Applicability of Lean toward improved efficiency in software development
– A case study at a Swedish ICT Company

Maria Dahlman
Ulrika Olsson

Master of Science Thesis IIP 2014
KTH School of Industrial Engineering and Management SE-100 44 STOCKHOLM
Abstract
In recent years, software development has been evolving around an Agile way of working to optimize software development processes. Simultaneously in other industries, the Lean concept have been adopted and used to make manufacturing and production of services more efficient. Software development companies has now started to realize that in order to become competitive in software delivery they need to optimize the entire end-to-end process and not only improve the development phase. Thus, an increased interest for Lean thinking has appeared in the industry.

Applying Lean successfully in a software development context is not an easy task. This since Lean is developed for and by the manufacturing industry its content is also dependent on the characteristics of this context. Thus, it is not applicable to other contexts without being adjusted. Though, literature argues that when the content of Lean is properly adjusted to nature of the relevant context it can contribute with the same benefits as in manufacturing. However, the literature lack in studies on how Lean can be successfully applied in non-manufacturing contexts. This study aims to contribute to this knowledge gap by adjusting and applying Lean tools in software development context in order to make the end-to-end process more time- and resource-efficient. This will be made through a case study at a software development department at a Swedish Information and Communications Technology (ICT) Company. This department develops software components that work as building blocks of the product sold to the end customer. More specifically, these components contribute with functionality that enables communications between the nodes in a network.

The review of the existing body of knowledge together with an analysis of the company specific problems indicates that the most sufficient tool to apply in this case study Lean implementation is Value Stream Mapping (VSM). Hence, the study focuses on adjusting the Value Stream Mapping framework to this new context in order to visualize the entire value flow and make improvements aligned with Lean. The adjustments of VSM are underpinned by relevant findings from previous research together with an analysis of the differences between manufacturing Lean and software development Lean.

The study shows that it is possible to reasonably adapt and benefit from Value Stream Mapping also in the context of an ICT Company. The outcome of the study is an overview of the current value flow at the company and suggestions on how, where and what improvements can be made. Though, all improvement suggestions evolve around the aim of improving the flow efficiency and eliminating waste by improving the communication, standardization, synchronization, resource allocation and developing proper measurements.

The conclusion is that Lean is applicable in the particular software development context with some necessary adjustments made to fit the current way of working and value offering. The usage of Lean tools such as Value Stream Mapping is possible, also here adjustments are a must and the VSM shows that the current state is better in providing a base for improvements than the future state map. This since improvements in this context is not always visualized and captured in the future state map, but the future map contribute to a vision for the value stream to aim towards.

Key words: Lean, Agile, Value Stream Map, Software development, Flow, Waste, Management
**Prologue**

This is a Master of Science Thesis within the field of Industrial Engineering and Management at the department of Industrial Production at the Royal Institute of Technology in Stockholm, Sweden. The study corresponds to 30hp and has been executed during January to May 2014 in close cooperation with a Swedish ICT Company.

We want to start the thesis by thanking everybody who has contributed through guidance, participation in interviews and enabled occasions for observations. Thank you for making it possible to obtain information and encouraging us to persistently pursue our search for deeper knowledge and understanding during the drafting of this study. Some extra credit should be given to our supervisor at the ICT Company for making it possible for us to perform this thesis and continuously helping us when road bumps appeared.

We would also like to thank our supervisor at KTH, Hakan Akillioglu, for interesting thoughts and ideas throughout.

Finally we would like to dedicate a thank you to Björn Langbeck at Swerea who helped us with the analysis and outlining of the Future state Value Stream Map.

Stockholm, 2 Juni, 2014

Maria Dahlman and Ulrika Olsson
Table of Contents

Abstract ................................................................................................................................. 2

1 Introduction .......................................................................................................................... 10
  1.1 Background .................................................................................................................... 10
  1.2 Purpose ......................................................................................................................... 10
  1.3 Research questions ....................................................................................................... 10
  1.4 Company problem ....................................................................................................... 10
  1.5 Scope ............................................................................................................................ 11
  1.6 Methodology and outcome ......................................................................................... 11
  1.7 Disposition .................................................................................................................. 11

2 Theoretical framework ....................................................................................................... 13
  2.1 The background and nature of software development .................................................. 13
  2.2 Lean Thinking ............................................................................................................. 14
  2.3 Applicability of Lean in software development ............................................................ 15
  2.4 The importance of making adjustments of Lean .......................................................... 16
  2.4.1 Adjustments of Lean principles ................................................................................ 16
  2.4.2 Adjustments of Lean tools ....................................................................................... 17
  2.5 How to succeed with a Lean implementation ............................................................... 18

2.6 Value Stream Mapping .................................................................................................. 19
  2.6.1 Benefits of VSM ....................................................................................................... 19
  2.6.2 Notions and icons in VSM ....................................................................................... 20
  2.6.3 Steps in the creation of a value stream map .............................................................. 21

3 Methodology ..................................................................................................................... 22
  3.1 Case study .................................................................................................................... 22
  3.2 Data collection ............................................................................................................. 22
  3.2.1 Literature review and theoretical framework ............................................................ 22
  3.2.2 Interviews ................................................................................................................ 22
  3.2.3 Observations ............................................................................................................. 23
  3.3 Data analysis ................................................................................................................ 23
  3.3.1 Selection of components ......................................................................................... 23
  3.3.2 Root cause analysis and fishbone diagram .............................................................. 23
  3.3.3 Value Stream Mapping ............................................................................................ 24
  3.3.4 Interview analysis .................................................................................................... 24
  3.4 Limitations ................................................................................................................... 25
  3.5 Delimitation ................................................................................................................ 25

4 Company context .............................................................................................................. 26
  4.1 Organization ................................................................................................................ 26
  4.2 Product ......................................................................................................................... 26
  4.2.1 Components ............................................................................................................ 27
  4.2.2 Component selection ............................................................................................... 28
  4.3 Communities ................................................................................................................ 28
  4.4 Roles and responsibilities ............................................................................................ 29
  4.5 Value stream ................................................................................................................ 30
  4.6 Process ........................................................................................................................ 30

5 Company problem ............................................................................................................ 32
  5.1 Identified problems in the Company .......................................................................... 32
  5.1.1 Limited holistic understanding of the value stream ............................................... 32
  5.1.2 Temporary lack of resources ............................................................................... 32
  5.1.3 Insufficient communication with Applications ...................................................... 33
  5.1.4 Somewhat insufficient quality control towards Applications ................................ 34
5.1.5 Resistance among the software developers ................................................................. 34
5.1.6 Lack of technical knowledge upstream ................................................................. 34
5.1.7 Unclear roles and responsibilities ........................................................................... 35

5.2 Comments to company problems ............................................................................ 35

6 Adjustments of Lean to fit the Department ................................................................ 37

6.1 Adjustments of flow ................................................................................................. 37
6.1.1 Product flow........................................................................................................... 37
6.1.2 Information flow .................................................................................................... 37

6.2 Adjustments of waste ............................................................................................. 38
6.2.1 Different kind of wastes ....................................................................................... 38

6.3 Adjustments of VSM ............................................................................................... 41
6.3.1 Icons ...................................................................................................................... 41
6.3.2 Data collection and measures .............................................................................. 42

7 Value Stream Mapping of current state ..................................................................... 44

7.1 Current state map Component 1............................................................................... 44
7.1.1 Opportunity submission ....................................................................................... 44
7.1.2 Opportunity analysis ............................................................................................ 44
7.1.3 Component Roadmap Meeting I ........................................................................ 44
7.1.4 Pre-screening ....................................................................................................... 44
7.1.5 Component Roadmap Meeting II ........................................................................ 45
7.1.6 Backlog grooming ............................................................................................... 45
7.1.7 Sprint planning ....................................................................................................... 45
7.1.8 Development process ............................................................................................ 46
7.1.9 Pre Release Control ............................................................................................ 46
7.1.10 Release ................................................................................................................ 47

7.2 Current state map Component 2............................................................................... 48
7.2.1 Opportunity submission ....................................................................................... 48
7.2.2 Opportunity analysis ............................................................................................ 48
7.2.3 Pre-screening ....................................................................................................... 48
7.2.4 Development process ............................................................................................ 48
7.2.5 Pre Release Assessment ..................................................................................... 49
7.2.6 Release .................................................................................................................. 49

8 Analysis of current state maps ................................................................................... 50

8.1 Flow ......................................................................................................................... 50
8.1.1 Information flow ................................................................................................... 50
8.1.2 Product flow ......................................................................................................... 51

8.2 Waste ...................................................................................................................... 53
8.2.1 Prolonged items on backlog ................................................................................ 53
8.2.2 Extra features ....................................................................................................... 54
8.2.3 Extra processing ................................................................................................... 54
8.2.4 Handoffs ............................................................................................................... 54
8.2.5 Task switching ...................................................................................................... 55
8.2.6 Delays .................................................................................................................... 55
8.2.7 Defects .................................................................................................................. 56

9 Discussion and improvement suggestions ................................................................... 57

9.1 Communication ....................................................................................................... 57
9.1.1 Improvements that favor communication ......................................................... 58

9.2 Standardization ........................................................................................................ 58
9.2.1 Improvements that favor standardization ......................................................... 60

9.3 Synchronization ....................................................................................................... 61
9.3.1 Improvements that favor synchronization ......................................................... 62
9.4 Measurements ........................................................................................................... 63
  9.4.1 Improvements that favor control and measurement ........................................... 63
9.5 Resource allocation .................................................................................................... 64
  9.5.1 Improvements that will better the resource allocation ................................. 65

10 Value Stream Mapping of future state ....................................................................... 66
  10.1 Future state map of the components .................................................................................. 66
    10.1.1 Communication Forum ....................................................................................... 66
    10.1.2 ERP system ......................................................................................................... 67
    10.1.3 Opportunity submission ..................................................................................... 67
    10.1.4 CRM I ................................................................................................................. 67
    10.1.5 Study ................................................................................................................... 68
    10.1.6 Pre-screening ....................................................................................................... 68
    10.1.7 CRM II ............................................................................................................... 69
    10.1.8 Sprint planning .................................................................................................... 69
    10.1.9 Development process .......................................................................................... 70
    10.1.10 PRC .................................................................................................................. 71
    10.1.11 Backlogs ........................................................................................................... 71

11 Conclusions .................................................................................................................. 72

12 Future research ............................................................................................................. 73

References ....................................................................................................................... 74
Appendix ............................................................................................................................ 77
List of abbreviations

AAC – Applied Architecture Council
AV – Process Availability Time
CO – Changeover Time
CBA – Component Based Architecture
CPU – Central Processing Units
CRM – Component Roadmap Meeting
IAC - Implementation Architecture Council
ICT – Information and Communications Technology
JIT – Just-in-time
KPI – Key Performance Indicator
LT – Leadership team member
MAC – Architecture Council
MPO – Main Product Owner
MR – Main Requirement
NVA – Non-Value Adding
NNVA – Necessary Non-Value Adding
PEST – Portfolio Efficiency System Study
PM – Program Manager
PO – Team Product Owner
POG – Product Owner Group
PRC – Pre Release Control
PSF – Program Steering Forum
PT – Process Time
RDA – Reference Deployment Architecture
RID – Release Intent Decision
SLES – SUSE Linux Enterprise Server
SM – Scrum Master
SPM – Strategic Product Manager
TPS – Toyota Production System
VA – Value Adding
VSM – Value Stream Mapping
List of figures

Figure 1. Flow of content and sections .........................................................12
Figure 2. Table of icons in a VSM .................................................................21
Figure 3. Research methodology overview ..................................................25
Figure 4. A schematic sketch of how a CBA based product is combined with the internal customer and the Department located.................................27
Figure 5. Fishbone diagram of company problem ........................................36
List of tables

Table 1. The concepts of Lean thinking .........................................................14

Table 2. The principles of Lean thinking .......................................................15

Table 3. Adjustments of Lean principles to fit the software context according to Poppendieck and Poppendieck (2007) ..............................................16

Table 4. List of earlier successfully implemented Lean tools in a software development context .................................................................17

Table 5. Seven wastes in manufacturing and software development ..............38

Table 6. The seven manufacturing wastes adjusted to the context of the Department .........................................................................................41

Table 7. Adjustment of the icons used in a manufacturing VSM .................41
1 Introduction

This chapter starts off with presenting the background and purpose of the study. Thereafter the research questions, which the study aims to answer, are presented, followed by the statement of the Company problem and corresponding scope when examining the problem. Moreover, the section briefly presents the outcome of the study and thereafter the disposition of the report to simplify understanding of the printed structure.

1.1 Background

In recent years, software development has mostly evolved around an Agile way of working to improve the product development process and the operational success (Dybå and Dingsøyr, 2008). However, Agile has been argued to lack in providing the big picture, which research states as necessary in order to succeed in software delivery (Biffl, Aurum, Grünbacher, & Boehm, 2005). Thus, to improve the Agile methods and processes in software development, companies have started to look at practices from Lean thinking. This since Lean, in contrast to Agile, has a holistic approach to product development, which is argued as essential to stay competitive on the future software development market. Though, applying Lean to an Agile software development context has turned out to be difficult, especially since how it should be applied depends on the context and objectives for the application (Poppendieck and Poppendieck, 2007; Staats, Brunner, and Upton, 2011). Moreover, Lean in software development is a new term that has not yet fully been understood and examined by the research. Thus, further studies in a non-repetitive development context would enrich the current research (Wang, Conboy, and Cawley, 2012).

However, since Lean thinking is developed by and for the manufacturing industry its principles and practices are adjusted to their repetitive, visible and somewhat rigid production process (Staats, et al., 2011). By adjusting Lean practices to the nature of the specific context, Lean has turned out to contribute with advantages to other industries than manufacturing as well (Åhlström, 2004; Song, Tan, and Baranek, 2009; Wang, et al., 2012). In this study, the applicability of Lean principles and practices in non-manufacturing industries are further investigated.

1.2 Purpose

The purpose of this study is to contribute to current research about how Lean can be applied in non-manufacturing context in general and a software development context in particular.

1.3 Research questions

In order to fulfill the purpose of the study the following questions are answered:

(1) How can the current software development value stream be improved by drawing on Lean thinking?

   a. Which are the current problems in the software development value stream?
   b. Which Lean practice is suitable to use when solving the current problems?
   c. How should this practice be adjusted in order to fit the specific context?

1.4 Company problem

Lean thinking aims to constantly improve value streams, by making each process as efficient as possible. However, Lean includes numerous of practices and in order to evaluate their applicability they need to be applied to a problem that suits the aim of the practice. The
particular value stream that is investigated in this study suffers in two major areas. Firstly, a lot of activities that can be classified as waste according to Lean thinking, exist in the value stream. This waste does not only affect time efficiency in delivering value, but also reduces the amount of value actually being delivered. Secondly, the flow of both information and products is intermittent, which indicates room for improvement.

1.5 Scope
The study is limited to study only one department at a Swedish Information and Communications Technology (ICT) Company. The department, where the case study is executed, is developing several software components. These components together act as building blocks in the product, called Component Based Architecture (CBA), which is offered to the end-user. This specific department will throughout the report be referred to as the Department.

Due to limited time and resources this study will only investigate the value stream of two of these components. Moreover, the study only investigates how the product development process can be improved why only activities and processes that are directly involved in this will be considered. Thus, maintenance activities, communities and sub-processes such as testing and auditing that are made by other departments, will not be deeply investigated. Though, since these tasks affect the value stream and the over all efficiency of the Department, they will be mentioned and solutions will be developed to favor the collaboration with these instances.

1.6 Methodology and outcome
The study relies on a case study. Data was collected by reviewing literature, conducting interviews and performing observations. First interviews and observations were made with an aim of gaining an in-depth understanding of the company context as well as the nature of the problem in the value streams. In parallel, a literature review was conducted in order to understand how to successfully draw on Lean in this context. From the findings a theoretical framework was built to describe which Lean practices could be useful to apply in order to solve the identified company problems. Then a second round of interviews and observations were executed in order to collect data about specific processes in the value streams. The data was then analyzed, by using an informal method of text analysis, and used as input in the Lean tool Value Stream Mapping (VSM). Finally, the findings from the VSM is discussed and presented regarding what specific actions to take, in order to turn the current state map of the VSM into the improved future state map. The outcome of this study is a visualization of the current value streams together with visualization of an improved value stream, where the current problems are eliminated or minimized. Improvement suggestions for how to operationally turn the current state maps into the future states will also be presented in the result.

1.7 Disposition
The flow and content can be seen in Figure 1 below. Section 2 presents relevant theories necessary to understand in order to ensure proper application of the Lean in this specific context. In section 3 a presentation of the methodology used in the study is made. Hence, here all methods that are used are presented and justified. In section 4 the Company context is explained and in section 5 the identified problems in the current processes of the Company are discussed. Section 6 is dedicated to adjustments of the lean philosophy to the context of the study. Section 7 presents the result through Current state maps for the value streams under investigation; the analysis regarding these maps is done in the following section, 8. The
result is then further discussed and improvements are established in section 9. Section 10 contains the Future state map of the value stream based on the earlier improvements and discussions. This is followed by section 11 where the final conclusions of the study are drawn. Furthermore, section 12 gives suggestions on future research.

Figure 1. Flow of content and sections.
2 Theoretical framework

This section presents relevant findings and theories from previous research, which is necessary to understand in order to execute the study properly. The theoretical framework also gives credibility to the relevance and outcome of the case study. Firstly, the backgrounds of software, Lean, Lean in software and necessary adjustments to make in order to succeed with a Lean implementation in the software development context are presented. Also some theoretical basis regarding Value Stream Mapping is presented, this to highlight the applicability and implementation of the tool.

2.1 The background and nature of software development

The software development industry constantly becomes more and more intense and competitive. Thus, the ability to develop and deliver software solutions in the most efficient way has become a desired target for any company in the industry (Biffl, et al., 2005). In recent years software development has been evolving around an Agile way of working to improve the development process in order to reach operational success (Dybå and Dingsøyr, 2008). Agile software development is a development paradigm, which relies on values and principles that enables the development process to respond flexibly and quickly to changes in customer needs (Petersen, 2010).

The essence of Agile is presented in the Manifesto for Agile Software Development. This manifesto is a result of 17 software practitioners, which followed a similar software development method, who came together and rationalized a common philosophy that they termed as “Agile” (Misra, Kumar, Kumar, Fantazy, and Akhter, 2012). The values and practices form a framework for Agile software development that is characterized by short iterative cycles, quick frequent feedback from customers and constant learning. However, Agile is only dealing with activities directly related to the limited development process and does not consider other, supporting, activities in the development process (Cohen, et al., 2003).

The recent years academic research regarding what makes software development companies successful in today’s society, highlights the importance of meeting customer needs and properly delivering value (Hodgetts, 2009). To ensure fulfillment of customer needs, focus cannot only be on improving the development phase of the product development process. This as software development only is a subset of an overall product development process (Poppendieck and Poppendieck, 2007). All steps in the process must therefore be taken into consideration, but in the widespread Agile methodology only tools and frameworks regarding the actual development phase exists. Hence, researchers have realized that the popular Agile methods are too narrow and feature development oriented to use by themselves, especially when striving towards developing and delivering software solutions that adequately meets customer demands and ensures value to the end-user (Cohen, Lindvall, and Costa, 2003).

This insight has turned software development companies towards looking at the advantages of investigating the development process as a whole. This new holistic approach to improving the delivery of software has resulted in an increasing interest in Lean thinking (Musat and Rodriguez, 2010). This since Lean naturally adopts a holistic approach and is focused on improving processes as a whole rather then focusing on improving core activities in a process (Hodgetts, 2009). Applying Lean thinking to a software development process might help to optimize the entire process and secure competitive advantage by proper fulfillment of customer needs and value delivering. Though, Leans applicability in other industries than manufacturing has been questioned due to the differences in nature between
manufacturing and other industries (Poppendieck and Poppendieck, 2007; Staats, et al., 2011). On the contrary, recent research has discovered that Lean can be successfully applied in other kinds of industries than manufacturing (Åhlström, 2004; Bowen and Youndahl, 1998; Song, et al., 2009). However, both wider and deeper studies on how Lean should be adjusted and applied successfully in non-manufacturing contexts are demanded in order to better the conceptual basis regarding the applicability of Lean. This study aims to contribute to this existing knowledge gap by investigating Leans feasibility in software development.

2.2 Lean Thinking

The origins of Lean thinking go back to the roots of Lean manufacturing, which evolved from the Japanese production system called Toyota Production Systems (TPS) (Ohno, 1988; Poppendieck and Poppendieck, 2007). Lean thinking is a philosophy that aims to improve operational excellence of organizations by putting focus on the value creation in a constant flow. Hence, Lean thinking is very customer oriented as the customer determines what should be classified as value and thus the customers’ perception of value is highly important to understand in order to become successful with Lean (Womack and Jones, 1996). Lean is characterized by its broad perspective to process improvement and its high focus on eliminating waste activities (Hall, 2004).

The implications for using Lean thinking comes from the success stories that Lean have brought to the manufacturing industry. The benefits are described in The Toyota Way. The Toyota Way includes a presentation of how Toyota used Lean to create success and become the worlds most profitable auto company in an industry where competitors had more economies of scale. In other words, the book illustrates the perks of using Lean to create operational excellence with scarce resources (Hino, 2008). The core is that Lean takes a holistic approach to improvements by focusing on the entire chain of product development activities. Thus, Lean touches all areas of the value stream of the Company and aims to optimize this as a whole. This approach enables Lean to provide benefits to companies even when competition increases. Thereby maximization of value from available resources is of high importance (Hodgetts, 2009).

Today, Lean is a well-known philosophy but the meaning and constitution of the term varies (Åhlström, 2004). One way to explain the constitution of Lean thinking is to split it into three elements; concepts, principles and tools. The concepts represent the base in Lean and are inherently linked focus areas that help organizations to improve their processes. The concepts are very general and constitute a frame for Lean thinking (Table 1) (Womack and Jones, 1996).

<table>
<thead>
<tr>
<th>Lean concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Value:</strong> defined by the customer and highly important to understand and recognize</td>
</tr>
<tr>
<td><strong>Value stream:</strong> all steps in a process that brings the product of service to the customer.</td>
</tr>
<tr>
<td><strong>Flow:</strong> ensure that the value is delivered to the customer through the process in a continuous way.</td>
</tr>
<tr>
<td><strong>Pull:</strong> that a product is produced only when an order from the customer exists. Thus, nothing is produced before it is needed.</td>
</tr>
<tr>
<td><strong>Perfection:</strong> continuously identifying and removing waste in the process in order to achieve perfection.</td>
</tr>
</tbody>
</table>

*Table 1. The concepts of Lean thinking (Womack and Jones, 1996)*
The principles of Lean thinking are building blocks of the concepts and deals with particular aspects of the manufacturing system (Åhlström, 2004). The description of the Lean principles can be chosen to be either simple and general or specific and more accurate. This results in a trade-off due to the fact that a framework never can be simple, accurate and general at the same time. The effect of this trade-off is that the principles presented in the literature either lacks in details regarding the specific nature of its associated tools and techniques or are extremely context dependent (Dean and Bowen, 1994). However, one general description of the Lean principles is suggested by Åhlström (2004) and presented in Table 2.

### Lean principles

**Eliminate waste:** Since the customer is not willing to pay for waste is should indisputably be eliminated.

**Zero defects:** To reach high productivity all parts need to be free from errors from the beginning.

**Pull instead of push:** A pull system ensures that the right part is provided to each operation in the process, in the right quality and at the right time.

**Multifunctional teams:** When each team is responsible and qualified to produce all the tasks in their part of the process flow, the full potential of each operator can be released.

**Decentralized responsibilities:** The responsibility should be pushed down to the lowest level in order to empower the multifunctional teams.

**Vertical information systems:** To ensure that the teams can perform according to the company goals, an information system that relies on direct information flows to relevant decision makers must be put in place.

**Continuous improvements:** In order to reach operational success the over all goal should always be perfection which only can be reached by making continuous improvements.

<table>
<thead>
<tr>
<th>Table 2. The principles of Lean thinking. (Åhlström, 2004)</th>
</tr>
</thead>
</table>

The third Lean element is called practices or tools and is possible activities to undertake in order to reach a desired improvement (Åhlström, 2004). A successful implementation of the Lean philosophy is heavily dependent on proper application of these Lean tools. Hence, Lean includes a predetermined toolbox useful to utilize when implementing Lean (Hall, 2004). The tools in the Lean toolbox helps the organization to specify what value is as well as identifying and lining up value-creating actions and conduct these activities without interruption (Womack and Jones, 1996). Today, around 101 different Lean tools have been identified in the literature (Pavnnaskar, Gershenson, and Jambekar, 2003).

### 2.3 Applicability of Lean in software development

As mentioned the term “Lean” is fuzzy and the available definitions stated in the literature differ. However, in contrast to this academic disagreement, researchers fully agree on that the focus of Lean traditionally have been put on production companies. More specifically, on high-volume, low variety suppliers i.e. mass producers (Åhlström, 2004). Due to the fact that Lean thinking is developed by and for the manufacturing industry its principles and practices are to a high extent created to fit repetitive, visible and rigid production processes (Staats, et al., 2011). This allows researchers to question the applicability of Lean in other contexts than manufacturing (Poppendieck and Poppendieck, 2007; Staats, et al., 2011; Womack and Jones, 1996). Since a perfect fit to manufacturing also hampers the possibility to successfully apply these directly to organizations and processes that differ from manufacturing (Wang, et al., 2012).
Software development and manufacturing differs a lot. Manufacturing processes include flows of both physical goods and information (Staats, et al., 2011). In contrast, a software development process rarely include flows of material, as software development is all about producing a product that consists of code i.e. a non-physical product. The product is in the heads of the developers or stored in a computer and cannot be collected as inventory. This makes it more difficult to follow the flow of the product throughout the process and it hinders the understanding and definition of measurements in the process.

A development process where the product evolves in people’s minds through manipulation of information is classified as knowledge work. In knowledge work information is the basis of value. To secure that features are produced to the lowest possible cost, but with highest possible quality, the information flow rather than the product flow should reach the right person in the right form at the right time (May, 2005). Therefore the tasks often require both a high-level perspective and understanding for low-level details, why the need for knowledge varies with the task (Staats, et al., 2011). This results in a process with a high level of demand and supply uncertainty in comparison to manufacturing processes. This higher uncertainty complicates the implementation of Lean thinking (Lee, 2002).

The complexity and high uncertainty is created through the customer’s requirements regarding the product changes more frequently in software development. This hampers the process of defining what value is, as the customers rarely know exactly what they want. Additionally, when customers are using new software, their ideas of what they want tend to shift why predictions of their needs are very complex. (Poppendieck and Poppendieck, 2007) In contrast to manufacturing, this makes repetitive tasks rare in software development processes, which makes standardization of the process difficult (Staats, et al., 2011).

Even though manufacturing and software development differ a lot, recent research indicates that the applicability of Lean might be bigger than one can predict. Several case studies show that Lean can be applied successfully in software development companies. Though, it turns out that an adjustment of the principles and practices to fit the characteristics of the specific situation is essential (Wang, et al., 2012).

### 2.4 The importance of making adjustments of Lean

#### 2.4.1 Adjustments of Lean principles

Based on the differences between the industries that are highlighted in the section above, Poppendieck and Poppendieck (2007) suggests another formulation of the Lean principles developed to fit software development processes (Table 3).

<table>
<thead>
<tr>
<th>Lean principles in software development</th>
<th>Lean principles in manufacturing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eliminate waste</td>
<td>Eliminate waste</td>
</tr>
<tr>
<td>Build in quality</td>
<td>Zero defects</td>
</tr>
<tr>
<td>Deliver fast</td>
<td>Pull instead of push</td>
</tr>
<tr>
<td>Respect people</td>
<td>Multifunctional teams</td>
</tr>
<tr>
<td>Optimize the whole</td>
<td>Decentralized responsibilities</td>
</tr>
<tr>
<td>Create knowledge</td>
<td>Vertical information systems</td>
</tr>
<tr>
<td>Defer commitment</td>
<td>Continuous improvements</td>
</tr>
</tbody>
</table>

*Table 3. Adjustments of Lean principles to fit the software context according to Poppendieck and Poppendieck (2007).*
It can be noticed that the different principles have an overlap. This is not strange but rather reflects the essence of the Lean philosophy; that the aim is to provide a holistic approach of how to deliver value to the customer by linking the processes to one another and thus the overlap is natural (Poppendieck and Poppendieck, 2007).

### 2.4.2 Adjustments of Lean tools

Just as for the principles, the tools in the Lean toolbox must be adjusted to the specific context in order to fulfill their purpose. As earlier mentioned, the number of available practices is numerous. However, all are not relevant to use in every situation or context where Lean is adopted. Wang, Conboy and Cavley (2012) has collected a list (Table 4) of relevant Lean practices that have been researched and applied in software development contexts, which is the context represented in this case study. The list of previously used tools are presented below:

- Address bottlenecks (Liker, 2003; Goldratt, 1992, 1997; Middleton et al., 2005; Poppendieck and Poppendieck, 2003)
  - Cumulative Flow Diagram (CFD)
- Avoid too much local 17nstill17tion (Poppendieck and Poppendieck, 2003)
- Defer decision making (Thimbleby, 1988; Poppendieck and Poppendieck, 2003)
- Develop appropriate incentives/rewards (Ambler and Kroll, 2007)
- Hansel: relentless self-reflection, to acknowledge one’s own mistakes and to commit to making improvements (Liker and Hoseus, 2008)
- Heijunka: workload 17nstill17uction smoothing. It aims at reducing muda (Liker, 2003; Middleton et al., 2005).
- Hide individual performance (Poppendieck and Poppendieck, 2003)
- Jidoka: intelligent automation, automation with a human touch. People should not serve machines but vice versa (Liker, 2003; Liker and Hoseus, 2008).
- Kaikaku: radical improvement within a limited time (Womack and Jones, 1996).
- Kano analysis: link voice of the customer to requirements (Middleton et al., 2005; Raffo et al., 2010).
- Make everything transparent (Womack and Jones, 1996): Make project status highly visible. Visualise all work items
  - Measure and manage (Anderson and Garber, 2007)
    - Employ queuing theory (Reinertsen, 1997; Goldratt, 1997, 1992) but measure the right things (Reinertsen, 1997)
    - First-In-First-Out (FIFO) queue
  - Move variability downstream (Poppendieck and Poppendieck, 2003)
  - Plan-Do-Check-Act (PDCA) cycle (Deming, 1986)
  - Pragmatic governance (enable first, manage/control second) (Ambler and Kroll, 2007)
  - Pull the andon cord: promote a “safe to failure” environment and 17nstill a “stop the line” mentality (Poppendieck and Poppendieck, 2003; Womack et al., 1990)
  - Quality Function Deployment: transform the voice of the customer into engineering characteristics and appropriate test methods (Raffo et al., 2010).
  - Reduce slack (Middleton, 2001)
  - Root cause analysis
    - The 5 whys? (Womack et al., 1990)
  - Use pull systems. Kanban board, Limited WIP, CONWIP (Sugimori et al., 1977; Bradley, 2007; Kniberg and Skarin, 2010)
    - Batch control processing (Bradley, 2007), Minimal Marketable Feature (MMF)
  - Value stream mapping: analyse and design the workflow required to bring a software or service to a customer (Womack and Jones, 1996; Liker, 2003; Poppendieck and Poppendieck, 2003; Mujtaba et al., 2010).

Table 4. List of earlier successfully implemented Lean tools in a software development context.
2.5 How to succeed with a Lean implementation

One of the major barriers to a successful Lean implementation is the misapplication of Lean tools. The misapplication can be of three kinds; using the wrong tool for a certain problem, using one tool to solve all problems and using all tools on every problem. Misapplying Lean tools may waste additional time and money and it may decrease the confidence employees have in implementing Lean manufacturing (Pavnaskar, et al., 2003).

In addition to the need of applying Lean tools in a proper way, research shows that the ability to understand the benefits from a Lean implementation in advance is highly important in order to succeed with the implementation itself. This is essential since a Lean transformation takes time and only when the incentives are important enough the company will succeed in keeping the Lean work a high priority task (Portioli, Staudacher, and Tantardini, 2010). The final aspect to consider when implementing Lean is how to establish the new mindset that Lean requires throughout the organization (Bhasin and Burcher, 2006).

In order to successfully implement Lean both a business and an operations strategy must be in place. The business strategy defines what type of customer need the organization wishes to satisfy. Hence, the business strategy is about defining what value is according to the customer and choosing and prioritizing objectives based on this definition. The operations strategy expresses how the organization should meet this need. In other words, this strategy helps realizing the business strategy by defining how value should be produced. (Modig and Åhlström, 2013)

A Lean operations strategy involves making a correct priority between flow efficiency and resource efficiency. According to the Lean philosophy, flow efficiency should clearly be prioritized. This since an organization can reduce a lot of superfluous work and waste by putting focus on flow efficiency. Reducing these superfluous and eliminating waste will in turn automatically lead to improvements of the resource efficiency. Thus, focus on flow efficiency fosters an improvement of resource efficiency, but the same does not appear the other way around. (Modig and Åhlström, 2013)

In order to ensure taking actions that are improving the flow efficiency, the strategy must foster that change are made in such a way that the entire value stream works more in line with Just-In-Time (JIT) and Jidoka. JIT and Jidoka are called the two pillars the in the Lean philosophy and are necessary to have in place in order to take appropriate actions that supports a Lean implementation (Ohno, 1988). Working in line with JIT means ensuring that the flow delivers what the customer want when the customer wants it. Working in line with Jidoka means fostering the overall flow by establishing transparency which ensures that everyone knows what to do at all times so that disturbances in the flow can be identified and eliminated immediately (Modig and Åhlström, 2013).

However, no exclusive Lean operation strategy, that always foster flow efficiency, exists. Though, during the part years several methods possible to draw on when starting a Lean implementation have been explored in different research studies. Smith (2012) has reviewed and summarized the findings in the existing research and the following paragraphs describe the approaches that Smith highlights.

Åhlström (1998) suggests that in the early phases focus should be on finding and eliminating defects and delays. When zero defects are achieved it is proposed that focus should shift towards continuous improvements. However, Åhlström (1998) does not promote any hands-
on method for which tools to use and in what order they should be used. Furthermore, the
benefits from following the advice presented in the article are not assessed.

Karlsson and Åhlström (1996) takes another approach and focus is put on implementing
“quality circles” i.e. small groups of people who regularly meet and discuss possible
improvements. This method relies on involvement among employees and gives some hints of
how to work more Lean. Though, their method lacks an explanation regarding what tools to
use and how to measure the effects of the actions.

Detty and Yingling (2000) suggest an ambitious method that relies on digital simulations of
the current and future state of the process. This results in a detailed visualization of a
possible future performance. However, the method is costly and requires a deep analysis of
the process as a first step of preparation.

Rother and Shook, (2003) states that Value Stream Mapping (VSM) is a suitable method to
use as a first step for any organization that wants to implement Lean. The method of Value
Stream Mapping indicates what and when a tool should be applied, as well as the current- and
future-state map giving a hint of the effects given by implemented changes. Though, VSM
does not consider any issues with the cultural change, which is an unavoidable issue in an
organizational transformation.

Finally, Productivity Press (2002) proposes a method that starts with creating awareness
throughout the company about the implementation of Lean being performed. Thereafter 5S is
used. The 5S implementation is then followed by the usage of several Lean tools. The journal
presents a basic framework for how to implement Lean, although it lacks in details regarding
what tools to apply and in what order to apply them. Thus, the feasibility of the method is
questionable.

None of the above-mentioned methods for Lean implementation is comprehensive. However, the method claimed by several researchers as the most suitable one to use, during
the first step of a Lean implementation, is Value Stream Mapping (Pavnaskar, et al., 2003;
Singh and Sharma, 2009; Vinodh, Arvind, and Somanaathan, 2010). This is due to the fact
that VSM gives an indication of what actions to take and where in the process to improve.
Additionally, a VSM provides the possibility to measure and evaluate the potential benefits
of the actions at a rather low cost. It is a simple visualization of the process also favors the
necessary cultural change as it provides essential understanding for the benefits gained by the
process improvements.

To further strengthen the argumentation for using VSM in this specific study visualization
has turned out to be supporting the software developers in handling the coordination of tasks.
This as the visualization provides them awareness of parallel activities, allows them to self-
coordinate by placing their work in the relation to other actions related to the development
process (Solding and Gullander, 2009).

2.6 Value Stream Mapping

2.6.1 Benefits of VSM
According to the first Lean principle all non-value adding activities should be eliminated as a
first step in working towards improving a process. These activities are called wastes
(Poppendieck and Poppendieck, 2007). To succeed with waste elimination the activities or
processes that ought to be classified as waste must first be identified. Value Stream Mapping is a Lean practice that helps in visualizing activities and processes in a value stream and works as a support for both identification and elimination of waste. VSM facilitates re-designing a process in a comprehensible and adequate manner with the aim of eliminating waste (Musat and Rodríguez, 2010).

Research also shows that the simplicity of Value Stream Mapping favors the necessary understanding among the management regarding the problem of the process and the need for improvements. Often, managers tend to blame decreasing competitive advantage on high costs (McDermott, 2002). Though, the real reason is often that resources are spent on the wrong things i.e. the process includes a lot of waste. A value stream map makes it easier for all employees to realize how their processes look like today and how an improved process could look like when waste have been eliminated. A VSM is cheap, fast and easy to understand (Serignese, 2010; Smits, 2007).

In order to successfully realize the identified improvements of a process, changing the beliefs and behavior among the employees is essential (Emiliani and Stec, 2004). However, driving people towards change is not an easy task and depends heavily on the corresponding leadership belief of the managers close to the process (Argyris, 2002). Value Stream maps help managers in reflecting over their fundamental beliefs, which can lead to changes in their behavior and in addition a changing behavior among employees and thus an improved process (Emiliani and Stec, 2004). Thus, Value Stream Mapping evokes changes and actions that eventually leads to a decreased time-to-market, better ability to produce Just-In-Time and an increased capability to produce cheaper products with the same quality (Musat and Rodríguez, 2010).

2.6.2 Notions and icons in VSM
The icons are illustrated in Figure 2 (below). A value stream consists of several activities classified as value adding (VA), non-value adding (NVA) or necessary but non-value adding (NNVA). Together, these creating the flow, value stream, that brings the product or service to the final customer. The stream includes both the flow of material and the flow of information and the VSM takes both flows into consideration when improving the value stream. When making a value stream map a predefined set of icons are often used. These facilitate the making of unambiguous interpretations regarding the VSM and make it easier to agree on purposeful decisions that would improve the value stream (McDonald, Van Aken, and Rentes, 2002).
2.6.3 Steps in the creation of a value stream map

The first step in Value Stream Mapping is the identification of an appropriate start and stop position in the process (Poppendieck and Poppendieck, 2007). In general, the start and stop position are both portrayed by the end-customer. Though, many value streams become too long and complex if they start and stop at the end-customer. Such value streams can preferably be divided into smaller parts and internal customers can be used when determining start and stop position.

Then a map of the current state of the value flow is drawn, which shows how and where value is created in the investigated value stream. This map is often referred to as a current stream map. The next step is to analyze the current stream map in order to identify which activities that add value and the ones that do not add value. When this is completed, a new map is drawn, visualizing the desired flow of value through the value stream i.e. a value stream that favors realization of the Lean concepts (Liker, 2004). This map is often referred to as the future stream map (Musat and Rodríguez, 2010).

In order to evaluate the value creation of each activity some standard measurements are often used in VSM (Rother and Shook, 1998). These are:

- **Process time (PT)** – the time it takes for the product to pass through the whole value stream.
- **Changeover time (CO)** – the time for resetting the process in order to enable another batch to run.
- **Process availability time (AV)** – the time that the process is available, thus when downtime is excluded.

Moreover, the total time spent on value adding activities and the total time for the whole flow should be measured and the ratio is to be calculated. A high ratio indicates that a large part of the time is dedicated to value adding activities i.e. the value stream is efficient. In contrast, a low ratio indicates that a lot of time is spent on non-value adding activities i.e. the value stream is inefficient (Poppendieck and Poppendieck, 2007).
3 Methodology

This section describes the methodology used in the study. It also highlights its limitations and delimitations.

3.1 Case study

The entire methodology is illustrated in Figure 3 (below). This study relies on a case study. In general, a case study approach has its starting point in the case and aims to gain in-depth understanding for the specific issue in order to add knowledge or strength existing theories (Collins and Hussey, 2009). Lean practices has previously been mostly applied and investigated in repetitive processes where the flow of physical products can be followed. Since the purpose of this study is to investigate the applicability of Lean in complex contexts that differs from manufacturing a case study is suitable to use. This as it enables testing of existing theories in a new environment and the findings will thus enrich the knowledge base and understanding for how Lean can be applied in other contexts than manufacturing.

The case study takes an inductive approach to the research being made. Hence the starting point when drawing conclusions about the results presented, is based in an actual context.

3.2 Data collection

3.2.1 Literature review and theoretical framework

A review of the existing literature was performed to gain knowledge about the existing research made in the area and answers the second (1b) and partly the third (1c) of our sub-research questions (Collins and Hussey, 2009). Based on the findings in the literature review a theoretical framework was created to support the study with knowledge about the nature of software development, Lean and provide deeper understanding for Value Stream Mapping. This was necessary to ensure having an educated knowledge base to rely on when performing research in the current context. Additionally, it was useful to sort out which Lean methods that are suitable to apply when aiming to detected company problems and how these should be adjusted. Thus, the theoretical framework works as a platform for both the current and future state maps of the value streams.

3.2.2 Interviews

Interviewing is a method used to collect data in order to find out what selected participants think, feel or perceive. In this way one can get in-depth understanding and ensure covering all angles of a problem. Hence, it prevents the risk of missing out of important aspects due to unawareness as a researcher (Collins and Hussey, 2009).

An essential task in this study is to ensure getting a correct understanding for the process in order to be able to map the value flow of the processes properly. Thus, making unstructured interviews with key people for all available components was a necessary first step to take. Unstructured interviews have the advantages of letting the researcher ask whatever question he/she wants during the interview. This method is therefore recommended to utilize when aiming for understanding of complex context and facilitates identification and formulation of the problem (Alvesson and Sköldberg, 1994). However, one should be aware of that unstructured interviews takes time to analyze and are complicated to execute in an unbiased way (Collins and Hussey, 2009). Hence, they are affecting the reliability of the study negatively but the validity positively as it favors a proper understanding of the situation.
Unstructured interviews were used to answer the first of our sub-research questions (1a) regarding current company problems. Moreover, data from these interviews were used to identify which two component processes that were most suitable to map. Key people related to the value stream of these two components where then participating in a second round of interviews. These interviewees were selected depending on the type of information that was needed and included team members, Product Owners, Program Managers, Strategic product managers and study manager. This second round of interviews was also done in an unstructured manner to ensure capturing all the details. Moreover, all interviews were recorded and transcribed directly after the session.

All participants were aware of their possibility to interrupt and leave the interview at any given moment and everyone was offered anonymity and confidentiality.

### 3.2.3 Observations

Observations are a method for data collection that is most often used in a laboratory or natural setting to observe and record people’s actions and behavior (Collins and Hussey, 2009). In parallel with the interviews observations where made in a non-participant way in order to gain information useful to answer all of the three research questions. The demo meetings, telephone conferences and daily activities among employees etc. were observed. These observations add credibility to the information gathered by the interviews and justify claims made by interviewees. Thus, the problem that might appear with participants experiencing a situation differently compared to reality, as a result of close involvement and emotional engagement, could be minimized (Collins and Hussey, 2009; Rother and Shook, 1998).

### 3.3 Data analysis

#### 3.3.1 Selection of components

Due to time constraints only two out of the four available components were investigated. The choice of components was based on the extent to which the components are influencing other components and the CBA product as a whole. The choice of components was made though analysis of data from interviews made by relevant people that are involved in the value stream. Additionally, the components were chosen in order to capture all kinds of problems that are experienced at the Department. In this way the result, i.e. improvement suggestions, can be argued as easier for other, less complex components, to utilize.

#### 3.3.2 Root cause analysis and fishbone diagram

In this study root cause analysis was made in order to identify the symptoms of and actual problems that has been disclosed by the interviewees. The relation between each symptom and problem was thereby analyses and key problems were identified. Root cause analysis is a tool that helps identifying what, how and why an event occurred. Understanding these three questions regarding an event is argued as necessary in order to ensure making effective recommendations and take proper actions in order to solve a problem. (Rooney and Vanden Heuvel, 2004)

The symptoms and problems that appeared in the root cause analysis were then grouped by using a fishbone diagram. The fishbone diagram method was created to identify and group causes, which generates a specific problem (Ilie and Ciocoiu, 2010). Hence, a fishbone diagram makes the relation between the symptoms and the problems, revealed in the root cause analysis, more clear, which makes them easier to understand and investigate.
3.3.3 **Value Stream Mapping**

In order to answer the main research question (1) a Lean tool was identified, adjusted and applied in this case study. The findings in the literature review highlights Value Stream Mapping as a suitable Lean tool to examine when starts working with Lean. In addition, the nature of the problems that have been identified in the case study company can be classified as problems regarding waste created in, or flow of, the process. Thus, Value Stream Mapping can be argued as a relevant Lean tool to apply in this case study.

VSM provided a visualization of the current and the future state of the value streams related to the two selected components. In this way waste activities and room for improvements could be identified. The benefits from the application of VSM to this context was then analyzed and evaluated and conclusions regarding the applicability of the Lean in this context were drawn.

3.3.3.1 **Adjustment of VSM**

Since the literature lacks in research regarding how VSM should be applied successfully to a complex context as the Department, the general VSM method had to be adjusted to the specific case context in order to be applicable. In order to adjust the general VSM framework to fit the specific context literature on how VSM have been successfully applied in different contexts before was examine. Additionally, how software product development differs from manufacturing product development in general was researched. Based on the findings from this research an adjusted framework for VSM was developed in order to fit the case company context to the largest possible extent. This adjusted VSM was then applied to the value streams of the two selected components.

3.3.3.2 **Current state map**

The first step when applying VSM was to choose a starting and stopping point for the value streams under investigation. Based on the data gathered from interviews and observations the process activities and flows between them were then identified and plotted in the map for each of the chosen components. These maps were then corrected and confirmed by key people that work with the components to ensure that the map matched their perception of the value streams.

3.3.3.3 **Future state map**

The next step was to identify problem areas in the current state map that stuck up the flow and activities that add value and the once that do not. The purpose of the future map is to visualize how an improved process can look like by eliminating waste activities and ensure that each activity is linked to each other in the easiest way (Rother and Shook, 1998). This map facilitates taking the right actions in order to improve the process. In order to create such an improved value stream, the current state map was analyzed and with input from literature, interviews and observation a future state map was drawn. Key people at the Department assessed the map according to its feasibility and relevance and provided feedback.

3.3.4 **Interview analysis**

In order to come up with feasible recommendations regarding actions to take to improve the value stream, the current ways-of-working, company culture etc. needed to be considered. Thus, the interviews were analyzed in a way to detect improvement possibilities that not only improves the value stream but also fits the existing culture, processes and people.
3.4 Limitations

Normally when a product development value stream is investigated a physical product can be followed. In these cases the researcher him/herself can follow the steps in the value stream with his/her own eyes but this study must rely on the perception of the people involved in the value stream. Hence, this study needs to takes on a qualitative approach and thereby relies heavily on interpretations of information collected through interviews. The data collection in this kind of research tend to be more biased due to small sample size and the complexity in analyzing answers critically and correctly. Moreover, there is a risk for post-rationalization among the interviewees, which might result in incorrect information (Collins and Hussey, 2009). The answers might thereby to lack in reliability and credibility due to the fact that the participants knows that they are under investigation (Collins and Hussey, 2009).

Moreover, the features developed in the value streams vary both in kind and complexity from time to time. This makes it impossible to generalize the flow to the same extent as in manufacturing. Additionally, the study could not map the flow of all the components due to limited time. Thus, the study is very context dependent.

3.5 Delimitation

The scope of this case study results in several necessary delimitations to be made. VSM was only applied to two out of four available components. The VSM maps the transformation of an opportunity into a feature. Thus, the VSM does not map the value flow all the way to the end-customer, instead an internal customer is defined as ordered and receiver. Moreover, only the main value stream and its processes were taken into consideration for improvements. Hence, improvements related to sub-processes were not considered. However, the effect that these processes have on the value stream will be discussed. A final delimitation is that the study is performed in one specific ICT company department placed in Sweden.
4 Company context

The aim of this section is to provide knowledge about the context of the study, but also to explain the product, process and roles of key-people involved in the value stream. The information is collected from interviews with employees at the ICT Company and observations made by the researchers.

4.1 Organization

The ICT company under study delivers communication networks, telecom services and support solutions to customers in over 180 countries. Globally the Company has about 110,000 employees and present a wish of being a driving force in a communication intense world. This ICT company will throughout the report be referred to as the Company.

The Company is divided into three major business areas where our study takes place in the largest one. Moreover, this business area consists of a multitude lower departments and only one is available and thus relevant for this study. The Department that constitutes the context for this case study acts as a factory for software components. These components are parts of software products that are sold to the end-customer.

4.2 Product

The value flows in this study will not be caused by a physical product itself, but is represented of software code used in masts supplying a network of nodes with digital traffic. A node is a receptor or giver of digital traffic. The core of the software development process is to have teams create lines of code to build up the software. The code is clustered and integrated into each other in different stages towards becoming the end product shipped of to the end-customer. First off all, code is represented in something called a component, which is the level of clustering that is considered in this study. The components location in the CBA offering is presented in Figure 4 below.

What is referred to as a product in this study is by the Company called a feature. These features are improvements or new functionality related to a specific component. The components are building blocks that can be combined in several formations in order to supply the customer with a demanded functionality. Offering a customized product through this kind of clustered component based architecture (CBA) is, and has been, popular among large software development companies. This since their portfolios are constantly growing and become more fragmented. Hence, the portfolios become more costly to maintain if the Company would use traditional platforms. In contrast to the platforms, CBA builds on a modular architecture that favors the extent to which the product can be re-used across organizational borders both from a software- and hardware point-of-view. This kind of modularization of a product enables the customer to buy a product adjusted to their specific need and secures a good foundation for lower total cost of ownership. This is done through an alignment between the operation, administration and maintenance of the product.

At the Company, CBA is often described as a sack full of Lego pieces. Each piece represents a component and by assembling the components into one another a sellable product is created. Thus, the components are not valuable to the customer but when they are clustered they generate value. The instances that are responsible of coupling the components into a sellable product are called Applications. However, in this study they act as internal customers who are demanding new features of the components created at the Department. Thus, the Department in this case is considered as a factory. Besides the software attributes, a CBA
product consists of hardware (the physical attributes) and middleware (connecting the hardware to the software). The hardware in a CBA is a cluster of blades called Central Processing Units (CPU), which handles the incoming data and transforms it into outgoing performance of the product. The CPUs’ functionality are controlled by which applications the CPU’s are connected to or containing.

![Diagram of component relationship]

Figure 4. A schematic sketch of how a CBA based product is combined with the internal customer and the Department located.

### 4.2.1 Components

The components available in the CBA are numerous. However, only four of these are available to this study. These are in this report referred to as Component 1, Component 2, Component 3 and Component 4. Though, one should notice that there is a component in very close relation to these four, called Component 5. Component 5 is however a bigger Component in comparison to the other four and even though it has an interface towards Component 3 it is handled separately in another department. Thus, it is outside the scope of this study due to organizational reasons. The four components available for this study are:

**Component 1** – A component that configures the general operating system to the specific CBA environment. Thus, it works as an adapter between the Company specific components and a general operating system. It is a rather small component with few features, although it is essential in many products.

**Component 2** – A component that coordinates logs and configurations from the different Central Processing Units (CPU, hardware that carries and executes the information in a computer program). These CPUs are located in an application and ensures that the applications have correct status concerning availability of the CPU. It handles communication between the different CPU’s and makes sure a configuration made to one CPU is added to the entire cluster. Moreover, Component 2 processes works as an adapter, which adjusts the general community code to the specific environment of the Company.

**Component 3** – A “glue” component between Component 5 and Component 2. It enables adaption between these two components and isolates Component 5 from the underlying implementation, to promote portability. It has a tight coupling to Component 5, which is the component that holds the modules.

**Component 4** – Component 4 is the most complex out of all components. The function of the component is to load balance incoming and outgoing traffic to reduce the number of IP addresses the nodes use to send traffic. The component offers a function that provides multiple of CPU’s with one contact address both for outgoing and incoming data traffic. Another big function is the distribution of incoming traffic to the correct CPU and to balance the traffic so, for example, CPU with back-end tasks do not get overloaded.
4.2.2 Component selection

After investigating each component on a rather high level, Component 1 and 2 were chosen for further investigation and analysis. First off, as these components are included in most applications and therefore have the possibility to make a greater impact than the others on the company result if improved.

Secondly they were picked because both components are linked to a community. This makes their value stream more complex than the value stream of other components. Thus, the result will be easier for other components to apply the result of this thesis due to the fact that it most often is easier to remove processes than add.

Thirdly, these two components have different Strategic Product Managers (SPM), which widens the sample size when performing interviews and thus increases validity and reliability. Especially since the SPM has a central role in the process and communication flow, hence the opinion and perception of these are important to have as unbiased as possible.

Finally, as Component 2 has started to investigate the advantages of visualization while Component 1 has not. The Component 2 visualization board acts as a good starting point for the analysis and will facilitate a VSM creation.

4.3 Communities

As earlier mentioned, both Component 1 and 2 are in a close cooperation with a community to be able to provide their final product to Applications. Software developers’ contributions to a set of code, which is commonly used and that has general accessibility, create the communities. Such code is called open source code and the entire base of open source code is available at the communities.

However, the Company needs to have a certain part of the open source code packaged and with guaranteed service and support. For this, a third party supplier is used. Component 2 uses a supplier called Oracle. Oracle handles open source code from OpenSAF and the product they are submitting to the Company and the development team is OpenSAFfire. This product is delivered every 6-12 months. In contrast, Component 1 uses Linux open source code, which is provided by a company called Novell under the brand SUSE. SUSE is a packaging of what is called SLES (SUSE Linux Enterprise Server) and SLES is the subset of the Linux code utilized by Component 1. SLES is delivered every 3-12 months. Red Hat is another provider of SLES, which Component 1 is using. Red Hat is the standard provider of Linux code to enterprises, since they own 60-70% of the market.

Although a product delivery from the communities occur 1-2 times a year, the development teams can get smaller deliveries of alfa-code straight from the community. These are, however, not possible to use in a final product being released since they are not supported by the third party supplier if malfunctioning.

A part of the agreement when using open source and communities is for the Company to contribute with code to the community. To determine what the community works with and to provide tasks, tickets are given. These tickets can be seen as wishes from the Company side of the cooperation.
4.4 Roles and responsibilities
This section describes the roles and responsibilities of the people that are involved in the product development process:

Strategic Product Manager (SPM) – The SPM is responsible for the budget and strategic alignment of the component and functions as a strategic hub between Applications and the development units. Thus, they ensure that the requested opportunity is properly taken care of. Each SPM is responsible for several components and since they are not working with development they are allocated in another department than the components.

Leadership team (LT) - The members of the leadership team are managers for the team members that develop features for the component. They are responsible for recruiting suitable team members based on the actual need and budget of each component. Normally a LT only handles one component.

Program Manager (PM) – The PM is responsible for administration of the development project i.e. ensures allocating the budget, writing status reports and secure that necessary resources are available. These persons are communicating the big picture to the software developers in each component i.e. communicating contextual information and changes made due to company wishes. The PM is also responsible for driving the Pre-release control (PRC) and scheduling release plans.

Main Product Owner (MPO) – The MPO is a standardized role in Scrum, which is a version of the Agile philosophy. He/she is responsible for technical investigation, sectioning and prioritizing the requirements coming from Applications. They are together with the SPM making an analysis of each opportunity and they are the owners of the opportunity backlog. Moreover, they are responsible for quality and release planning.

Product Owner (PO) – The product owner is a standardized role in Scrum, which is a version of the Agile philosophy. This person supports one or more development teams. They are most likely senior product developers with a high level of technical expertise and understanding for the component. The PO relieves the MPO by taking the responsibility of breaking down user stories in even smaller tasks which explains how and when the team should perform a certain task.

Development team – The development teams are software developers, which are transmitting the requirements in to coded features. They are responsible for the design and the development of the feature as well as identification and implementation of quality improvements. At the Department, 3-4 development teams, containing in average 6 developers, are available to dedicate to development tasks related to either a community or the Company component. Since 2010 the Company has had a wish of working in line with the Lean and Agile philosophies. However, the implementation of Lean thinking at the Company is in an early state, and even if it is ongoing, it is not as established as the Agile philosophy. All software development teams are working in line with Agile today and the software development process is characterized by Agile practices and procedures.

Scrum Master (SM) – The scrum master is also a role from Scrum and is a “servant leader” that helps the rest of the Agile development team to follow the process. They are responsible for facilitating regular meetings such as demos, stand-ups and retrospectives and they ensure
that the team focuses on the right tasks and do not get disturbed during their development process.

*Study manager* – Responsible for organizing and planning the studies related to any CBA component. This person is responsible for updating and prioritizing the study backlog. The study manager is, together with LT and MPO, putting the study teams together.

*System architects* – These are people with extraordinary expertise and knowledge in software system architecture. They work at different levels in the organization. Some architects are located at “business area level” and does thus have knowledge in the overall architecture while others work at department level and are experts in the architecture regarding specific components.

### 4.5 Value stream

The value streams of the components under consideration differ in both kind of tasks and time. However, the base in the structures of the value streams is similar. The value streams starts and ends with the internal customer, which is referred to as Applications. When an Application perceive a need for a function or detects a problem with a component, they place an order to the Department in form of an opportunity that describes their wish. The opportunity is then turned into a feature that meets the requested requirements from the customer through a process including several activities. The general activities included in the process is opportunity analysis, study, preparation, development, testing, PRC and release.

### 4.6 Process

In Lean manufacturing the value-adding activities are often executed by machines that perform specific operations. These operations are executed in processes, which include a step in the value stream that the product has to pass in order reach the customer. However, in this context all operations are executed by people and not by machines. Thus, the investigated processes in this study are heavily relying on people and do not include any automatic machine operations. The only machine used is a PC computer but that is rather a tool used to realize the product that has appeared in a developers mind. The processes that have been identified in this specific value stream are the following:

*Meeting* - A gathering of people with the aim of discussing a certain topic in order to reach a decision, agreement or understanding for the topic. Meetings can be executed with all kinds of people at any stage of the value stream.

*Development* - The processes including production of code together with supporting and activities of preparation. This kind of process involves software developers only.

*Study* - The further examination of the design of a product. This is a sub-process that sometimes involves only a few members from a development team or people with specific expertise from other parts of the organization.

*Testing* - The verification and checking of the code that is made in order to minimize errors after implementation. Testing is made in parallel with the development process as well as in the end of the development phase where if often intensifies. The testing, that is made in parallel to development as a sub-process at another department that only works with testing. The Program Manager executes the final testing, sometimes with help from the Main Product Owner and the Strategic Product Manager at a phase called Pre-release control (PRC).
**Auditing** – Before the feature can pass PRC it need to be audited by a group of system architects at a higher level in the organization. This is made to ensure that the design is aligned with other components. However, this is also a sub-process to the value stream.

**Release** – This process includes documentation, final testing and informing about that a feature is available for implementation in a product. Thus, this is the final process in the value stream.
5 Company problem

The following section describes the Company problem. Seven major problems have been identified after performing a root cause analysis. Each problem is reflected by symptoms that are visible in the everyday business. The connection between the symptoms and the problems are presented and described in the following paragraphs together with a corresponding fishbone diagram.

5.1 Identified problems in the Company

The identified problems and symptoms are visualized in a Fisbone diagram on page 36. The problems where established during a root-cause analysis. The symptoms and problems where obtained in the first round of interviews and during observations, and no symptom where excluded when performing the root-cause, merely sorted to find the actual problems within the Company.

5.1.1 Limited holistic understanding of the value stream

The wish of the Company to work with Lean creates a problem when the strive for “continuous improvements” is set in the limelight. A problem for the Company today is that this wish is not unified with knowledge about the Company’s own processes. Hence, improving the value stream as a whole becomes hard. Previously, improvements have therefore been done to the later part of the value stream by isolating problem areas and solving the specific problem for this process. The focus of the Department has mostly been on trying to improve the development teams and their efficiency. However, the unified perception at the software development department is that the necessity of the activities performed in the software development process has not been identified or evaluated. Hence, they experience a lack of information regarding what to improve and how to improve it. Although the Company expresses awareness about bottlenecks in the value stream, these are rather unexplored regarding their existence and effects on other parts of value stream.

Today no measurement of the performance of the whole value stream is made. This results in releases being postponed due to non-ability to adjust the value stream when it deviates from its intended performance. Another indicator is that defects are discovered late since each person in the flow is only focused on his/her own task. This is due to lack in knowledge about following process steps and what these demand from the earlier tasks being performed. Furthermore the interplay with sub-processes is insufficient. Thus, the blame for errors are constantly pushed from the components on to Applications, the community, testing, auditing or study, and vice versa.

All these symptoms indicate a lack of holistic understanding and results in that no improvements are made to the value stream as a whole. This is additionally due to lack of knowledge in what such an improvement should or could be.

5.1.2 Temporary lack of resources

An obvious issue at the Department is their lack of resources in, mostly, personnel. Today there are vacancies in the teams not being filled and some components, such as Component 1, are not able to allocate resources to the tasks being planned or performed. This is due to that people with the correct competences in either open source or software development at the Company is hard to acquire.
One symptom related to this problem is delays of the releases due that a task demands a certain amount of man-hours. When this task is handled by fewer men then desired, it will take more time to complete. Another symptom is that studies are not being conducted due to lack of available people who has the time or knowledge to handle them in parallel with their everyday work. Additionally, these long studies contribute to the releases being delayed. It can also lead to errors being detected later in the value stream. This since proper resources is not allocated to the early stages in the value stream and hence the error detection is tardy. Moreover, the teams do not have as many software developers as included in the budget. This is due to that the leadership team cannot find suitable people to recruit for these positions.

5.1.3 Insufficient communication with Applications

Since Applications are viewed as the customer in the value stream, the communication between the Department and Applications is rather critical. Today, the initial communication is handled through an SPM and no direct contact between Applications and the corresponding component is established. Hence, communication between components at the Department and the Applications is strictly bound to personal relationships or just contacting whomevers contact details can be obtained the easiest.

The result of this in the Company is firstly that the components produce features, which are not used immediately by an Application. This since no dialog about the utility and usefulness of the features is made during the development. The demands (opportunities) sent from Applications are non-defined or unclear and cannot be clarified until they reach someone with the correct technical competence. Additionally, the fact that this backlog exists is a symptom of poor communication with Applications. They apparently add opportunities in the opportunity inbox without confirming or understanding the relevance of the whishes. The insufficient communication also creates an environment where a lot of handoffs are present. These handoffs are time consuming and create risks for misunderstandings.

A bad relation, and communication, between Applications and the components creates a lack of knowledge regarding each other’s processes; hence applications will not take the releases being put out by the components since they are not synchronized with their own workflow and output. The lack of synchronization between the departments also contributes to changes in demands being done at a late stage, which causes inefficiencies or delays of the releases.

When a product is released Applications are guaranteed maintenance for a certain amount of months towards end-customers. This means that each release causes a maintenance track, or version of the software to maintain. This maintenance takes resources from the software development process of the components and the software developers are occupied with maintenance and support related work. However, this is not entirely understood by Applications as they are not informed about the resulting effects that these maintenance tracks have on the overall feature development capacity.

Since a pull flow is strictly defined as acting on demand a push flow is created when the contrary is done in the development of a product. The insufficient communication with Applications can sometimes contribute to the product becoming dated and unwanted. Even if the intent was to act on demand, the product is being pushed throughout the value stream and also sometimes altered along the way. This results in the market not wanting the final product. Sometimes the miscommunication can also lead to that a “not-wished-for feature” being produced. The developed feature can then, in retrospect, be seen as pushed towards the market.
5.1.4 Somewhat insufficient quality control towards Applications

Today, no sufficient quality control of the code is done towards Applications. This is due to that the assessment of the applicability of the code is first finalized in the PRC step of the value stream. Not having sufficient quality until the end of the value stream is a result of Applications unwillingness to accept releases. This since the releases does not attain the same level of quality as previous version already being used by the customer. Moreover, it is due to that some of the final testing is extensive and affects components. These operations are therefore preferably done at one single occasion for all features that should be included in the coming release. Since the final testing is done at such a late stage, faults can be detected close to a release date, which might cause delays and put a lot of pressure on the developers.

In addition, the view upon quality differs. Some Applications consider the Testing shipments as sufficiently tested to initiate a build upon, even if it cannot be released to end-customer. The view upon the quality of Testing shipments also differs among the employees in the component. Quality in itself is a very subjective term.

5.1.5 Resistance among the software developers

The software developers present a unique set of competence and knowledge. Though, they are currently not managed in how or what they should do at the Company. One can notice an unwillingness to do less exiting tasks among the developers together with a form of resistance to changes that commonly exist in all organizations (Fleming and Spicer, 2006).

This unwillingness hinders processes and can contribute to a big backlog stacking up due to that software developers are not taking on tasks. Especially in the end phase, documentation can be quite heavy due to the resistance of performing the documentation meanwhile developing features. The fact that the developers can somewhat influence the tasks being put on the backlog can be beneficial. However, it can also contribute to features, not necessary to the customer, being developed. Lastly, the studies being performed needs resources and if the software developers do not wish to perform studies, this will disrupt the carrying through of the studies. Delayed studies will lead to delayed product releases.

Occasionally, the software developers engagement in contributing to the Company is not evident. This leads to them suggesting and developing features or upgrades not connected to any customer request. This can mostly be seen in teams that work for and in the communities, such as the OpenSAF team. Thus, their only aim is to contribute to the community rather than to the company.

5.1.6 Lack of technical knowledge upstream

Today, people with sufficient technical knowledge is not getting in touch with the feature until late in the value stream. A symptom of this is that features are often sent backwards in the value stream due to that a need for clarification is not detected until the developers get in touch with the opportunity i.e. in the development phase. This results in that the features might need to be analyzed once again or is rejected later in the value stream. Thus, unnecessary time is spent to the feature.

Another symptom is related to the SPMs. Their function is to act as a mediator between Applications and components. However, in general they have very little technical knowledge about the product. This lack of knowledge hinders proper communication between Applications and components. In turn this will lead that errors are detected late in the value stream due to that the SPM act as a gatekeeper that can pass by errors or insufficient information. If the demands are insufficient from the beginning it can lead to requirements
being changed during the development process. Such hinder in communication and changes in demand can in addition lead to postponements or delays of releases. Since the SPM does not have technical knowledge a handoff is made to ensure the technical quality of opportunities, which as mentioned earlier is time consuming and increases the risk of misunderstandings.

5.1.7 Unclear roles and responsibilities

During the recent years, the way-of-working and the roles and responsibilities of the employees have changed several times at the Department. As for all changes they do not happen over night and need time, communication and involvement from the whole department. However, at the Department the changes have failed in many ways and caused more confusion than structure. This is heavily due to that the involved people have not understood the need or responsibility for their new role or the forum that they from now on should attend or drive forward.

Another symptom of the fuzzy roles and responsibilities is that the people in general lack in knowledge regarding who is responsible for what. Moreover, the description of the responsibility that comes with a role differed a lot depending on who is asked about it. Additionally, the team members that work for the Company, but are hired as consultants, do not even know who their closest manager is. These confusions in responsibilities also wastes resources as people participate unnecessary meetings i.e. meetings that they are not affected of or can contribute to.

Another big symptom is the lack of evident follow-up and control of implemented changes to the Company structure. Lack of Key Performance Indicators (KPI) and who is responsible for driving change or new ways-of-working are evident. Many employees’ even state that even the wish of working Agile is not fully implemented or enough evaluated before implementation.

5.2 Comments to company problems

It is quite apparent that there are many interrelations between the symptoms and problems of the Company. Hence, the complexity of the problems requires a systematic approach to ensure a holistic solution. Merely solving the symptoms will not ensure a more efficient value stream and additionally solving the problems without a holistic picture in mind will not ensure an improvement the operational outcome of the value stream.

As argued in the literature review, Lean has been proven to improve the operational outcome of product development by embracing a holistic approach. Thus, it is relevant to use when optimizing a value stream that includes several complex problems. Drawing on Lean thinking is also necessary in order to reach the aim of this study. In order to “leanify” the problems detected in the value stream, guidance from the Lean framework, which is presented in the theoretical framework section, is used.

The symptoms and problems in the value steam can be classified into two “Lean groups”. The first includes symptoms and problems that hamper the flow of the value stream and the second includes factors that can be seen as waste instead of value creating. Further, the flow-related issues are divided into two sections, one regarding the information flow, the other one concerning the flow of the actual product. By also highlighting the importance of good management, the study ensures covering all the aspects that should be consider when solving the problems by applying Lean.
Figure 5: Fishbone diagram of company problem.
6 Adjustments of Lean to fit the Department

This section aims to describe the adjustments that have been made to general Lean concepts and the classic VSM method in order to ensure the best possible fit to the specific context.

6.1 Adjustments of flow

A development process includes several kinds of flow. Products, material, people and information creates flows in all value streams. However, the nature of the flows is not similar in all contexts and to avoid a misunderstanding the different flows should be identified and redefined for each value stream. The following two sections describe the interpretation of the meaning of a product and an information flow in the value streams considered in this study.

6.1.1 Product flow

Product flows can be intermittent due to bottlenecks. In manufacturing the bottlenecks are represented by, for example, machines being ordered to perform more operations or working faster than possible. In the software development context, bottlenecks occur when people or processes experience more work than they can manage or produce. A bottleneck is hence characterized by a long waiting time before the process itself, not that the process is time-consuming. However, as in manufacturing, a bottleneck still indicates an issue in capacity or planning.

In manufacturing a pacemaker is the process that regulates the product flow. It controls the takt for the material flow, typically upstream, in order to ensure meeting the customer demands regarding when the product is scheduled to be delivered. This process adjusts the ingoing material flow into the process to smooth out the flow based on the customer demand rate (takt time). In software development the customer demand rate will vary a lot due to the uniqueness of all features that is demanded. A pacemaker-process in software development is hence depending on the outline of the production processes. Meaning a pacemaker can be whichever process that sets the takt and controls what is taken into production. Thus, in this context a pacemaker can control both upstream and downstream processes since it acts as a hub for the “material” that passes through the value stream.

6.1.2 Information flow

Both in manufacturing and in software development information flow to, from and within the value stream. In manufacturing a distinction between manual and electronic information is made. Manual information is people sharing information, person to person or person to machine, and electronic information is when information is shared through an electronic source. The second is usually the case when sending orders or feedback.

In the context of this study the product is realized in an electronic environment and hence the information trade is most frequently electronic. This is also due to the fact that the Company is rather big and geographically divided, which makes it difficult for people to physically meet and thus manual information flows becomes rare. Hence, in our context, manual information is seen as information sent from one person to another through e-mail, telephone calls, or face-to-face meetings.

In contrast, electronic information flows are classified as when information is sent between instances and thus is available for a lot of people. One example of this is information sent between two instances such as community and the Department. Thus, all feedback directly affecting the product and that is available for every one is classified as electronic information. These definitions are thereby strictly bound to the nature of the sender and receiver of
information and very little regarding the channel they arrive through, which is the case in a manufacturing context.

6.2 Adjustments of waste

6.2.1 Different kind of wastes

The most distinguishing principle of Lean thinking is the one regarding waste elimination (Monden, 1983). What should be classified as waste depends on the specific process. Though, literature on Lean manufacturing highlights seven common wastes that can act as a general framework to rely on when identifying non-value adding activities in any kind of product development processes (Poppendieck and Poppendieck, 2007). Poppendieck and Poppendieck (2007) have investigated and adjusted these seven wastes in order to make them more relevant to a software development value stream. Table 5 shows the seven wastes in the different contexts.

<table>
<thead>
<tr>
<th>Manufacturing</th>
<th>Software development</th>
</tr>
</thead>
<tbody>
<tr>
<td>In-process inventory</td>
<td>Partially Done Work</td>
</tr>
<tr>
<td>Over-production</td>
<td>Extra features</td>
</tr>
<tr>
<td>Extra processing</td>
<td>Relearning</td>
</tr>
<tr>
<td>Transportation</td>
<td>Handoffs</td>
</tr>
<tr>
<td>Motion</td>
<td>Task switching</td>
</tr>
<tr>
<td>Waiting</td>
<td>Delays</td>
</tr>
<tr>
<td>Defects</td>
<td>Defects</td>
</tr>
</tbody>
</table>

*Table 5. Seven wastes in manufacturing and software development. (Poppendieck and Poppendieck, 2007)*

The value streams that this study takes into consideration do not only include development of software. It also includes clarification and design of tasks etc. why the wastes present in our value streams will be drawing on both the seven wastes in manufacturing and the seven wastes in software development. The following section aims to discuss the available definition of kinds of waste in order to elaborate a list of the seven wastes that fits our specific context the best. These final wastes are presented in Table 6 on page 41.

6.2.1.1 Prolonged items on backlog

Since software features are not physical products no material inventory exists in a software development value stream. Thus, a better waste activity to look for in this context is partially done work (Poppendieck and Poppendieck, 2007). Here the objective is to minimize the risk that the product is getting stuck in the process and that aim is to integrate, test, document code in a single rapid flow to ensure that all product are taken through the whole process and nothing is left unfinished in the process. To accomplish this, small iterations or batches must be used. Because, small iterations means smaller amounts of information and data, which will prevent that the development gets stuck in the process (Bauch, 2004).

Working with small batches comes naturally when embracing the Agile philosophy. Since the development phase in our case study is working in an Agile way, their process involves splitting tasks into smaller pieces that can be developed during the time of one sprint. These small items cause sort of an inventory, as they are stored on a backlog while waiting for the developers to become available. Hence, inventory in our value stream is in form of items on backlogs. This inventory is not physical as in manufacturing and is not as costly to store. Though, they cause bottlenecks and an undesirable intermittent flow when they get stuck on a
backlog for a long time. Thus, in this study the first waste is labeled as prolonged items on backlog.

6.2.1.2 Extra features
Making things that is never used is often referred to as the worst wastes of them all and overproduction occurs both in manufacturing and in software development (Poppendieck and Poppendieck, 2007). The cause of overproduction is lack of synchronization of up and downstream processes, including both the planning and execution stages of the process. Though, due to the low repetitiveness and the fact that information has a central role in software product development the synchronization of the processes does not look the same in development as in manufacturing. Hence, some aspects have to be adjusted, added or subtracted (McManus, 2004). For example, in development processes synchronization of the distribution of information is much more important than in manufacturing processes (McManus, 2004).

In software development over-production results in that features are produced even though a demand does not exist i.e. when features are developed but never used by the customer (Poppendieck and Poppendieck, 2007). To ensure a proper evaluation of whether a feature should be developed a correct interpretation of customer demands and sufficient and accessible information is essential. Since the final product in this case study is a coded feature the waste suggested by Poppendieck and Poppendieck (2007) is suitable to use in our specific context.

6.2.1.3 Extra processing
Waste in form of extra processing occurs in manufacturing when oversized equipment, not properly maintained equipment or equipment that are not designed for handling the task is used. This results in an excessive processing time and thus cost compared to if the most proper equipment had made the operation (McManus, 2004). In software development the processes are located in peoples minds and hence one major issue is to secure a rigorous approach to capturing proper knowledge associated with the specific task. In knowledge work contexts, this favors by engaging people, with essential knowledge, early in the development process. This is necessary to secure that the most suitable person with essential knowledge is giving their input on the task at the right place in the right time (Poppendieck and Poppendieck, 2007).

In our specific value streams both extra processing and relearning occur when people with wrong or insufficient competence is placed on a certain task. We will use extra processing as the term for this waste. This as capturing and sharing knowledge rather is a way of decreasing the need for extra processing than a waste in itself.

6.2.1.4 Handoffs
Since software is a non-physical product no waste from transportation of material appears. However, several persons with different skills and qualities handle the information regarding the feature before it becomes a finished product (Bauch, 2004). Thus, the value stream includes unavoidable handoffs, which demands a learning process where all complex tasks are handed over. To ensure the most efficient handoffs, they should be made personally and not through documentation to the largest possible extent. In this way the time to capture knowledge will decrease and thus a structured way to hand over knowledge is necessary in order to maintain the development process. One way to do this is to use cross-functional teams, so that people can teach each other naturally along the process. (Poppendieck and Poppendieck, 2007)
The value streams under study do not include any transportation of physical goods. However, many handoffs appear due to that several people are involved in the process. The most suitable definition of this waste will hence be handoffs and not transportation.

6.2.1.5 Task switching
Unnecessary motion of the product can never be seen as anything else than a waste. In development processes the waste from movement is usually considering the movement of people caused by lack of access to information e.g. data or people. (Bauch, 2004) Each new task includes a learning phase where you have to search for information and understand the task before starting the development. Hence, task switching is a cause of waste in all development processes. Minimizing the number of tasks that a developer has in his/her track is therefore desirable in order to minimize the time spent searching for information. By doing so the full potential of the developer can be utilized, which in turn will provide the best possible quality on the code. (Poppendieck and Poppendieck, 2007)

The value stream in this study touches both of the above descriptions of waste. However, the study will use the term task switching as it is more relevant in knowledge work and since motion easier can be related to movement of a physical products and thus cause confusion.

6.2.1.6 Delays
In a development process waiting is defined as the idle time due to unavailable manpower, information or computing resources (McManus, 2004). Poppendieck & Poppendieck (2007) claims that delays are a better description of the waste than waiting in a software development context. One cause for delays is when people are not available when needed. Another reason for delays is the lack of understanding and hence need for extra research before making a decision. In order to decrease the delays one should work in a collected manner, in short iterations and with regular feedback. Additionally, ensuring that knowledge is available to all people involved in the process is essential in order to minimize the time needed to search for knowledge. (Poppendieck and Poppendieck, 2007)

In our case, both waiting and delays are present. However, waiting can be included in long lasting items on backlog why we will use delays as the sixth waste.

6.2.1.7 Defects
Defects can appear both in code in a software development environment and in physical products in manufacturing. In order to ensure that defects are eliminated to the largest possible extent from the code the process must include frequent and extensive testing activities. Whenever a defect is found, a test should be created to secure that the defect does not appear again. In general, focus should firstly be on extensive mistake proofing tests and secondly on finding defects as early as possible. Thus, the earlier in the value stream that testing become involved the better (Poppendieck and Poppendieck, 2007). Since coding is an important part of the value stream under study, defects is a waste that must be investigated in this value stream.
Table 6. The seven manufacturing wastes adjusted to the context of the Department

6.3 Adjustments of VSM

6.3.1 Icons

The standardized icons that are used in a VSM are made by and for manufacturing and aims to map a physical flow of material. Thus, some of the icons must be adjusted or redefined to fit a software development context. These adjustments mean redefining the meaning and content the icons to better fit the nature of the context. Table 7 gives an explanation to the meaning of the icons in this specific study.

<table>
<thead>
<tr>
<th>Manufacturing</th>
<th>Software development</th>
<th>Explanation</th>
<th>Icon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual information</td>
<td>Manual information</td>
<td>All exchange of information between two persons is classified as manual information.</td>
<td></td>
</tr>
<tr>
<td>Electronic information</td>
<td>Electronic information</td>
<td>Electronic information is information that is not meant to travel between two people but rather two instances e.g. between the community and the Department.</td>
<td></td>
</tr>
<tr>
<td>Shipment arrow</td>
<td>Shipment arrow</td>
<td>A shipment is the delivery of a finished and quality tested product to a customer. This is not context dependent even though the shipment in software development is not physical.</td>
<td></td>
</tr>
<tr>
<td>Shipment truck</td>
<td>Shipment</td>
<td>The symbol for a shipment is a truck, which highly emphasizes the manufacturing relation of the Lean concept. In software delivery no logistics of this kind is required. Hence the symbol of the truck will remain but it will mean a non-physical shipment. The product sent is digital.</td>
<td></td>
</tr>
<tr>
<td>Customer</td>
<td>Applications</td>
<td>The value streams that are taken into consideration in this case study do not include direct contact with the end-customer. Hence, in this VSM a proper “conceptual” customer to use is Applications.</td>
<td></td>
</tr>
<tr>
<td>Process</td>
<td>Process</td>
<td>A process is when an operation is made on the product and thus adding value to it. During a process someone is actually handling the product. Hence, processes require people to be actively involved. The number within the circle is the number of people directly involved in the process.</td>
<td></td>
</tr>
</tbody>
</table>
Production control | ERP system | A Production control facilitates planning and control of material, time and flows throughout the value stream. To be labeled as a production control it must thus have control over the process and utilize an efficient production regarding time and cost.

Push | Push | Push means creating or add value to something not asked for, such as acting on own demand i.e. improvements or wishes from the software developers. The entire flow is regarded as push when passing a product, which is not asked for by the upcoming process.

Pull | Pull | Pull is simply letting the product further down the value stream when the next process is ready.

Inventory | Backlog | Since no physical products exist, the definition of an inventory is; a place where a product in the value stream is waiting to be refined in an activity. Hence, the opportunity inboxes and backlogs are the best representation of “the waiting place”.

Go see | AAC/IAC/MAC, RDA, PRC | Go see is a manual quality control. Software development environment and the nature of the product demand a (to the largest extent) digital control. Thus, steps of quality control of all kind will be marked with a go see anyhow. This just to show where actual batch quality controls are made.

Supermarket | Release | In production a supermarket is holding a fixed variation of products. However, in the software development nature the releases are unique and customer specific. In software development the supermarket is more loosely defined as where ready to ship products are being stored or held. Thus, in out case this means complete inventory.

| 6.3.2 Data collection and measures |

In a standard VSM data is collected by observing the activities and operations that are performed by machines. These repetitive operations can be measured over and over again and thus data can easily be collected by observing and measuring the machines when executing their operations. In this study the processes are not as visible as in manufacturing why observing and clocking the operations and activities is impossible. Hence, the best way to collect data was through in-depth interviews with key people involved in the processes. This forced the study to become more qualitative than a standard VSM and heavily rely on people’s perceptions of process activities.

In a standard VSM measures such as process time, changeover time and process availability times are used. However, all these measures are not easily applicable to the processes in a software development value stream. The changeover time in this context is the same it takes for a person to switch between two tasks. This time is very difficult to measure and will always include high uncertainty. The same applies for process availability time. Since no machines are performing the tasks in our process the availability time in our context is the time that each employee is actually working i.e. 40 h per week minus lunches, coffee breaks,
toilet visits etc. Making a correct estimation of these down times is very difficult, as people's perceptions of their time management most often differ from reality.

However, the process times can be estimated as well as the waiting time i.e. when a task is ready to be handled but is not. Moreover, the time people spend on tasks that are not related to the feature will be roughly estimated but not expressed as specific measures such as availability time or changeover time. This will enable to evaluate the overall efficiency of the value streams the ratio between time spent on non-value adding and value-adding activities will be estimated.
7 Value Stream Mapping of current state

This section presents the outline for the current state maps. The maps can be found in Appendix A and B.

7.1 Current state map Component 1

7.1.1 Opportunity submission

The value stream starts when an opportunity is sent to the Component 1 department from the internal customer Application. The opportunities are descriptions of new features and improvements that need to be added to the current functionality in order for the product to evolve or meet customer requirements. The submission of opportunities is done through an ERP system and the opportunities are collected in an opportunity inbox. Each opportunity is getting an ID and the name of the submitter is always visible. However, requirements of the design of the submitted opportunities are beyond this very unrestrained. Though, a submission template is in available but most often not used.

7.1.2 Opportunity analysis

The opportunities are then analyzed, regarding their feasibility, relevance and clarity, by SPM, MPO and when necessary, system architects on component or higher, BNET, level. This process is called Opportunity analysis and is executed during a one-hour meeting. The wish is to hold Opportunity analysis meetings every second week but currently they are executed more rarely. Based on that the meeting takes place every second week and the assumption that all new opportunities are analyzed at the next Opportunity analysis meeting, the waiting time before an opportunity is analyzed is at most 14 days and at least 0 days. Thus, the average waiting time is 7 days. The opportunities can go four separate ways after this meeting. A relevant and understandable opportunity is turned into a Main Requirement (MR) or multiple requirements if broken down, and put in an inbox called MR-inbox. A MR is a translation of the opportunity that better fits the lingo of the department and includes more detailed relevant to the development of the feature. Opportunities that affect several components or need further investigation, before passed through to the backlog, are sent for study Product Owner Group/Portfolio Efficiency System Study (POG/PEST). If an opportunity has a fuzzy description and need clarification, the person who submitted it or another person that works at the Application, is personally contacted and asked clarifying questions. The last scenario is that the opportunity is rejected due to the fact that it already exists in the functionality of the product, or because it is impossible to develop due to technical constraints.

7.1.3 Component Roadmap Meeting I

The MRs in the inbox are then prioritized in a proper order. This is done by the MPO and SPM at a Component Roadmap Meeting (CRM), which takes place every second week and lasts for an hour. The current level in the MR-inbox is 25 MR and at each CRM approximately 5 MRs are handled. The longest waiting time will hence be 35 days and the shortest 0 days. Thus, the average waiting time in the MR-inbox is approximately 17 days. The MRs handled on the CRM is put in a prioritized MR-inbox while waiting for Pre-screening.

7.1.4 Pre-screening

The Pre-screening is then made in another meeting where MPO and PO and sometimes team members are participating. The participation of the team members is on a voluntary basis. Although the participation of the team members is voluntary in this stage, they often do participate. During the Pre-screening each MR is broken down or grouped, as well as time
estimated and evaluated regarding its dependency or impact on other components. Sometimes
further clarification by Applications can be needed at this stage. A sketch on how the MR
should fit into the product backlog is drawn and the MR is broken down into smaller items,
called user stories. Pre-screening meetings are executed when needed and is depending on the
current number of heavy MRs on the backlog. However, all items must be pre-screened
before going into the development process, why at least one Pre-screening meeting is held
every third week to ensure that material is always available when the development process is
ready. This gives an average waiting time on 1,5 week for each MR at the prioritized MR-
backlog.

7.1.5 Component Roadmap Meeting II
The Component Roadmap Meeting is a process dedicated to approving pre-screened items
and put them on the Product backlog. Thus, the MR is passing this process twice during its
progress through the value stream. At the second pass through CRM, SPM and MPO identify
and send the MRs that affects other components than just Component 1 to Program Steering
Forum (PSF) to secure alignment with other components.

If the MR is not of a cross-component nature, and passes through the technical councils and
thereby becomes ready to implement, information regarding it is sent to the PM who updates
the Release Intent Decision (RID), of the feature. The MRs given a high or medium (green or
yellow) priority after the RID is then approved by the SPM and put on the product backlog
with a high priority. This indicates that the item will be included in the coming release.

When the MR has successfully passed the second CRM it is put on the Product backlog. The
top of this backlog contains descriptions of features that are scheduled to be released to
Applications in the coming release and it presents a detailed description of the user stories
that express how and what that should be developed. The Department itself can also add
items that express wishes about what is needed to simplify their own work and evolve the
component further on the product backlog. These demands are strictly internal and not
required by an outside customer. About 20% of the time should be dedicated to development
of these kinds of tasks.

7.1.6 Backlog grooming
The items on the Product backlog are successively assigned to available teams. The teams are
working in an Agile way i.e. divides their work in batches to fit a three-week sprint.
However, when the teams start to investigate an item, their first step is to make some
preparation. This preparation process takes place every week and aims to groom the product
backlog to narrow each item and make them as well defined as possible. Here the user story
is broken down into even smaller parts in order to better fit the format of the sprint. The
preparation meeting is called Backlog grooming and is driven by the Scrum Master and done
by the teams, PO and sometimes the MPO at a one-hour meeting and when the items have
been groomed they are put back on the Product backlog. At each Backlog grooming meeting
approximately 4 items are getting groomed. Since the current level in the Product backlog is
43 items the average waiting time before an item is groomed is 35 days.

7.1.7 Sprint planning
Before a new sprint is started, a process called sprint-planning is also executed. This sprint-
planning meeting takes place every third week and aims to choose items from the product
backlog that should be handled during the coming sprint. This meeting lasts for
approximately 2 hours. The current level of groomed items on the Product backlog is one out
of three and each sprint is handling approximately 2-10 items. Thus, the average waiting time is 24 days.

### 7.1.8 Development process

After the sprint planning the feature enters the development process. The development of a whole feature takes in average 12 weeks and includes 4 sprints, both value adding and non-value adding time. Since the sprint backlog should be empty after each sprint, the average waiting time for an item is 10.5 days on the sprint backlog before it enters the development process. The development process starts with prototyping of the part of the feature that will be developed during the coming sprint. This prototype is, together with descriptions on how to develop and implement the feature, then sent to technical councils shortened Applied Architecture Council (AAC), Implementation Architecture Council (IAC) or Department Architecture Council (MAC) depending on the organizational level of the council. These councils audit the feature design and give the team approval on continuing with development of the MR.

Even though 12 weeks time should be dedicated to development most of this time is assigned to tasks that are not adding value to the product. Firstly, the developers are only available 8 hour per day and 5 days a week. Secondly, the developers takes coffee breaks, participate in unrelated meetings etc. Thirdly, they are getting interrupted often, as they need to help colleagues and answer questions. How much time that is spent on these non-value adding tasks is not measured to day. However, a general estimation has been made based on the perception of the developers regarding how their spend their time and according to that estimation about 1/3 of the time is spent value-adding activities.

During the development process the teams are continuously, every second week, getting the opportunity to ship their code to a testing department called Reference Deployment Architecture (RDA). This instance provides several testing tracks. The one used for Component 1 (and all the other components in the Department) is Testing shipments and it delivers feedback, in the form of a report, to the teams regarding the adequateness of the feature. However, sending items with Testing shipments is not compulsory but preferable to do at least once with each feature in order to secure sufficient quality.

Component 1 uses a third party supplier, which provides the development teams with shipments of code to rely on in their development. These deliveries arrive every 6 months. The third party supplier is the supplier of the code produced by the community. The cooperation with the community is a mutual exchange between the Company and the community itself, meaning that the Company has to contribute to the community with improvements and features and the community provides code and requested features, which their components heavily relies on, to the Company. More about communities and the cooperation between communities and the Company is described in section 4.3.

### 7.1.9 Pre Release Control

Every 6 months a release of the features developed during the last months is made. Though, before the features can be released it must pass the Pre-release control (PRC), which checks that the feature meets all requirements. The PRC happens every 6 month. Hence, the average waiting time become 3 months, i.e. 90 days. In the PRC the developed features are tested towards Applications in order to find out limitations in its functions. If no limitations are found the feature is passed forward to the release state, where they are available for all Applications to use to a full extent and promote in products towards the end-users. The PRC is driven by the PM and involves activities that control how well the feature meets the
requirements of the customer. Parts of the PRC activities are executed in parallel with the
development (the documentation related to the feature) but some could only be done while
multiple features are finished due to compatibility testing and some parts of the PRC are not
beneficial to perform more often. Thus, somewhere between four days and two weeks is
dedicated for PRC activities after the feature is developed and before it is released. The part
of this time that is spent on actual value adding activities is estimated to 1/3 of the total time,
due on the same reasons as in the development process.

7.1.10 Release
If the product passes the PRC testing it is released i.e. available for Applications to utilize in
their offerings to the end-customer. If not, the team is set on troubleshooting and to fix the
code so that it has sufficient quality to later be tested and released. When the feature passes
the PRC it is classified as ready to use for the Application. This process is only a matter of
changing the status of the feature from PREL (preliminary) to PRC (PRC approved). Hence,
this process takes no longer than 10 minutes. The over all efficiency of the Component 1
value stream is calculated to 9%.

When Applications have explored the released features they provide feedback to the
components at the Department through requesting maintenance and support of the product.
This results in a maintenance track for each new version of a feature. These maintenance
activities are provided by people from development teams at the Department and are
executed in parallel with development of new features. Some maintenance is also pushed
towards the third party supplier of the community-developed code.
7.2 Current state map Component 2

7.2.1 Opportunity submission

Today an opportunity is submitted by an Application and arrives, through the ERP system, in an opportunity inbox. As for Component 1, the requirements regarding the design of the opportunity is very unrestrained and the available template is rarely used. The time that each opportunity stays in the opportunity inbox varies and is heavily depending on how keen the Application is on getting the feature implemented.

7.2.2 Opportunity analysis

All opportunities are however, sooner or later, investigated by the MPO. Here the MPO makes a first analysis of the opportunities. This analysis can have four different outcomes. The first option is that the opportunity is accepted and thereby is turned into a Main Requirement (MR). The second option appears if the opportunity is clear and understandable but is discovered to have an impact on more then one component, or interfere with the overall strategic direction of the component, the opportunity will be sent for POG/PEST study. The third scenario is that it can go back to the Application for further clarification of the demands and requirements. This clarification is made by contacting the submitter of the opportunity or another person that works at the Application. Finally, the opportunity can be rejected due to insignificance or technical inability to implement such a demand. This transformation is made by the MPO and includes reading, understanding and writing the opportunity in a MR format i.e. expressed in a more understandable way for the developers. The total time to write an MR is approximately an hour and this is made in an ad hoc manner when the MPO has time. All written MRs are put in an MR-inbox in a prioritized order. This operation is also done by the MPO.

7.2.3 Pre-screening

The next process is the pre-screening process that is currently executed in a rather unstructured way. The Pre-screening means making a rough time estimate and resource allocation of each MR and this is done by the MPO with help from system architects. This is usually made in an intermittent way when the development for the coming release is finished and the department can switch focus from the current coding to future developments of features. In general, about 30 MRs are moved from the MR-inbox every 6 months, which gives and average waiting time of 6 months based on that it is 60 MRs in the inbox right now.

During the pre-screening it might sometimes be detected that the task is not feasible from a development point of view or that the task is complex and will affect more then one component, and then it is sent back for POG/PEST study or further clarification depending on the nature of the problem. If the task is clarified enough, the pre-screened MR is put on the team backlog related to the team with most relevant competence. This backlog has not the same function as the sprint backlog in Component 1. The team backlog includes tasks suitable to be executed by that specific team, but does not state when the task should be finalized. Tasks on this backlog is impossible to time estimate due to lack of information regarding how often the items are handled.

7.2.4 Development process

The next process is the development phase, which naturally takes between a couple of weeks up to a couple of months for each feature. To estimate the total time spent on development of a feature the worst case scenario will be used i.e. 2 months. As for Component 1 the time of this process is not only assigned to value adding activities. An estimation of the time shows that one third of the total time spent in the development process is assigned to value adding
activities and two thirds are spent on non-value adding activities. Moreover, the development does not only include coding the feature. It starts with a preparation where the MR is broken down in user stories and described in more detailed. During the development phase the teams work quite differently. Some are planning their works in sprints some are working with only with post-its on a “Kanban” board. During the early stages of the development process, MR’s also has to be sent off for approval by the relevant technical council (IAC/AAC/MAC) to ensure technical assessment of the future feature.

As for Component 1 the development process receives shipment deliveries from the community through a third party supplier. However, in contrast to Component 1 the Component 2 supplier is Oracle, which is described in more detail in section 4.3. Oracle delivers shipments of code every six months. However, the team can fetch Alfa-versions more often and test them as a sideline activity to enable such a continuous development process as possible. In parallel with the development process the code is sent by Testing shipments for testing. The shipments going on the Testing shipments takes place every second week, just as for Component 1.

7.2.5 Pre Release Assessment
When all MRs related to a feature is developed the feature becomes ready for PRC. This process takes 2 months and includes similar procedures as for Component 1 plus a lot of documentation related to the feature. The PRC is set to occur every 6 month, so the average waiting time here is the same as in the previous value stream, 90 days. As for Component 1 most of the time in the PRC step is dedicated to non-value adding activities, due to waiting time, questions, meetings etc. An extraordinary observation in Component 2 is that during the PRC step, all of the development stop and everybody devote fulltime to the PRC.

7.2.6 Release
When the feature has passes PRC it is ready to be released, which as for Component 1, is only a matter of changing the status of the feature. After the product release, feedback from Applications is given through a request of maintenance of the existing commercial product. The development teams handle the maintenance, and the priority of the maintenance corresponds to the severity of the problem detected.
8 Analysis of current state maps

This section analyses the current state value stream in order to reveal flow and waste issues. The analysis is rather bound to what is visualized by the maps and the description of current state made in the previous section.

According to the Lean philosophy, improving the efficiency of the value streams means increasing the ratio between value adding time and non-value adding time. Thus, the first thing to look for when analyzing the current state maps is where the value-added and the non-value added time is spent. This as all changes that increases the time spent on value adding activities or decrease the time spent on non-value adding activities improves the flow.

One can notice that the development and PRC process are the two most time-consuming processes in both value streams. All time-consuming processes naturally include some necessary non-value adding activities. However, all the time spent on non-value adding activities are not necessary and by reviewing the activities in the process, time spent on non-value adding can be spent on value adding tasks instead. Thus, these processes represent a significant opportunity to increase the time spent on value adding time in value stream. Additionally, the biggest possibility to decrease the time spent on non-value adding activities are by minimizing the time a task spends on each backlog. As these backlogs contributes with nothing but non-value adding time.

8.1 Flow

8.1.1 Information flow

A lot of the information flow appears inside each process of the value stream. Hence, it is not extensively visualized by the VSM. However, this is natural as VSM takes a holistic approach for the whole value stream, why communication with other departments becomes more apparent then the information handled within meetings.

Component 1 has a structured way of handling meeting processes, which gives the overall information flow a more structured setup in comparison to Component 2. This can be seen in the VSM by looking at the number of processes that each component have in place. Moreover, who should participate in each meeting and why is also clear and agreed on, which favors an organized information flow.

Component 2 has a more ad hoc structure of their information flows, which makes them rather unstable. An indication of this ad hoc-ness is illustrated in the VSM by tasks that either stand still (many items on backlog) and by the amount of information that is available regarding each process. Assessable information regarding the processes in Component 2 is less extensive then the information available from Component 1. This can be exemplified by the opportunity analysis meeting in Component 1, where system architects, SPM and MPO have a close collaboration and foresees that the right information is handled and passed along too the right instance and hence is available and known by everyone involved. In Component 2, however, the communication between MPO (who handles most of the analysis on their own), SPM, PM and the teams are done in an ad hoc manner, which creates a messy information flow. This rather messy flow indicates wastes such as task switching and extra processing.

The electronic information attached to a release can sometimes be superfluous or on the contrary, unclear and insufficient. Much documentation is well needed in order to understand
and properly use the product, but some information is questioned regarding its relevance by the components, other is missing is stated by Applications. Information that does not contribute with any value can be seen as excess work. Superfluous information can also be time-consuming for the receiving party to go through. Moreover, insufficient information can be regarded as a cause for task switching for Applications.

Information in the form of feedback being delivered from sub-processes such as RDA or the auditing is necessary to maintain a good quality product. Although such feedback can be disturbing during the development phase and is thereby, for example, linked to the waste of task switching. The information and feedback coming from the technical auditing is handled during the development process. This may result in delays since the coding cannot start until the feedback has arrived. To get clearance by auditing in an early stage, before development, would decreases lead-time since it minimizes actual waiting in the development process.

The clarification arrows between opportunity analysis and Applications that appear in both Component 1 and Component 2 correspond to a total waste and depict structural errors in the opportunity handling process. This information flow increases the lead-time since it causes waiting. How much actual time the clarification takes is not known today. Though, it is known that the time for making a clarification varies a lot and the process of clarifying an opportunity is complex. Much of the complexity is caused by lack of information about who to contact and how to get in touch with a person that has the authority or right competence to contribute in the clarification. This lack of information can be assumed as an effect of the unrestrained requirements regarding the design of an opportunity.

The clarification process affects the value stream quite a bit. Especially since many of the opportunities need clarification. Sometimes an undefined opportunity slips through the up stream processes and causes for developers to have to clarify it later in the value stream. Also the fact that even a clarified opportunity can be unclear which, in the worst-case scenario, can lead to an unwanted or faulty feature being produced.

#### 8.1.2 Product flow

The product flow in Component 2 is highly intermittent, while it in Component 1 is somewhat intermittent. However, the Component 1 flow is intermittent to such an extent that it can be classified as necessary due to the Agile way of working. This since Scrum in Agile promotes 3-week sprints that are continuously prepared and followed through. Intermittent in Component 2 means that there is a work overload in the end of the development right before a release is made. Additionally, all planning starts shortly before finishing the previous release. Thus, Component 2 has a quite intermittent linear flow and Component 1 works more iterative and in line with Agile e.g. continuously have backlog grooming, sprint planning etc.

Pacemakers can be identified in both the Component 1 and Component 2 value stream. The pacemakers set the takt time for the work as a whole on the Department, after the corresponding definition presented earlier in the study. In Component 1 the Component Roadmap Meeting acts as a control for evening out the flow and sorting out tasks to sub-processes, upstream or downstream in the flow. The Component 2 pacemaker is the PRC that takes place every 6th month. This is the major planning point, which controls the entire upstream process.
Currently, items can only be released every 6th month. This makes it tricky to make the flow completely even throughout the entire work-period, as continuous flows can only be in pace if the product is delivered frequently to the customer in small batches. Besides the obvious 6-month delivery, there are a multiple of sub-processes that needs to be considered when planning and structuring the value stream. Since the sub-processes are not synchronized, they have their own time-schedule and incentives when performing tasks. They do, thereby, affect and contribute to an intermittent flow and wastes being created in the components workflow.

An example of this is the fact that the Testing shipments to the RDA department for testing are supposed to be sent every second week. However, the Agile way-of-working at the Department itself is constructed in 3-week sprints. It thereby becomes more difficult to plan and synchronize the work and adds time, in the form of waiting, to the total lead-time of the product i.e. a non-value adding activity.

Another sensitive cooperation is between the components and the communities. In Component 2 the OpenSAF community is to a large extent owned and controlled by the Company. Hence, the community shipments can be affected, controlled and planned by the Company. This can therefore be a possible factor to adjust in order to even out the flow inside the development process. Component 1 has a third party supplier that does not rely on the Company to such a great extent. Hence, this community is more segmented with multiple of companies and persons contributing to the core open source code. This makes it more difficult for Component 1 to affect the shipments from this supplier and the community as a whole.

A few bottlenecks among the value streams have been identified in the VSMs. The first one is the study process. Studies sometimes are sometimes complex and take thus long time to perform. This together with lack of resources makes the waiting time for a study long, which indicated by the heavy study backlog illustrated in the current state maps. Noticeable is that all components in the Department are using the same study pool hence the workload on this is major and the resources therefore become scarce.

The opportunity analysis in Component 2, made by the MPO solely and a bit ad hoc, also creates a bottleneck. In general, one person can never possess the same level of knowledge as a team of people where all the members have relevant competence. Thus, this process has a time-consuming nature when the task is handled by the MPO alone as he/she needs to search for information more often in order to make a proper evaluation. Moreover, the pre-screening of Component 2 MRs are portraying the characteristics of a bottleneck. Hence, the MR-inbox contains a lot of tasks that are now set to wait.

One can notice the RID is sat in a more unstructured way at Component 2 in comparison to Component 1. At Component 1 setting the RID is a part of the CRM II meeting while Component 2 defines the RID sometime between the opportunity analysis and the development process. This is due to that it is done when the SPM asks for it and this happens on an irregular basis. Though, since the SPM is more engaged throughout the value stream in Component 1 the RID can be demanded in a more regular way. However, at both Component 1 and Component 2 the RID is available for anyone through a document on the internal web site. Though, currently the RID is not updated either regularly or frequently. The time between best-case scenario and worst-case scenario, regarding the release date, is therefore often wide.
At Component 1, the Product backlog includes many items. However, as stated, this backlog contains items and several items can correspond to the same MR. Items are broken down MR’s and the big number is not to be confused with the same waiting time that is created in, for example, the Component 2 flow by the MR-inbox. However, the fact that there are a lot of items it does indicate a bottleneck in the flow. This bottleneck is more likely caused by development process itself due to lack of resources e.g. developers or time. This is applicable at both Component 1 and Component 2, as both development phases are time consuming and can only be affected by each developer producing faster, or having more developers. However, due to the nature of knowledge-work in general, and software development in particular, it is hard to affect the pace of the development phase. Creative work must take time.

Due to the insufficient time estimates in Component 2 it is impossible to calculate the ratio for value adding over non-value adding activities. However, in Component 1 the ratio is about 9%. The meaning of this number is fairly hard to interpret due to the lack of possibilities for comparison with similar value streams. Though, the ratio can be compared with the ratio in the future state map, which gives an indication of the possible effectiveness.

### 8.2 Waste

#### 8.2.1 Prolonged items on backlog

Many items on a backlog can indicate two things. Either that there is a bottleneck process after the backlog or that the item on the backlog is only a part of an MR i.e. the MR has been broken down in the previous process, resulting in more but smaller parts.

At Component 1 the items are broken down throughout the value stream. Each division favors movement of the task, as it makes it easier to understand and manage. Hence, it can hampers items to stay on a backlog for a long time. Many items on Component 1’s Product backlog are related to the same MR and the number is not as appalling as one can first think. At Component 2 on the other hand, no division or atomization of the MRs are made. Thus, the items in the MR-inbox and in the team backlog are each and every one representing. Moreover, the average waiting time for the MRs in the Component 2 MR-inbox is 180 days in contrast to 17 days in Component 1. This indicates that the process after the MR-inbox, the Pre-screening, is a bottleneck in the Component 2 value stream but not to the same extent in Component 1. Thus, waste from prolonged items on backlog is greater in Component 2 than in Component 1.

The study backlog is heavy in both the Component 1 and Component 2 value stream. This is due to lack of resource or that the MR is affecting many components. The second scenario makes the study more complex, as it needs to involve more people and instances. Therefore, these cross-component studies naturally take time. However, in either situation, the study process hampers the product flow and contributes to waste in the value stream by store the MRs on the backlog for a long time.

The presence of the clarification backlog testifies that many opportunities are too unclearly described when submitted. Since the number of items on the backlog is relatively high, in comparison to other backlog levels, one can assume that clarification is currently a common outcome as a next step after an opportunity has been analyzed. The handling of clarifications only make items stand still or move in the opposite direction of the value creation.
8.2.2 Extra features

The collaboration with the communities might cause production of extra features. This as developers at the Company has their focus on contributing to the community, which makes them develop code that is not always beneficial for the Company. However, this is an obligation in order to get access to the community.

At Component 1 extra features might also appear due to the developers and system architects putting their own items on the product backlog. This means that something is developed without an actual demand from a customer or upcoming process and is illustrated in the VSM with push arrows from the development phase to the product backlog. However, these items might be relevant to implement and can possibly enable Component 1 to develop features to meet a customer need that the customer has not yet detected it has. Since people with high technical expertise submit the items pushed to the product backlog, the probability that proper predictions regarding the direction of development of the components are made is rather high. Hence, the fact that Component 1 dedicates 20% of the developers’ time to work on these kind of tasks might be a strategic action, even though it means acting on a push and thereby creating the risk of producing extra features. The activity is regarded as Necessary but Non-Value Adding thus it does not contribute with direct value to the customer.

At Component 2 extra features can be developed due to the opportunities or MRs getting out of date. Since the items in the Component 2 MR-inbox is many and can be up to two years old the risk that these items become irrelevant increases. Developing such items would result in extra features that no one will use.

8.2.3 Extra processing

The need for clarification of opportunities causes extra processing in both Component 2 and Component 1. In the current state maps this is illustrated by the clarification arrow and the high number of tasks in the clarification inbox. The extra processing is a result of the issue of finding the right person at Applications that can clarify the opportunity. Often, the submitter is not technically competent enough, why a lot of time will be spent on finding a person that can give a more detailed description of the opportunity.

Since the MPO is exclusively responsible for the analysis of the opportunities at Component 2, a heavy workload on one single person is caused. Additionally, this cause a need for information search as the MPO not possibly could know everything necessary to make a proper analysis of every opportunity. At Component 1 a team of SPM, MPO and system architects have a meeting to make the analysis. This makes more knowledge and information available during the analysis and favors a fast but still sufficient opportunity analysis as well as a decreased need for information search.

8.2.4 Handoffs

Since the value streams involves numerous of people and processes they include a lot of handoffs. Both within and between processes handoffs are made. One can notice that the number of processes in the Component 1 value stream is higher than in the Component 2 value stream. Each process indicates planned handoffs, but necessary handoffs are included in each process as well. For example the team member are sometimes developing different parts of a feature why handoffs become unavoidable. Hence, one can assume that Component 1 are dealing with more handoffs than Component 2 due to more processes in the value stream.
In both value streams a lot of information passes the MPO. Hence, this is a key person who is involved in many handoffs related to a feature. However, a difference between Component 2 and Component 1 is that Component 1 have several forums for the handoffs being made between people, while Component 2 are working in a more ad hoc manner. This enables Component 1 to handle the handoffs in a more structured way in comparison to Component 2. Thus, it is not obvious that Component 2’s handling to handoffs is better, even if they are fewer.

Another handoff appears between the OpenSAF team and the development team at Component 2. Currently, the OpenSAF team is only coding for the community and then the development team is adjusting the general OpenSAF code to Component 2 standard and performs the testing.

The fact that Applications are submitting unclear opportunities also causes handoffs. The handoff as such is unavoidable but the bad quality of the description related to the opportunity makes this handoff extensive, as it has to go back and fourth between people at the departments several times.

### 8.2.5 Task switching

The efforts regarding clarifications causes room for task switching. This as the person(s) involved in the analysis must stop the processing and start searching for a person who can clarify the opportunity instead. As mentioned, this is an issue present in both value streams and that are illustrated in the current state maps by the clarification arrow and the clarification backlog.

Both in Component 2 and Component 1, feedback that comes from an audit or test causes task switching. This since the feedback does not arrive instantly after the code is sent for testing or auditing. Thus, the developer will start working on a new item while waiting for feedback. When the feedback later on arrives, the developer must switch what he/she is doing in order to change the item in line with the feedback.

Moreover, the maintenance arrow from Applications indicates a possible task switching in the development process. When the maintenance is urgent extra resources might be needed. This might force the development teams to switch from their ordinary tasks and help out with maintenance instead. However, this will only be the case when the maintenance is very urgent and thus highly prioritized.

Approximately only one third of the total time dedicated to development of the feature is assigned to Value Adding activities. Which is a relatively ordinary approximation in other industries as well. The rest is Non-Value Adding time, which often is a result of that the developer is getting disturbed. The developers are helping each other a lot, which means that they sometimes need to stop coding (Adding Value to the feature) and help a colleague (Non-Value Adding time). All these disturbing activities results in task switching’s.

Finally, the value streams include several processes in form of meetings, each meeting will of course cause a task switching. The Component 1 value stream includes more meetings then Component 2 why this value stream includes more tasks switching of this kind.

### 8.2.6 Delays

Room for delays are present in both value streams. An item can be delayed due to that other more important tasks are submitted, tasks which get a higher priority on the backlog. This
complies with the Agile philosophy regarding how to best handle an item, as Agile advocates the wish of always handling the most important item first. Moreover, the need for clarification of opportunities stops the opportunity from continuing through the value stream. The time that is takes to clarify an opportunity varies a lot. Thus, it becomes difficult in both value streams to include the time needed for clarification in the estimation of a release date for an opportunity.

Currently, both Component 2 and Component 1 are dividing the development process in 3-week sprints. Though, the shipments that are sending the code for testing are going every second week. Hence, often code that has been developed during one sprint is finished one or two weeks ahead the next Testing shipment. If the shipments and sprints were in sync, no waiting would appear and feedback would be provided faster. However, today this can cause delays.

Delays due to inadequate time estimates are an issue in both the Component 2 and the Component 1 value stream. Today, time estimates are not yet fully implemented and are thereby being done in a rough and ad hoc manor. This results in insufficient time plans that are difficult to meet. The timeline in the Component 2 current state map is not even possible to completely fill in. Today, no measurements of the value streams are made in either Component 1 or Component 2 and in Component 2 key persons in the processes are not even able to make estimations of the time spent on some of the tasks.

The shipment and feedback from auditing is made in the development phase in both value streams. The amount of changes that need to be made after feedback is difficult to estimate and might thus cause delays. Moreover, the time it takes from shipment to the audit or testing department until feedback arrives depends on the occupancy of the Department at that moment. These departments are handling testing and auditing for department other than the Department as well and hence their response time varies a lot. Though, the component cannot release a feature before they have gotten feedback from testing and auditing, why these sub-processes can cause delays. This creates a risk for devastating delays for both components as the time between feedback from auditing and release date is smaller and hence necessary changes might not be possible to finalize in time.

Finally, the collaboration with the communities can cause delays for both components. When a shipment arrives from the communities the existing features must be adjusted to the new shipment, which takes time and might result in unexpected delays.

8.2.7 Defects
The maintenance tracks in the VSM indicate the presence of defects. Though, the continuous testing by shipments to RDA indicated that both value streams have a structured way of handling and minimizing defects. One should notice that in software development in general, and in Agile software development in particular, defects are unavoidable. This is due to that it is more important to get the product out to the customer in a sufficient quality on a short matter of time than releasing it in a perfect condition later on. The changing demand among software customers makes the strive for fast releases natural and hence total elimination of defects impossible.
9 Discussion and improvement suggestions

This section highlights issues revealed in the previous sections as well as issues that do not fit into the format of VSM. The outcome is a proposal of improvements to the current practice that would improve the current value stream. The improvements are grouped in five categories depending on what or how they improve the value streams.

According to the Lean philosophy the current value streams will best be improved by focusing on improvements that fosters flow efficiency. As presented in the theoretical framework this is preferably done by changing the processes in such a way that the entire value stream works more in line with the Lean pillars of JIT and Jidoka. The following section discuss and presents solutions that aim to improve the value streams, by eliminating the wastes and things that hampers the flow, this is done by drawing on the mindset of JIT and Jidoka.

Issues related to management are not visible in the value stream maps. However, management is heavily affecting the value streams, as the managers are ultimately responsible for the overall look of the value streams. Hence, the usage of management in a correct way is essential to successfully implement Lean and facilitate the below outlined solutions.

9.1 Communication

The ability to deliver Just-In-Time depends heavily on performance of communication. The same appears for Jidoka. If the processes can communicate perfectly the flow will automatically become more efficient. However, both Component 1 and Component 2 value streams lack in communication, why this is an area with room for significant improvements.

The issue related to the lack of holistic understanding for the process is to a big extent created by the deficiencies in the communication. The Agile way-of-working at the Department contributes with a well-established and functioning horizontal communication. The developers are satisfied with the teamwork and collaboration. However, Agile prevents an understanding for the value stream as a whole. The close collaboration between the team members also has its drawbacks. This causes task switching, as a big part of the developers’ time is spent on helping other team members with their tasks. Enabling the developers to work undisturbed would increase the amount of time spent on value adding activities and thus improve the flow efficiency.

Attention should also be directed to improvements regarding the communication between the Department and other instances as well as between the development teams and other people involved in the process. This would foster both a better understanding for the overall value stream as well as better the communication with Applications. A better communication between Applications and the Department is linked to many detected issues in the VSMs for example the clarification part of the process.

Clarifications are currently not handled in a smooth way and are heavily depending on personal relationships between people working at the Department and people working at Applications. This creates vulnerable communication channels and causes delays and extra processing when no personal relationship is available.

One reason that the communication between Applications and the Department is deficient is due to lack of understanding for each other. For example, Applications plans for the
implementation of a release based on the information in the RID related to the item. However, the RID is finalized relatively late in the value stream, which hampers Applications from taking releases as early as possible. Another indicator in the lack or understanding is the nature of the documentation sent with each shipment and release. According to Applications the information attach to the code is difficult to understand or not what they are looking for. Information will only be valuable if it is properly understood and useful for the receiver. Additionally, the necessary documentation required to send the code for auditing is perceived as too extensive by the developers. Thus, the current condition of the documentation related to releases, testing and auditing sometimes becomes nothing else then a time consuming waste activity as no one really uses it.

Today, a lot of extra processing appears in both processes due to that it is difficult and takes time to find proper information fast. One reason for this is the lack of clearness regarding the roles and responsibilities. This forces people to search a long time after the right person to ask and thus time is spent on non-value adding activities. Currently, roles and responsibilities are not communicated to the people in the process in an active way. Though, they can be found in slide packs on the intranet. However, the overall awareness of what people are meant to do is very low among the people in the value stream.

9.1.1 Improvements that favor communication

- Streamline communication between people with technical expertise by establishing a Communication Forum between Applications and the Department including relevant people from both instances. This Forum should be owned and driven by several Application Drivers. This would make the communication channels less personal dependent and vulnerable.

- Update RID frequently and continuously throughout the value stream. This would further enable the communication between the departments

- Management should put focus in communicating what and why a task should be done, standard be followed etc. to people involved in the value stream. This to facilitate a total implementation on an operational level.

- Evaluate and adjust the documentation that is sent with releases and shipments to fit the needs of the receiver e.g. adjust the content, level of details and the wording. This to avoid misunderstandings or an insufficient information flow.

- Create a call group among the software developers were one person per day is responsible for answering questions and support colleagues while the others should not be disturbed if not necessary. This is done to streamline the communication between the developers.

9.2 Standardization

When aiming to increase the flow efficiency, the flow must to some extent be standardized. This is necessary to ensure that people involved in the value stream have the same understanding for the value stream and its different processes (Modig and Åhlström, 2013). Hence, standardization favors Jidoka, as when everyone understands and knows how the value stream should look like they can more easily detect deviations and take proper
adjustment actions. Moreover, standardization increases the ability to control and predict the performance of the value stream, which favors the ability to deliver JIT.

However, many of the complications in the Department can be derived to a lack of structure in the value streams. Additionally, a big part of the Lean approach in creating a smoother flow and eliminating waste is the standardization of tasks and processes. Due to the nature of software development some parts of the value stream is not beneficial to standardize and even though an intermittent flow is impossible to avoid, the variation can be minimized or reduced with structuring, planning and good time estimates. With all this in mind the following solutions will not involve standardizing the software development itself since this might hamper necessary innovation.

If the handoff is made ad hoc and vital information can easily be forgotten. Hence, wastes such as handoffs become bigger with lack of standardization. This is visible in the opportunity handling and clarification. Furthermore, extra processing is, for example, created when the lack of structure regarding the opportunity analysis handling in Component 2 causes for the MPO to obtain information on their own. The lack of structure in the processes and meetings contributes to a waste when people dedicate their time to tasks they are not able to contribute to. This mostly occur in Component 2, as Component 1 has managed to set a structure that is empowers the employees and which they understand as well as manages the product and information flow in a smoother way.

The absence of structure in Component 2 is also indicated by the few number of processes included in the value stream. At first sight, this might give the impression of a Lean and well-structured value stream. Though, in reality it means that each process includes many tasks, which naturally are executed in a more or less ad hoc manner. This since the more tasks a process contains the more complex it becomes to structure and standardize. This inability to structure hampers the risk management of the value stream. If a value stream is divided into many small processes the ability to manage tasks that might have high risk impact, such as feedback from auditing and testing, higher up in the value stream will be favored. This in turn, prevents delays, as the time slot available to make corrective actions is larger then if the feedback comes close to a release.

Standardizing by dividing the current processes into smaller is one way to improve the flow another is to standardize the way that different tasks are executed by drawing on best practice. Today, both Component 2 and Component 1 complain on the heavy, but necessary, documentation in connection to the development and PRC of a feature. Currently, the best way of doing this documentation has not been investigated and thus the components handles it in different ways. At Component 2 a lot of documentation is pushed a head during the development why the amount of documentation during the PRC phase becomes massive. By evaluating how the necessary documentation could be divided into smaller blocks that can be handled in parallel to the development process, the flow could be smoothened. Moreover, this would make the perception of the effort put towards documentation less painful and boring. Hence, the possibility to get the developers to do some of the documentation on their own increases, which would decrease the number of handoffs and relieve the MPO.

To eliminate bottlenecks through standardization of processes and structure within the processes to make them less time consuming is a possible solution. This is in manufacturing done to reduce the queues, inventory, before each process. In manufacturing the inventories accumulate costs, this is not the case in software development. To reduce the inventory as
such is not the aim, it is rather to handle the orders quickly to produce JIT and to not create waste in the shape of extra features by the opportunity becoming dated. This becomes an issue at Component 2 as they are not handling their opportunities or MR on a regular basis. Thus, a lot of their items on their backlogs become old as they do not perform any frequent analysis of them. Having many items on the backlog also hampers making a proper analysis, as it is more difficult to evaluate and prioritize a backlog with many items.

Adding a structure will facilitate making quality improvements, within a reasonable amount of time, to handle the tasks ordered by the customer faster and in a more precise manor. However, the quality aspect is already handled in the Company by the continuous integration initiative; and defects are not perceived as a big problem today. Hence, the solutions presented in this study will not handle this specific area in more detail.

Before introducing a new structure, careful evaluation must be done regarding the applicability and suitability of the structure as such. For example, some critique can be made to the software development at the Company when implementing the Agile way of working. They adopted some parts of the philosophy, but some vital parts such as continuous, smaller, deliveries are not set in place. This should not be the case with the Lean implementation why much thought is given to the specific context before determining the solutions. Also, the standardization may vary between the components due to their different preconditions.

Some standards are already set in place, such as a standard template for the opportunity submission. This template is, however, not used, and a standard can first be called “a standardization” when adopted on an operational level. A standard template will reduce the risk of delays and need for extra processing as it facilitates the clarification of opportunities that needs clarification.

9.2.1 Improvements that favor standardization

- Define reasonable process steps for a components entire value stream. This by first determining best practice solutions by benchmarking other components. This would foster a basis for knowledge sharing and enable measuring and benchmarking throughout the Department.

- Make the CRM the pacemaker to create a pull flow and ensure that not to many items are in the process. This will increase the ability to control the work in progress and minimize the risk that items get stuck in the value stream, become out of date, but even though are developed.

- Structure the meeting with aims, decision points and participants. This would ensure value being added in each process and streamline the actual processes.

- Structure the handoffs between the teams, especially between Component 2 development team and OpenSAF. To avoid tasks not being performed properly and causing a extended lead-time.

- General philosophies such as Lean and Agile should be adjusted to the context of the Department. Be pragmatic and evaluate the applicability and adjust to the specific conditions at the Department.
• MR’s that have been studied in POG/PEST should not go directly to the MR-inbox. This to enable a second round of evaluating before entering the value stream.

• Evaluate the necessity of the documents being produced and sent with each release. Standardizing necessary documentation by clustering it in order to enable making it continuously during the development to eliminate the heavy documentation in the PRC step. To streamline documentation would enable better resource allocation and increase the actual value adding time.

• Make standard template compulsory for opportunity submission. This should be easily obtained by Applications and the Department, understandable and complete. If the form is incomplete it should not be possible to submit to the opportunity inbox. To avoid clarifications and time delays in handling the opportunities.

• Standardize the information that is being sent with each shipment. It should be understandable from Applications point of view. This would create a clear view of what is included in the shipments hence make it easier for Applications to build upon.

9.3 Synchronization

JIT is all about timing. Therefore, the fit between each process and instance in the value stream affects the ability to produce in line with JIT. The better the synchronization between the processes in the value stream, its suppliers and customer are the better the ability to deliver JIT will be. Synchronization also favors Jidoka, as it increase the transparency between the instances and avoids the risk that issues are getting ignored due to lack of ownership and understanding.

Currently, the synchronization and thus transparency between other departments/instances and Component 1 and Component 2 is limited. However, each component has several instances, which they have to interact with, communication wise, but also in actual shipments of the product. Hence, synchronization between the Department and the other departments and suppliers are vital to manage in order to create a smooth value stream and prevent waiting time and other wastes. A better synchronization can create a more holistic understanding for the components and the processes connected to their value stream. Additionally, it will foster the communication with Applications, which as earlier mentioned, causes massive problems in the current value streams.

The planning should not only involve synchronization of the internal processes but should also be made according to interest of sub-processes and Applications. The RID is a part of establishing a better synchronization with Applications. As earlier mentioned Applications have needs for receiving information regarding when features will be available for release. The refinement of the RID or to in some other way confirm the status of the opportunity continuously is essential, as well as making the opportunities traceable throughout the entire value stream. Today, the IDs are sometimes changed during the progress through the value stream as the MRs are broken down. This makes it difficult for Applications to follow the feature. Another planning issue is the synchronization and information regarding tasks that have been sent for study. The study process has a heavy workload and the time it takes to study a MR varies a lot. Making a perfect prediction of when the MR will be ready will be
difficult. Though, the study pool could communicate the progress of the MR continuously
and thus facilitate the preparation and planning regarding the MR at the Department.

The sub-processes, if not synchronized, have their own time-schedule and incentives when
performing tasks. Thereby, they do affect and contribute to an intermittent flow and wastes
being created in the components workflow. This could to some extent be eliminated by
coordination, communication and time-management. An unsynchronized in-flow from sub-
processes lead to task switching if the incoming task is urgent or ranked higher than ongoing
tasks.

A telling example of this mis-synchronization between departments is detected in the
shipments. The Agile way-of-working consists of three-week sprints, but the Testing
shipment is made every second week. This causes for a more complex planning and further
tells about the lack of evaluation regarding the implementation of philosophies in the
Company.

The reduction of maintenance tracks, to be able to avoid both task switching, resources being
occupied as well as not being able to put out new features by still coding on the old ones, will
in the future be necessary. Hence, to communicate the need for this to instances that have the
possibility to help in the reduction, such as the departments where the SPMs are located and
Applications, is vital. Fewer maintenance tracks would ensure a good synchronization within
the Department and within the teams located in geographically diverse places, even in the
future.

9.3.1 Improvements that favor synchronization

- Evaluate the possibility to make Testing shipments in sufficient quality for
Applications to use. This is a way of working towards small batch deliveries and
would synchronize well with both the Lean and Agile philosophy.

- Aim towards 1:1-mapping of the opportunities were ID:s are traceable between
Applications backlog and a backlog owned by the component. This would create an
opportunity for Applications to continuously trace an opportunity and favor planning
of the implementation of the product.

- Update RID frequently and continuously throughout the value stream. This would
further enable the synchronization between the departments.

- Synchronize tasks and time-management with study pool as much as possible. To
avoid stoppages or unaccounted for tasks entering the value stream, i.e. risk
management.

- Synchronize the frequency of Testing shipments with length of the sprints. This
would make planning easier as well as facilitate a reduction of the lead-time due to
the pooling of processes. Furthermore the two-week sprints are also contributing to
smaller batch deliveries of a higher quality. Also the shorter sprints favor the handling
of highly prioritized tasks not being entered in the middle of a sprint causing for
delays or work overload on the software developers.
• Clearly communicate the need to reduce maintenance tasks to SPM and Applications with the aim of, in the future, cutting the oldest maintenance tracks.

### 9.4 Measurements

In order to improve the flow efficiency of any process, elimination, reduction or management of variation is essential. However, managing variation is only possible if the variation is known. Thus, measuring the current situation is the key to ensure taking appropriate actions to compensate for the variation.

The overall variation within Component 2 and Component 1 is high due to the uniqueness of the features that are developed. Today, no measurement of the process time is made and KPIs related to the value stream is not in place. This makes it impossible to control and evaluate value streams and take proper actions to correct unwanted deviations. Developing and implementing such measures and setting KPIs for the value stream, is a task that should be driven by management. However, the developers should preferably be involved in the KPI development as well. The opportunity to be a part of affecting how and what that should be measured will increase their understanding for the process and the goal that they work towards. This engagement will to some extent empower the developers, which in general is highly correlated to the motivation among employees. Additionally, this understanding, empowerment and engagement will decrease the resistance among the software developers.

Currently, the components are failing with fully implementing new initiatives such as ways-of-working, teams, meetings etc. The slide packs and proposals are many, but most often they never become successfully realized and implemented. This makes the improvement initiatives counterproductive, as they rather cause confusion among the employees than improve the current value stream. The inability to implement changes is caused by the absence of commitment among management but also absence of measurements related to the change. According to Lean, the management act as ambassadors for the Company and should align their employees with the vision of the Company, which is essential to succeed with any improvement. To enable evaluation of the performance among leadership, managers must be measured according to how much time they spend on value creating activities. In turn, this requires an evaluation of which activities that actually creates value. Though, implementing changes is always complicated, but if the aim and goal with the change is clearly communicated the realization will be easier. Hence, defining the goal and relevant measurements that can be used to evaluate the outcome of the change will foster a successful implementation.

At Component 2 both managers and employees are aware of the obvious issues related to the heavy MR-inbox, absence of Pre-screening etc. However, no pro-active actions are taken in order to solve the problems. Here selective measure is probably necessary to eliminate the heavy backlog that affects all the other steps in the value stream negatively. When making this effort, measures are once again the key to success. With well-defined measurements a better level of maximum items on the backlog can be set and thus resources can be assigned and correction actions be taken in a proper way.

#### 9.4.1 Improvements that favor control and measurement

• Develop KPI’s that favors stability and control of the value stream. The KPI’s should correspond to the strategy of the Department and available resources.
• Develop some of the KPIs together with the developers. When they get the opportunity to influence the business and set goals towards which they should work themselves, their motivation will increase.

9.5 Resource allocation

When smoothing out a flow to improve the flow efficiency, access to the right amount and type of resources is essential. Allocating resources is also one of the things that Modig and Åhlström (2013) highlights as an activity that favors low efficiency. This as it increases the capacity and reduces the cycle time.

The resource issue at the Department is only regarding a temporary lack of personnel. Today they cannot manage to spend their entire budget regarding vacancies in their work force. Considering the previous solutions presented, especially regarding allocating people with technical competence higher up in the value stream, a need for more people with the right competence becomes crucial. This puts a higher pressure on management who is responsible for resource allocating and acquiring.

As noticed in the analysis, a heavy backlog can be an indicator of lack in resources. To minimize the prolonged items on the backlog a good effort could be to allocate resources to the bottleneck processes. This as it would increase the capacity and thus lessen the waiting time for each item on the backlog. Initially, the usage of selective measures to handle critical inefficiencies in the flow will take resources from other processes.

A part of the resource allocation is also handling the repercussions of the handling itself. If a resource is allocated to perform a certain task, some demands from the management side of the development are needed. This is necessary to tame the natural resistance among the software developers that appears when they are faced with tasks they are not used to or have a bad attitude towards.

When allocating resources in form of manpower, an important part is to allocate the right competence to the correct process in the value stream. Placing wrong people in important places can hinder the flow. It is, for example, important to have people with good managing or planning skills in the roles of MPO, PO and PM as these are involved at many places in the values stream. By doing this in a proper way issues related to lack of technical knowledge upstream can also be eliminated.

Today, most of the employees are experts in their own component. This makes it difficult to get people to the study teams regarding cross-componential studies, as you need people form all components and thus need much manpower. Moreover, there is a resistance among the employees to perform studies. Many of the developers experience the studies as boring. However, the ability to learn new things is mentioned as highly motivating. Participating in a cross-componential study team would give them the opportunity to acquire new knowledge and information, which could be stated as an incentive to make the software developers participate in studies.
9.5.1 *Improvements that will better the resource allocation*

- People with technical competence that are interested in participating earlier in the value stream should get that opportunity. However, this resource allocation must be made with regards to the trade-off off taking time from actual development.

- Allocate extra resources, as a selective measure, to bottleneck processes. This as problems of an urgent character cannot wait for a systematic change of the entire value stream.

- Management needs to have the ability to put demands on the software developers on what they should produce or where they should be allocated. This to favor the overall efficiency, not only the efficiency within a process.

- Management should recruit more people to support the development process, and to enable allocating technical knowledge to the processes upstream.

- Evaluate the match between necessary qualities for the role as MPO, PM, PO and SM and the qualities of the person that currently has this position. To coach these leading positions in an adequate way.

- Use and promote the study pool as a place for knowledge sharing. Make it compulsory to participate in a study team on a regular basis. This to ensure strengthening the software developers’ competences and work towards a more cross-functional team approach to the highest possible extent.
10 Value Stream Mapping of future state

By drawing on solutions presented in section 9 this section presents as an improved value stream in form of a future state map (Appendix C). This section also presents comments of how the solutions are visualized in the map and points out how they solve the problems of the Company.

10.1 Future state map of the components

The future state value stream represents an improved standardized value stream that contributes to a reduction of wastes and a better flow throughout the value stream. The value stream is divided into structured processes to help carry an opportunity to a feature and to help eliminate the problems the Company is currently suffering from. Moreover, some new sub-processes have been introduced that will facilitate the flow by favoring the communication between Applications and the components. A Leaner value stream with feasible solutions implemented and a holistic picture in mind would result in an efficiency ratio of 18%. This is an improvement of about 9 percentage points compared to the current Component 1 value stream and probably an even bigger one compared to the current state of Component 2.

This standardized approach of the value streams in the Department will contribute with several benefits. It provides a better basis for outlining roles and responsibilities in each process is possible, a better understanding for the process as a whole and an improved ability to measure and control the performance of the value stream. However, this study is not expressing how the change from current state to future state should be realized. In general, management should be responsible for driving this kind of radical change. Thus, their understanding and approach to the new structure is essential in order to successfully realize it.

Furthermore, a basic assumption when applying Lean in this Agile context is that no task should pass the gatekeeper if not intended to be executed in the near future. This is done to avoid prolonged items on backlog and developing features not being wanted by the customer when exiting the value stream. To have a gatekeeper also ensures the Value Stream being changed from push to pull. The upcoming process needs to ask for tasks to perform, hence the pull flow is created. This is one of the most important changes towards a leaner value stream.

10.1.1 Communication Forum

An add-on to the future state map is the introduction of the new C&A (Components and Applications) Communication Forum. The need for a structured way of handling a continuous communication to shorten the waiting-time and misunderstandings has been highlighted throughout the entire study. The vision is that all communication or at least the coordination of the communication should be handled in this new forum. The people allocated to act as drivers of this forum, need to be good at communicating and actively work with obtaining detailed information about competences, responsibilities and contact information of each employee. This is done to enable a quick and streamlined communication and to provide confirmation of information handling to make the handoffs more efficient and avoid the waste of extra processing. The forum should not be a standardized part of the value stream but should be available for all processes and people involved in the value stream. The win of this change will firstly be the have people who can facilitate the holistic understanding and communication between Applications and the components at the Department. Moreover, it will bring a time-gain by the reduced waiting-times and thus delays due to lack of information.
10.1.2 ERP system

In order to favor the communication and synchronization between Applications and the Department some fixed points for interaction between the instances need to be introduced. The planning and ability to implement new shipments and releases among Applications are heavily dependent on the extensiveness of the information regarding when the feature will be available. Hence, in the future confirmation regarding the progress of the opportunity throughout the value stream should be more extensive in the future. The first confirmation should be sent when the opportunity has passed the first process, CRM I, in order to communicate that the opportunity is now under processing. Moreover, since the update and correctness of the RID is of importance to Applications the RID should preferably be updated more frequently in the future. To enable a more correct and frequent update of the RID someone must feel ownership of performing this task. The most suitable person for doing this is the PM, who is currently responsible for making the first RID. Another change that will favor the communication and synchronization with Applications is the improvement of traceable ID’s on each opportunity. In the future each opportunity should be linked to their ID that it is given during the submission, even when it is broken down to items and user stories. This as it is of importance to favor the planning at Applications.

The confirmation of the progression of the opportunity can already today, to some extent, be done in the ERP system. A tool already implemented in both departments and used today for opportunity submission etc. However, currently the ERP system is used insufficiently to facilitate the attributes regarding traceability and planning of the opportunities. By expanding the information available in the ERP system the instance can resembles what in manufacturing is labeled as a production control. Such instances aims to synchronize the planning by using customer information as the basis for process schedules. In this case, the ERP system could be developed to get sort of the same, but simplified and reversed function. This would facilitate the planning of Applications by delivering information from the Component regarding the progression feature. This would in turn increase the ability of Applications to take new releases.

10.1.3 Opportunity submission

No changes have been made regarding where the opportunities are submitted, as submission through the ERP system is currently good. Though, some changes have been made regarding the requirements of information that need to be filled in, in order to submit an opportunity. Currently, a standardized template for opportunity submission is set in place, but today it is to tedious to use why it needs to be redefined to be easily accessed and relevant. An improved standard template could further reduce, or at best eliminate, the need for clarification. This would result in a reduced waiting-time since tasks submitted into the value stream, that later has to travel all the way back to the customer for clarification, are minimized. It will hence decrease the waste of extra processing and delays and at best totally eliminate the current clarification backlog.

10.1.4 CRM I

The first process in the future value stream is the Component Roadmap Meeting (CRM). This meeting has replaced the Opportunity Analysis Meeting and is divided into two parts, CRM I and CRM II. This favors the iterative cycles in the value stream, which enables a minimization of the number of processes. The CRM meeting should act as pacemaker. Much like in the Component 1 current state map, this pacemaker controls processes downstream and upstream. This seems to be both functional and feasible in the current environment, why it is applied straight of to the future state map. The participants in the CRM should be at least MPO and SPM. However, system architects and PM should be invited when the items under
consideration requires their expertise in order to get properly analyzed. Currently, due to lack of time, PM and system architects have not always participated as planned in the opportunity analysis meeting or CRM. In spite of this, the quality of the meeting has turned out to be sufficient in most cases. Though, when a complex opportunity arrives, more extensive knowledge will be required why system architects and PM should still be prepared to participate. By having this pragmatic attitude to the CRM, required knowledge and skills would always be available at the meeting, which will make it as efficient as possible.

The scheduling of the CRM is one hour every second week. This has turned out to be a good frequency and time according to Component 1 who already has this meeting in place. Every part of the meeting handles tasks and MR’s in different stages of the value stream. The first part of the CRM, called CRM I is the previous Opportunity Analysis and acts as a gatekeeper for what tasks might enter the value stream. Hence, a first analysis of the opportunities is done here.

To avoid task switching and reduce waiting time on backlog, opportunities that passes the CRM I are immediately turned into prioritized MRs. Thus, the non-prioritized MR-backlog can be eliminated, as the priority is made in conjunction with the analysis.

At the CRM I meeting the opportunities can now take 4 alternative ways. They can be turned into a prioritized MR and put on the prioritized MR-backlog, sent for study, pushed back to the opportunity inbox to be handled or accepted in the future, or rejected. The alternative to send the opportunity back to Application for clarification is no longer necessary. This is due to the extensive opportunity submission template that eliminates the need for clarification. However, if some clarification would be required the opportunity should be sent to the C&A Communication Forum who should provide the necessary information or contact details to a proper person at Application. In this way a lot of extra processing can be avoided as the total time spent on finding information will decrease.

10.1.5 Study
Currently, the MRs that are sent from the PEST/POG study are submitted to the MR-backlog. In the future state map these MRs are sent to the Opportunity inbox instead. This will secure that they analyzed again and ensures that they still fulfills the requirements to pass on to the pre-screening. In this way the waste of extra features can be minimized. Moreover, the number of people available in the study pool is 17 in the future state map instead of 12 as in the current state maps. This is the number of resources that according to the study manager is necessary to have in place to ensure the ability of finalizing the studies within the demanded 7 weeks time frame. These resources become available due to an increased efficiency of the development process and by making it compulsory for the developers to participate in the study pool on a regular basis.

10.1.6 Pre-screening
The Pre-screening is set to occur every other week to meet the takt of the CRM. It is preferred that the meetings are time consuming rather then frequent. This to avoid task switching in the participants daily work. The pre-screening should be made by the MPO and PO or if necessary the developers should be involved to ensure that enough technical knowledge is available. Here the items are roughly analyzed regarding its time and resource requirements. Another aim in the future state is that all items in need of a study should be submitted to the study pool before the CRM II meeting. This requires technical competence in the pre-screening. However, it would favor the reduction of the non-value adding time further down the value stream and can thus be seen as a beneficial trade off.
10.1.7 CRM II
In the second part of the CRM the MRs should be checked for cross-componential alignment by being send to necessary forums. They should also be fit into a coming release and put on the Product backlog. A new compulsory step in the process is to update the information regarding the feature and make it available for Applications. This could preferably be done by updating a new “status field” in the ERP system or if that is not feasible by updating the RID document. Since this is an iterative meeting the CRM II takes place with the same frequency and for the same time as CRM I e.g. one hour every second week.

10.1.8 Sprint planning
Another standardized improvement, which favors synchronization, is to work in two-week sprints during the development process instead of three-week sprints. This would better adjust the Agile philosophy into the current context, as working in two-week sprints would enable sending shipments with each Testing shipment and thus facilitate the planning. Though, a present problem with adopting Agile at the Department is the fixed 6-month releases. This is against the Agile philosophy, which encourage organizations to work in close contact with the customer and make small but frequent deliveries. Additionally, the software developers view on the quality of the shipments could be changed since it is currently not regarded as important when only being submitted for shipment in the middle of a sprint. In other words this will, in the long run, eliminate the current issue of no sufficient and continuous quality control.

The Backlog grooming meeting at Component 1 is a great way to divide the tasks early in the process in order to facilitate the development of the feature later on. Additionally, the sprint-planning meeting is a great forum for preparation of the items before they are entered into the development phase. Thus, these two meetings secure that the tasks that are on the team backlog can be coded right away e.g. no extra processing needed.

The current relation between Backlog grooming (every week) and the sprint planning (every third week) is set in place since new tasks who are prioritized higher keeps entering the value stream in the development phase. Hence, these items need to be handled and could be included in a current sprint why the grooming meeting is taking place more often then the sprint planning meeting. However, when the development is made in two-week sprints the planning would become more frequent hence prioritized tasks submitted in an ongoing sprint could wait to be handled until the next one. The introduction of two-week sprints would enable a beneficial pooling of the backlog grooming and sprint planning. This is preferable as the same people are involved in these meetings today and by pooling them the number of task switching and the total time spent in meetings can decrease. When the backlog grooming operations are added to the sprint planning meeting the length of the meeting have to be extended. However, since everyone is already focused on the meeting the total time can be limited with half an hour. Hence, the sprint planning takes place every second week and lasts for 2.5 hours.

Today, a sprint review is taking place after each sprint. This is a meeting where the team, PM, PO and MPO meet to discuss what was good and what can be improved based on the leanings from the finalized sprint. When the sprints are shortened to two weeks a proposal is to have sprint reviews every second sprint instead of after every sprint. This, in order to minimize the number of meetings and hence decrease the task switching and increase the time that developers are spending on coding.
10.1.9 Development process

Today, the processes before the development process is not extensive enough to ensure that all necessary information regarding the MR is available when the development starts. This forces developers to send the MR back for clarification late in the value stream, which causes delays as well as extra processing. However, the new structure should prevent that deficient MRs are passing through the value stream, as the developers should get involve earlier in the process and thus detect the MRs that are unclear or lack in details earlier. In the future state map the only thing passing from the development phase should be questions to Applications. This could be regarding, for example, technical issues and specifications. But no clarifications of the task itself, rejects or studies should be detected at this point.

A push-arrow from the development process to the product backlog is also present in the future state. It illustrates submission of items submitted based on self generated demands of the developers. This ensures that the Department not looses focus from constantly improving their ways of working. This is in line with the Lean principle “continuous improvements” as well as favors incremental innovations. Component 1 are already encouraging submission of self-generated items and their goal of dedicating 20% of their time to these items seems to be a proper way of stimulate development of these items.

The idea of introducing two-week sprints is good for synchronizing the development team with the RDA. Additionally, making the sprint planning more frequent is also a part of introducing the two-week sprint. To actually make the sprints more efficient a reduction of the task switching need to be made. Involving people with technical competence higher upstream would result in a task switching but solve the current problem with lack of technical knowledge upstream. Though, the developers currently state that they spend as much time as 30% in helping other developers. This is good since to favor knowledge sharing among the developers, but to reduce the number of times a developer has to change tasks, the task of helping others is viewed as a call duty. If the information is to specific it could pass through the person with the right competence, but in case of universal questions one should contact the software developer who is on call.

The call group will make the development more efficient and together with the reduction of the number of meetings the software developers have to attend by performing backlog grooming and sprint planning in one session would make it possible to allocate them elsewhere. Resources from the development phase can hence be allocated to perform studies to reduce the waiting-time of the studies and get a better flow throughout the value stream. In order to be able to allocate resources for the study pool the total time spent on developing a feature will be the same as for current state of Component 1 e.g. 3 months.

Another synchronization within the development is coming from Component 2. Currently, the OpenSAF team does not contribute to the development of new feature in the extent that they can. Moreover, they cause for the development team in Component 2 to perform their testing instead of letting them handle their own tasks. This causes for a disturbance and a planning point in the Component 2 feature development team. To avoid this OpenSAF should be separated from the process and regarded as a third party supplier of open source code. This requires for the code to be of sufficient quality, tested and relevant to the Company. In this way, the handoff between the community and the Company will be much more efficient.

One can notice that no visible changes related to the auditing sub-process are present. It is still done at the same stage in the value stream as before. This is due to that auditing cannot
be performed before a prototype is set in place and this could happen first in the development phase. Thus, it is an operation that unfortunately has to be done in the late phase of the value stream. However, the level of documentation that is necessary to send with each auditing shipment can preferably be evaluated. This would facilitated the prototyping and minimize the time spent on documentation related to a design that in the end is not approved by the auditing instance. Hence, avoiding this documentation would decrease waste related to of extra features.

**10.1.10 PRC**

In the improved value stream the PRC is, to the highest possible extent, done in parallel with the development process. Some documentation and approvals need to be passed in the end of the development. This is due to that PRC need to be done on a 6 months basis. Hence, the waiting time before the PRC is the same in the future state map as in the current states map. However, to portion the PRC in parallel with the development would result in an improvement regarding the total lead-time. An evaluation of the documentation and steps in the PRC might further streamline documentation. This could reduce the actual time spent on the PRC related tasks and will favor a continuous flow. The same is for the testing related to the specific feature. The extent to which this testing is made in conjunction to the development the shorter the final PRC process will be. Today, Component 1 is spending way less time on their PRC why their time spent in the PRC phase can act as a feasible benchmark. The value added time is therefore estimated to 10 days when the further improvements have been implemented.

**10.1.11 Backlogs**

The lowest or highest accepted level of opportunities, MR’s or user stories in each backlog is not defined today, but should to some extent be determined in the future. This is necessary to enable control by measuring the value stream and getting an indication regarding the need of resources of each process. The only backlog that should not been set with a predetermined level is the opportunity inbox, this since the customer should not be restricted in putting orders onto the backlog. Hence, the time an item spends in this backlog is not possible to estimate as it depends on the rate of opportunity submissions. The intension is that the Communication Forum will provide a long-term planning perspective and understanding between the instances, which will favor a balanced opportunity inbox even without a maximum input level.

Since CRM I works as a gatekeeper it ensures that the opportunities are pulled into the value stream. This ensures that the level of items on each backlog can be kept low and thus the waste of prolonged items on backlog will be avoided. The prioritized MR-inbox should hence have a maximum level of “2 weeks work”. The following processes will thereby be supplied with work, but with a reduced waiting time for the MR’s. This makes the average waiting time 7 days as the Pre-screening takes place every second week. The same appears for the Product backlog. At best the items should pass this backlog within two weeks, which gives an average waiting time of 7 days. The team backlog should in the future include items to fit the time frame of a two-week sprint. Thus, all items will be handled within two weeks, but the team backlog also include the items groomed for the upcoming sprint and will thus be finished within a four week period. The average waiting will time thereby become 14 days when making this calculation for both groomed items, and items being handled in the upcoming sprint.
11 Conclusions

In this section conclusions are drawn regarding the applicability of Lean and Lean tools in a software development context.

Lean can be applicable in order to create flow efficiency and eliminate waste even in a software development context. Although the tools need to be used in a pragmatic way and adjustments are necessary in order to succeed. The Value Stream Mapping tool helps in detecting bottlenecks and inconsistencies in the flow, and to some extent the wastes being created throughout. However, all wastes are not detectable, especially when occurring inside a process. On the other hand, the VSM contributes to a holistic view and help knowledge and information sharing within the company. Moreover, the VSM can be a good tool for indicating what to measure and facilitating a follow-up of the current value stream and changes made to it. To use the tool of creating a fishbone diagram by conducting a root cause analysis was easy and gave a comprehensive view upon the issues, hence it is feasible in the current software development context.

The VSM was lacking in its ability to visualize information flows. In contrary to a manufacturing type industry, the information could be done in numerous ways and when using the VSM it needed to be grouped by rather rough estimates.

The future state map lacks in providing all the solutions, especially the ones not strictly related to an operational process. Such an example is the C&A Communication Forum. Furthermore to reduce the number of meetings, when the meeting structure is iterative, does not seem to be clearly communicated by the VSM. The software development way of working also makes time estimates and backlog levels quite hard to establish, this since each MR is broken down into smaller parts, i.e. the product does not stay intact throughout the value stream. However, the future state does provide a goal to strive towards and a good basis for communicating the necessary improvements to higher instances.

The efficiency ratio is hard to evaluate due to the lack of benchmarking opportunities, and also due to the hard time estimates. Although, the VSM will sufficiently show improvements being implemented into the value stream.

Though, a small remark must be made in this study since the method of VSM was adjusted by the researchers. This could have some impact on the result by making it more context dependent. The main aim for using this method and the current and future state maps are not external benchmarking since it might not be feasible.

With all that stated, the conclusion is that Lean is applicable in a beneficial way also in a software development environment. Classic Lean tools can thus be used but only after necessary adjustments to fit the nature of the context.
12 Future research

Suggestions on future research in the academic field are presented in this section.

Even if the applicability of Value Stream Mapping and an implementation of Lean in this context are evaluated in this study, further studies are required to improve the reliability. Moreover, the findings from this study is very context dependent and thus more studies on how VSM can be applied in software development is demanded in order to increase the generalizability. More research could also be done to evaluate the effect that Lean has on the performance and benefits of Agile and what necessary structures that need to be in place for a successful implementation of the philosophies at the same time.
References

Book

Conference proceedings


**Figures**


**Internet sources**


**Journals**


**Reports**


Appendix A
Current state Value stream map figure Component 1
Appendix B
Current state Value stream map figure Component 2
Appendix C
General future state map for the components