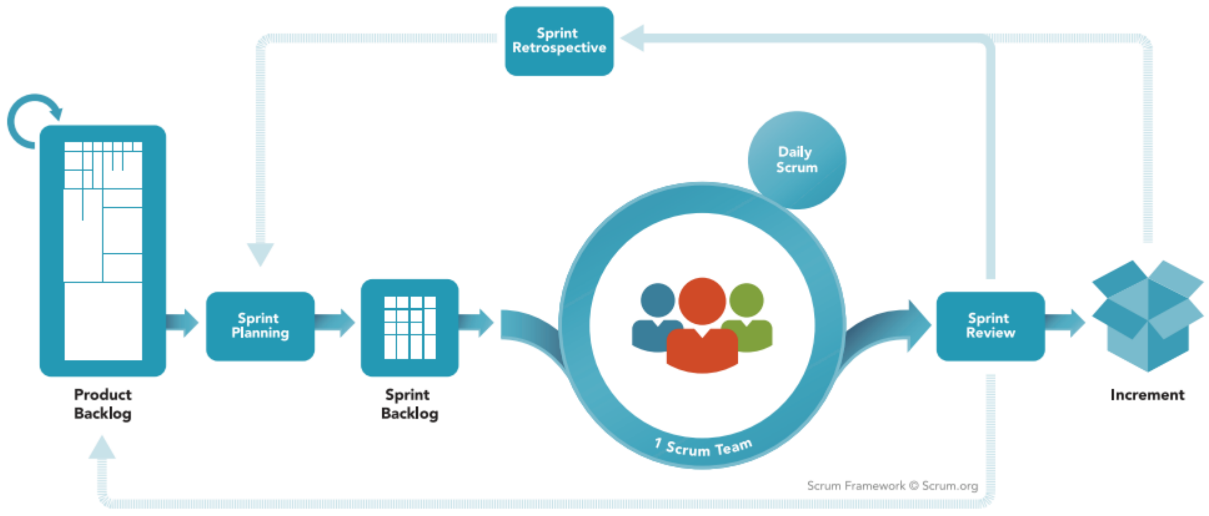


Figure 10: Illustration of the main events and artifacts in Scrum (Scrum.org, 2016)

Sprint

The main event in Scrum is the *Sprint*, which itself contains several events. The sprint is a 1-4-week event where all the stories chosen for the sprint backlog is developed. The sprint consists of a *Sprint Planning, The Development Work, Daily Scrum, Sprint Review and finally the Sprint Retrospective.* Figure 10 illustrates the Scrum Framework with all the events and artifacts in a sprint. The Sprint Planning is a (up to) eight-hour meeting where the Scrum team plans the coming sprint, breaks features or user stories down to smaller tasks and decides how the sprint goal will be achieved. The SM is responsible for making sure the sprint planning occurs, but is not responsible for the planning itself. The topics for the sprint planning are what can be done in this sprint? And how will de work get done? And by the end of the sprint planning the development team should be able to inform the PO and SM how they intend to self-organize the sprint.

As the Development work starts, each developer chooses a story to begin with. The development team's progress in a sprint is visualised with a *Scrum Board* to ensure sprint goal will be reached. Figure 11 illustrates a typical Scrum board where the stories are taken from the backlog and how the stories are broken down to small task and placed in the "To Do". As a developer begins with a task it is placed in "In Progress" and after development and testing is performed by the team it is moved to "Done".

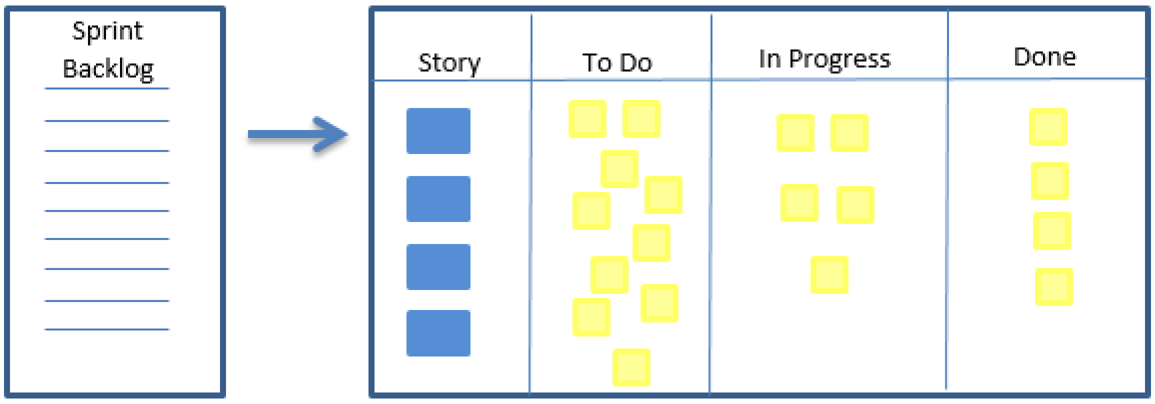


Figure 11: Example of a Scrum Board

Every day the Scrum Master calls for a Daily Scrum, this is a 15-minute meeting for the development team. In the daily Scrum every developer answers three questions.

What did I do yesterday to help the team reach sprint goal?

What will I do today to help the team reach sprint goal?

Has any impediments occurred that prevents sprint goal?

Sprint Review

At the end of a sprint, the PO invites the Scrum team and the key stakeholders to a *sprint review*. The sprint review is a (up to) four-hour meeting where the PO informs what items from the product backlog is "Done" and what is not "Done". This is not a formal meeting, it is an opportunity to get feedback and collaborate. The development team explains how the work has been performed, what problems they have encountered and how it was handled during the sprint. The PO explains the priorities in the product backlog and they all collaborate on what the next step will be, which will be a good input for the Scrum team's next sprint planning.

Sprint Retrospective

After a sprint review, a sprint retrospective is performed before the Scrum team begins the next sprint planning. This is an opportunity for the Scrum team to inspect themselves and reflect on how they can improve before the next sprint. The self-reflection regards everything from people to processes and tools used during the sprint. The Scrum team then tries to identify improvement possibilities and how they can implement these in the coming sprint.

Team Progress towards sprint goal

A common tool for a team to monitor its progress toward a sprint goal is by a burn-down chart. Figure 12 illustrates a typical burn-down chart where both the planned work pace and the team's actual work progress is visualised. This is so the team is ensured that they are on track and not falling behind in the on-going sprint. One of the most common ways to work with burn-down charts are to estimate the user stories or task using story points. This is a simple way for the Scrum team to visualise how many story points are completed relative to the sprints total amount of story points. Burn-down charts are often used both for individual sprints and a "main" burn-down chart covering the progress of the entire product backlog.

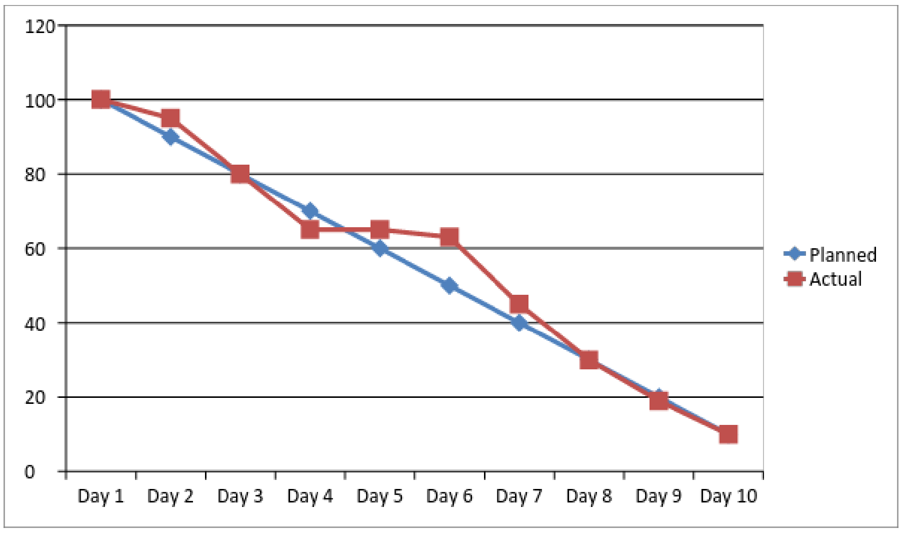


Figure 12: Example of a Burn-Down Chart

The Scrum Guide by Schwaber & Sutherland (2016) includes five values which need to be embodied and embraced by all team members for them to become a successful team. The values are: commitment, courage, focus, openness, and respect.

Commitment: All team members must commit to the sprint, and the sprint goal, during the sprint planning. The commitment is also to the team, if the members does not commit to the team, the team is not likely to succeed

Courage: The members must possess the courage to do the right thing and be open to changes, even if the team is going towards another direction then a member's personal opinion.

Focus: The team members need to be focused on the ongoing sprint and the sprint goal they committed to.

Openness: The team and the members in the team must be open to each other and stakeholders about challenges and obstacles which constrain them from reaching sprint goal.

Respect: There needs to be a mutual respect among all team members, there is no room for judgement when lacking capabilities, instead there should be a sharing of capabilities, teaching each other and sharing knowledge.

3.4.3 Kanban

Kanban is Japanese for "signal card" and the main idea is to visualise the work flow. As in Scrum, the work should be broken down to smaller tasks, where each task is written down on a small card. All tasks should then be visualised on a Kanban board or on the wall so everyone easy can see all tasks. Kanban does not work in short sprint or iterations, the work flows by a pull system where there always is a limited amount of work in progress (WIP) (Kniberg, 2011).

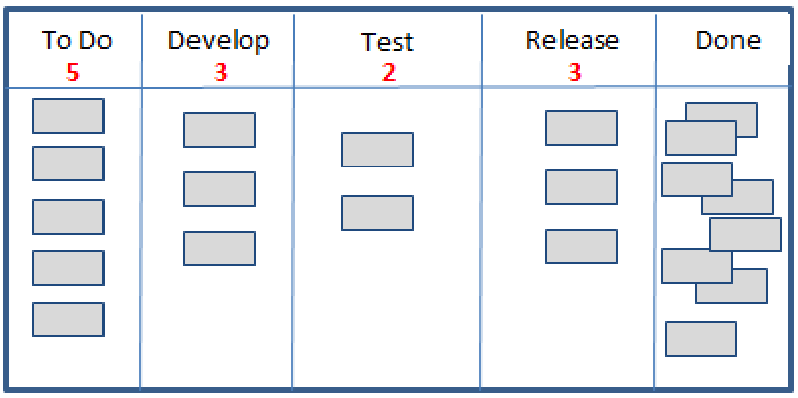


Figure 13: Example of a Kanban board

The contents on a Kanban board can vary to fit the project or organisation, Figure 13 illustrates an example of a simple Kanban board. As a task occurs it is put on a card and placed in the backlog and when it is prioritised it is moved to “To do”. When a developer begins a task, it is moved as “on-going” through Develop, Test and Release phases and when task is completed it is moved to “Done”. The red numbers on each slot is the limited WIP and there can never be more cards in a slot then the red number (Kniberg, 2011).

Kanban is based on six core elements: (Al-Baik & Miller, 2015).

- Visualize the workflow: so everyone can see what work is done to get a better understanding of how their work is put in context to what others are developing.
- Limit work-in-progress (WIP): the pull system is to ensure there won’t be an overflow of work for the developers.
- Manage flow: set up measures and metrics so the development teams know how much work they need to commit to and how long lead time they have over a longer period of time.
- Make policies explicit: All policies needs to be documented so it is easy for development teams to work smoothly, this could be for example how a specific decision should be made and by whom.
- Implement feedback loops: this promotes the learning for employees at all levels within the organisation. The feedback loops facilitate learning and it makes it easier to see the outcome from changes made within the process.
- Improve collaboratively: this is one of the fundamentals of Kanban, to continuously improve. The organisation must create a culture where everyone always tries to improve and evolve.

Kanban does not include specific roles or processes as in Scrum, but one can choose to include these roles if they are needed. When initiating working according to Kanban, the teams can be organised as they were before, they do not need to reorganise or assign new roles. The teams evolve and improve one step at a time, and they continue to do this, forever (Measey, P, 2015).

3.4.4 Summary Agile Methods
The three methods and frameworks explained have several similarities and some differences. Both Scrum and XP are focused around working in short iterations or sprints while Kanban uses a continuous flow through the development process. Scrum monitors efficiency and establish high quality by working with several events, artifacts and roles. XP uses guidelines and practices to establish this. Kanban monitors efficiency by measuring lead-time for the features but also ensures quality by having the developers focused on the “active” work, using the WIP-limit. It has also been discovered that it is common to use combinations or “hybrids” of Scrum and Kanban when working in agile teams. A hybrids can for example be that a team choses to use the continuous flow from Kanban but include some events or roles from Scrum.

3.8 Scaling Agile Development
There are three well established frameworks that are most commonly used when scaling agile development today. The frameworks are Scaled Agile Framework (SAFe), Large Scale Scrum (LeSS) and Nexus, they have been used by many organisations when attempting to scale agile development through the entire organisation and not just working with agility in the actual development work.

3.8.1 SAFe-Scaled Agile Framework
To be able to scale the agile methodologies several frameworks have been developed by practitioners, one of the most commonly known is the Scaled Agile Framework (SAFe) (Turetken, Stojanov, Trienekens 2016). The SAFe framework provides a structured way to scale agile development to several levels and departments in a development organisation. Since the SAFe-Framework has not been thoroughly investigated by the academic community the main source for this part of the framework will be from the company that has developed the framework, Scaled Agile.

Principles
A framework can usually not be adopted off the shelf, companies differ a lot from one another and therefore some tailoring is necessary when adopting a framework such as SAFe. To successfully adopt and customize SAFe it is important to understand its fundamental nine principles (Scaled Agile, 2016). The information in this following chapter about SAFe is based on the SAFe 4.0 Introduction (Scaled Agile, 2016), unless stated otherwise.

Take an economic view
To support decentralized decision making the strategy for incremental value delivery must be communicated throughout the organisation. This strategy should define the trade-offs between risk, cost of delay, operational and development costs.

Apply systems thinking
In order to properly adjust the SAFe-framework to your organisation a system thinking has to be applied. This means that problems that are experienced within the development project are

not a people problem but a problem within the system that the developers use to perform their work.

Assume variability; preserve options
Lean system developers try to maintain multiple design option further into the development process to use empirical data to narrow them down when this is available. This prevents the development project to lock into solutions that then become costly to change.

Build incrementally with fast, integrated learning cycles
Within the SAFe framework incremental development should be used, meaning that solutions are built in iterations. Each iteration should conclude in an integration that can be used to gain feedback of the system. This provides the fast feedback loops that are essential in agile development that allows the project to change direction if necessary.

Base milestones on objective evaluation of working systems
Each integration point should be used to evaluate the system in terms of financial, technical and fitness-for-purpose governance.

Visualize and Limit WIP, Reduce batch sizes and manage queue lengths
There are three keys to achieve a continuous flow of new capabilities being developed. First, limit demand to actual capacity by visualizing the amount of work-in-progress. Second, reduce the amount of work in each work item to achieve a more reliable flow. Last, reduce wait times by managing the queue lengths.

Apply cadence, synchronize with cross-domain planning
Integration is a key factor and to be able to integrate the rhythm and synchronisation of the project must be handled. Setting a cadence for the development and handling planning activities cross-functionally will enable for integration to be synchronized.

Unlock the intrinsic motivation of knowledge workers
To reach a higher level of employee engagement Lean-Agile leaders must provide autonomy, mission and purpose and minimize constraints for the teams.

Decentralize decision making
A decision that is not strategic, global in nature or has economies of scale is best handled by the people affected by the decision. A decentralized decision making can reduce delays, enable faster feedback, improve the flow and provide more innovative solutions. Therefore, it is important to construct a decision making framework for the organisation to rely on (Scaled Agile, 2016).

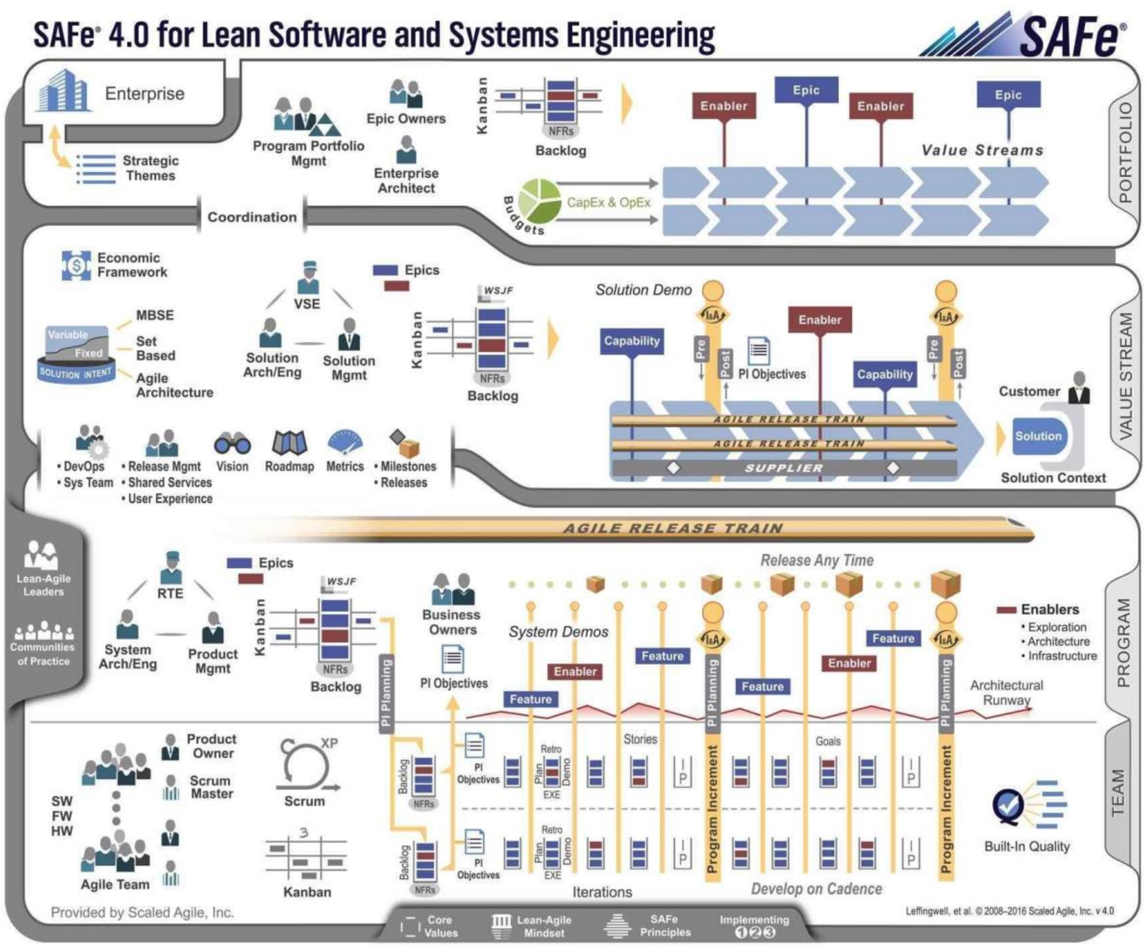


Figure 14: An overview of the Scaled Agile Framework (Scaled Agile, 2016)

The framework is divided into four levels: Portfolio, Value stream, Program and Team, see Figure 14. To understand how the framework is used all levels will be explained.

The Portfolio Level

In the portfolio level the organisation is centred on the flow of value in one or several value streams. It also provides governing tools and budgeting to assure that the value streams meet their strategic objectives. A value stream is a long-lived program that develops and delivers a continuous flow of value to the customers. The main actions of the portfolio level are handled by the Program Portfolio Management (PPM), they handle the funding of the value streams. The budgeting is performed with a Lean-Agile budgeting approach which provides fast decision-making.

Another main responsibility of the PPM is the administration of major development initiatives that can provide new capabilities or are cross-functional between value streams. These initiatives are handled in the Portfolio Kanban system. The input to this Kanban system can come from the strategic themes of the organisation, market demand, need for cost savings or many other places. This Kanban board is not limited by work in progress, instead all Epics will be considered. The PPM then reviews these ideas together with stakeholders both within and outside the value streams. The epics that receive an approval are added to the portfolio backlog where they wait to be implemented. The portfolio backlog then serves as an input to the Agile Release Trains and the value streams where they are presented at the Program Increment Planning meeting.

The Value Stream Level

The value stream level is an optional level, it is only required for complex solutions and large system with multiple agile release trains. It is not necessary for systems that are independent or can be built with a few hundred people. The main purpose of this level is to build the solution for the customers. To handle this the solution intent must first be specified. The solution intent is a way to communicate the future state of the product and uses tools such as the product roadmap and vision. This is broken down into capabilities that are administered on the value stream Kanban board. Another important task in this level is to synchronize the releases of several Agile Release Trains (ART). The planning for each increment is performed within the ARTs but to align them and create a single plan across all train value stream PI objectives are constructed. In the end of the increment a demo of the solution should be performed to the stakeholders to receive feedback.

The Program Level

The program level organizes the resources, mainly development teams, to ongoing development projects. It is connected to the portfolio level through the stakeholder needs, which should be reflected in the vision and the roadmap of the program. The work within this level is centred on the agile release train. The ART is a self-organising team of agile teams which delivers the primary value to the project, see Figure 15. It can be seen as a virtual organization that delivers a continuous flow of incremental releases. Therefore, it becomes the program levels responsibility to manage the flow of these releases. This is done in Program Increments (PIs) of 8 to 12 weeks, during each PI several deliveries from each team contribute to a release with features according to the vision and roadmap of the project. The specifics regarding each PI is communicated in the PI objectives, which are developed in collaboration with the teams. In case a value stream level has been implemented, the roadmap and vision must be developed in collaboration with the other ARTs to synchronize deliveries. The planning for each program increment is performed during a two-day meeting where the ART meets face-to-face. During this meeting the capabilities that are going to be developed and dependencies between teams are identified.

One of the primary ways this is done is through the development of the enablers and features that are required for the realisation of the vision and roadmap. These are put in a common backlog that is then handled by the system Kanban board. To make sure that the roadmap and vision is followed, a collaboration is needed between the Troika consisting of the Release Train Engineer, System Architect/Engineer and the product management. The Release train engineer acts as the Scrum Master for the release train and which means that he optimizes the flow of value through the program the flow. The product manager is the internal voice of the customer and he adds the functionalities that needs to be developed into the backlog. Product management is also responsible for defining and validating the system features. The system architect should adopt a system thinking and define the architecture for the system. This includes non-functional requirements and defining the subsystems, including the interfaces and collaboration.

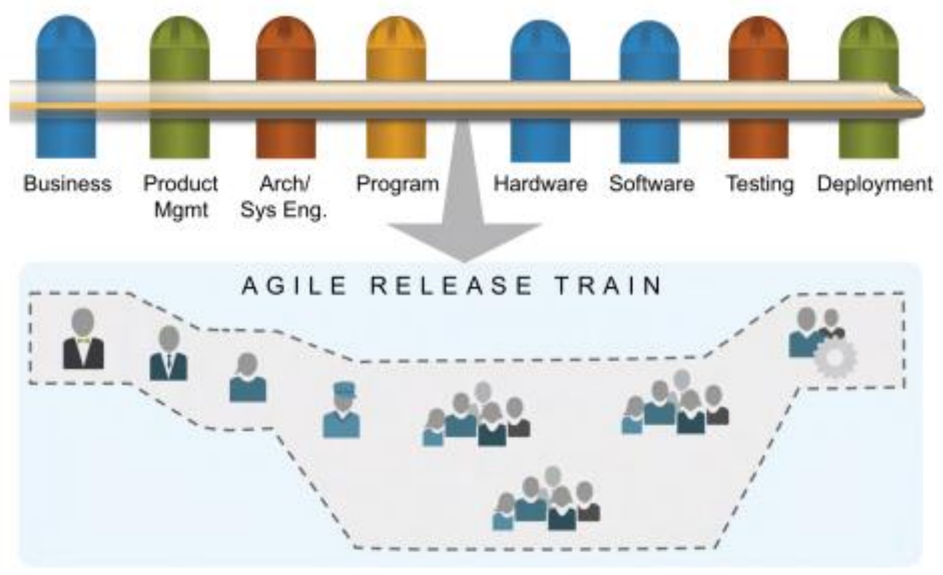


Figure 15: An agile release train with members from all part of the organisation, (Scaling Agile, 2016)

The Team Level

The team level consists of agile teams working together within the Agile Release Train. These teams work according to Scrum or Kanban in sprints of 2 weeks. At the start of each PI the team participates in the PI Planning. This is one of the most important events when working in the sprints, it is a face-to-face meeting between all members of the program. Here the roadmap is broken down into the PI objectives for each team, these objectives are then aggregated to the features which become the Program PI objectives. A vote of confidence is given to the program objectives from all program members to ensure that the objectives are feasible. The Team PI objectives are then used as the input for the teams to create their stories.

Each two-week sprint should provide a valuable increment of new functionality to the product. This is ensured through a series of steps that is performed in each sprint. First the sprint is planned to commit to the functionality that will be developed. Thereafter the stories are developed and tested, followed by a demo of the functionality. When all stories are finished, a retrospective is performed to further improve until the next sprint. At the end of the sprint a system demo is performed which is a critical integration tool for the ART. In the end of a PI there is an Innovation and Planning (IP) sprint. This sprint allows the team members to plan, reflect, improve on their work and also acts as a buffer. In case there is a system release this also provides time for system verification, validation and documentation is performed to release the functionality. This means that each PI is not planned at full utilization which increases flow, throughput and delivery reliability.

Implementing SAFe

To implement SAFe, Scaled Agile suggest a three-step adoption. First the change agents and implementers will be educated regarding how to lead an agile transformation and how to launch agile release trains through inspection and adaption. This provides the change leaders with the tools to evaluate the organisation and create a vision of how the organisation will look in the future. Second, the management and executives is educated in how to adopt an agile mindset and the different principles and practices in the framework. This creates a buy-in from management and allows them to help the change agents in creating the necessary environment for a successful implementation. The last step is to train the teams especially regarding how to work within the release train and explains the Scrum master and product owner roles. The first PI planning meeting is also performed to directly be able to start working in the framework. When the work starts, it is important for management and the change leaders to inspect the organisation and quickly adapt to avoid problems.

3.8.2 Large Scale Scrum

Large Scale Scrum, LeSS is a framework used for scaling agile development or scaling Scrum and was established by Craig Larman and Bas Vodde. It is well established in organisations which have multiple Scrum teams or complex developments and is used to manage and reduce the complexity (Larman & Vodde, 2010).

The LeSS framework is divided in two different structures, called the LeSS and LeSS Huge, where the first is appropriate for smaller organisations and Less Huge for larger more complex developments. The information in the following chapter about LeSS is based on Scaling Lean and Agile Development by Larman & Vodde (2010). The first framework includes only one product owner and one product backlog for the entire product. The second one is very similar but also includes “area product owners” with area product backlogs.

LeSS Framework 1

The first framework promotes only using one PO responsible for the entire product and up to ten Scrum teams. The framework can be used for more than ten teams, but only to the extent that one single person can have an overview of the entire product and can interact with all teams. The PO has the same responsibilities as in traditional Scrum, the only difference is that handling many Scrum teams means more features in the product backlog and therefore more work for the PO. As said above, in the LeSS framework there is only one PO and one product backlog. If the workload is too high for a single person, the PO can choose to either delegate part of the work to the teams, or use the second framework (Larman & Vodde, 2010).

The Scrum team and the SM also have the same structure and responsibilities as in traditional Scrum. Each team has their own Sprint backlog, consisting of stories from the product backlog.

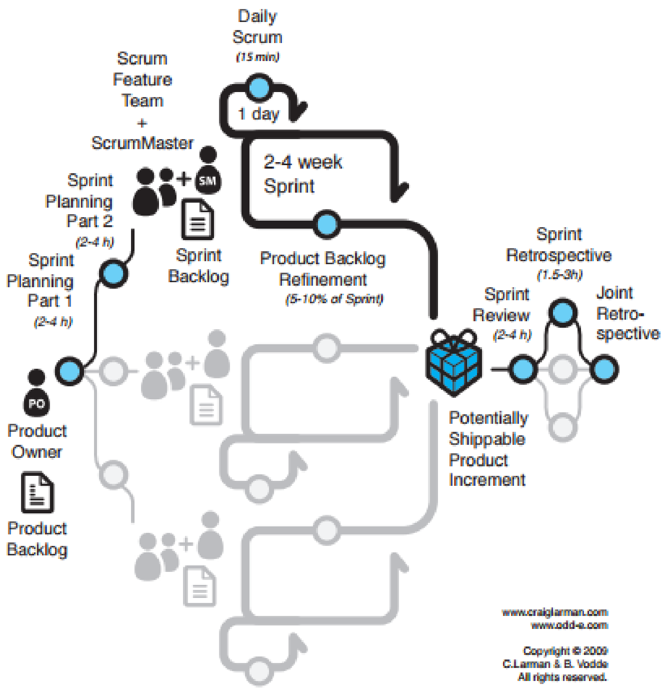

Daily Scrum (15 min)
1 day
Scrum Feature Team + ScrumMaster
SM
2-4 week Sprint
Sprint Planning Part 2 (2-4 h)
Sprint Backlog
Product Backlog Refinement (5-10% of Sprint)
Sprint Retrospective (1.5-3h)
Sprint Planning Part 1 (2-4 h)
Sprint Review (2-4 h)
Joint Retro-spective
PO
Product Owner
Potentially Shippable Product Increment
Product Backlog
www.craiglarman.com
www.odd-e.com
Copyright © 2009 C.Larman & B. Vodde All rights reserved.

Figure 16: Schematics of LeSS, Framework 1 (Larman & Vodde, 2010)

Figure 16 illustrates the roles and events of LeSS, from PO and product backlog down to shippable product and retrospective. The sprint planning in the LeSS framework consists of two parts, sprint planning part 1 and 2.

Part 1 is a two to four-hour session where the PO and representatives from all Scrum teams attend. During this planning, they go through the product backlog and decide which features needs to be done in the coming sprint and teams can volunteer which features they would like develop. At the end of this planning all features should be divided among the teams as equally as possible.

Sprint planning part 2, all Scrum teams have their own sprint planning and is conducted in the same ways in traditional Scrum planning. The sprint review is also similar to traditional Scrum, but in addition to PO it also involves members or representatives from all teams. The sprint retrospective can be done the traditional way, but it is suggested to also include a joint retrospective. The joint retrospective is held after the team's retrospective so all teams can improve together. If there are many teams the joint retrospective can be performed with representatives from each team, preferably SM and one or two other team members (Larman & Vodde, 2010).

LeSS Huge Framework (Framework 2)

When using more than ten Scrum teams it is often difficult for a single PO to be responsible for all teams and the product backlog. The second framework recommends the identification of the products major requirement areas. The requirement areas then have their own product area backlog, consisting of features from the product backlogs. Each area has an Area Product Owner, APO, which is responsible for the Scrum teams working in the specific area. Figure 17 illustrates the schematics of the framework including the new role APO. The PO and all APO's together form a "Product owner team", and together they are responsible for all product area backlogs (Larman & Vodde, 2010).

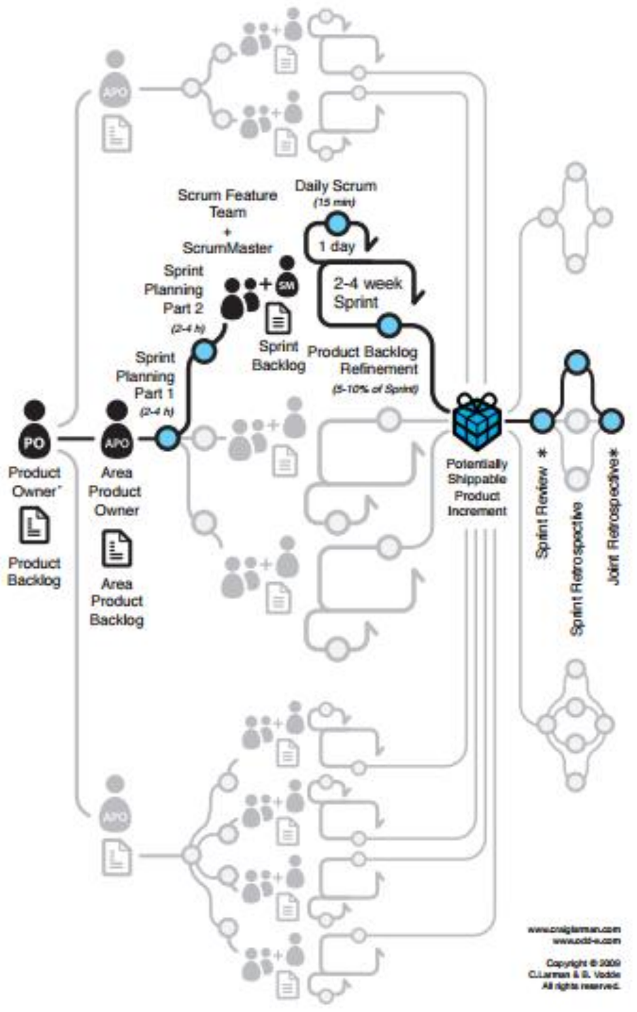

Scrum Feature Team + ScrumMaster
Daily Scrum (15 min)
1 day
2-4 week Sprint
Sprint Planning Part 2 (2-4 h)
Sprint Planning Part 1 (2-4 h)
Sprint Backlog
Product Backlog Refinement (5-10% of Sprint)
Product Owner
Product Backlog
Area Product Owner
Area Product Backlog
Potentially Shippable Product Increment
Sprint Review
Sprint Retrospective
Joint Retrospective
www.craiglarman.com
www.odd-e.com
Copyright © 2009 C.Larman & B. Vodde All rights reserved.

Figure 17: Schematics of LeSS Huge, Framework 2 (Larman & Vodde, 2010)

Pre-sprint planning

Before each Scrum team begins their sprint planning, the "product owner team" has a pre-sprint planning. In the pre-sprint planning the team needs to prioritise the product backlog, and the area product backlogs. This so the APO can discuss the progress for all teams together and decide what is to be prioritised in the coming sprints so the product is growing as a whole.

The second sprint planning is separate for each requirement area where members or representatives from each Scrum team within that area attend. The sprint planning is performed in the same way as in the first framework, in two parts, one for prioritising and dividing the features among the Scrum teams, and one individual planning for each team.

The sprint reviews are held within each requirement area, including members or representative from all Scrum team. It is optional to use joint reviews and retrospectives for all teams in the requirement areas, using a few representatives from each team. The joint reviews are good if the Scrum teams have finished features affecting other teams or areas so they can be discussed together and everyone gets an overview of the progress. The same goes for the joint retrospectives, they are good to use if requirement areas or Scrum team has issues or improvements regarding the overall product level.

3.8.3 Nexus

Nexus was developed by Ken Schwaber and Scrum.org to be able to manage several Scrum teams working together with the same product. The framework is based on the roles, artifacts and events from Scrum, which are illustrated in Figure 18. The following review is based on the nexus guide produced by Scrum.org (2015).

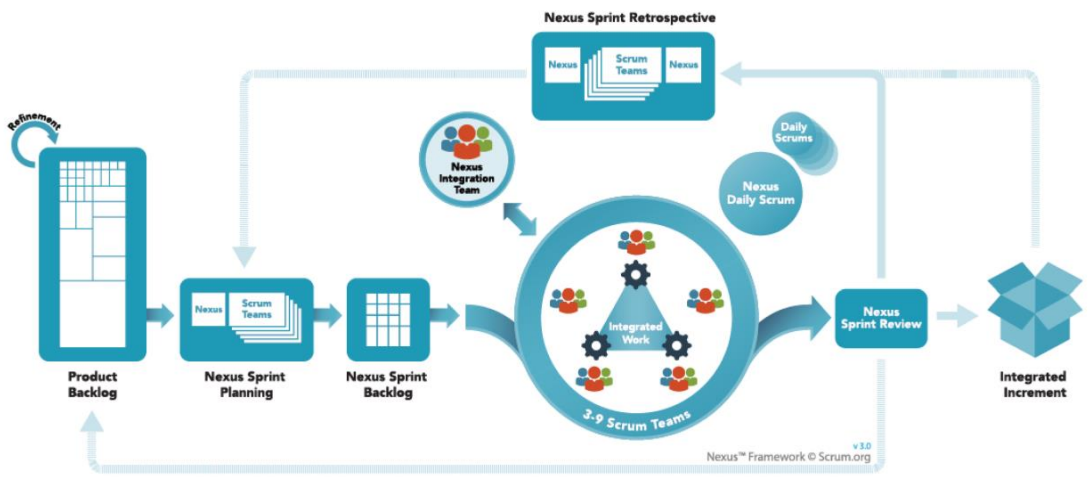


Figure 18: The Nexus Framework (Scrum.org, 2015)

Roles

There is one new role introduced within the Nexus framework, the Nexus integration team. The teams shall make sure that an integrated increment is delivered at least once every sprint. The team have a product owner, a Scrum master and team members. These can also work within the Scrum teams but must prioritize the work in the integration team. Tasks that are performed by the integration team are coaching, consulting, mapping of dependencies and resolving cross-team issues. They are also supposed to assure a successful integration of the work performed by the Scrum teams and expected to handle integration issues.

There is only one product owner within a Nexus since all work is performed from one product backlog. The Product owners responsible for the product backlog and makes the final decision regarding its contents. The target for the product owner is to maximize the value from each integrated increment that is produced by the nexus.

Events
The events in the Nexus are correlated to the events in Scrum and will help both the overall effort and the effort of all teams and the individual teams.

The Nexus sprint planning is a meeting to coordinate the activities in a sprint between the different Scrum teams. During the meeting the product owner makes the priority decisions and guides the selection of activities. All member of the teams should attend the meeting to reduce the risk of communication issues. One of the goals of the meeting is to create the Nexus sprint goal which describes the purpose of the activities in the sprint. When this is understood, the teams will perform their own sprint planning and if new dependencies are found these shall be communicated. The product backlog should be refined prior to the Nexus sprint planning

Nexus daily Scrum is a meeting which is attended by representatives from all teams. The meeting is focused on the different team's impact on the integrated increment and information that needs to be shared between teams. The purpose is to identify cross-team dependencies and integration issues. Activities that have been identified during the meeting will be brought back to the individual team's daily Scrums.

After each sprint a Nexus sprint review, which replaces the team sprint reviews, should be performed. The purpose of the review is to capture feedback from the key stakeholder for the integrated increment produced. A Nexus sprint retrospective should also be performed to inspect and adapt the internal processes. The retrospective consists of three parts, first representatives from all teams meet to identify issues that have impacted more than one team. Thereafter the teams perform their own team retrospective according to the Scrum framework. Lastly the representatives meet again to decide how to track and visualize the decided actions.

To be able to worked in a scaled agile environment the items in the product backlog must be adequately independent. This means that the story can be worked on without conflict between the different teams. To keep the items in the product backlog independent refinement is necessary. Refinement aims to decompose the items in the backlog down to enough detail to be understandable by the developers and to define the sequence of the actions. Dependencies also must be identified and visualised so that the teams can allocate their work to minimize cross-team dependencies.

Artifacts
There is only one single product backlog for all teams and each team has a sprint backlog for their work in the sprint. To help the teams see what the other teams are doing a new backlog called the Nexus sprint backlog is introduced. This backlog contains all items of the individual team's sprint backlogs.

The Nexus goal is also introduced which is the goal for the entire sprint. This is the composition of each team's sprint goal and thereby contains all the functionality developed in the sprint. This functionality is released to the integrated increment which is the sum of all integrated work. The integrated increment should be a releasable product. To ensure that the product is usable the nexus integration team sets up a definition of done. This definition must then be applied by all teams to their own deliveries to release the function to the integrated increment.

The Nexus framework is built on transparency. This means the integrated increment, product backlog and stories should be visualized and understood by all member of the nexus. A decision made in the nexus will only be as good as the transparency, if all information is not understood it could lead to a faulty decision. The lack of complete transparency makes it impossible to minimize risk and maximize the value of the Nexus.

3.8.4 Summary Scaling Agile Frameworks

The three agile frameworks presented in this review has been chosen to represent different approaches to scaling agile, more frameworks do exist but share similarities with the ones presented above. The SAFe framework has the most defined processes and requires more roles than the other frameworks. This do provide the possibility to scale it up to top management and thereby create an agile enterprise. LeSS does not enable the possibility of creating an agile enterprise; however, it is not as rigid as the SAFe framework which allows for more freedom. This could create more agile teams since they will be able to focus more on the first agile value: *Individuals and interactions over processes and tools.* By using the LeSS Huge framework it is still possible to encompass an entire development organisation regardless of size. Nexus is not as suited for large development organisations with dependencies between departments. However, it is a lightweight framework which focuses on enabling team efficiency with less synchronisation and planning activities. This makes it suitable for smaller development organisation. This study has not found any process to choose between frameworks, therefore this will be further studied.

3.9 Organizational Culture

Iivari and Iivari (2010) claims that one of the main difficulties when implementing agile methodologies is the organisational culture. The culture should correspond to the agile values and allow for an agile mindset. There are several frameworks in use today that tries to categorise the culture in an organisation. In this study the framework developed by William Schneider will be used.

Organisations seeks success and formulates and implements their fundamental method of operations which will turn into the organisational culture. The definition of organisation culture can be defined as the way an organisation operates to reach success (Schneider, 1994). Schneider (1994) defines core culture as the centre of an organisation’s culture, what drives the other expressions of culture. He claims that there exists four such core cultures, see Figure 19, and although they are all reflected in organisations, one or two is usually dominant.

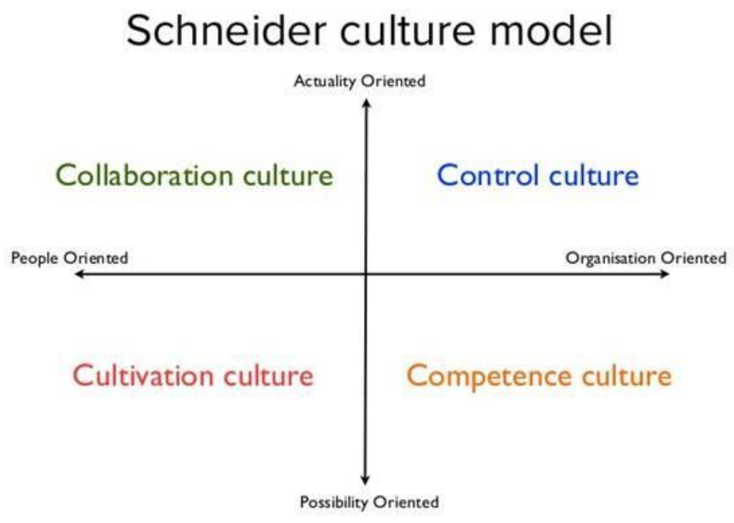


Figure 19: Schneider's culture model (Schneider, 1994)

The Control Culture

An organisation with a control culture aims to be successful by having more control than the competitors. This can be found in military organisations and is also common within companies who strive for market domination, to be the only player on the market. One common effect of this culture is the fight against vulnerabilities, a lot of effort is put towards security and feeling secure. The individual motivation within this culture is the need for power since power enables the control of a person, situation or organisation. Within an organisation with such a culture dominance is valued among the leadership and they control the workers through this. The structure in an organisation with control culture is hierarchical and bureaucratic. The customers and suppliers of these companies often must adapt and play by the control cultures rules. Some of the benefits of a control culture is that it is effective at planning and making people feel safe since they know what is expected of them. The weaknesses are that it can create dysfunctional competitive behaviour and make people hesitant to share bad news because they are afraid of the ramifications (Schneider, 1994).

The Collaboration Culture

The collaboration culture is strongly influenced by sports, success is claimed by developing and making use of an effective team. An organisation with this culture tries to gain synergies within their work, meaning that the effect of two persons work together is greater than their individual effects. The collaboration culture tries to utilise diversity through creating teams with different backgrounds and capabilities and letting them work towards a desired goal. The leadership in this culture is focused on team building and fostering cooperation. Another important task of the leader is to create commitment to the organisation and that the people feel ownership of their work. The structure of a collaboration culture will in reality be cluster-like no matter how it looks on paper. The collaboration culture fits in an organisation with close relationships with their customer where they work together to incrementally reach their goal. The strengths of the collaboration culture are that it is effective at managing diversity, handle conflict and increase communication. The weaknesses are that it can get too supportive and people don’t hold each other accountable and can also lead to group-think there no one wants to differ (Schneider, 1994).

The Competence Culture

The competence culture values expertise and advancement of knowledge, it is a culture of ideas, concepts and technologies. To gain success within a competence culture the idea is to be superior to the competitors, to create a product that is unequalled in the market. Therefore, the organisation keeps building expertise in order to have better knowledge than their competitors. The leadership in a competence is focused on setting standards and establishing visions for the work. Leaders are often visionaries or architects and one of their main issues is to be able to sell their vision to the employees. The competence culture is often a meritocracy where your merits is what will give you respect, raises and promotions. The structure in a competence culture is less important, instead leaders have an attitude to organise according to the current conceptual system. Task forces are common within this culture where teams can be formed, solve a problem and then be disbanded. Competence culture is suitable for organisation that have created market niches for example state-of-the-art or one-of-a-kind products. The strengths in a competence culture is the high-performance standards and the institutional knowledge it creates. The weaknesses include over analysis and over planning as well as creating a stressful environment for employees (Schneider, 1994).

The Cultivation Culture

The cultivation culture creates an organisation of faith. The people in the organisation will believe that they accomplish their goals. The success is created because the organisation expected it and put trust in the people. Within in these organisations the purpose is of importance, the employees believe their work is valuable to themselves and others. When the mission is realised it has also fulfilled a purpose for both the customer and the employees. This culture creates commitment because they feel both individual and collective fulfilment with their work. Leaders within a cultivation culture should act as catalysts to stimulate the ambition to reach a certain goal. They should be responsive and listen to ideas to harvest these for further development. Leaders must also see the possibilities in all scenarios in order to get the organisation to work towards one vision. The structure within a cultivation culture is in reality circular. This means that people are allowed to interact with anyone and that most activities are decentralised. The cultivation culture fits best within organisations that try to make people and the world better, mainly products that try to accomplish higher order purposes for its customers. The strengths of the cultivation culture are that it builds commitment and offers opportunities for growth in their employees. The weakness is the lack of direction and that it is prone to be inefficient (Schneider, 1994).

For an organisation to evaluate their core culture Schneider (1994) created a survey that will aid in the evaluation. The survey asks the respondent to describe different events and values within the organisation where an alternative corresponds to one core culture, see Appendix E. Most organisations that has existed for more than 50 years will find one of the cultures receiving more than 50 % of the answers. If one core culture cannot be identified it is possible that different cultures have been created in various parts of the organisation.

3.10 Effective teams

Teams are the heart of agile development, or any kind of development process, and creating effective teams is a challenge for many organisations. In agile development, many frameworks and methodologies advocates the importance of establishing and maintaining effective and self-organised teams. There is a difference from working in a team and as a group of individuals. A group of individuals have a common goal in which they provide part of the solution as individuals and has individual responsibility areas (Wheelan, 2015). A team has a common goal but they also share the vision to reach the goal and has established effective methods to reach the goal together. When a group transitions to a team they become more productive and effective, but studies also show that the members feel more engaged and appreciated. S. Wheelan (2015) has established a framework for how to transition from group to team by working with four stages including how the members, leaders and the organisation can act to support each stage to transition towards the following.

Stage 1 Dependency and inclusion: In the first stage members depend a lot on the team leader and relies on his/her directions. In these groups the productivity is rather low and the members work more individually to reach goals. Identification markers for these groups are personal security that members feel they have acceptance from others, they need direction from the leader and has no clear picture of the group's goal. These group has very limited amount of conflicts among members and subgroups are rarely created (Wheelan, 2015).

Stage 2 Counter-dependency and trust: When a group reaches the second stage members begins to disagree among themselves about group goals and procedures. These conflicts often regard the group's differences in norms and values and how tasks are performed. These conflicts are necessary for the group to establish trust among themselves and create an environment where members are free to share opinions and disagree with each other. Some group get stuck in conflicts, creates obstacle to reach the next stage, if the conflicts get personal the trust among members will disappears. Conflicts are crucial to establish teams, and only by the resolution of the conflicts the team will develop a common vision of the team goal and establish a genuine collaboration. Once the trust increases within the group they usually try to separate from the dependency to leader. Identification markers for these groups are conflicts and disagreement about tasks and procedures and that the group begins to establish common goals and roles. In this stage subgroups are usually created, causing more conflicts and intolerances among members (Wheelan, 2015).

Stage 3 T*rust and structure:* When a group reaches stage 3 the members are beginning to create positive work relationships. If the team has resolved many of the conflicts in the previous stage, the members trust towards each other increases. These groups are more engaged and are better at cooperating with each other. The communication will be more open and task oriented, providing room for members to express their opinions. The members have more mature negotiations regarding their roles and the procedures to reach the group's goal. Identification markers for this stage are that roles and tasks are adapted to fit the group goal and the leader is less controlling and functions more as a coach or consultant for them. There can still be some subgroups within the group but they collaborate more. Once group's reaches this stage members tend to be more satisfied with the work environment and more engaged in the group (Wheelan, 2015).

Stage 4 Work and productivity: In the final stage the group has made the transition to team and has solved most issues from previous stages and are now working with more productivity and effectiveness. They are functioning more as a unit and share the responsibilities to reach the team's goal. In this stage, there is also a shared vision and understanding of the team's purpose and goal where the members make informed and sound decisions together. The tasks performed in the team rely on team effort and not individual contributions and the leader delegates many responsibilities to the team (Wheelan, 2015).

Ten keys to productivity for high-performance teams.
According to S. Wheelan, there are ten keys to establish high-performance team and increase their productivity, these are presented below.

Goals
Most important is that all member have a clear view of the team goal, but even if they have the same goal there might be differences about how to reach them. This is what separates high-performance teams apart, they agree on a team goal and how to best reach it.

Roles
Once the team has agreed on the goal they must decide what need to be done to reach the goal and by who. Each member must know what their role is, be sure they possess the skills necessary to fill the role and feel comfortable and accept the role they taken.

Mutual dependencies
In a team with mutual dependencies the task to reach goal are established so it cannot be performed by individual efforts, only by team effort.

Leadership
In a high-performance team the leadership is flexible and adapts when necessary to meet the team's needs. In a high-performance team the members take responsibility for some of the leader's responsibilities in the earlier stages and for this to work the leader must leave some of the control to the members and function more as a coach or consultant for the team.

Communication
In high-performance teams, there is a very open communication where everyone has a right to express their feelings and opinions. The members often provide each other with constructive feedback on individual performance and accomplishments and use it to improve.

Planning and decision-making
High-performance teams put an effort in planning how work is to be conducted and how decisions are to be made, before they make decisions. These teams also spend time on discussing big decisions or problems before a final decision is made, so all member have the opportunity to express their opinion and everyone has an understanding about the issue.

Implementation and evaluation
When a decision has been made, high-performance teams take action and implement the decision. The also follow up and evaluate the results of the decision and make corrections if necessary.

Norms
High-performance teams have established norms to encourage success and quality. It is difficult for a team to be high performance if the members are not expected to perform.

Structure
High performance teams are structured in as small teams as possible, but as large as necessary to be able to perform the tasks to reach goals. Teams with about six members have much higher productivity than larger teams. These teams are also dedicated to one team for a longer period of time, and studies show that it takes about eight to nine months for a team to become successful as a team.

Collaboration
High-performance teams often function as a unit where everyone cooperates to reach goals. A high-performance team does not become successful by having one or two highly skilled members, they are successful because they collaborate and work as a unit.

3.11 Implementing change in an organisation

Implementing agile methodologies in an organisation is a big organisational change and the transition needs to be thoroughly prepared and maintained. Kotter (2007) created an eight-step model describing common pitfalls that can occur in this type of organisational change and how to avoid them.

The eight steps are:

1. *Establish a sense of urgency,* this must be established in the very initial phase of transition. If management does not communicate why the organisation must change the personnel will not be motivated and accept the change.
2. *Forming a powerful guiding coalition*, management must establish a core group responsible for the change. The core group are motivated and engaged in the change and has the power and support to lead the organisation through the change.
3. *Creating a vision*, a clear vision of the goal and objective for the change must be established with a strategy for how to successfully achieve the goal.

4. *Communicating the vision*, the vision and strategy must be communicated to all personnel affected by the organisational change to avoid resistance due to lack of information.
5. *Empowering other to act on the vision*, change structures and systems to remove obstacles for the change. Encourage the people in the organisation to come with suggestions and non-traditional activities that can benefit the change.
6. *Planning for and creating short-term wins*, establish clear short-term wins for the organisation. Plan and measure small performance improvements and celebrate and reward the personnel when they are fulfilled.
7. *Consolidating improvements and producing still more change*, if system and processes is hindering the vision they must be removed. If the personnel within the organisation is lacking in power or skills to implement the vision the organisation must develop, promote or hire personnel who can.
8. *Institutionalizing new approaches*, communicate the relation between the organisational success and the new behaviour that has been incorporated.

Following these steps, or similar, before and during the transition is crucial for a successful implementation of agile methodologies or frameworks.

4. Empirical Data

To scale the agile development techniques in the baseline program the current situation regarding how they organise their work and problem areas had to be identified, in order to know what was needed to be changed in the organisation. The first section of this chapter consists of the information found in interviews conducted at the baseline program at Saab. The second section consists of the information found in the investigated case studies.

4.1 The Baseline Program

As earlier stated, the baseline program was founded in 2014 to create a base product which could be customized for different customers. The main reason behind the founding was to avoid duplicate tasks being performed in different projects and perform all development of the base product in one program. When the base product is developed, the program will sustain and maintain the product. During its founding, it was decided that a program manager was going to lead the program instead of a project manager. The main difference between these two roles is that the program manager will not only focus on what shall be delivered but also on how the development is performed. The day-to-day operations is headed by subprogram managers which together with the program manager is called the operative management. The program manager reports to the Sponsor who is supported by the external steering committee, which consists of leaders within the organisation. An overview of the baseline program's schematics is illustrated in Figure 20.

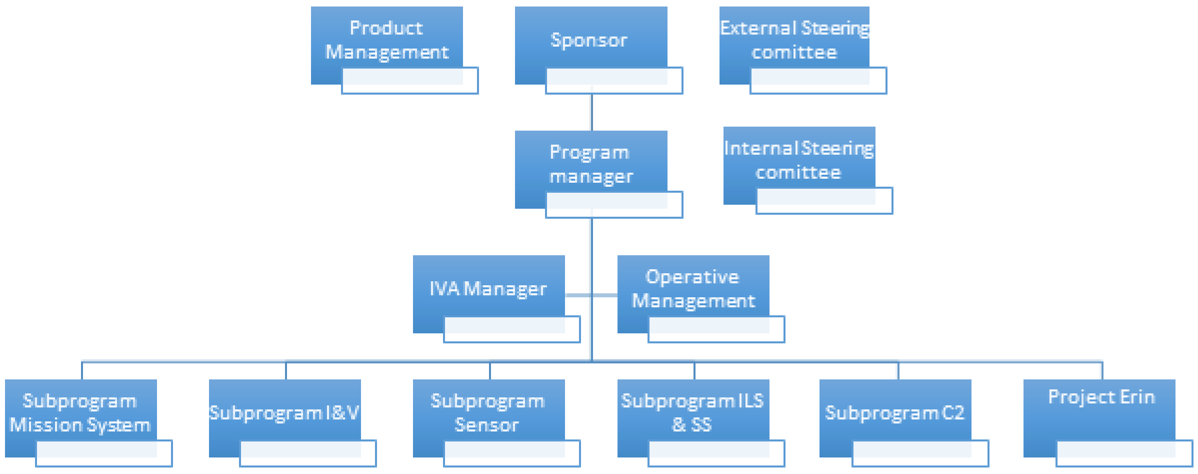


Figure 20: Schematic overview of the baseline program

Development plans

The technological roadmap is the overall development plan and is handled by the product manager and his product management team. This team is responsible for the commercial view of the product and decides what functionality the product should have. In what order the different functionalities should be developed is also handled by the product management through the release plan. The product management also heads the configuration control board

(CCB) to handle change requests from the baseline program. The release plan is the base for the master schedule which contains all important activities and overall objectives. The master schedule is then broken down into increment plans. The baseline program currently works with three increments per year and during each increment certain functionalities are developed and integrated into the system rig. When developing the increment plan the Integration and verification manager (IVA) looks at the master schedule to see what functionalities are supposed to be delivered at the end of the increment. He then checks with the system teams and subprogram managers to see the feasibility of this plan, they respond with what they will be able to develop during the increment. These answers are then compiled into the increment plan.

The program is divided into five different subprograms that will work continuously with the development and maintenance of the product. However, these programs are dependent of one another and their decisions affect the other subprograms. Therefore, collaboration is needed in order to deliver a good product; this is the reason behind the creation of the system teams. The system teams are cross-functional teams that are focused on a specific subsystem or function within the product. They mainly consist of systems engineers, I&V employees and participants from the development teams and ILS & SS. The work in these teams is focused around requirements and verification and the actual work is mainly not done within the teams.

4.1.1 Subprogram Integration and Verification

Mission
The main task for subprogram Integration and Verification (I&V) is to integrate the deliveries from the different subprograms and to verify the system's functionality and find faults. The tasks also include maintenance and development of the system rig. This consists of an actual radar where all deliveries are integrated and verified. Within the baseline program the task is not to handle customer acceptance, instead the task is to deliver a product to the customer project according to the agreed requirements. These requirements are derived from both the internal development plan and from the customer requirements.

Subprogram structure
The I&V department is divided into two groups, one group that is working within the baseline program while the other is working with the customer projects. The personnel that work within the baseline program are assigned to one or more of the system teams. Within these teams, they focus mainly on how the system shall be verified. Test cases are designed and put together in verification specifications. These specifications are then the foundation upon which the system verification is performed. However, all requirements are not tested in the system rig and therefore they also have to control the verification that is performed by the subprograms. The subprograms test all possible settings of the subsystem while I&V are verifying the system as it is supposed to be operated and the interfaces between the different subsystems.

The integration of the system is another important factor for I&V, the different subprograms deliver their products or prototypes to the system rig in order for it to be tested in the correct environment. To see that the different subsystems work together they have to be integrated. The other subprograms have their own test rigs, but in them the other subsystems are generally simulated and therefore they do not offer the same level of realistic integration. Therefore, the feedback from the system rig is crucial for the developers and system engineers to make sure they have designed and developed a working product. The integration and verification is performed in three phases: pre-integration, integration and verification. During pre-integration I&V integrates updates and new functionality that needs to be tested. During the integration phase all deliveries included in the increment plan is received and installed in the system rig. The verification phase is when I&V runs tests to ensure all functionality is working according to requirement specification. The system rig is one of the bottlenecks in the development process, due to time consuming installations the system rig is closed for updates and new deliveries during two to three weeks when tests are performed, the verification phase. All subprograms, including I&V desires to work more with continuous integration but due to time constraints it is not possible at the time. An obvious solution would be to invest in another system rig where integration could be done continuously but the system rig is very complex and expensive.

The management of the I&V subprogram consist of the line manager, the subprogram manager and the Integration and Verification Responsible (IVA) Manager. The line manager signs an agreement with the program manager regarding the responsibilities for the I&V subprogram. The Line manager has appointed a subprogram manager who handles the day-to-day management of the subprogram. The IVA-manager is responsible for the verification of the product. It's his responsibility to generate the increment plan and make sure that verification will be performed ahead of the system releases. He instructs the system teams on what needs to be delivered in each increment in order to verify the functionality.

The system teams have started the journey to work more cross-functionally however the work is still performed within the subprogram. There has been no effort to start working in sprints or to educate the employees in agile methodologies. This means that the problems that agile development can solve must be communicated and the staff needs to be educated to understand the ideas behind it.

Identified Problem Areas

Based on the interviews with the personnel at I&V several problem areas were identified and are described in the categories below.

Ways of working

Some of the people working within the system teams have experienced uncertainties regarding what is expected from them and the system team. There are no leaders within these teams and the management creates uncertainties concerning who and where to report problems and progress. It can also create a problem regarding who will develop a time plan and control that they are on schedule. This has led to that the tasks in the early stages of development have been thoroughly analysed while the tasks in the later stages have had to be stressed. The final tasks are currently not always a group activity within the teams, instead I&V personnel are left alone to handle them while other team members continue with the next assignment or increment. This approach causes the I&V work to be more individual rather than a team effort and the personnel may feel isolated in their assignment with big responsibilities. Many of the interviewees has said that the lack of time is the main reason that causes people to work within there are of expertise. The first requirements have recently been verified after years of development, this causes stress and difficulty to see their overall progress. Other uncertainties within the system teams include who has the overall responsibility to make sure that nothing falls between the cracks of the teams. Some of the personnel at I&V are members of more than one system team and consider it to be difficult to prioritise between the tasks in the two teams. This has resulted in the system teams more being used as meeting forums rather than actual cross-functional teamwork. The interviews have shown that many of the subprograms have different ways of working causing problems, since I&V is the receiver of the deliveries they are more affected by this than others. It is desired to agree on a common way of working and delivering functions.

Communication and information

Another issue with the deliveries to the system rig is that they are not always delivered according to plan. Several people who work in the rig claims that they are often surprised of what is delivered and what is missing. These people claim that there is a need of a better communication between the subprograms. There are also communication problems between the baseline program and the customer project. Some people within the customer project claims that it is hard to get an overview of the progress of the baseline program. To get an idea of the status of the development they must attend several meetings and create a picture from these. There are also problems the other way around where the customer project has activities with the customer without notifying the baseline program.

There have been several large changes within the department in a short period of time. The baseline program has been created, the system teams have been introduced, a new methodology has been introduced with agile product development and at the same time the product is supposed to be significantly re-engineered. These changes have caused confusion among the engineers within the baseline program, they wish that the changes would have been incremental and better managed.

The interviews and meetings have shown that there are uncertainties regarding the roles within the baseline program. People are used to a project organisation with a clear project manager; however, within the baseline program there is a program manager, subprogram managers, IVA manager and line managers. On top of this there is also a customer project organization with corresponding roles. This has caused an uncertainty regarding who has the right to make certain decisions and who can change already made decisions. These decisions does not always reach all the affected teams. Having corresponding roles within the customer project and the baseline program causes more reporting for the personnel, since there are more stakeholders to report to. The lack of progress communication has also caused more stress on stakeholders leading to more progress reports for them to feel comfortable.

Requirements and testing
One of the main problems for I&V subprogram is the fact that they have many deliveries coming in from the different subprogram to the system rig. Many of these deliveries are problematic and time consuming to install and the operations in the rig stops during these installations. During these stops verification is not possible. If the deliveries are frequent they do not have time to verify anything, instead all activities in the rig is focused on installation instead of running tests. There are several stakeholders that uses the rig to test certain functionalities, they require help and this takes time from the I&V employees. Frequent customer visits also limit the capacity in the rig since nothing new can be installed prior to the visit to guarantee a well-functioning system. To handle this problem I&V have divided the increments in three phases: Pre-Integration, System Integration and Verification. During the Verification stage, no deliveries are accepted and thereby time is created for verification. However, many of the subprograms wants to be able to deliver continuously and oppose the fact that they can’t deliver during this period.

4.1.2 Subprogram Sensor

Mission
Sensor is one of the subprograms within the baseline program. The developers in this subprogram consist of both software developers, software testers and system engineers that work together in teams. The system engineers maintain and manage the system within the functional area of sensor. The main responsibilities for sensor is to design, develop and test Signal Data Functions (SDF). They are also responsible for creating analysis, investigations and validations for the radar work.

Subprogram structure
The developers are currently organised in four teams, one of the teams consist of only system engineers and is organised as a version of Kanban. The remaining three teams are cross-functional with team members from both developers and system engineering and works according to Scrum.

The first team consists of only system engineers working in small temporary teams, often with tasks reaching over a longer period of time than the sprints. The majority of the system engineers in this team are also a part of the system team where they work in collaboration with I&V. Their main responsibilities consist of making analyses, investigations, validations and design of system tasks. They get their tasks from the same product backlog as the other teams but do not work according to Scrum or in sprints to the same extent as the other teams. This is due to that the tasks for these teams are often larger and cannot be completed within a sprint. For each new task or package brought to the team, the members pair up according to what type of function the package is and work temporary in that group until the task is finalised.

The remaining teams work more cross-functional consisting of both software developers, software testers and system engineers. Their main responsibility is to build systems, design, develop and test SDF. The teams are organised according to Scrum methodology with a PO and

SM for all teams. They work in sprints of three weeks where task or epics are provided from a main product backlog for all teams. The main product backlog is prioritised according to business perspective by the PO in collaborations with the Scrum teams. Each team also has a "team" backlog consisting of epics based on the main product backlog. Each team conducts a sprint planning, daily Scrum, sprint review and retrospective. The teams also have a joint retrospective called "information sharing" summoned by the SM to evaluate overall improvement areas. The team has a backlog refinement event once a week with the PO to update and reprioritise the backlogs content. The SM's main responsibilities are to keep the teams together and ensure there are no impediments so the developers can work efficiently. The SM also summons to daily Scrum and other meetings affecting the teams. The PO is responsible to bring tasks and epics to the main backlog and prioritise it according to a business perspective. The PO is also used as a filter from the external environment, external subprograms cannot add tasks to the developers without going to the PO who then can add it to the product backlog and prioritise among the other tasks. The PO is also responsible to bring up issues from the team and communicate the team's needs and decisions to other subprograms and program management.

The subprogram has managed to implement an agile environment with their internal work and is constantly improving their ways of working. The employees are assigned to teams and all the work is performed within these teams. The team members are also educated within Scrum and agile values which has led to that all employees are able to contribute to the continuous improvement. However, the teams are not cross-functional and this means that they cannot develop a feature from start to finish.

Identified Problem Areas
From the interviews conducted with the engineers and managers at sensor some problem areas could be identified.

Ways of working
The majority of the engineers within Sensor are very experienced and possesses high knowledge within their areas of expertise. The assignments given to the engineers often cover a wide technical area and it has shown to be difficult to cover the competence necessary within a small team. It was brought up in several interviews that there is a conflict, small teams are desired but it can be difficult to cover the competence needed in a team if there are many single competences among the developers in Sensor. It is today not possible for the engineers to dedicate 100 % of their time within the Scrum team. This is due to additional assignments occurs frequently on individual developers which is not included in the team's backlog. Some of the developers, especially system engineers, are members in more than one team, one team at sensor and also a system team for example the Radar System Team. Being in more than one team can make it difficult to decide how much resources should be dedicated in each team and the system teams have been used mainly as meeting forums from which they get more assignments to complete.

Many of the interviewees agreed that it is beneficial to work in three-week sprint, to focus on a smaller amount of work at a time in which they plan, develop and test the functions within the sprint. The drawback is that it has shown to be more difficult to get the overall view of the programs progress which is desirable to know among many developers. Many engineers have also mentioned a desire to include I&V in the Scrum teams. This would facilitate a better collaboration between the subprograms and sensor would get feedback continuously. They also think it would ease the work for I&V when planning verification process if they were part of the teams building the functions that shall be verified.

Communication and information

Sensor has one meeting a week where they discuss the teams' progress and share information and knowledge. Many feel it would be appreciated to have more presence from stakeholder in these meetings to include information about the overall program progress and customer projects. Stand-up meetings from stakeholders are another suggestion that has come up during the interviews. Stand-up meetings could be conducted frequently to communicate overall progress and information from customer projects.

Transitioning to a baseline program from multiple customer projects running in parallel have resulted in the introduction of more and new roles in the organisation. The introduction of agile frameworks has also introduced new roles such as Product Owners and Scrum Masters. The number of roles in the organisation has created some confusion among the developers, the purpose of each role and who is responsible for different areas. Having the baseline program and the customer project, which conduct the final phases before delivery to customer, also leads to many roles being duplicated. This leads to confusion on whom to bring up issues to, should they for example contact the verification manager for the baseline program or from the customer project? Having large amounts of roles also results in difficulties to know who can make tough decisions, which decisions are we allowed to make in the team and if not, who can the team turn to? And if a decision has already been made, who is allowed to change a decision? This confusion leads to many decisions needs to be made higher up in the organisation and the feeling is that the hierarchy has grown from a flatter organisation by introducing the baseline program. Today there are several steering committees to report to continuously during development, many developers feel that a lot of time is spent on reporting which takes focus from the daily work. They feel that there should be more trust in the teams from management that they are developing on time, according to increment planning, and not have to spend so much time and effort on reporting progress.

Requirements and testing

The Scrum teams in Sensor work in short sprints and there is a big desire to deliver more frequently to the system rig. The developers want to test their functions in a more "real" environment to get feedback if the functions are operating as planned. The system rig is a big problem area today. There are many deliveries to the rig from multiple subprograms and the installation of all deliveries takes so much time and resources to install that I&V frequently has to close the system rig for new deliveries. There is a big desire to find a solution so the system rig can handle continuous integration from Sensor so the developers can have faster feedback loops from I&V. From many interviews, it could also be concluded that many developers feel that a lot of resources are spent on formal verification of requirements. Many feels more resources should be spent on building and testing functionality in the early phases and only prepare the test cases and verification document in parallel but the formal verification should be conducted later in the process.

4.1.3 Subprogram Mission System

Mission

Mission system is a subprogram that is focused on systems engineering in the baseline program. The system engineer's role is to design the system on an overall level and provide the developers with specifications. This means that they are responsible to provide a design that will allow the system to meet the customer requirements.

Subprogram structure

The engineers in the subprogram are assigned to a system team together with people from the other subprograms. The system teams are focused on different functionalities within the system. Within the team the systems engineers' major task is to provide design specifications and requirements to the developers in order for them to know what the desired functionality is and what boundaries exist. The work of the system engineers is based on both customer requirements and requirements from product management, where the product management requirements focus on building a product according to the product roadmap. These requirements will provide the functionality that is necessary to develop in the program and from this the systems engineer will produce the requirements for the different subsystems. The requirements are however not definitive, feedback on their feasibility will be provided from both the I&V sub-department and from the development teams.

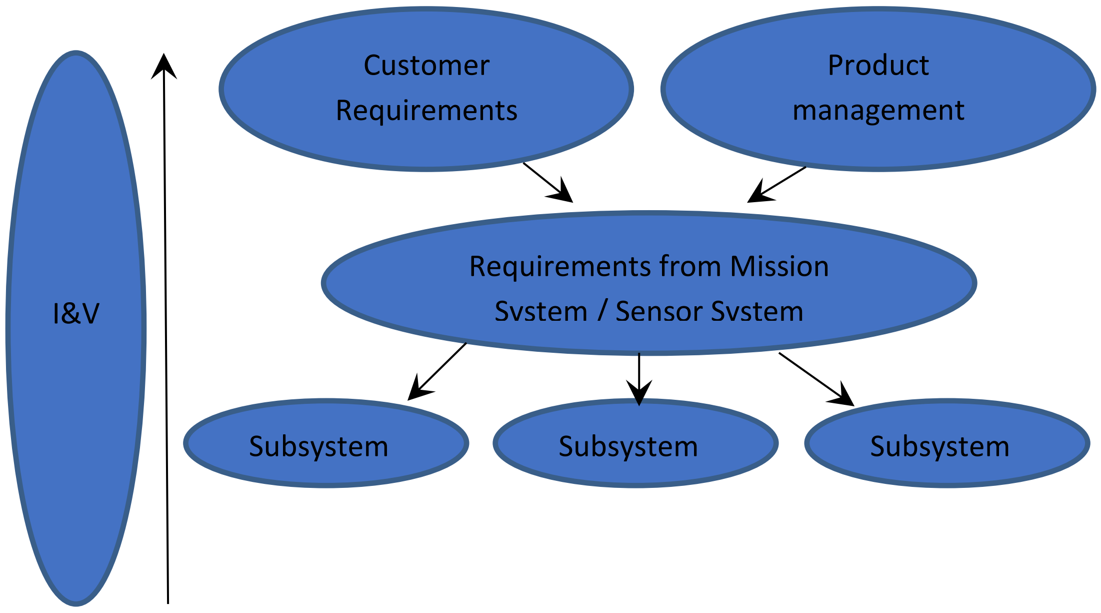


Figure 21: Schematic overview of the requirement process

The reason for introducing the system teams was to increase the cooperation between the different subprograms that worked within the same functional area. This means that the systems engineer now have the opportunity to receive quicker feedback on the feasibility of their requirements. It also provides them with the possibility to affect the verification and testing cases.

There have been several attempts to implement a more agile way of working in the subprogram. A Kanban board was implemented to start working with stories and epics, however the meetings became too long and the board is now sparsely used. The system teams were introduced to increase collaboration and to work more cross-functionally. This has successfully increased collaboration but the work is not performed within the teams. This shows that there is a drive to become more agile within the subprogram but the persistence and education might have been lacking.

Identified Problem Areas

The transition from customer projects to the baseline program has not affected the system engineers work assignment to a large extent. Many experience the assignments are relatively similar to previous projects. During interviews, it has however surfaced that some feel the development work is falling back to old processes and patterns, focusing more on delivery projects rather than the baseline program. The issues found in during interviews are presented in four categories below.

Ways of working

The system engineers operating within Mission System are organised in various system teams, with representatives from all sections within the baseline program. Many are positive to working more cross-functional and including I&V more in the teams; to ensure the requirements can be verified at an earlier phase. It often occurs that for example I&V are not able to participate in earlier phases and it would be beneficial to engage in a better collaboration in the teams. The system teams were initially meant to cooperate to deliver various functionalities, but today the teams are more used as meeting forums in which the members can discuss work assignments which they then bring to their own section to perform. Another subprogram, ILS & SS, whom are responsible for the manuals and safety aspects of the products are often included in the team's work late in the process. There is a desire from some to also include them earlier in the process so that manuals can be developed in parallel with the development work. This would create a better communication between the engineers developing the functions and the engineers writing the manuals which would enable more feedback earlier in the process.

There is a desire to get more efficiency in the teams, that all teams have a function they are responsible for and collaborating more when performing the activities within the function. Having clearer function areas for the teams could help ensure that all assignments are covered. Today some feel that assignments may sometimes fall in the area between two teams and it can lead to not all assignments are covered. Many feel it could be more clearly stated which the team members are and which competences are necessary to cover the team's assignments. Some of the system teams are larger than others, and small sub-groups are automatically established within a team. The lack of collaboration within the teams also leads to many system engineers being isolated within their own small function area. This is due to the fact that there are some single competences within the team, but also the fact that many assignments are so large that it takes several months to complete.

Some experience that the development process is rigid according to waterfall processes, and that many assignments are conducted by different sections at different times. This lead to long feedback loops and a more flexible and parallel development process is desired. For example, if an engineer makes a correction in a function, they cannot integrate it in the system until the next delivery; which can be up to three months later. Some feel that to only work with three increments per year is not enough. Shorter loops would increase the possibility to receive crucial information before continuing the development process. Some also feel it is difficult to implement more agile teams in an environment based on the waterfall process. The entire baseline program must change to enable the teams to change. It can also be difficult for engineers to work in agile teams if they do not know and understand what agile really is. The word is often used and people within the baseline program express a desire to work in an agile way but if they are to make the transition they need to be educated and trained first.

Kanban

The subprogram has made previous attempts to work with Kanban, and they still do but not to the same extent. Many feels that using a Kanban board to visualise work assignments are good to increase the communication within the subprogram. However, it has not yet had the outcome they were hoping for. Visualising the assignments on a board, showing who is working on what is good in theory, but the assignments are usually large so that they sometimes can remain on the board for up to 6 months. The stand-up meetings at the board has been very appreciated, it is a simple and fast way to get information about what everyone in the team is working on. The benefits of Kanban have varied over time, for some periods the stand-up was misused, there were often discussions about current problems leading to the stand-up being a long meeting rather than a short status report.

Some feel that most assignments are too large; conflicting with the pull system in Kanban. The assignments are also difficult to break down to smaller packages, which would enable the use of Kanban and also provide a better situation for sharing assignments between personnel and incorporate more collaborated work. Some feel it could be beneficial to make an attempt to break down assignments to smaller parts, and try to divide them among the engineers. If the decision is made to keep the Kanban board, some feel that it is crucial that all work packages are prioritised and visualised on the board so everyone can get the information. Previously they also had colour codes for the assignments, green is if work is conducted according to schedule, yellow if slightly falling behind, and red if they will not be able to complete assignment to the set delivery date. Using the colour codes made it easier for management to get a progress status and see if more resources where necessary for an assignment. Many feels that Kanban is a good idea in theory, but that a large part of the engineer’s time is not spent on the assignments on the board. There are many additional activities such as consulting with other teams and departments or going on meetings, and these activities are hard to visualise on a Kanban board.

Communication and information

Today it can be difficult to get an overview of the progress in other subprograms. There is no simple way to find this information except to attend several different meetings to create the overall picture. The customer project doesn’t continuously communicate its progress and this information also must be sought out individually. Several people have expressed a desire to more easily access this information, either through one single meeting or a condensed report.

There is no clear way to map dependencies between the different teams. This leads to sub-optimal prioritization of tasks since one team is not always aware of what the other team is doing and what will be delivered to the system rig. At the moment, it is even hard to keep track of what functionality is in the rig, not to mention what will be delivered during the next increment. If the dependencies where clear and communicated better it would be easier to plan the work in order to deliver a complete functionality. This could be done through better communication when development plans are being established.

Some feel that there is an uncertainty regarding what mandate the system teams have and exactly what they are expected to produce. This has however been solved by internal communication and hasn’t caused any problems.

Requirements and testing

There are some differences on how the system engineers specify requirements, and also a difference on what they define as a requirement. Some want the focus to be on design specifications and others on the requirement descriptions. There is also a conflict on if a design specification is a requirement or not. It would be beneficial to find a mutual understanding on what requirements really are and decide on a common way to work with requirements and design specifications.

The interviews showed a desire to get better systems to test functions before they are sent to the system rig. Today too many errors are found in the system rigs and it would be more beneficial if more errors could be found and corrected before they are integrated in the rig.

The majority of the verification today is to ensure the requirements are fulfilled, some feel that there should also be a verification procedure for the functions. Simply having a pass/fail for requirements are not enough to ensure the system fulfils the customer’s needs; there should also be a pass/fail on functions and the functionality.

4.1.4 Subprogram Integrated Logistic Support & System Safety

Mission

Integrated Logistic Support and System Safety, ILS & SS, is one of the subprograms within the baseline program. The subprogram is responsible to ensure the product fulfils the safety aspects and all the manuals and training necessary for the customer to operate and maintain the product.

Subprogram structure

ILS & SS is a fairly small subprogram, consisting of eleven engineers and they are responsible for and divided between four different areas:

Reliability and maintainability: ILS is responsible for the products reliability and maintainability. This includes the maintenance of the product such as spare parts and support equipment's at the customer to ensure the product is reliable at all times.

Technical Publications: ILS is responsible to provide the customer with manuals and descriptions for operators, maintenance, functions, service and more. This to ensure there is information available for the customers regarding how to use and maintain the product after delivery.

Customer Training: ILS is responsible to train and educate the customers in the delivered system. This includes training for operators, maintenance and instructors so the customer has the knowledge necessary to use the system.

System Safety: They are responsible to ensure the product fulfils the safety requirements. This is conducted through safety analysis and hazard logs which is communicated to R&D.

Currently there is no agile methodologies implemented within the subprogram. There has not been any education and no attempts to start working according to these methods. The subprogram is currently isolated from the other subprograms since their main work is performed when the product is ready for delivery. This means that they are not as affected by the current issues and do not have the same need for change. In the interviews, several people have mentioned that they would like to increase their collaboration with ILS and get their input earlier. This could be done by involving them more in the system teams.

Identified Problem Areas

The interviews conducted at ILS & SS has shown that their subprogram does not experience the same issues as many of the other subprograms. ILS & SS have a rather low level of dependency towards the other subprograms, so they are not affected by the communication and integration difficulty to the same extent as others. This since they are not involved in the development as early in the process, their work is limited in the beginning and grows exponentially towards the end when manuals and educations must be planned and prepared. In the early stages of development, they mainly attend meetings and communicate with personnel from the other subprograms to take part of how the system is developed and provide feedback

The work conducted at ILS & SS has not been affected by the transitioning to a baseline program, the work assignments are similar to before when they worked in various projects. The only difference is that they now have to identify what work is customer specific and what work is general for the base product. When working with the base product they try to establish for example manuals which can be reused for different customer deliveries. Previously they worked only with documents; where they now try to establish modules, which can be assembled depending on the customer and delivery.

Some of the personnel in ILS & SS are included in the system teams, but mainly to take part of information and decisions. This is since ILS & SS work assignments in the system teams are not necessary this early in the program. The plan is that when the program moves over to the next phase ILS & SS will work more in collaboration with the system teams.

The only problem area identified in ILS & SS regards communication, some has experienced difficulties to get information about whom is responsible for various things. This could be communicated better so it is easier to know who is responsible for what and whom to reach out to in various assignments.

4.1.5 Subprogram C2

Mission
The Command and Control (C2) subprogram develops the Man-Machine Interface (MMI) for the base product. This includes all user interfaces that are visible for the customer and the fusion of the data that is visualized.

Subprogram Structure
The C2 subprogram consists of about 40 people and is the only part of the baseline program that is not co-located with the other subprograms. This is solved through video conferencing and is not considered to be a problem by the people that were interviewed. Within C2 the agile methodologies have been extensively developed. The development is handled in four cross-functional Scrum teams, see figure 22, with members from systems engineering, software development and integration and verification. The teams are not constant but vary over time to match the current tasks. The Scrum teams were initiated by the developers and they managed to get their co-workers on board quickly, which they think was one of the reasons they had a successful transition. They did educate all people within the organisation prior to the implementation. When the Scrum teams were introduced they felt that they became isolated within the team and it was hard for them to know what the other teams were doing. This caused problems when decisions had to be made and led to the creation of different meeting forums. These meeting forums made it possible to take overall decisions and to conduct more long term development of their own process.

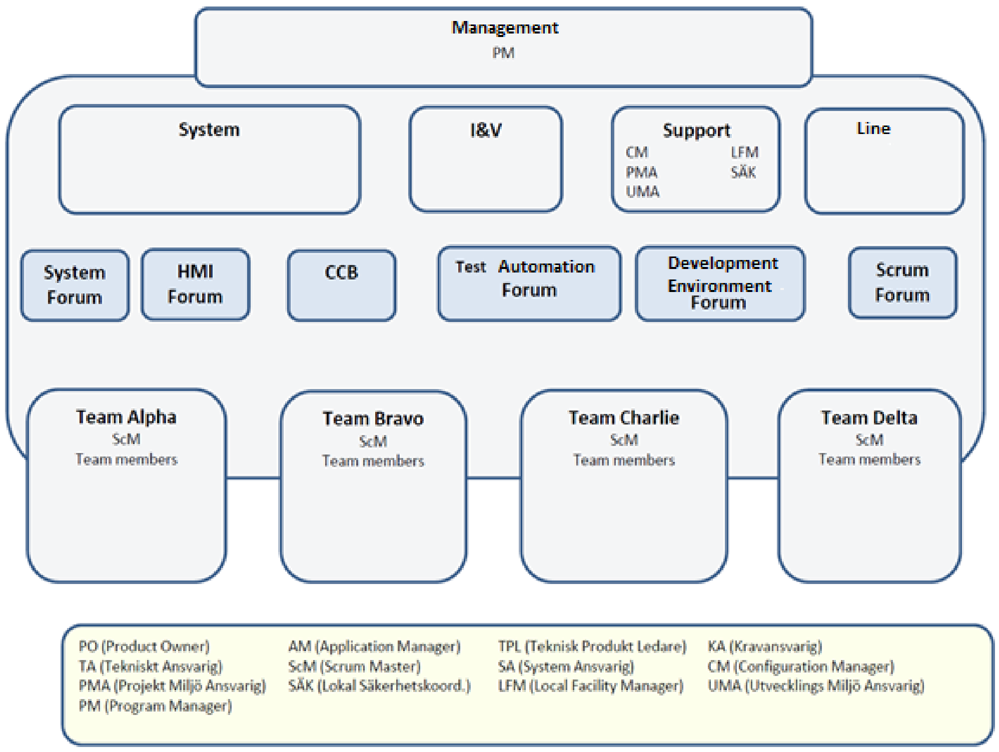


Figure 22: The organisation in Subprogram C2

The implementation of Scrum has been considered a success by the management of the subprogram. They feel that they have become more effective and also delivers code with better quality, earlier simple errors usually reached the subsystem rig but know that rarely happens. The collaboration with the other subprograms works well. They deliver their code to the system rig two times per increment, one pre-integration drop and one final delivery. Earlier they only delivered one time per increment and this caused long installation times due to more errors and lesser knowledge of the software in the system rig. The system I&V also visits 2-3 weeks prior to delivery in order to learn the new functionalities.

The geographical distance to the other subprograms has allowed C2 to develop their own way of working without interference from others. The result is a well-functioning agile organisation built around four Scrum teams. They have managed to become more agile than most parts of the baseline program and to improve they need to continuously integrate at the system level and work more cross-functionally.

Identified problem areas

One of the main problems that have been discovered is that even though C2 has started to deliver one pre-integration drop, this is not enough. The software still causes problems when it is installed in the system rig. More frequent deliveries would shorten the feedback cycles and also increases the knowledge of the functionalities among the system I&V employees.

The distance between C2 and the other teams is also considered to be a difficulty, especially when problems occur in the system rig. The other subprograms are able to come to the rig and look at the problems, but this is not possible for C2 which makes it more difficult to solve these problems.

4.1.6 Summary empirical findings Saab

The findings presented above vary between the different subprograms. The findings common for several subprograms are presented below and attributed to different problem areas. These problem areas, see Figure 23, will be the base for the future solutions and recommendations.

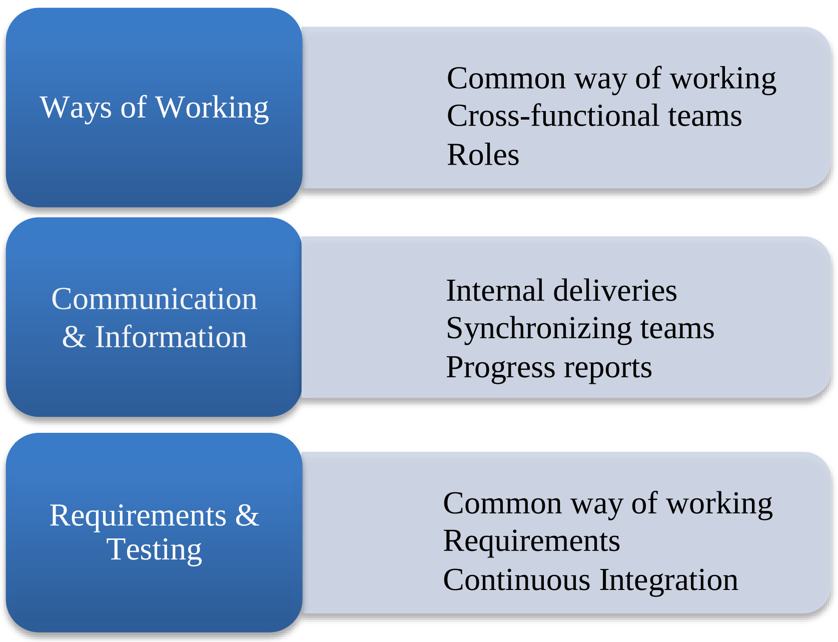


Figure 23: Identified problem areas

The majority of the identified issues regard integration and communication within the baseline program. There is also a desire to finds a common way of working and achieving more collaboration and cross-functionality. These issues are described further in the section below.

Ways of Working

There is a desire to establish a common way of working within the baseline program; so all system teams and/or subprograms follows the same process and develop and deliver in the same way. There is also a desire to work more within cross-functional teams to increase the collaboration between the subprograms. Concluded from the interviews is that the biggest desire is to include I&V more in the system teams; that they share the responsibility and establish requirement and test cases together early. This can ensure that the functionality is built according to a common understanding of the requirements and ease the verification process of test cases. Some of the interviewees feel that there is an uncertainty regarding the roles within system teams and in the overall organisation. Especially I&V whom has frequent contact with both program management but also the customer projects; there are several corresponding roles leading to confusion on who has responsibility and mandate on certain areas. The conclusion is that many in the baseline program desires to find a better team structure which is more cross-functional; and to evaluate which roles are necessary and what mandate the roles have. The evaluation of roles is both for system teams and organisation. What roles are included in a system team and what roles will lead and manage the entire program?

Communication and Information

The system teams currently work somewhat independently from each other and there is a desire to find a way to synchronise the teams better. There are many dependencies among teams and these are not communicated well enough. It can be difficult to find out what the other teams are currently working on and if they will deliver according to plan. This creates uncertainties and makes it more difficult to prioritise the work in the system teams. There should be an easy way to communicate what the teams are working on and what, when and how will be delivered. Increasing this communication can improve the process of integration and also help the teams to identify the dependencies among teams and thereby the enable the prioritising of work in the teams. As stated before, from interviews it could be concluded that I&V sometimes can be surprised of what is delivered to the system rig. Sometimes functions are not delivered according to the incremental plan; functions can be removed or added in a delivery. This makes the integration more difficult, this could also be improved by inserting a better way to communicate; so that all are aware of progress and exactly what is to be delivered and when.

Requirements and Testing

One of the big conflicts found during the interviews is that I&V need to close the system rig for deliveries and update. This is due to many installations are time consuming and to actually be able to conduct tests and verification they cannot insert new deliveries or updates during these times. Many of the other subprograms desire to make more frequent deliveries to the rig and update functions which have been resolved from minor errors. There is a desire from all parties to find an easier way to integrate more frequently in the system rig but the right conditions must be in place for this to be possible. It should be investigated how the baseline program could find a better way to deliver and integrate in the system rig; and how they can be assured that the functions fulfil the quality criteria before they are integrated. The interviews also showed an issue regarding how requirements are established. The subprogram Sensor desires that requirements should focus on describing functionalities and design rather than having stricter requirements on a detailed level. The interviews in subprogram MS also showed that there is not a mutual understanding on how requirements are established. Some engineers prefer to work with design specifications describing the desired functionality while others work with requirement lists. It would be beneficial if the personnel responsible for requirements can decide on a mutual way to create and communicate these.

4.2 Organisation survey result

To evaluate the core culture within Saab, Schneider's (1994) culture survey was used. The results from this can be seen in the table and figure below.

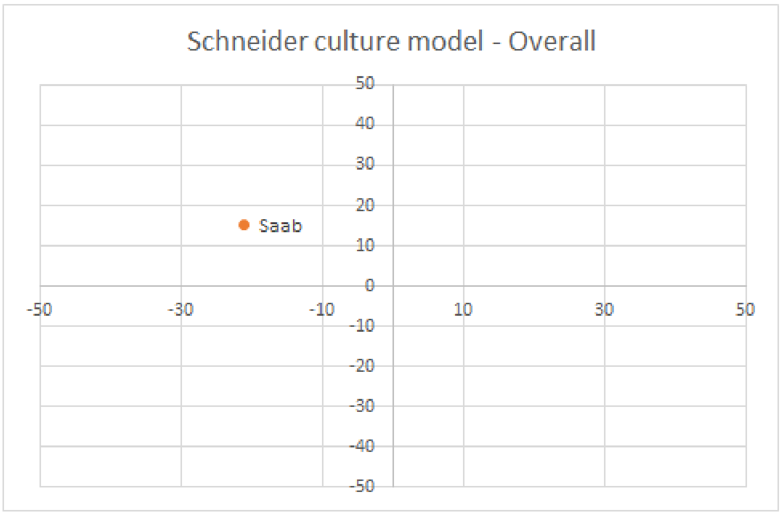


Figure 24: Saab's positions in Schneider's culture model

Control	26%
Collaboration	24%
Competence	19%
Cultivation	31%

Table 2: Distribution of answers from Schneider's culture survey

As can be seen Saab does not have a dominant core culture which according to Schneider usually is a sign that multiple cultures exist within the organisation. The reason behind this can be that there are different cultures within the teams and as the organisation as a whole. What can be seen is that control and cultivation is stronger than control and competence. This suggests that the organisation is more oriented around the people than the organisation itself. This result did surprise many of the employees who felt that control is an important part of their culture. However, when the results were divided between management and engineers another discovery was made, see Figure 25.

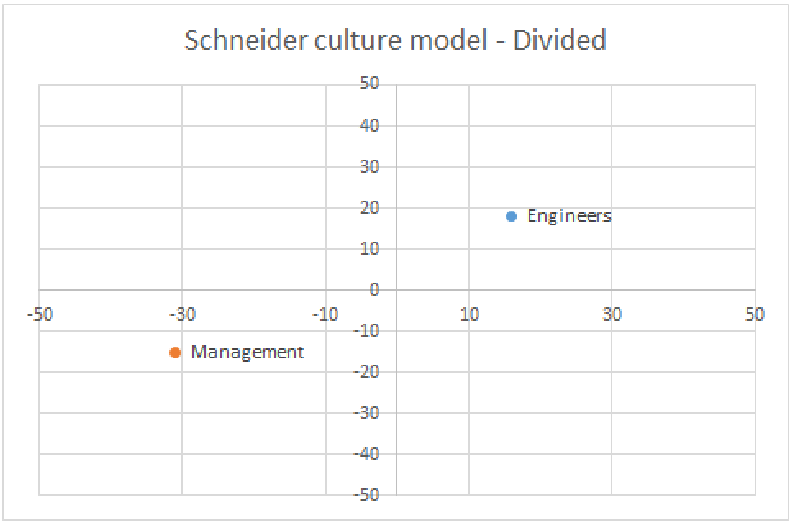


Figure 25: Management and Engineers responses compared to each other

As can be seen there is a difference between how the two groups considers the current culture. The engineers consider it to be more control oriented and management thinks the culture is more towards cultivation and collaboration. This might be because management wants to create a collaborative environment and believes that they are successful while the engineers still feels there are a lot of control mechanisms in place. However, no culture still reaches the significant level of 50 % which implies that several cultures exists.

4.3 Case studies

During this research four case studies has been conducted to analyse previous attempt to scale agile development. The cases studied are two separate departments at Saab which are part of other business units then the baseline program. The other two cases were conducted at departments at Ericsson and Volvo Group. Additionally, three expert interviews are analysed as a case study since their combined experience provided deep insight to how this type of transformation can be conducted.

4.3.1 Saab Department X

During interviews, other agile implementation projects within Saab were brought to attention. One of these was investigated to provide insight and learnings for this thesis. The case study is based on interviews with two project managers from the department in question.

The department develops products similar to the one investigated in this thesis and include integration of both hardware and software. When it was decided to introduce agile methodologies in the R&D process, an external consultant was used to educate the employees in Scrum methodology. After all employees had under gone this education the Scrum teams was established and they started with a series of test sprint. The department did not get support from the consultant to adapt the Scrum methodology to fit their organisation, and they experienced some problems initially.

The department started to adapt the Scrum methodology on their own and they had multiple pitfalls along the way, they did however make the adaption one step at a time and let the change take time. The approach was to simply test different structures regarding team and events in Scrum and if it was not an appropriate fit they rearranged the structures and made another attempt. After a series of test runs they found a good structure with the appropriate level of agile methods and tools for each team.

The department initially started with some, but limited, cross-functional teams. This did not have the effect desired so they decided to have teams formed according to functionality and instead had teams depending on each other to work close together instead. The design engineers work according to Scrum methodology, working with backlog, sprint planning, daily Scrum and so on. The system engineering teams work more according to Kanban, this was a better fit for system engineers since their work packages normally are more time consuming and it showed to be difficult to always finish a package within a sprint. The teams working with hardware development have different structures among the teams. All teams work with different variations of Scrum, some do Scrum by the book and some have only chosen some events or artefacts used in Scrum. The teams formed in I&V initially tried some agile approaches, but it showed not to be appropriate for their work so they are now operating in more traditional approaches.

The adaption of Scrum resulted in many different work structures among the teams, and they do not use cross-functionality to the extent which agile methodologies suggest (Schwaber & Sutherland, 2016). The department did however place the teams close to each other to increase collaboration and communication between teams. All teams currently work in continuous pace or sprints, of three weeks. The teams are very self-managed, but an obstacle appeared regarding how they could synchronise the teams. They tried to introduce meetings where each team reported their progress and synchronisation could be discussed. With up to 15 teams, this showed to be very time consuming and it did not provide the desired outcome. The solution for the synchronisation and an immense boost for the entire progress resulted in an integration plan. The teams are now synchronized by a visual integration plan where all deliveries from both hardware and software are illustrated and described.

An example of how the integration plan can be visualised is shown in Figure 26. The plan is visualised one sprint at a time, and there should always be a planned integration plan for the coming three months so all teams can plan and prepared in advance. The main delivery or integration for the end of the sprint is shown with a “star”, and it should include information on what is to be delivered, at what time, and who has the main responsibility. Each work package that is included in the main delivery is also visualised, in what order and how they are affected by each other. Each work package is broken down to smaller task or packages and is visualised as a branch. For each task or package include what to be developed, what time it should be completed and which team is responsible for the delivery. The large work package at the top of the branch also includes what to be delivered, at what time it should be completed and who has the main responsibility.

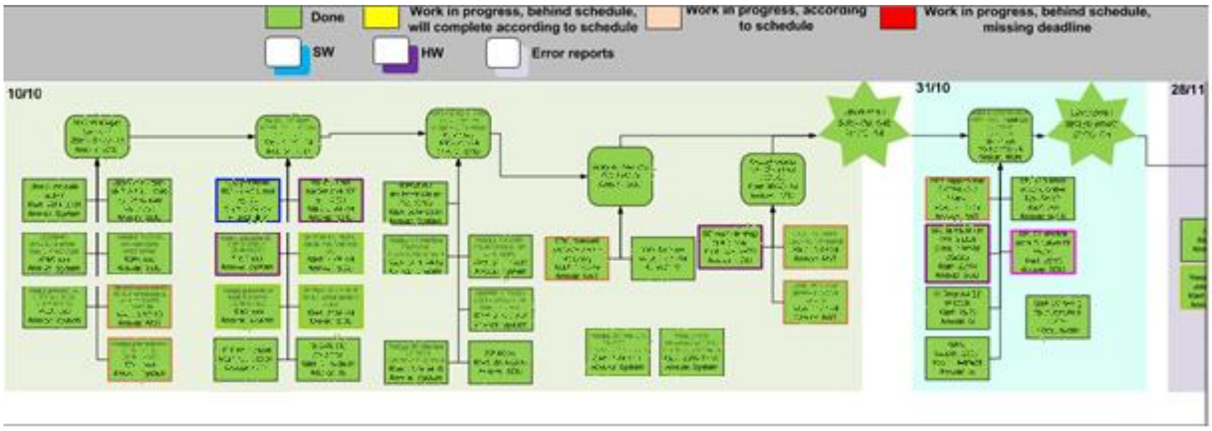

Done
Work in progress, behind schedule, will complete according to schedule
Work in progress, according to schedule
Work in progress, behind schedule, missing deadline
SW
HW
Error reports
10/10
31/10
28/11

Figure 26: Example of an integration plan

The integration schedule is updated continuously and progress is visualised using colours, the meaning for each colour is explained at the top of the illustration.

In Figure 27 an example of what the integration plan can look like if development is falling behind the time schedule. The red boxes are developments and integrations falling behind schedule, the red boxes in the branch to the left will not be completed on time. Since the packages to be developed in this branch not will be completed in time, neither will the big packages which is to be integrated later in the plan, since these are affected by the first work package. By updating these in the integration plan, all teams affected by this can easily see the progress of other teams and how it will affect their own teams. This also brings clarity to the receiver of the main delivery at the end of the schedule, of what actually will be in the end delivery.

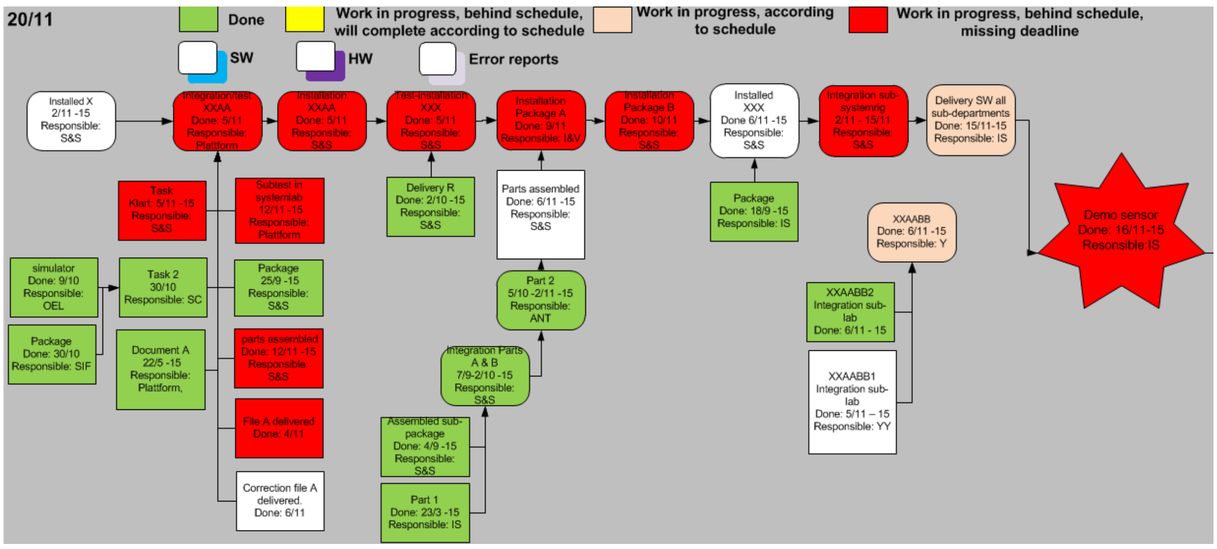

20/11
Done
Work in progress, behind schedule, will complete according to schedule
Work in progress, according to schedule
Work in progress, behind schedule, missing deadline
SW
HW
Error reports
Installed X 2/11 -15 Responsible: S&S
Integration/test XXAA Done: 5/11 Responsible: Plattform
Installation XXAA Done: 5/11 Responsible: S&S
Test-installation XXX Done: 5/11 Responsible: S&S
Installation Package A Done: 9/11 Responsible: I&V
Installation Package B Done: 10/11 Responsible: S&S
Installed XXX Done 6/11 -15 Responsible: S&S
Integration sub-systemrig 2/11 - 15/11 Responsible: S&S
Delivery SW all sub-departments Done: 15/11-15 Responsible: IS
Task Klart: 5/11 -15 Responsible: S&S
Subtest in systemlab 12/11 -15 Responsible: Plattform
Delivery R Done: 2/10 -15 Responsible: S&S
Parts assembled Done: 6/11 -15 Responsible: S&S
Package Done: 18/9 -15 Responsible: IS
XXAABB Done: 6/11 -15 Responsible: Y
Demo sensor Done: 16/11-15 Resonsible:IS
simulator Done: 9/10 Responsible: OEL
Task 2 30/10 Responsible: SC
Package 25/9 -15 Responsible: S&S
Part 2 5/10 -2/11 -15 Responsible: ANT
XXAABB2 Integration sub-lab Done: 6/11 - 15
Package Done: 30/10 Responsible: SIF
Document A 22/5 -15 Responsible: Plattform,
parts assembled Done: 12/11 -15 Responsible: S&S
Integration Parts A & B 7/9-2/10 -15 Responsible: S&S
XXAABB1 Integration sub-lab Done: 5/11 – 15 Responsible: YY
File A delivered Done: 4/11
Assembled sub-package Done: 4/9 -15 Responsible: S&S
Correction file A delivered. Done: 6/11
Part 1 Done: 23/3 -15 Responsible: IS

Figure 27: Example of integration plan when development is falling behind schedule

The department also introduced demos after each sprint where than teams demonstrate what has been developed during the sprint. The demos showed to be very beneficial, especially in the development of hardware. To each demo, all teams, PO's, project managers, economy departments and so on are invited, and whoever is interested can attend. The demo can be by showing a drawing, 3D-model or prototype, and the developers can explain how and why the development was established. This provides the opportunity for all else to understand the delivery, provide feedback and ask questions. Using demos has increased the communication and understanding among teams and divisions within the department and has resulted in an immense reduction of after work in the integration process. Production has been very surprised of how smoothly everything is assembled and they have never seen this successful integration in previous projects. The department has not yet had the same success in the demos of software developments, this is since it is more difficult to visualise and explain a code if the whole function is not yet developed.

The demos and integration schedule has improved the installation and integration performed in the system rig. Another success factor they noticed is that I&V participate in all tests performed in sub-system rigs as well. Having I&V in the sub-system rig provides them with a greater understanding for what is to be delivered in the system rig, and it also provides a faster feedback for the construction and system engineers. The overall feeling in the department is that there is considerably less "big bang" happenings in the system rig since they can catch errors earlier in the development. If errors are found in the sub-system tests it can be moved to resolve the error in the coming sprint so no, or only few, errors are sent up to the system rig. Another positive outcome for including I&V in the sub-system tests is that collaboration and communication has increased immensely and it has resulted in a better work environment.

There are several PO running the teams in the project, where each PO is responsible for a number of teams. The PO currently acts as a team leader and is responsible for bringing new tasks and work packages to the team and prioritise in each backlog. Before a new sprint begins all PO's' performs a PO sprint planning where they plan which task to place in which teams' backlog and how it should be integrated in the integration plan. The PO's also have integration meetings once a week, one with all PO's and once meeting a week each PO has an integration meeting with their teams.

The department has come a long way in the transformation toward agile development but they have some areas where the feel further need to improve. One of them are to evaluate which roles are needed in the project since there are some confusions on which responsibilities are on each role. What is for example the difference between a PO and a subproject manager? Are both necessary and beneficial to have? To reduce this confusion, they need to evaluate the roles, divide responsibilities and decide which are necessary to efficiently run the project.

4.3.2 Saab Department Y

Department Y at Saab runs a project with about 1500 engineers, around 500 of these are software engineers. The software engineers have worked with agile development for the last 5 to 8 years, depending on the team. This has led to a well-functioning structure from which several things can be learned.

The organisation consists of three levels. The highest level, design management, is still working according to traditional product development strategies. This means that they put up milestones and rigid development plans. These plans are then translated by the second layer, the product owners, into epics and incremental plans. The product owners plan the increments, which are three months long, one year ahead and before the start of a new increment the plans are redesigned. These plans are then further broken down into a product backlog. This means that the product owners become a key factor within this system. The product owners create the possibility for the development teams to work in an agile environment. Since they break down the overall targets and make decision regarding prioritization and they need to be senior employees with technical competence. Another important task for the product owners is to make sure that the teams are not disturbed with questions from management and other departments all the time. Instead these questions should come to the product owner directly and he will add stories to the backlog or prioritize tasks that might be important.

The teams consist of about 7 members which have been deemed the best size after testing out different options. They try to not change the teams to keep them constant since they have seen that this increases the efficiency. Most teams use Scrum or variations of Scrum as their method of choice. Scrum in the lower levels started the agile transformation but there is no constraints regarding how to apply this, the teams get to adapt whatever method works for them. The teams are not cross-functional between system engineering and software development instead certain teams for system engineers have been created. The reason for this is that the system has been considered to be too complex for the system engineers to work alone or in pairs in the teams, they need to work together to achieve a more holistic view. The development teams are specialized within certain technical areas and contain only developers. The development is built around functionality instead of requirements. This means that the system teams start to investigate the needed functionality and delivers this to the development teams. These teams then develop a prototype that can be tested to receive early feedback. The perception is that this helps them to build a better product with more advanced functionalities; however, it takes time to get the prototype code ready for production. The tests are performed in simulators that try to emulate the real environment, however all functionalities cannot be tested in these simulators. Therefore, integration in real rigs and airplanes are needed, this is handled by specific integration teams who are not part of the agile organization.

Each product owner has several teams which he handles. The breakdown of the product backlog down to a sprint backlog is handled by the product owner together with the teams. The teams have stand-up meeting every morning to report the progress and to receive help with potential problems. After these meetings team leaders, product owners and technical leaders meet in order to identify if there are any re-prioritization or technical decision that needs to be handled. In order to handle dependencies a synchronization meeting is held between the product owners and team's leaders where they develop a backlog with activities that are needed from other teams. The stories in this backlog is integrated with the stories from the product backlog and prioritised into the sprint backlog.

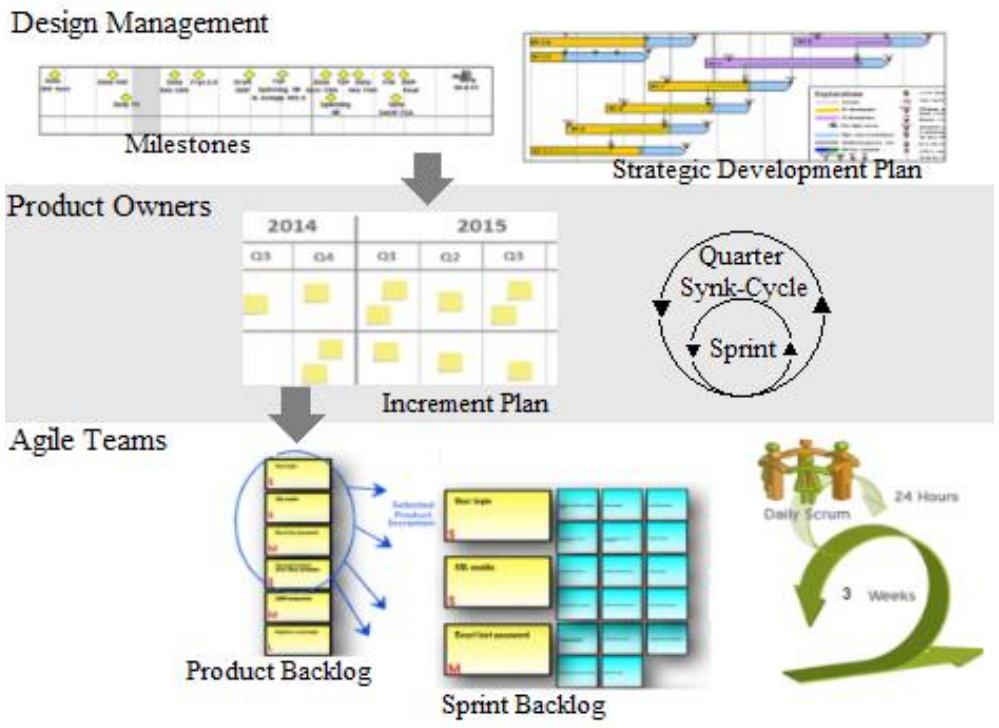


Figure 28: The three levels of the organisation with their typical activities

4.3.3 Case Study at Ericsson

Ericsson is one of the world leaders within information and communication technologies (ICT) with approximately 116 000 employees of which 24 000 works with R&D (Ericsson, 2015). The investigated department at Ericsson employs about 1000 people located in 5 sites all over the world. The department mainly works with software development of mobile networks. The data in this case study is collected from interviews with one product manager, who was part of the core team of the agile transformation at the department, and two Scrum Masters.

Ericsson agile journey, see Figure 29, began in 2010 when top management made the decision that the entire department would incorporate agile methodologies. In 2011 all personnel at the department underwent a Scrum course and they shaped their first cross-functional team, XFT. In 2012 they expanded the agile organisation and introduced multiple XFTs, this was also a big change for line management; who now was responsible for a number of XFTs instead of a traditional section or component team.

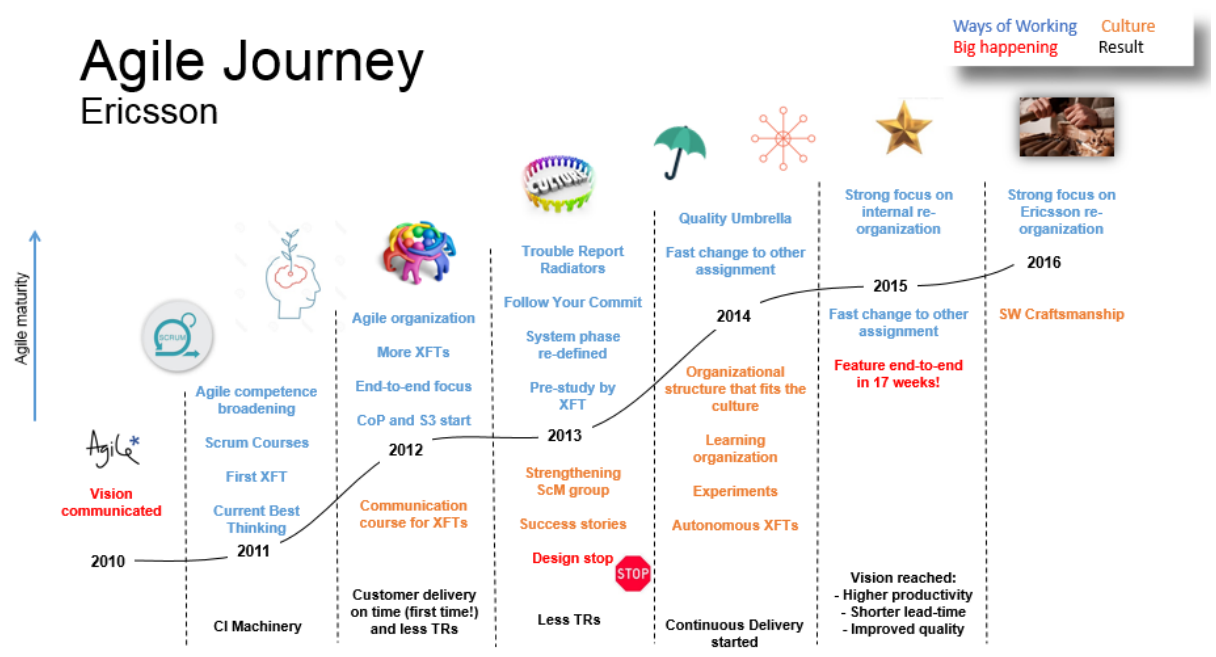


Figure 29: Ericsson's Agile Journey

Up until 2012 the R&D department worked more or less in a traditional project organisation. The projects were divided into different network generations and the organisation was functionally divided with the developers mainly working with only one component. Because of the component focus the organisation became very complex and it was hard to get an overview. This created specialist within each specific function but didn't enable collaboration between functions. The organisation was also considered rigid and decision making became slow since it had to be approved in many levels. The project manager was responsible to deliver updated version of the software twice a year.

The rigidness and slow decision making was part of the decision to introduce an agile transformation. However, the main objective for the transition was that the organisation desired to shorten lead-times and increase productivity, quality, and employee motivation. When a feature had been developed, it was first tested within the sub system, thereafter it was tested within the section and lastly it was tested at the top level. This meant that the feedback from the top-level integration could take up to 6 months before it reached the developer. By this time he or she had already began working on the next release and often did not remember the code. This meant that bug fixes took longer time than necessary and that the software in the release wasn't the newest one.

These problems lead the organisation to believe that they had to start working with continuous integration. They wanted to shorten the feedback-loops and decrease the lead-time. To do this a development environment was built where all code could be integrated and tested. The test cases were to be developed by the XFTs but to make sure that the complete environment was always up and running a CI-team was created to handle these issues.

Change process

Ericsson desired a change that could shorten their lead time and feedback loops so continuous integration could be possible. They were inspired by a department at Ericsson in Finland that had successfully implemented agile methodologies and had performed some major changes in their organisation. The department began with researching the area in 2011 with the purpose of investigating how the department could implement continuous integration and create a more agile organisation. The actual transition and implementation began in 2012.

The desire to change was divided in different level in the organisation and after the research it was decided by top management that they wanted to scale agile development throughout the entire organisation, within this department of Ericsson.

Core team

Ericsson set up a core team which could be responsible to plan, implement and support the transition. The core team consisted of a change manager, representatives from line organisation, project managers and technical experienced developers. The core team investigated agile fundamentals and came up with the concept Current Best Thinking (CBT). CBT is a concept which is a plan for how to operate in the best way so development can be effective, their agile core. CBT is continuously updated so the development process always is being improved. They decided to move away from component based sections and implement cross-functional requirement areas, in which cross-functional teams could operate.

When transitioning to cross-functional team which should work more agile, they had training and education for all employees so all would possess the agile mind-set. This since it is important that everyone knows the new methodologies and also why they are transitioning to this new way of working. The developers had training in Scrum, and a few selected had the SM training to gain knowledge of what the SM responsibilities are and how to support teams by coaching rather than leading. The OPOs had additional training and educations to ensure they possessed the competence and technical knowledge necessary to coach the teams. The typical OPO previously worked as a system engineer, team leader or project managers. They also had some workshops with management where they could learn the purpose and fundamentals of Scrum.

The department initially had many difficulties with the transition; it is a difficult change involving many people. They had external consultants and agile coaches to drive, motivate and support the employees over an extended period of time after the transition. The consultancy was also used to ensure that employees do not fall back into old processes and patterns when an obstacle appears. To ensure a smoother transition they held several kick off meetings, discussing what the new roles in the organisations would be and ensuring that each team knows their functions within the organisation. To create empowered teams they used a bottom up approach for solution, some issues needed to be lifted to management but if possible they tried to let the teams find solutions and take responsibility. Since the previous organisation had been broken up COPs, communities of practices, was used to identify the obstacles that were common for different levels or teams and focus on finding solutions for these. These COPs included the line organisation, OPO and SMs. In COP they also discussed definitions of done, how release programs should be organised and the quality of these deliveries. To reduce the lead-time, they tried to remove as much overhead and handovers as possible. They changed the functional R&D organisation into programs with program managers instead of the previous projects.

Team formation

The process of constructing teams was one of the aspects that the core team focused on in order to get a successful implementation. The teams were previously organised in various component areas or silos, A-E in Figure 30. When forming the new teams, they placed members from each silo in the new XFTs. The XFTs responsibility is to design, develop and test all features within the team. They previously had several system teams responsible for designing and providing the teams with specification but these teams were removed and the system engineers were placed in the XFTs. The department only kept one small system team, responsible for the pre-studies of market demands, called System Early Phases. There were previously also several integration and testing teams which also were removed and placed within the XFTs, only one System I&V team still exist and is responsible for the final verification of the entire product. Placing this amount of responsibilities in the XFTs was a big change for the department and they experienced some resistance in the early phases. To ensure the quality of the product would not decrease, specialists from each silo, or component area formed a Product Maintenance team. The Product Maintenance team not only ensured quality in their area of expertise but they could also provide the XFTs with their competence and support when the XFTs encountered assignments outside their competence areas.

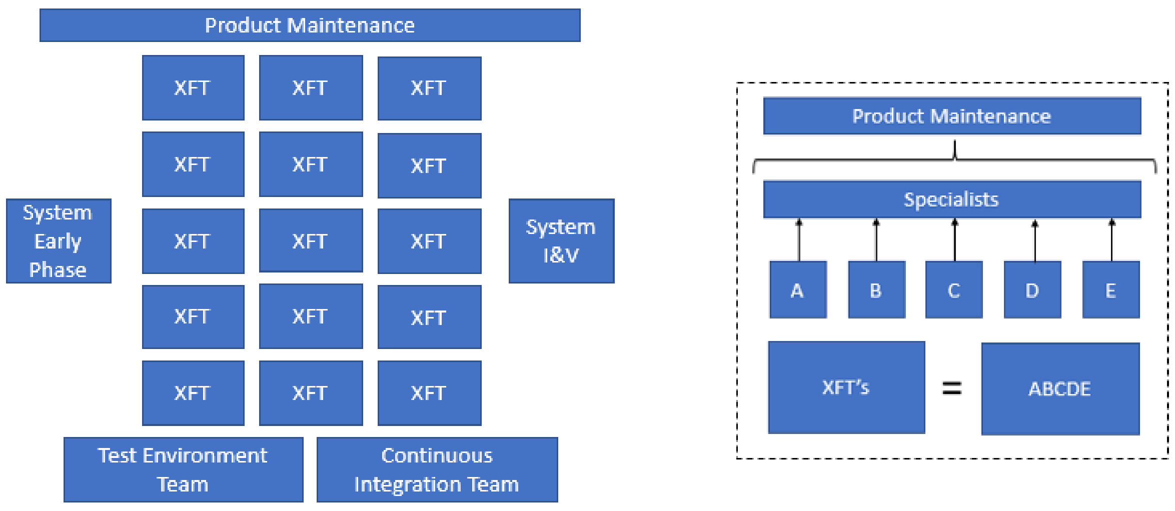


Figure 30: New structure at Ericsson

They contemplated the trade-offs that exists when forming teams. They wanted the teams to cover the necessary competences and still be able to work as one unit. The organisation realized that teams should have 7 members because bigger teams caused sub-groups and smaller teams didn’t meet the required competence. Smaller teams are also more sensitive to increased workload if a member is away as there is no spare room to switch tasks if the team is too small. However, they discovered that competences weren’t the most important when forming teams. Some teams with senior and skilled engineers were outperformed by teams with less experience and knowledge. This realization made them consider the personalities of the team members and found that the constitution of different personalities was one of the most important factors. Therefore, they looked at Belbin’s team roles in order to build teams that were more efficient and willing to learn in order to solve problems.

The new structure

Ericsson introduced a core team which set the overall agile principles that were implemented at the department. They also used external consultants for the evaluation and implementation of agile methodologies. The consultants worked as coaches for the team's, training them in agile methodologies and measured the departments agile maturity. The chosen approach is based on LeSS, Large Scale Scrum, with some adaptations to fit their organisation. The LeSS framework was chosen since the organisation is very large and complex and they needed a structured way to scale the agile development between teams and up to management. The implementation resulted in a new way to organise the work and new roles to manage the development process. The department previously had their teams divided between several components, they shifted this to instead have the teams organised in requirement areas. Each requirement area is represented by several teams which are responsible for developing the functions for these requirements. An illustration of the roles and teams implemented is illustrated in Figure 31.

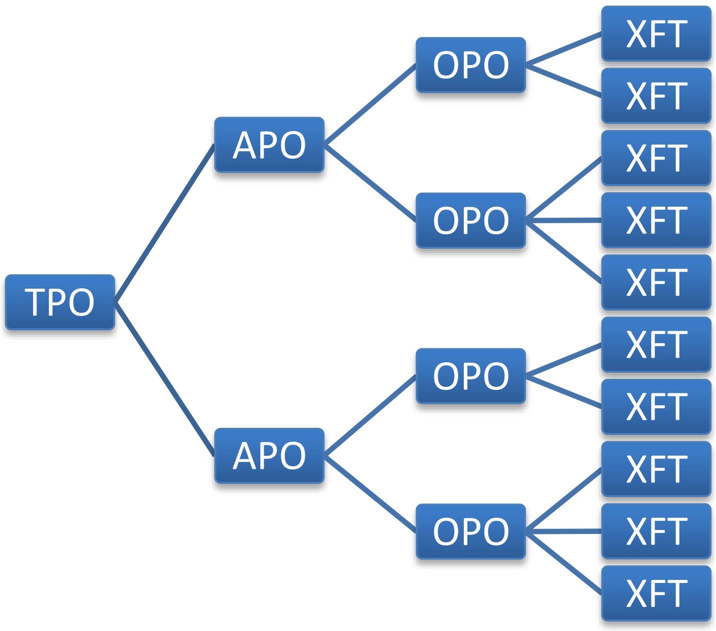


Figure 31: New matrix organisation at Ericsson

Total Product Owner (TPO)
The Total Product Owner is the overall person responsible for all development within the departments all RAs. The TPOs responsibilities are to ensure the development are aligned with the strategic business goals. The TPO has the overall backlog for the entire development; this is managed and distributed to RAs in collaboration with the APOs.

Area Product Owner (APO)
The APO is responsible for one Requirement Area, RA, and all teams working within it. The APO is also part of product management, investigating market needs and continuously following and updating the product's roadmap. The APO has 2-3 OPO's.to delegate and communicate with the teams. The APO has a main backlog for all teams within the RA based on requirements from customers and product management. The APO then, in cooperation with TPO and OPOs, breaks down the main backlog and divides the requirements to each OPOs' and their teams.

Operational Product Owner (OPO)
The OPO is responsible for 2-3 Cross functional teams and is a link between the XFTs and the APO. The OPO has a good overall view of what teams are working on and has continuous communication with the APO. The OPO breaks down the backlog items provided by APO to epics and features and assigns them to the teams after prioritisation.

Cross Functional Team (XFT)
The XFTs consist of seven engineers, five designer engineers, one system engineer and one testing engineer. Each XFT is collocated and all team members are dedicated 100% of the time to the XFT, team members can share competence to other teams but all development is conducted within the XFT. The teams are responsible to create specifications, designing, developing and testing all features in the backlog. The XFT is also responsible for the verification of all features before delivery. Each XFT has decided on how to organise their work. The majority of the XFT works according to Scrum with all events and artifacts, however, some XFTs do Kanban or hybrids of Scrum and Kanban. Even if the XFTs have organised their work according to different frameworks, they all work in the same sprints cycle of three weeks. At most they were about 100 XFTs are operating in the same sprint cycle and develops features according to their backlog.

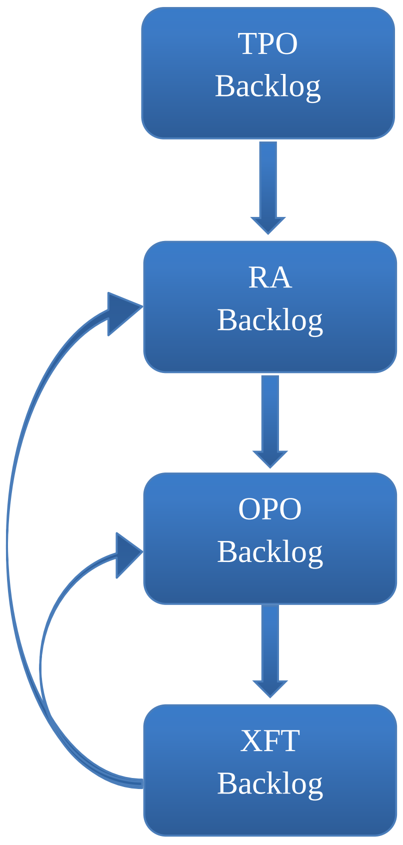


Figure 32: How a TPO backlog is broken down to XFT backlog

The schematics of how the TPO backlog is broken down to XFT backlog is illustrated in Figure 32. The TPO has a main backlog including the overall requirements within the department. This is to manage the development and maintain an overall perspective of the department's progress. The TPO communicates with the APOs and discuss what requirements belong in the different RAs. The RA Backlog is managed by the APO with input from customers and early phases (ER) department, which generates technical concepts and ideas. The RA backlog includes all requirements included in the RA and the APO is responsible that the requirements are correctly prioritised. The APO then breaks the backlog into OPO backlogs, where each OPO has a backlog containing the requirement within the operational area. The OPO is responsible for breaking the requirements in to epics and features that is to be developed. The OPO is also responsible to prioritise and estimate the size of features and then divide them among the XFTs. When breaking down requirements to features, the OPO always evaluates dependencies among features. They try to limit the dependencies among XFTs as far as possible to ensure team members can dedicate 100% of their time to the XFT. The XFT evaluate and estimate the features provided in their backlog and breaks them down to user stories. The XFT can also provide feedback on the feature in the backlog, if they should be corrected or is not estimated correctly. The feedback is sent back to OPO or APO depending on the feature size or how important the decision making will be.

Progress report

There are two types of progress reporting conducted at the department. One is a simple table where each XFT has a line, and all features are represented with a column. They then have a short stand-up meeting once a week where all teams and OPOs are participants. All XFTs then reports if progress is: following time plan with no impediments (visualised with colour green), behind time plan or some impediments but likely will deliver on time (visualised with the colour yellow), or if XFT is behind schedule or have big impediments and will not deliver on time (visualised with the colour red). This meeting was initially very time-consuming since XFTs often wanted to discuss impediments that might have arisen during the sprint. This has now been conducted for three years and normally takes around 10 minutes, and they have introduced other forums where XFT's can discuss arisen impediments.

The OPOs also has an additional progress report to conduct. Every week all OPOs meet to report and evaluate progress in ongoing sprint release and the coming sprint release. Which features the XFTs are developing, when they will be finished and what they will develop after that feature. This is done in collaboration with the APO, whom then can report to the release program when, what and how the features will be delivered and if any changes are necessary.

4.3.4 Benchmark Volvo

Volvo Group is a global manufacturing company producing trucks, buses, construction equipment and marine drive systems. The case study was performed on one of their R&D-departments mainly working with software development.

The Agile journey

The department had started an agile pilot project three years earlier to be able to perform specifications, development and testing within the same team. The suggestion to start working according to agile principles came from the developers who saw this as a solution to some of the problems they faced at the time. The pilot project performed their work according to Scrum and this worked so well that they decided to institute more teams. They had difficulties with finding the right functional areas for the teams and have experimented with different team setups. They are now trying to make sure that the teams deliver one product together, which means that if something needs to be changed they should be able to handle it within the teams. When forming teams, they focused on finding the right competences however people then stayed in their own roles and divided the work according to their former roles. They have tried to erase these former roles so that people don’t consider themselves responsible for certain areas within the teams, everyone should just consider themselves a team member. They have found that the teams need to be formed with the right personalities that complement each other. In order to be successful, they gave everybody a two-day Scrum education and used agile coaches to help the teams.

When the number of teams grew, they started to face problems with synchronisation. To solve this, they started up synchronisation meetings but they realised that this was not enough and therefore looked at different scaling frameworks. They chose to implement SAFe, both because it was the most used and because it was possible to implement step-wise. To implement SAFe they had help from external consultants who held educations, acted as coaches and took key operative roles in the organisation. They made sure to educate key stakeholders outside their organisation for them to know how their new organisation would work. The complete framework has not yet been implemented however the team and the program level is currently working well. They do not perform a two-day meeting for the increment planning but instead they use a big room planning event at the start of the increment.

They have instituted several new roles in the organisation such as product owners and release train engineers but people outside the agile organisation do not understand these roles. Because of this several people have two titles one title internally and one externally to ensure the rest of the organisation feels safer. It is hard to be agile in a waterfall organisation and currently this is solved by the leaders working as a screen to create an agile environment. Internally they work according to SAFe and Scrum but externally they still report in the same way as the rest of the organisation. The view outside the organisation might be that there is no planning within their agile teams even though they plan more now compared to previous projects. This can be since many in the rest of the organisation do not have any experience in agile methodologies and have created their own prejudices. They also perform demos for people within and outside the organisation. These demos have decreased the feedback length and increased the knowledge regarding their development throughout the organisation.

4.4 Expert interviews

To gather more data regarding success factors and potential pitfalls when performing agile transformations three consultants were interviewed. All of them had been part of agile implementations in different leading companies and having different roles. They also represent three different companies with different approaches, they are Scrum.org, Knowit and Cybercom. These interviews will be summarised together since no specific companies were investigated instead their combined experience and knowledge was gathered and is presented below in categories.

Implementation Strategy

When implementing agile development two different methods was discussed with the experts. Two of the experts proposed a step-by-step implementation were either certain method is implemented stepwise or letting different parts of the organisation adopt agile in steps. This creates the possibility to create internal success stories and also doesn't shock people with too many new concepts at the same time. It does take a longer time but does not create the same amount of chaos than an all-at-once strategy might create.

The other strategy proposed is the big bang strategy where all education and planning is done in advance and at one certain point the organisation shifts to agile development. This might create more uncertainties and a lower efficiency during the first sprints but it also lets the entire organisation work it out together. During the first sprints the teams will face problems to self-organise and the leadership must inspect and adapt to give the teams the best possibilities to succeed. This strategy allows the organisation to become agile faster but might be hard to perform in a large organisation.

Organisation Culture

All experts claim that the organizational culture is of high importance in order to become agile and not just doing agile. Scrum.org recommended Schneider's culture model as one that has been successfully used by consultants to quickly get an overview of the current core culture. According to him, the culture doesn't necessarily have to be a true collaboration culture but in order to get effective teams elements of this culture are necessary. The agile mindset is also closely linked to the culture and too much control can harm the effectiveness of the organisation. Knowit's consultant suggested that management should attend management 3.0 education in order to better support the teams.

SAFe
When discussing the methods and frameworks used for scaling agile development, SAFe became a centerpoint. All consultants claimed it was the most popular framework and that their customers often requested it. However, the views of the framework varied between the experts. One expert didn’t like the controlled processes in the framework and thought that they opposed the first agile value statement “*Individuals and interactions over processes and tools* (Fowler & Highsmith, 2001)”. He wanted the teams to be able to set their own rules and find the best way to collaborate instead of pushing a rigid framework onto them. The other experts however had a more positive view and had been part of successful implementations. They claimed that the processes supported empowered teams and allowed for control without controlling the ways of working within the team. The framework should however not be implemented without first studying the organisation and adapting it to fit the work and people in the specific situation. All roles, processes and rules are not necessary for all organisations and others might be.

The main theme of the expert interviews was to not be too focused on how other organisations had solved the problems. The key is to find what works for the specific organisation. This can only be done through trying and adapting, therefore courage to test ideas i of high importance.

4.5 Summary Takeaways Case Studies

The main takeaways will be presented below in different categories:

Teams
Volvo and Ericsson stresses that the teams are of most important part of agile transitions. To establish strong cross-functional collaborative teams that can learn and evolve together. Empower the teams and trust that they can make the right decisions, also make sure to provide them with the necessary support. One successful support function for Volvo and Ericsson was the use of operative coaches who can support the teams and help them when they encounter problems. They also gave the advice not only look toward competence when putting together a team, it is very unlikely that you will be able to cover them all in all teams, also consider personalities and how people will work together.

Agile Implementation
Suggested from all case studies is to have the courage to be agile when implementing agile methodologies. Do not be afraid to make changes but evaluate the result and change again if necessary. Also, communicate with the people outside the department during the transformation to make sure that they understand your new ways of working. It is important to communicate that there still is planning and documentation within an agile development team. Ericsson and Volvo also stressed how important, and successful, the use of external consultants was for their implementation. To use the consultants to educate and train but also to have them as support so the department does not fall back to old processes when obstacles occur. Volvo also recommended having the external agile coaches acting in an operative role; this helps the agile coach to better understand the development process rather than acting as an observer. Department X at Saab only used external consultants in the preparing phase of the transition

and also for educating all members in agile mind-set and Scrum. Not having the consultants as support during the transition made it more difficult for them to align the new methodologies with their department.

Scaling agile

Ericsson used LeSS to manage and scale their agile development, introducing PO’s in various levels in their organisation and using them to synchronise teams and deliveries. Volvo has begun to implement SAFe, using an external agile coach as their agile release train engineers to manage the synchronisation of teams. Department X at Saab has not implemented any specific framework to manage and scale their development, they have created an integration plan instead. Their integration plan has been a great success for them in their agile journey and is an easy way to plan and communicate progress and dependencies among teams. Department Y at Saab uses a structure reminding of LeSS but only with one level of PO’s. The PO’s continuously get together to prioritise backlogs and synchronise the teams. Department Y at Saab do not have the same issues with dependencies among teams since the department is very large and they keep the dependencies among teams very limited.

5. Analysis

The analysis is a combination and reflection of the theoretical and empirical data found during the investigation. The analysis is described in four sections: agile methods, scaling agile development, organisational culture and building effective teams.

5.1 Agile methods

Several agile frameworks and methodologies have been introduced and explained in the theory chapter. Agile development sets apart from traditional waterfall development by working in short sprints or cycles. In each sprint or cycle the teams design, develop, and test functioning software in collaboration by working in cross-functional, self-organised teams. Three of the most used today are Extreme Programming (XP), Scrum and Kanban. They originate from software development but several attempts have been made to incorporate them in other fields.

From the interviews at Saab it was clear that the various subprograms and system teams work in different ways and there is a desire to find a common way of working. The majority of the teams at the subprogram Sensor work according to Scrum, with all events and artifacts. Some of the teams at Mission System work with Kanban, not using the Kanban board to the full extent but work according to pull system and has stand ups where they communicate what they are working on and the progress. Other departments work more traditional with similarities to waterfall process and only follow the increment plans. It is common that teams work with different methods within an organisation. All investigated case companies have teams which operate in different ways, but they have some rules which creates a common frame to hold the teams together. Department X at Saab has teams working according to Scrum, Kanban or combinations of both. They keep the teams together by working according to the same cycle time and by the integration plan. The majority of the teams at Ericsson works with Scrum but some teams work according to Kanban or combinations. They also have all teams working in the same cycle, and they also had Current Best Thinking, CBT in the beginning of their agile journey. The CBT was “the best way of working right now” and it was followed by all teams. The C, current, means it evolved over time and today it is no longer necessary since all teams are so used to the new way of working. All the interviews performed during this investigation has showed that there is no need for having all teams working according to the same framework or methodology, all teams must find their own way of working but there needs to be a supporting system which holds them together.

Many subprograms and teams at Saab has a desire to work more according to agile methodologies to shorten feedback loops, reduce lead times and increase to collaboration between subprogram. The case studies show that Scrum and Kanban are possible methodologies, even though the majority of the development in the studies regards software it is integrated with hardware which suggest it would be possible for Saab to incorporate these methods. A suggestion from Ericsson and Volvo is to have the courage to try, inspect and adapt. It will most likely not succeed at the first attempt but in the end, it will be worth the effort since both Volvo and Ericsson has improved both quality and productivity since they made the transition. The agile manifesto’s third principle, respond to change over following a plan, is not only for the development of the product but also the development of the teams. After initial attempt the teams must evaluate and have the ability to change their way of working to fit their purpose (Hunt, 2009). One of the pillars of Scrum is inspect and adapt, this must be one of the fundamentals for the teams, to have retrospectives and evaluate their work and make continuous improvements.

Introducing Scrum or Kanban in the teams at Saab is appropriate to initiate the agile transition, so all better can understand the agile mindset. Using Scrum or Kanban helps the teams to break down their work into smaller assignments or user stories, which is necessary to even make it possible to work in short sprints or cycles. Especially one subprogram, Mission System, find it difficult to break down their assignments to smaller tasks or user stories. Some feel that they should make the attempt to break the assignments down, not only to enable shorter cycles but to make it easier to collaborate on the work and not work individually.

Having the teams evolve and find their way of working is also how self-organised teams are established (Schwaber & Sutherland, 2016). The teams get assignments from management but it is up to the team to decide how to organise the work to be successful. S. Wheelan (2015) also states in her framework for creating effective teams that it is crucial in team forming that the team decides together on team procedures to reach their goal. This builds commitment to the team and its purpose and will increase their long-term productivity (Wheelan, 2015). Transitioning to agile methodologies is not only a change for the engineers, it is also a big change for management. To establish self-organised teams, management has to delegate responsibilities and trust the teams. Pushing technical decisions down to the teams can be scary, but management can keep control by introducing a system which provides good progress reports and communication. This can be done in several ways, increasing face-to-face communication, using visual aids and by having a common framework for scaling agile development.

5.2 Scaling agile development Frameworks

The three agile frameworks that were presented in the literature review represents different levels of control and overhead. However not much research regarding how to choose a framework for scaling agile development has been found and a reliable recommendation to Saab couldn't be found. Therefore, this study interviewed practitioners using each of the frameworks in order to investigate them further. The key factors investigated were the benefits and drawbacks with each framework and in what situations they were suitable. SAFe was studied in practice at Volvo and further investigated during interviews with consultants. Ericsson was used as a benchmark company to investigate LeSS and in order to examine Nexus a representative from Scrum.org was interviewed.

The necessity of a framework is not obvious, if teams simply choose tasks from their prioritised backlogs the overall development would still progress. However, in order to be efficient when several teams are working together coordination regarding dependencies, deliveries, cadence and integration are needed. If teams work without consideration of other teams they sub-optimise and this leads to problem in the aforementioned areas. A framework is not supposed to interfere with how the teams work internally but rather provide a set of rules to enable a more efficient development.

SAFe

SAFe is the most commonly used framework for scaling agile outside of Scrum-of-Scrum techniques (Version One, 2016). It has been adapted by several large companies mainly because of its clearly defined processes which makes it easier to handle complex systems. However, this has sparked a debate among practitioners of agile since agile development was created to get away from rigid processes, extensive reporting and excessive documentation. Processes such as a two-day incremental planning meeting does not support the main idea behind agile, to focus on delivering a working product through solving problems within the team. However, many of the opponents of SAFe are working in smaller organisations which are not as affected by the problems that agile aims to solve, coordination and communication issues. When talking to organisations that uses SAFe the main reason for implementing it was to be able to synchronise their teams both in regards to deliveries and dependencies. The other frameworks provide opportunities for this as well, however SAFe provides tools for visualization and a structured two-day event during which increases communication. Another reason for implementing SAFe were the release train which enables a new way to handle deliveries and prioritise the backlogs. The possibility to further scale the agile principles and keep the same processes in the lower levels was also considered a reason to use SAFe.

To summarise, SAFe is considered to be a good alternative for large organisations. It allows for control and transparency in the management level while still creating an agile environment for the teams. However, it is considered by some to be too rigid and requires too many administrative activities.

LeSS

The two versions of the LeSS framework creates the opportunity to fit both large and small organisations. It doesn't require as many activities as the SAFe framework but it does provide a structured way to deal with dependencies and prioritise the overall development. The study at Ericsson has showed that the framework works in a large organisation of 1000 people which means that the frameworks is a viable candidate for Saab. However, it should be noted that Ericsson has modified the framework and created communities of practices in order to combat some of the communication issues. The framework contains more activities than Nexus even in its smaller version. Therefore, it is not as focused on the development activities and requires more time spent on administration and planning. The framework can be modified to fit many companies in different sizes. However, no information regarding how it would work in a more complex environment with both hardware and software development was found.

Nexus

Nexus is one example of a framework that was developed for software companies that wants to stay true to the agile principles. To enable several Scrum teams to work together with one project coordination is necessary and Nexus tries to use the techniques in Scrum to accomplish this. The number of extra activities outside of the normal Scrum activities are kept to a minimum for the teams to be able to focus on their development. This creates an environment where the teams can operate without much outside interference since cross-team dependencies are kept to a minimum. However, this method is not as applicable when the organisation grows and with more complex products. In these scenarios dependencies will exist between teams and more communication will be necessary.

Implementing a framework

During the study two different schools of thought were found regarding how to implement these frameworks. Some practitioners argued for a big bang implementation while others suggested that it should be adopted in steps. The main arguments for a big bang implementation is that it is quicker and allows the organisation to see the new structure immediately. However, it requires the organisation to inspect the transition and quickly adapt in the areas where problems occur. According to the proponents of this practice this enables the organisation to quickly reach a higher efficiency within the new framework. The other idea, to implement the practices step-by-step, claims that large organisations can't handle a big bang implementation. They claim that if implemented all at once the new structure will create confusion and significantly hurt the development during the implementation. This can create resistance towards the new framework and might cause the implementation to fail. Instead they propose to start with a few teams and make sure that they adapt it smoothly. This can then be used as a success story in order to motivate the other teams to adopt the framework. This means that the implementation will be longer but allegedly increases the chance of success.

5.3 Organisational Culture

The different core cultures are useful in different contexts and an organisation will build their success from this culture. In order to transform to an agile organisation, it is important to have a culture that supports the agile methodologies. To determine what culture is preferable to support an agile organisation the values from the agile manifesto will be used:

Individuals and interactions over processes and tools - To promote interactions and teamwork a collaborative culture is beneficial. This culture will enable collaboration both within and across teams in order to solve problems instead of for example following a problem-solving methodology (collaboration culture).

Customer collaboration over contract negotiation – To gain a closer collaboration with the customer there is an obvious need for a collaboration culture. Instead of controlling the customer with contracts and agreements it is important to work together to create the best product.

Working software over comprehensive documentation – To create working software both competence and collaboration is needed. Competence regarding development of the product is necessary but collaboration between people with different competences opens new possibilities. However, the control culture would stress comprehensive documentation and therefore goes against this value statement.

Responding to change over following a plan - To be able to quickly change an organisation must depend less on plans and focus on what is most important for the customer now. This means that management must let go from their control and let the development organisation focus on the highest prioritised task (Highsmith, 2001).

What can be concluded from the above analysis is that organisations can't have a controlling culture in order to successfully becoming agile. The culture that supports the transition is collaboration which allows the teams to perform their work undisturbed and enable collaboration both within and outside teams. The process of changing the core culture of a company is closely inter-linked with creating the agile mindset. Companies who have a control or competence culture will have a longer journey towards an agile implementation. To create an agile mindset and a collaboration culture the driving forces in agile must be understood. The main idea is that a team that works together over time and share a mutual goal will be more effective than individuals working by themselves. An organisation can adopt agile tools and methodologies but still not become an agile organisation. In order for this to happen the culture and mindset of people must support teamwork and empowered teams. To change this both the experts and the case companies claims that coaching is necessary. An objective coach that can support teams and leaders to remind them of why they perform their actions and provide support for struggling teams. Practitioners also suggest not to talk about changing the culture with the team members, instead management and leaders should be educated to support a new culture.

It is however important to work with the current culture and not oppose it. This means that the company will not be forced into a new culture but instead will be guided to start exploring new cultures. For example if an organisation currently have a control culture Kanban can be implemented which support that culture better. However, Kanban can increase collaboration and also allow certain functions to go further and start working with Scrum. This creates a road to a more agile mindset for the organisation. It must be noted that all organisations don't need an agile mindset, it might be better for them to adopt Kanban and make use of the benefits in this method. Ericsson tried to change their core culture from control to collaboration, they were successful since management lead the change and supported the growth of a new culture. It was however not an easy transformation and it didn't become as successful in other parts of the organisation.

In the case of Saab, the results from the culture survey suggest that there might exist multiple cultures within the organisation. Therefore it is likely that cultures have developed within the subprograms that are not the same as the overall baseline program. When discussing this with the respondent they claimed that they would have answered different depending on whether the entire baseline program was investigated compared to their subprogram. Some subprograms currently work with agile methodologies and has adopted an agile mindset it is therefore expected that they have a collaboration culture. This can be useful for Saab when they put together their cross-functional teams, the employees for these subprograms can be given responsibility to spread the agile mindset. However, leaders are still influencing the culture more than anyone. The survey revealed that management think that the current culture is more collaborative than it is. This implies that to further enable a collaboration culture the management team needs to adapt a new philosophy based on trust in their team's capabilities. Control mechanisms needs to be kept to a minimum to let the teams focus on their main tasks. To change a culture is hard, but there are champions of a new culture within Saab today and by utilising them the transition will be easier.

5.4 Build effective teams

Both the literature study and the case studies of other companies stress the importance of incorporating effective, cross-functional teams. For any development, the teams are the foundation of the process towards the delivery and goals. Agile methodologies and framework states that the teams should be cross-functional to ensure quality and productivity in the development (Schwaber & Sutherland, 2016). The first principle of the Agile Manifesto, individuals and interactions over processes and tools, states that the people and the teams is what has the biggest impact on success. Many focus on introducing new processes and tools to become more effective and increase quality. Having methods and tools which support the development is important, but many organisations do not put the same effort and resources to create effective teams (Hunt 2006). Another big factor stressed in most methodologies is that the teams should be autonomous and self-organised to reach maximum productivity. The case study at Ericsson showed that their biggest success factor are their teams. Transitioning from component teams to cross-functional teams were a big challenge but the outcome has been tremendous. Scrum and Kanban includes many visualisation and communication tools to make teamwork more transparent, but they are both also founded on the establishment of effective, cross-functional, self-organised teams (Schwaber & Sutherland, 2016). From the interviews at Saab it was concluded that they had taken the first step towards cross-functional teams by introducing the system teams, which representatives from various subprograms. However, the system teams do not collaborate to a large extent when conducting the work, much work is performed individually and the system teams are more used as meeting forums in which they can receive new assignments or report progress. This is characteristic for the first stage teams, in the framework by Susan Wheelan, Creating effective teams. The teams in the first stage is not a team, it is a group of people with the same goal but where the work is conducted by individuals and not by collaborated teamwork (Wheelan, 2015).

There is a desire among the personnel at Saab to increase the collaboration and work more cross-functional, especially to increase collaboration with I&V. Many of the subprograms want to include I&V in the earlier phases of development, to use their competence and plan for test cases earlier. On the other side, I&V often feel somewhat isolated in the final phases, left alone with the responsibilities while other subprograms begin new assignments. Scrum and other agile frameworks promotes cross-functional teams where each team is responsible for all phases of development. This also decreases the lead time and shortens the feedback loop, by limiting the number of hand-overs. Ericsson implemented a new team structure during their transition to agile development. They previously had component teams but changed to cross-functional teams including one system, five design, and one testing engineer. Introducing the new teams were initially a challenge but it has resulted in great collaboration and shorter lead times for the teams. Saab has a need and desire to introduce more cross-functional teams, but it is difficult to cover the competence necessary in each team. Many of the engineers at Saab has long experience and possesses a high knowledge in their area of expertise. If introducing cross-functional teams they must find a way to spread the competence or the engineers must be a member of more than one team. Ericsson had the same issue in the beginning, they chose to introduce a "Product Maintenance team", consisting of experts from various areas. The product maintenance team is responsible to ensure product quality in the various areas, and support teams that need extra resources or consulting in projects. All case studies showed similar difficulties, and their recommendation is to have the courage to try. If teams are reorganised to cross-functional, there will be a big challenge in the beginning, but the engineers are forced to learn new areas to deliver their tasks. As said above, both Volvo and Ericsson had difficulties to cover the competence necessary when introducing cross-functional teams. They instead put some effort in evaluating what type of personalities were necessary in each team, and match this with their personnel. Both companies agreed that evaluating personalities were beneficial, focusing on establishing teams which could grow and learn together. If a team consist of people who enjoy working together and complement each other they will evolve and learn new skills easier.

Some of the engineers at Saab are members of more than one team, this makes it more difficult to prioritise the work. Both the literature and the case studies show that it is beneficial for teams to be formed into a unit or small "family", it makes them much more efficient. For a team to be formed into a collaborated unit they must work together over a longer period of time. Being in more than one team makes it more difficult for the member to be formed in the team correctly. Not having functioning teams makes it difficult for some members at Saab to understand the team's purpose and what is expected from them. It also makes it difficult to understand their role within the team, and the organisation. According to S. Wheelan (2015), one of the foundation for effective teams are for the team to understand their purpose and create a mutual understanding for the team's goal. It is hard to identify a member's role, and accepting the role, if the purpose or goal for the team is not understood.

From all subprograms at Saab it was stated that there were some communication difficulties. The engineers are often not informed about other team's progress or what others are working on, this makes it difficult to identify dependencies among teams, and what is to be delivered to the system rig. Agile frameworks such as Scrum and Kanban uses visual aids, such as Scrum boards to visualise progress. They also promote face-to-face communication as often as possible. Having daily stand ups, where everyone shares what they are currently working on and if they have any difficulties. They also use reviews or demos, where the team presents what is delivered to everyone interested. This increase the communication and makes it easy to access for everyone (Schwaber & Sutherland, 2016). One of the studied departments at Saab, department X, has introduced these types of demos and it has improved their development a lot. Visualising what has been developed to everyone provides the opportunity to ask questions and give feedback, and it has made it easier for them to find errors and defects earlier in the process. Department Y also work according to an integration schedule instead of the increment plan. This was the big breakthrough for Department X, it made it much easier to follow progress and identify dependencies among teams.

6. Results

The results from this research is presented below answering the sequence of the research questions.

What different Agile Product Development methods exist and how can they be applied for development programs which integrates hardware and software, such as at Saab's baseline program?

There are several agile frameworks and methods were three of the most established are Extreme Programming (XP), Scrum and Kanban. Both from the literature study and the case studies it was found that many organisations and teams work mainly with Scrum and Kanban, but it is also very common that they use combinations or hybrids of the two. The case studies confirmed that organisations can have successful development using Scrum, Kanban and hybrids of the two. Even though the studied companies mainly develop software, they all frequently integrate their product with hardware, such as at Saab. All the visited companies stated that it was a difficult transition, but that the results have made it worth it. Their teams are functioning as more collaborated, cross-functional units, and their quality and productivity has increased. When implementing agile methods to teams, it is important to let the teams be self-organised and find a method that fits the team. This is most likely why so many teams today use hybrids of Scrum and Kanban. Some teams might have assignments that are difficult to break down to user stories small enough to complete within a sprint so they choose to use Kanban instead. But Scrum offers several other events that are beneficial, such as daily Scrum or retrospective, or having a Scrum Master (SM) which handles impediments or external interruptions. So many teams may work with Kanban pull flow but include events from Scrum. Finding the right fit for a team is something that must be continuously evaluated and improved, what works for a team today may not be the best fit next month. Most of the studies companies had all teams begin with Scrum since it trains the teams in many of the agile events and artifacts. After a period of time the team may switch to Kanban, and one of the foundations of agile is to be agile, to try, inspect and adapt.

What are potential benefits and drawbacks from scaling the agile development to the different subprograms, from developers up to program management?

To find the benefits, the identified problem areas were used in order to see which of them could be solved by scaling agile development. The first problem area was ways of working, where unclear roles, functional teams and a not having a common way of working was considered a problem. This can all be helped by scaling agile development, by using one of the presented frameworks all teams will follow the same process while still being able to self-organise internally. This will also remove the number of roles within the organisation, since the team will mainly report to their product owner and that the old-line organisation will mainly allocate resources. The framework will also provide a simpler structure than today's complex organisation which will make it easier to find the responsible person. When scaling agile development cross-functional teams are necessary in order to be feature-oriented. The idea of one team developing one customer feature from start to end has been one of the advantages in the case companies.

The second problem area was communication and information. The problems identified here included a lack of synchronisation between teams especially regarding dependencies. There is also no way to easily see what other teams are doing and if they will deliver according to plan, this creates problems with integration and testing. To solve this the different framework has various solutions, however all three stresses the importance of transparency between teams and the need to identify dependencies. The need for cross-team communication will also become smaller since the number of handovers will be fewer when more work is performed in cross-functional teams. The activity to identify dependencies will still be important and knowing what team are dependent on your activities provides more information in order to prioritise activities. To be able to see the progress of all teams will make it easier to plan the installations in the system rig to avoid queues. The possibility to easier see the entire program's success also provides the employees with a better knowledge regarding how they contribute to the overall goal and increases motivation.

The last identified problem area was requirements and testing. To be able to verify requirements the rig can’t accept deliveries since many installations are time consuming, this prolongs the feedback cycle. There is also an issue regarding how requirements should be developed, some deliver detailed requirements while other start with describing functionalities. The integration will be helped by the fact that the testing will be performed by members of the team which means that they have knowledge about the delivery and are able to easier handle problems that arise. The cross-functional team will also have more knowledge regarding testing which means that the verification in the subsystem rigs can be performed with better precision. This leads to higher quality deliveries and less struggle in the system rig. The frameworks also add the possibility to create a support team that is responsible for integration issues, this has been a success at Ericsson. The cross-functional teams will also simplify the development of requirements since this is handled within the teams. This creates the possibility to iteratively establish the requirements, where broader design specifications can be used to start the development and what is learned from this can be used to further specify the requirements.

The drawbacks from scaling agile development, or implementing agile methods is that it is time consuming and resource demanding. Ericsson has very successfully implemented agile methodologies into one of their department. But their agile implementation has been running for six years. The implementation at Ericsson did however see the benefits from agile methodologies early in the transition, but the first major success stories occurred after about two years. Department X at Saab also had many difficulties in the initial phase of the transition, when making adaptions of the methodologies to fit their department. The project managers at the departments stated that it will take time, and it you will not get it right the first time, you must try, inspect and adapt.

Introducing new methodologies is also resource demanding, if the teams are expected to work with agile methodologies, all members must be provided education and training. The education and training is also for management, so the leadership is aligned with the new way of working. Agile leader's delegates large amount of responsibilities to the teams and acts more as a coach or consultant supporting the teams. It is difficult and scary to let go of responsibilities and put the trust in the teams, but delegating responsibilities does not mean the leaders will lose all control. The literature study and case studies showed that there are many ways for managers and leaders to keep control and have good overview of the development progress using frameworks. Ericsson uses the LeSS framework to manage and maintain their development while Volvo uses SAFe. When incorporating a new framework into an organisation, it is important to use external support such as agile coaches who educate, evaluate and keeps the organisation from falling back to old processes when obstacle occurs. Ericsson had their agile consultants in the organisation for about a year, Volvo still has agile coaches working in an operative roles full time. Education and the use of external consultants is expensive and resource demanding, which is one of the largest drawbacks from transitioning to agile.

Even though the journey toward an agile organisation is long and difficult all benchmarked companies were pleased with the results. They knew that their organisation had become more effective because of the transition. Ericsson mentioned that earlier it would take about a year for a feature to be developed from customer order to delivery. Recently they were able to do this in 17 weeks which can be attributed to the reduction of handoffs and shorter feedback cycles. This is just one of many agile success stories and in order to efficiently scaling agile it is important to know communicate the expected results to receive buy-in from key individuals.

What are the important factors to consider for Saab when scaling Agile Product Development and how can they make the transition?

Preparation phase

Any organisation that want to implement agile development must first investigate what problems the implementation is trying to address and understand the current core culture. Then an evaluation must be made to determine if agile development will solve the problems and if the organisation is ready for the implementation. It is also important to analyse if agile development will help the company reach its business goals. If an implementation is considered useful a sense of urgency for the adoption of scaling of agile development is necessary. The organisation must communicate the existing problems and how these will be solved by the implementation. Instead of creating the willingness to change through fear of competition it can be done by getting the people intrinsically motivated to work in a more agile environment. Success stories from other companies can be used to show the benefits of working in an agile organisation. A change team consisting of informal and formal leaders can be used to spread the message effectively. This team should create a vision of how the organisation will work when agile development has been adopted. It is up to them to find a scaling framework that fits the organisation needs. The evaluation of the organisation will provide input to this choice, the higher need for control the more rigid framework should preferably be chosen. The more rigid frameworks are less effective but offers more synchronisation and visualisation activities. If managers within a control culture see that there is still organised processes in the organisation even if the teams are self-organised, the transition might seem less threatening.

When the vision is developed, it must be communicated in the organisation to make sure everyone understands the reasons behind the choices in the new organisation. However, people within the organisation might not have sufficient knowledge regarding agile development and won't understand the reasoning. Therefore, education is of high importance, everybody in the organisation should be educated in agile development to not only understand the principles but also which methods exists and why these are needed. To follow the education schedule suggested by Scaled Agile (2016), where the core team is educated first, then the management and lastly the teams, is suggested in order to receive buy-in from the right people. The biggest barrier to agile adoption is the organizational culture (Vision One, 2016) and therefore this must be a main focus in the transition. Education is useful, but to change behaviours and mindsets constant reminders of the principles behind the methods are needed. Before the transition begins the strategy of the implementation must be determined. Either the entire framework is adopted at once everywhere in the organisation or certain parts of either the organisation or the organisation is implemented step-by-step.

Transition phase

When the actual transition begins, even if all involved personnel has undergone education and are informed of how the change will occur, it will be difficult and many obstacles are likely to appear. The case studies and expert interviews showed that it is extremely beneficial to use external agile coaches to support and train all involved in the new way of working. It is important that the core team always support and motivate the teams, ensure the teams conducts retrospectives and self-assessments to find their own way of organise.

Team level

Initiate the transition one sprint at a time, after 2-3 sprints there should be a large evaluation process to inspect benefits and drawbacks after which improvements should be planned, scheduled and implemented. The teams are likely going to need much support from management and agile coaches, otherwise people often fall back into old processes when obstacle arises. Ericsson and Volvo highly recommends having agile coaches over a longer period of time to support and maintain the new way of working. Volvo, who uses the SAFe framework for scaling and monitoring their development, recommends having the agile coaches working permanent in operative roles. Using the coaches in operative roles involves them more in the organisation, the organisations obstacles becomes their obstacles too, rather having team observing as outsiders and not understanding the organisation or its development issues. As in Saab's case, SAFe is an appropriate framework for scaling and managing agile development. However as SAFe brings several new roles and tools, such as Agile Release Train and Agile Release Train Engineer, having an agile coach in this position is a recommendation to support these new tools and roles. The experts interviewed in this investigation recommends initiating all teams with Scrum, so all teams are trained in the same way and learn all events and artifacts of Scrum.

Program level
It is important for management to show support for the change, if they start to show distrust in the methods it might spread within the organisation. Ericsson and Volvo admitted that the first years contained struggles and difficulties. It is in these times that management must act as leaders to assure continued belief in agile development from everyone in the organisation. Instead they must trust in the change and it is also important that they refrain from controlling the teams and instead offer support and guidance, the teams must be given time to self-organise.

Maintenance phase
When the transition is complete it does not mean that the work is done, organisations using agile development aims to continuously improve. The main activity to assure continuous improvement is retrospectives, they should be held after every sprint. It is essential that the ideas that are discussed in the retrospectives are taken care of. If the teams do not see changes occurring from their ideas, they will not be inclined to come up with new ones. All ideas do not have to come from within since it is easy to evolve a group-think culture in any organisation. It is part of the leader's responsibility to stay up to date with the latest methods and ideas that professionals and academics presents and determine whether they are useful to their organisation. The use of pilot projects lets the organisation test new ideas in a smaller scale before implementing them. It is important to have the courage to launch these kinds of projects and let them succeed or fail. Inspect and adapt are very important keywords also within the maintenance phase.

7. Recommendation Saab

Saab's agile journey has already started but to take the next step to cross-functional teams and a scaling agile development further work is necessary. The recommended framework for Saab is SAFe since it is concluded that this framework is most aligned with their organisation. It has also been concluded that it is most suitable for Saab to make the transition step-by-step and not all at once, which is possible with SAFe. The recommendations are divided in short-, and long-term, where the short-term recommendations are focused on establishing effective collaborative teams and the first steps towards an agile release train (ART) which is part of the fundamentals in SAFe.

Short-term recommendations

If Saab decides to strive for more agile development it is highly recommended to let all personnel within the baseline program to take a basic education in agile development. It is very difficult to implement this new way of working if the personnel are unfamiliar with the methodologies and does not possess the agile mindset. So before introducing any the long-term recommendations Saab should have a minimum of one day training for all. Saab has today already incorporated cross-functional teams to some extent in their program, but the collaboration and shared responsibility for the assignments has not been successful since the system teams are mainly used as meeting forums. A possible reason for this could be that each subprogram has a line manager in which the team members report, delivers and can also get new assignments, see Figure 33. If the members of a team have separate managers to report and deliver to, it is more difficult for the team members to develop and deliver as a team. To create effective teams the teams must be formed together, and feel committed to the team and the conclusion drawn from the researchers is that the members are committed to the subprograms and not to the team. A recommendation for Saab is to restructure some of the management of the teams, and introduce a PO or team leader which is responsible for a set of teams and provides them with assignments. Having PO's responsible for the teams, right side Figure 33, provides the opportunity to better collaboration and responsibility sharing in the teams, having assignments provided to the team and not to individuals in the team. The line managers will keep their traditional responsibilities such as distributing resources hand handling salaries and new employments but must let go of some of their control in monitoring its personnel's development progress.

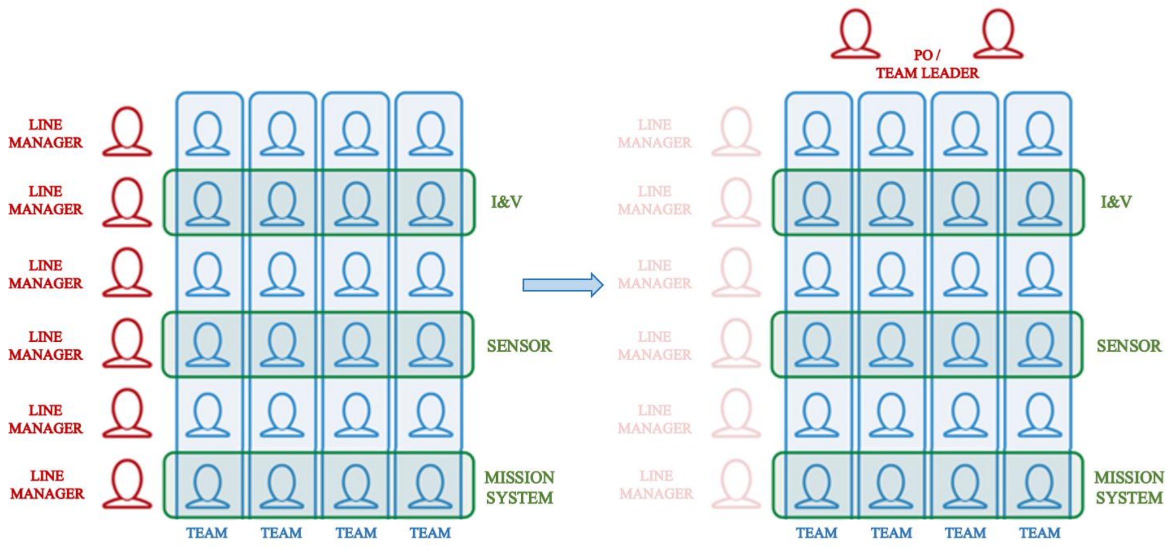


Figure 33: Proposed new structure for managing teams

Another characteristic for effective teams is that they are self-organised and decides within the team how to organise and conduct the work. Having the teams self-organise does not mean to let go of all control, the team needs to be provided with a set of rules or a guideline which is compulsory for all teams. The guideline should include a cadence so all teams can synchronise their deliveries and a clear definition of done. Today integration is only conducted three times a year and it is recommended to increase the number of integration so the time for installations can be less difficult and time consuming in the system rig. Saab could for example begin with a takt time of four weeks, where each team has a set of assignments to deliver and integrate at the end of the fourth week. The definition of done is an agreement between all teams which ensure that the assignments are truly done, regarding not only functionality but also the quality. When working according to definition of done it is very important that all involved understand the definition and that it has the same meaning for everyone.

To more clearly map up dependencies and create a clear picture of the overall development in the baseline program it is recommended that Saab introduces the PI planning activity from the SAFe framework. The PI planning is a big room planning event where all teams plans their work which is to be performed in the increment. Planning the work means discussing the assignments, what they will include, what dependencies it has from and to other teams and a rough estimation of effort and time. If all teams plan at the same time and close to each other it is easy to discuss dependencies and ensure the sequence of assignments are correct. After the planning is done all teams will shortly present to the other teams what they will do during the increment and in which sequence. The PI planning ends with a vote of confidence, meaning that each team member votes that the amount of work is possible to conduct during the increment, there is no point to have an increment plan if the members do not believe the plan is possible to follow and complete. This two-day event might initially seem like a time-consuming waste but it provides the opportunity to replace several of the meetings that are held today. Several of the planning and prioritisation meetings that are held today can be erased and people will not need to search for information regarding status for certain features. This activity both increases the cross-team communication and allows for more correct prioritisation. The planning activity can then be used as the foundation to develop an integration plan such as department X at Saab. The advantage of this visual plan is that, if continuously updated, it provides an instant overview of the state in all development teams. That overview can be used in the system rig to better plan the activities and avoid queues or unnecessary work. The integration plan requires the teams to start working in the same cadence to be able to plan the deliveries. This might be a too large step for some teams and they might need to only deliver every other sprint. A suggestion is that the IVA manager in collaboration with the system managers will be responsible for creating the integration plan after the PI planning event. Since the subprogram managers today are responsible for synchronising deliveries between the teams the suggestion is that they are responsible for continuously updating the integration plan.

Long-term recommendations

The long-term recommendation for Saab is to implement the SAFe framework for scaling agile development. SAFe is recommended since the framework provides the opportunity to make the implementation in smaller steps and not all at once. This allows for development to continue during the transition period and creates a smoother transition. SAFe also includes tools, such as the agile release train, which provides management with complete overview of development progress which enables delegation more responsibilities to the teams. The short-term recommendations cover the first steps in the SAFe implementation; the team level in the framework which enables teamwork and integration according to the chosen sprint and cadence. The next step in implementing is to introduce an agile release train, since the organisation is currently only working on one development project this release train will incorporate all activities. The number of people currently working in the baseline program is possible to incorporate within one train the recommendation is to continue this train even when the program will transfer into maintenance of the product. The agile release train will require several new roles to be created in the organisation. Many of these roles can be created by changing the title and some tasks from roles in the old organisation, for example the system manager can become the release train engineer. The framework also creates a new way to handle backlogs for the teams where the program backlog feeds the team backlogs with stories. Another benefit from the framework is the retrospective and demos that are held both at the end of sprints and increments. The demos will provide early feedback and increase the knowledge of the deliveries throughout the organisation. The SAFe framework facilitates action from the problems raised in the retrospectives. Easier, team specific problems are solved by the teams while larger problems that concerns the program level will be handled by a task force of volunteers. This allows the program to inspect and adapt in a more efficient way.

The core of agile development is the teams and an agile organisation will be as effective as the teams in it. Effective teams are not only dependent on the members, management is just as important. The Schneider core culture survey showed that the engineers at Saab feel that there currently is a control culture within the baseline program. Effective teams need to focus on reaching their goal and therefore needs authority. Management must take a step back, remove control mechanisms and let the teams make decisions. The most important task for management should be to facilitate the team's work and take action on their experienced problems. The product owners should tell them what to do while the teams must be able to decide how they are going to do it. There should also be a clear engagement to change the culture within the organisation towards a more collaborative culture. To create this new culture and team structure it is necessary to do away with the old hierarchical project structure that still influences the baseline program. This can be done by changing the old view that the time and scope of the project is set and that the resource are variable to the agile mindset that the resource and time are set and the scope is variable. This mindset will enable teams that are constant over time working on one base product instead of focusing on the customer projects.

The short-term recommendation includes new management structure for the teams, and the long-term recommendation is to evaluate if the teams should be organised differently, as discussed in the workshop. Before this evaluation is conducted they can work in their current teams to see the outcome from switching managers from subprogram to teams. The evaluation should be performed by an assigned team, with mainly representatives from the teams but also program manager and system managers. The eventual need for support teams, like the ones used by Ericsson, should also be investigated. Support teams could be used to handle for example integration issues or the test environment. Once the team structure is established, structure regards what areas/functions teams should be formed around, what responsibilities each team will have and what competence is needed in each. Saab can choose to either delegate members to the team or have self-designing teams. Self-designing teams are suggested in LeSS framework (Scrum Alliance 2013) and is a three-hour session where the members design their own teams. The managers place the different teams on the walls, describing team purpose, responsibilities and competences necessary in each team. Each member then writes their name on a post it and places on the team they want to join. When all has chosen a team, a discussion is conducted to see that each team fulfils the description of the team purpose. When they discover that they do not fulfil the description, they reorganise again, and this is usually done in three to four iterations until the teams are complete. This is a good example of how to delegate responsibilities to the teams and also the members have the opportunity to choose who to work with which likely will lead to better team dynamic. Since the engineers working at Saab are very experienced in specific fields, they are likely to choose a team that is aligned with their competence area and therefore the exercise will probably be quite pain free.

Future questions

Some questions have not been studied in this report but will be important for Saab to answer in the future to successfully scale their agile development.

How shall the demerging to the customer projects be handled without disrupting the teams in the baseline program?

Can subprogram C2 be included in virtual teams to create teams who can take on an even wider span of tasks?

8. Discussion

This study was conducted within a field that lacks academic work. Therefore, case studies, expert interviews and practitioner solutions has been the main data collection for this report. This means that the data in this report to a large extent is subjective and a more objective analysis is needed to generalize the results. The study aimed to find a more structured process to scale agile development and to identify core questions that need to be answered before the implementation. The report does provide value for companies with the same problems as Saab and will provide insight regarding the tools and methods in use today.

The interviews with the case companies and experts vary in time which means that some companies/projects have been more thoroughly investigated. This might have caused the study to go more in the direction of the companies and experts who were able to more in detail explain their choices.

Further research into this subject should incorporate a larger amount of companies that have scaled their agile development to make general conclusions. Questions that are of interest in the future are:

- In which settings are the different frameworks suitable? Is it dependent on size or culture?
- How can a large organization become an agile enterprise?
- What are the benefits of scaling agile further than a specific development program/project?

References

Al-Baik, O. & Miller, J. 2015, The kanban approach, between agility and leanness: a systematic review, *Empirical Software Engineering*, 20(6), pp. 1861-1897.

Beck, K. (2000), *Extreme programming explained : embrace change*, Boston, USA: Addison-Wesley

Beck, K. (1999). Embracing change with extreme programming, *IEEE Computer Society*, 32(10), pp. 70-77

Blankenship, J., Busa, M., Millett, S. (2011), *Pro Agile .NET Development with SCRUM*, Berkeley, USA: Apress L.P.

Bryman, A. (2012), *Social research methods*, 4th ed, Oxford, UK: University press

Bryman, A., & Bell, E. (2015), *Business research methods*, 4th ed, Oxford: Oxford University Press.

Cobb, C. G. (2015) *The Project Manager's Guide to Mastering Agile*. New York, USA: Wiley

Cohen, G. (2010). *Agile excellence for product managers.* Cupertino, CA: Super Star Press.

Coehlo. B, Baso.A. (2012) *Effort Estimation in Agile Software Development using Story Points.* International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868 Foundation of Computer Science FCS, New York, USA Volume 3– No.7, August 2012

Cooper, R., G. (2014). What’s Next? After Stage-Gate. *Research Technology Management.* 57(1). pp. 20-31

Cooper, R., G. (2009). How Companies are Reinventing Their Idea-to-Launch Methodologies. *Research Technology Management.* 52(2). pp. 47-57

Easterby-Smith, M., Thorpe, R., & Jackson, P. R. (2015). *Management & Business Research*, 5th ed,. London: Sage

Ericsson (2015), *The Company*, https://www.ericsson.com/se/thecompany [Accessed on: 2016-12-23]

Fowler, M. and Highsmith, J. (2001) The Agile Manifesto. [online] Available at: http://www.drdobbs.com/open---- source/the---- Agile---- manifesto/184414755 [accessed on: 2016-09-11]

Hunt, J. (2006). *Agile software construction.* London, UK: Springer-Verlag

Iivari, J. & Iivari, N. (2011), The relationship between organizational culture and the deployment of agile methods, *Information and Software Technology*, 53(5), pp. 509-520, .

Kniberg, H. (2011) *Lean from the trenches: Managing Large-Scale projects with Kanban*, Raleigh, USA: Pragmatic Bookshelf

Kotter J. P., (2007), "*Leading Change: Why Transformation Efforts Fail"*. Harvard Business Review.

Larman, C. & Basili, V.R. (2003), Iterative and incremental development: A brief history, Computer, 36(6) pp. 2-11

Larman, C. & Vodde, B. 2010, Practices for scaling lean & agile development: large, multisite, and offshore product development with large-scale Scrum, Upper Saddle River, USA: Addison-Wesley,

Measey, P., (2015) Agile foundations: Principles, Practices and Frameworks, Swindon, UK: BCS

Neuman, W.L. 2011, *Social research methods: qualitative and quantitative approaches*, 7.th edn, Allyn and Bacon, Boston, Mass

Royce, W. W. (1970) Managing the Development of Large Software Systems, *Proceedings of IEEE WESCON 26 (August*), pp. 1–9

Saab Group (2016) http://Saabgroup.com/about-company/company-in-brief/ [accesed on: 2016-09-21]

Saunders, M., Lewis, P., & Thornhill, A. (2009). *Research Methods for Business Students*, Harlow: Pearson Education Limited.

Scaled Agile (2016), www.scaledagileframework.com, *Various pages*, [Accessed on: 2016-10-23]

Scaled Agile (2016), *SAFe 4.0 introduction*, http://www.scaledagile.com//wp-content/uploads/delightful-downloads/2016/07/SAFe_4_whitepaper_digital_7-16.pdf [Accessed on: 2016-12-22]

Schneider, W. (1994), *The Reengineering alternative*, New York, USA: McGraw-Hill Education

Schwaber, K. Sutherland, J. (2016) *The Scrum Guide:The definitive guide to Scrum, the rules of the game.* http://www.Scrumguides.org/docs/Scrumguide/v2016/2016-Scrum-Guide-US.pdf [Accessed on: 2016-12-25]

Scrum.org (2016) *What is Scrum?* https://www.Scrum.org/Resources/What-is-Scrum [Accessed on: 2016-12-25]

Scrum.org (2016) *The nexus guide* *https://www.Scrum.org/Resources/The-Nexus-Guide* [Accessed on: 2016-12-25]

Scrum Alliance (2013), *How to form teams in large scale Scrum? A story of self-designing teams.* https://www.Scrumalliance.org/community/articles/2013/2013-april/how-to-form-teams-in-large-scale-Scrum-a-story-of [Accessed on: 2016-12-23]

Sommer,A. Dukovska-Popovska, I. & Steger-Jensen, K (2014). *Agile Product Development Governance- On governing the emerging Scrum/Stage-gate hybrids.* IFIP International Federation for Information Processing.

Tathagat, V. (2015). *Agile Product Development: How to Design Innovative Products That Create Customer Value.* New York: Springer

Target Process (2014) Upcoming Feature: Epics as one more hierarchy level. https://www.targetprocess.com/blog/2014/09/upcoming-feature-epics-as-one-more-hierarchy-level/ [Accessed on: 2016-12-22]

Turetken, O., Stojanov, I., & Trienekens, J. J. M. (2016), Assessing the adoption level of scaled agile development: a maturity model for Scaled Agile Framework. *Journal of Software: Evolution and Process*, 28(7)

Tiberius, R., & Silver, I. (2001). *Guidelines for conducting workshops and seminars that actively engage participants*. Department of Psychiatry, University of Toronto.

Version One (2016), *10th annual State of Agile Report,* http://www.agile247.pl/wp-content/uploads/2016/04/VersionOne-10th-Annual-State-of-Agile-Report.pdf

Wheelan, S. (2015), *Creating effective teams: a guide for members and leaders*, Fifth edn, SAGE Publications, Inc, Thousand Oaks.

Appendix A: Interview guideline System engineer

This Appendix illustrates the interview guideline with the engineers. This guideline was used when interviewing the system engineers, the other engineers received similar questions with some tweaks to specific questions.

Ways of working

- What are your responsibilities and tasks?
- How has this changed since the transition to the baseline program?
- Who do you interact and cooperate with to solve your tasks?
- How would you like to work? Cross-functional teams? Functional teams etc.? If cross-functional, how would you divide the teams?
- Do you think the Kanban board was a good tool?
- Why is it not used today? What were the problems?

Organisation and communication

- Are the roles in the baseline program clear? Do you know who to contact at all times?
- Is it possible to get an overview of the progress in the baseline program?
- Do you think the management can do something else to better support your work?
- What do you expect from your closest boss?
- Do you know who is dependent upon your work?

Base Product

- How do you work to create a base product and not a customer specific product?
- Does the requirements need to be different to create a base product?

System team

- Does the work in the system team function according to plan?
- Are the work tasks clear?
- Have the system teams made it easier for you to solve your tasks?
- Do you help I&V with the verification?
- Does I&V help you with the requirements?
- Does the team work without a leader?
- Are your mandates clear? Do you know what you are allowed to decide and not?
- How do you report your progress in the team?

Requirements and testing

- Do you validate your work in the system rig?
- What is prioritised? That the customer receives the right functionality in their product or that the product gets built according to the requirements?

Problem areas

What are the biggest problems within the baseline program today?

Appendix B: Interview guideline Management (I&V)

This appendix illustrates an example of the guideline used for interviews with management for the I&V subprogram. Similar guidelines was used for the interviews with management for the other subprograms but the questions are formed around the specific area for each subprogram.

Ways of Working

- Describe your responsibilities and tasks.
- How has the responsibilities been divided between the managers?
- How has the work process at the I&V subprogram been changed after the introduction of the baseline program?
- How did you evaluate the organisation and decide what needed to be changed during the shift to the baseline program?
- How have the system teams affected you way of working?
- How do you handle the verification specifications?
- Who is responsible for the system rig?
- How is the schedule for the rig developed?

Requirements and Testing

- What is I&Vs responsibility? To test the requirements or verify the functionality?
- How do you get feedback from the customer? Is it only through fulfilment of requirements?
- How are requirements broken down? By whom? Is today's way the best one?
- Is it necessary to start the official verification early?

Organisation and communication

- How do you communicate with the other subprograms?
- How do you know what will be delivered in the next delivery?
- How do you measure the progress within I&V and how is this communicated?
- How would you like to be organised? Do you have ideas for a better way?
- How is the collaboration with the main suppliers to the system rig handled? Do you check the quality before delivery?

Problems

- What are the biggest problems within the baseline program today?

Appendix C: Interview Template Expert interviews

Explain our thesis. Talk about the problems with scaling that we have encountered.

Background

- What is your background?
- What are your current role?
- How many agile transformations have you been a part of?

Agile ways of working

- How do you work to create effective teams?
- How do you work with management to create empowered teams?
- What methods do you propose the teams to use?
-

Scaling Agile Frameworks

- How do you analyse what is needed within a company in order to scale the agile ways of working?
- What are the important factors to identify?
- Which frameworks do you usually use?
- What frameworks are suitable in a complex R&D environment with both hardware and software?
- How do you create an agile mindset in the organisation?
- Can you reduce dependencies and how do you identify the ones that still exists?
- How do you convince management to let go of their control?
- How do you handle progress reporting?

- Can you take us through one of your agile transformations and explain the choices you made?

Appendix D: Template Questions Case Study

Organisation

- What product are you developing and how are they structured?
- How big is you development organisation?
- How does your current organisation look?
- What did your old organisation look like?

Change Process

- When did you agile journey start?
- Who drove the change?
- What were the main problems in the early phases?
- How did you analyse the organisation to find what methods should be used?
- Did you consult experts to help with the change?
- How did you educate the employees?
- Did/Do you use agile coaches? If Yes, What are their tasks?
- What were the main problems during the change process?
- What did you learn from your agile transformation? What are the most important factors to consider before changing?

Agile ways of working

- How do you handle progress reports? Both vertical and horizontal?
- How do you synchronise deliveries?
- How do you handle dependencies?
- How do you break down your epics to stories and assign them to the right team?
- How is the functional growth developed? Do you use a roadmap? If Yes, Who is responsible for this?
- How do you prioritise and estimate backlogs?
- What are your main communication channels when conveying decisions and overall progress between teams and management?
- What roles do you currently have? PO, SM, Line organisation etc.
- What are the fundamental principles when creating an effective team?

Appendix E: Core culture questionnaire

by W.E. Schneider - The reengineering alternative (1994)

For each question ask yourself: “When I boil it down and get to the heart of the matter, which of the four possible answers most accurately describes my actual experience in my organization?” Answer every question, and select only one answer for each. Indicate your answers in the table on the last page.

Questionnaire

1. When all is said and done, the way we accomplish success in the organization is to:

a. Get and keep control.
b. Put a collection of people together, build them into a team, and charge them with fully utilizing one another’s resources.
c. Create an organization that has the highest possible level of competence and capitalize on that competence.
d. Provide the conditions whereby the people within the organization can develop and make valuable accomplishments.

2. What do we pay attention to primarily in our organization and how do we decide about things?

a. We pay attention to what might be and we decide by relying on objective and detached analysis.
b. We pay attention to what is and we decide by relying on what evolves from within the hearts and minds of our people.
c. We pay attention to what might be and we decide by relying on what evolves from within the hearts and minds of our people.
d. We pay attention to what is and we decide by relying on objective and detached analysis.

3. The people with the most power and influence in the organization:

a. Are charismatic, can inspire others, and are good at motivating others to develop their potential.
b. Have the title and position that gives them the right and the authority to exercise power and influence.
c. Are both contributors and team players, who are an essential part of the team. People like working with them.
d. Are experts or specialists, who have the most knowledge about something important.

4. In our organization, “success” means:

a. Synergy. By teaming up with one another and our customers, we accomplish what we are after.

b. Growth. Success means helping others more fully realize their potential.
c. Dominance. Success means having more control than anyone else. Complete success would be for the organization to be the only game in town.
d. Superiority. Success means that the organization is the best, offering superior value. The organization is “state of the art” in all that it does.

5. In our organization, leadership means:

a. Authority. Leaders are regulators and call the shots. They are commanding, firm, and definitive. What they say goes.
b. Setting standards and working hard to get people to achieve more. Leaders are intense taskmasters, who always challenge workers to do better.
c. Being a catalyst. Leaders cultivate people. They create conditions in which people are inspired to fulfil their own and others’ potential. At the same time, leaders build commitment to the organization.
d. Building a team that will work well together. Leaders are coaches. They behave as first-among-equals. They strive to represent the people in the organization.

6. When we worry about something in the organization, it is usually about:

a. Losing. We worry most about being also-rans or having our reputation harmed because we couldn’t deliver as well as, or better than, our competitors.
b. Stagnation. We worry most about failing to progress, simply existing from day to or even going backwards.
c. Vulnerability. We worry most about being in a position where others have more power or market share than we do.
d. Lack of unity. We worry most about the team being broken up or alienating our customers. We worry about a lack of trust among ourselves.

7. Our organization’s overall management style is best described as:

a. Enabling. Empowering. Commitment oriented.
b. Challenging. Goal oriented. Very rational and analytical.
c. Democratic. Highly relational. Highly participative.
d. Prescriptive. Methodical. Policy and procedure oriented.

8. The essential role of the individual employee in our organization is to:

a. Collaborate. To be a team player.
b. Be an expert. To be the best in your specialty or area of technical expertise.
c. Perform according to policy and procedure. To meet the requirements of the job as outlined.
d. Be all you can be. To change, develop, and grow. To be committed to the organization and its purposes.

9. What counts most in the organization is:

a. Winning. Being recognized as the best competitor around.
b. Not losing. Keeping what we've got.
c. Evolving. Realizing greater potential. Fulfilling commitments.
d. Accomplishing it together. Being able to say "we did it together."

10. Which of the following best describes how you feel about working in your organization:

a. This is a caring and "spirited" place. I feel supported.
b. People are able to count on one another.
c. Things are no nonsense and restrained.
d. Things are rather intense. I feel like I have to be on my toes all the time.

11. What counts most in the organization is:

a. Security.
b. Community.
c. Merit.
d. Fulfilment.

12. Which of the following best describes the primary way decisions are made in the organization?

a. We pay close attention to our concepts and standards. We emphasize the fit between our theoretical goals and the extent to which we achieve them. Our decision-making process centres on how systematically our conceptual goals are achieved.
b We pay close attention to our values. We emphasize the fit between our values and how close we are to realizing them. Our decision-making process centres on the congruence between our values or purposes and what we have put into practice.
c. We emphasize what the organization needs. Our decision-making process centres on the objectives of the organization and on what we need from each function within the organization.
d. We emphasize tapping into the experiences of one another. Our decision-making process centres on fully using our collective experiences and pushing for a consensus.

13. Overall, life inside our organization is:

a. Spontaneous, interactive, and free and easy.
b. Intellectually competitive, rigorous, and intense.
c. Objective, orderly, and serious.
d. Subjective, dedicated, and purposeful.

14. In general, our attitude toward mistake is:

a. We tend to minimize the impact of mistakes and do not worry much about them. People who make mistakes should be given another chance.
b. Mistakes are inevitable, but we manage by picking up the pieces and making the necessary corrections before they grow into bigger problems.
c. Mistakes are nearly taboo. We don’t like them. A person who make mistakes is looked down upon.
d. We pay attention to the kind of mistake. If the mistake can be quickly fixed , we go ahead and fix it. If the mistake causes a function to get in trouble or could cause the organization to become vulnerable, we marshal all our resources to fix it as quickly as possible. Mistakes that affect the organization as a whole could get someone in trouble.

15. Concerning control, which of the following is most emphasized?

a. Concepts and ideas. We control everything that is critical toward achieving or preserving our superiority in the marketplace.
b. Everything critical to keeping us working together in the organization and retaining close ties with our customers.
c. Just about everything. Getting and keeping control is central to what the organization is and does.
d. As little as possible. We are put off by the notion of control. We prefer to leave things up to the commitment and good will of our people.

16. The essential nature of work in the organization emphasizes:

a. Functionalists. Individuals stay within their function. Specialties are subordinate to the service of functions.
b. Specialists. Individuals stay in their technical or other specialty. Functions are channelled into the service of specialties.
c. Generalists. Individuals move in and out of numerous functions and specialties.
d. All of the above. Individuals do all three.

17. The people who primarily get promoted in the organization are:

a. Generalists. They must also be capable people who are easy to work with.
b. Those who have performed consistently well in their function for many years and have demonstrated that they can seize authority and get things done.
c. Those who know the most about their area of expertise and have demonstrated their competence.
d. People who can handle responsibility and who want it. We don’t use the word “promotion.”

18. The compensations system in the organization is most similar to which of the following?

a. We emphasize fair and equitable pay for all. We also emphasize the long-term perspective. We plow a lot of money back into the organization to ensure continued growth and success, so personal financial compensation tends to be secondary to other more important matters.

b. Our compensation system is highly individual and incentive-oriented. Uniquely capable people who are recognized experts can make a lot of money.
c. Our compensation system is highly structured. The larger your role and function in the organization, the more money you make.
d. Our compensation is tied primarily to team eff ort. If the whole organization does well, we all share in the wealth. If the whole organization does poorly, we all sacrifice.

19. Which of the following best describes our organization's primary approach in dealing with customers or constituents?

a. Partnership. We team up with our customers or constituents. We want to be able to say "We did it together."
b. We emphasize uplifting and enriching our customers or constituents. We concentrate on realizing the possibilities and potential of our customers or our constituents more fully.
c. We emphasize gaining the greatest market share that we can get. We would like to be the only game in town for our customers or constituents.
d. We emphasize offering superior value to our customers or constituents. We try to provide state-of-the-art goods or services to our customers or constituents.

20. Which phrase best describes our organization?

a. "We believe in what we are doing, we make a commitment, and we realize unlimited potential."
b. "We are the best at what we do."
c. "We are the biggest at what we do."
d. "United we stand, divided we fall."

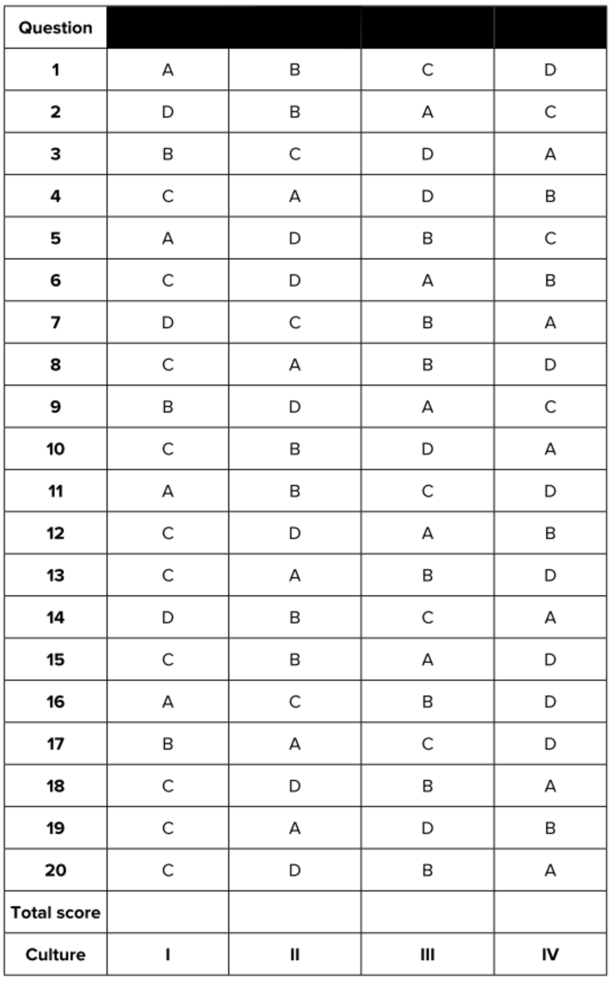
Directions for Scoring the Questionnaire
Record your answer to each question (a, b, c, or d) on the scoring table by marking your chosen answer at the proper question number. Do this for every question. When finished, add up the total number of marked answers under each roman numeral at the top of the scoring table.

I – Control Culture

II – Collaboration Culture

III – Competence Culture

IV – Cultivation Culture



Question				
1	A	B	C	D
2	D	B	A	C
3	B	C	D	A
4	C	A	D	B
5	A	D	B	C
6	C	D	A	B
7	D	C	B	A
8	C	A	B	D
9	B	D	A	C
10	C	B	D	A
11	A	B	C	D
12	C	D	A	B
13	C	A	B	D
14	D	B	C	A
15	C	B	A	D
16	A	C	B	D
17	B	A	C	D
18	C	D	B	A
19	C	A	D	B
20	C	D	B	A
Total score				
Culture	I	II	III	IV

Figure 1 Scoring table for the survey