Interdisciplinary Requirement Engineering for Hardware and Software Development - A Software Development Perspective

by

Bilal Tahir Sheikh

Supervisor(s): Lena Buffoni
Examiner(s): Kristian Sandahl
Abstract

The software and hardware industries are growing day by day, which makes their development environments more complex. This situation has a huge impact on the companies which have interdisciplinary development environments. To handle this situation, a common platform is required which can be acted as a bridge between hardware and software development to ease their tasks in an organized way.

The research questions of the thesis aim to get information about differences and similarities in requirements handling, and their integration in current and future perspectives. The future prospect of integration is considered as a focused area. Interviews were conducted to get feedback from four different companies having complex development environments. The conclusions of the thesis are as follow:

- Hardware and software are using different development processes, which makes it difficult to agree on common platform. Although hardware side can get benefit from software side through adopting agile development approach in development environment.
- The utilization of different development processes in hardware and software fields cause different time frames, which are hard to combine.
- It is a good sign that companies have internal standards to handle overall processes, documentation, and modification handling.
- Traceability between requirements is tried to covered from software theoretical framework, but according to the companies feedback they are striving to have a solid traceability mechanism.
- Balancing and compromising of requirements is a key issue which was endorsed by interviewed companies as well.
- The feedback from companies about increased interdisciplinary development suggests more obstacles than enablers. The inter-departmental dependency is one of the major obstacle and companies are looking forward to minimize it.
- Communication and collaboration are key enablers, highlighted by almost all interviewed companies to increase interaction between development teams to make interdisciplinary working environment more productive.

**Keywords:** Requirements handling, interdisciplinary development, integration, hardware, software, development, agile, engineering, system, platform, process, interview, communication, collaboration, traceability
Acknowledgments

First of all, I am thankful to Almighty Who has given me strength to complete this thesis report. I am always thankful to my family whose unlimited moral support and encouragement motivated me to give my best.

I am extremely thankful to my formal thesis partner Hanna Johansson, who has given her support from beginning to till submission of this thesis report. Most of the work in this thesis is done together with Hanna Johansson, studying the education program Design and Product Development. The contribution to the chapters in this thesis are distributed as follows.

1. Introduction: The approach section in the introduction chapter is written by me. The rest of the contents are written together with Hanna Johansson.

2. Theoretical framework: The theoretical framework about software engineering, requirements handling in software engineering, and system engineering are written by me. The thesis presented by Hanna Johansson covered hardware engineering, requirements handling in hardware engineering, and requirements engineering.

3. Method: The methodology chapter is used as a common platform between this and Hanna Johansson thesis. Hanna Johansson explicitly worked on this chapter with my feedback and discussion.

4. Results & Analysis: The compilation and analysis of results regarding research questions 1 and 2 in section 4.1 are done together with Hanna Johansson. The section 4.2 according to software engineering is done by me. Hanna Johansson given her input according to hardware engineering prospect in section 4.2. The sub-section 4.3.2 is written by Hanna Johansson and rest of the sections are written together. The results which are presented regarding research questions 5 and 6 from the interviews are written together with Hanna Johansson.

5. Discussion: The discussion about theoretical framework (section-5.1) and methodology (section-5.2) is written together. The discussion in sub-sections 5.3.2 and 5.3.15 is written by me. The discussion in remaining sub-sections of section 5.3 is written together with Hanna Johansson. The discussion in sub-sections 5.4.1, 5.4.3, and 5.4.5 is done with Hanna Johansson. The sub-section 5.4.2 is written by Hanna Johansson. The discussion in sub-sections 5.5.3, 5.5.6, and 5.5.8 is done by me. The sub-sections 5.5.1, 5.5.2, 5.5.4, 5.5.5 and 5.5.7 is written together with Hanna Johansson.

6. Conclusion: The organization of conclusion chapter is done by both of us. The text which was presented in connection with hardware engineering is written by Hanna Johansson. I had written conclusion related to software engineering.

7. Future work and Recommendations: The chapter future work and recommendations is written together with Hanna Johansson.

I am grateful to the companies which given time for interviews.
I am also thankful to the examiner Kristian Sandahl and supervisors Lena Buffoni, and Sara Nilsson for their valuable advices and feedback.

I also want to acknowledge the guidance of Marjorie from Academic English Support at Linköping university. Umair, Tauqueer, and Usman were also helpful in this regard. The valuable advices from these mentioned people were helpful for me to improve my English writing skills.
Abbreviations and definitions

Abbreviations

CCB Change control board
CDS Component design specification
HoQ House of Quality
HW Hardware (development)
PDS Product design specification
QFD Quality function deployment
RQ Research question
SDLC Software development life cycle
SRS Software requirements specification (software)
SW Software (development)
TDD Test driven development

Definitions

Actor: Often referred to as a stakeholder. Could be a person, a company, a legal entity. Anyone involved or effecting a development or requirement engineering process.
Agile: In software engineering, the agile development approach focuses on testing, development, and integration of proposed software in continuous way.
Complex system: In this thesis, a complex system is a system that requires more than one development department’s contribution for completion.
Hardware engineering: In this thesis, hardware engineering and hardware development concerns the development of all physical aspects of a product or system, and not only computer hardware which may be the use of these expressions in other fields.
Liaison: A liaison is a role in big companies acting as middle man or facilitator, whose duty is to make cooperation and communication between internal and external stakeholders in an efficient and collaborative environment.
Organizational system: The definition of an organizational system differs depending on company. It however is more complex than just entering requirements into a spreadsheet and keeping them manually in folders. An organizational system allows for storage of requirements, but also storage of related documents such as drawings. It has the possibility to link different entries, and to handle different versions of both requirements and related documents.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>i</td>
</tr>
<tr>
<td>Acknowledgments</td>
<td>iii</td>
</tr>
<tr>
<td>Abbreviations and definitions</td>
<td>vii</td>
</tr>
<tr>
<td>Contents</td>
<td>ix</td>
</tr>
<tr>
<td>List of Figures</td>
<td>xi</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xi</td>
</tr>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Aim</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Research questions</td>
<td>2</td>
</tr>
<tr>
<td>1.4 Approach</td>
<td>2</td>
</tr>
<tr>
<td>1.5 Delimitations</td>
<td>3</td>
</tr>
<tr>
<td>1.6 Guide to the report</td>
<td>3</td>
</tr>
<tr>
<td><strong>2 Theoretical Framework</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 SW engineering</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Requirements handling in SW engineering</td>
<td>17</td>
</tr>
<tr>
<td>2.3 System engineering</td>
<td>24</td>
</tr>
<tr>
<td><strong>3 Method</strong></td>
<td>35</td>
</tr>
<tr>
<td>3.1 Interview methodology</td>
<td>35</td>
</tr>
<tr>
<td><strong>4 Results &amp; Analysis</strong></td>
<td>47</td>
</tr>
<tr>
<td>4.1 RQ 1 &amp; RQ 2 - Differences and similarities between different development processes’ requirement handling</td>
<td>47</td>
</tr>
<tr>
<td>4.2 Link between different theoretical domains, interview questions, and answers</td>
<td>56</td>
</tr>
<tr>
<td>4.3 RQ 3 &amp; RQ 4 - integration state, current and target</td>
<td>61</td>
</tr>
<tr>
<td>4.4 RQ 5 &amp; RQ 6 Obstacles and enablers, simultaneous HW &amp; SW engineering</td>
<td>63</td>
</tr>
<tr>
<td><strong>5 Discussion</strong></td>
<td>65</td>
</tr>
<tr>
<td>5.1 Discussion of theoretical framework</td>
<td>65</td>
</tr>
<tr>
<td>5.2 Discussion of methodology</td>
<td>66</td>
</tr>
<tr>
<td>5.3 Discussion of results for RQ 1 and RQ 2</td>
<td>68</td>
</tr>
<tr>
<td>5.4 Discussion of results for RQ 3 and RQ 4</td>
<td>73</td>
</tr>
<tr>
<td>5.5 Discussion of results for RQ 5 and RQ 6</td>
<td>75</td>
</tr>
<tr>
<td><strong>6 Conclusions</strong></td>
<td>79</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Approach the thesis ................................................. 2
1.2 Graphical representation of thesis ................................. 3

2.1 Connection between theoretical fields ............................ 5
2.2 Waterfall process model ........................................... 9
2.3 Comparison between Waterfall & V-model ....................... 10
2.4 Boehm spiral model ............................................... 11
2.5 A comparison b/w Agile & plan-driven development processes 13
2.6 Scrum Process .................................................... 16
2.7 Requirements engineering process ................................ 17
2.8 Requirements engineering - Spiral model ....................... 19
2.9 loop of elicitation & analysis .................................... 19
2.10 Requirements modification management process .............. 23
2.11 Activities of system acquisition process and interaction with life cycle 26
2.12 Requirements driven system engineering approach ........... 32

3.1 Comparison induction and deduction .............................. 36
3.2 Method triangle, knowledge depth ................................ 36
3.3 Process of interview methodology ................................ 37
3.4 Development of the interview guide .............................. 40

4.1 Process of company W, mainly developing SW .................. 49
4.2 Process of company X, mainly developing HW and electronics 49
4.3 Process of company Y, mainly developing HW and services .... 50
4.4 Process of company Z, developing both HW and SW .......... 50

List of Tables

2.1 The waterfall model - stages & tasks .............................. 8
2.2 Differentiation between V-model & traditional waterfall model 10
2.3 Elaboration of user stories through use cases .................... 15
2.4 Elements of a partial PDS ......................................... 28
2.5 Component Design Specification elements ....................... 29
1 Introduction

This thesis is a part of a subsection of the Mistra REES program (Resource-Efficient and Effective Solutions) which aims to advance the transition of the Swedish manufacturing industry towards a circular and sustainable economy. The subsection to which this thesis belongs, concerns Product and Service Design Methods focusing on the development of a sustainable requirement specification, which involves both customers, and actors, looking from the entire life cycle perspective. The focus of this thesis keeps on requirements handling, and actors who are involved in it. Software (SW) engineering is covered throughly in this thesis and hardware (HW) which has covered by [Johansson (2017)] thesis. The integration of these two is tried to covered through an overview of system engineering.

1.1 Motivation

Today is the era of industries revolution where the role of SW and HW engineering has great impact to organize their development environment. A general perception exists in the mind of HW people that SW engineering is not competent to meet latest challenges in SW development, but [Sommerville (2016)] has a different opinion in this regard who says that, it is true that pretended competency issue of SW engineering is raised due to increasing system complexity and failure to use SW engineering methods. SW engineering provides in detail information about how SW is produced with the help of different development processes, tools, techniques, and models to increase the standard of SW products [Pfleeger and Atlee (2006)]. The purpose of requirements handling is well defined by [Ambler (2002)] as a discipline that handles the requirements of proposed system (SW, HW), which starts from interaction with stakeholders to know what stakeholder’s expectations from proposed system are. It gives right direction to development team to look what are the requirements which need to be fulfilled. It gives support to do estimation in terms of time and resources which are required to build the system. The use of system engineering for development of product is becoming a helping hand to increase collaboration in an interdisciplinary development environment. It gives a better management opportunity to handle complexity specially in terms of time and cost factors [Grabler and Yang (2016)]. The worth of integration in interdisciplinary working environment is defined by [Boyd L (2013)], who says that integration can acts as a compass of HW, SW, and systems work in line as a single unit. The integration methods can play a vital role to bridge the gap in interdisciplinary development environment. The integration method is basically a plan which guides how to integrate different system components together as a complete package [Pfleeger and Atlee (2006)]. To have a solid integrated method in integrated development environment is a key of success. The role of requirements handling in SW engineering process models has a vital role to handle the complexity of interdisciplinary development environment. That is the center of attention of this thesis, to investigate and analyze the requirements engineering process in context of interdisciplinary development environment.
1. **Introduction**

1.2 **Aim**

The major aim of the thesis is to understand the importance of handling requirements in SW engineering from both theory and practice perspectives. It investigates the possibility of combining both HW and SW development environments. The other objective of this thesis is to examine current state of integration between different development environments in industries to analyze their approaches and look current obstacles along with enablers to increased integration in the future.

1.3 **Research questions**

For the thesis to be considered complete it needs to answer the below research questions. The research questions come in pairs, with the first pair concerning differences and similarities in HW and SW engineering in interdisciplinary requirement handling. The second pair concerning the current and future wanted state of integration for companies which develop complex systems. The third pair concerns current obstacles and enablers for working towards wanted future state.

RQ 1 What differences in HW and SW engineering need to be considered during interdisciplinary requirements handling?

RQ 2 What similarities in HW and SW engineering support interdisciplinary requirements handling?

RQ 3 What is the current state of integration in companies which develop complex systems?

RQ 4 Are companies which are working with complex systems striving to have an interdisciplinary development environment in the future?

RQ 5 What obstacles exist for simultaneous HW and SW engineering?

RQ 6 What enablers exist for simultaneous HW and SW engineering?

1.4 **Approach**

The approach which has followed in the thesis is presented in Figure 1.1. The topics which are covered in this thesis are; SW engineering, requirements handling in SW engineering, system engineering. The detail about HW engineering is given in [Johansson (2017)](Johansson2017) thesis. Interviews are performed at 4 different companies on the basis of research questions (RQs).

![Figure 1.1: Approach the thesis](Own figure)
The interviewees at the companies had roles, or experience, covering mechanical, electronic, SW, and service development. Some companies wished to remain anonymous and therefore all companies have been left unnamed in this thesis. Once the interviews were done and summarized for each company, the results were summarized in the results chapter. The final analysis is performed using both data from interviews and the theory collected in the theoretical framework. Research questions 1 & 2 will be answered using both theory and interviews. The comparison of theoretical frameworks will also include related interview questions and answers for these questions. Research questions 3-6 will be answered using results from interviews.

1.5 Delimitations

It was agreed that a reasonable time per individual interview was 60 minutes. The amount of interviews was set to 2-4 per company, and 4 companies were visited for interviews. The responsibility for selection of interviewees was given to the companies as it was believed they would know who possessed the correct expertise within the company. Within the theoretical framework, the focus was upon the parts of SW and HW development which work with requirements. As system engineering was not mentioned in depth by the companies, the theoretical framework on requirement engineering was seen as enough coverage.

1.6 Guide to the report

Figure 1.2: Graphical representation of thesis

The thesis begins in a traditional formatting with an introduction, theoretical framework, and a method chapter. This choice has been made as the overview of different technical fields is needed for all research questions, and similarly the interviews contribute to all research questions. The end of the thesis becomes more and more divided by research questions. The final chapter of future work and recommendations returns to traditional formatting as this concerns several areas and research questions.

1. Theoretical framework

The report begins with a theoretical framework, which introduces SW engineering, requirements handling in SW engineering, and System engineering provides an overview of methodologies used to work with complex systems. HW and requirements engineering are covered in the thesis by Johansson (2017).

2. Method

The methodology presents theory related to the method and the method which has
been applied in this thesis. A very brief description is provided for how literature was collected, the interview methodology is more extensive.

3. Results & Analysis
First the in-depth studies from theory for requirement handling from different areas is presented. This is followed by an analysis which compares the different areas and connects the theory with the interview questionnaire. After this analysis, results from the interviews are presented sorted under the relevant research questions.

4. Discussion
The discussion covers methodology, ethics, and clear link for results going into the conclusion.

5. Conclusion
Conclusions are given in the format of answers to the six research questions. Conclusions have a clear connection to the results presented in the thesis.

6. Future work and recommendations
In the final chapter future work and recommendations are stated, based on both thesis conclusions and needs for future research expressed by interviewees in the interviews.
2 | Theoretical Framework

SW engineering, requirements handling in SW engineering, and system engineering along with requirements handling in system engineering are covered in theoretical framework of this thesis. In computer world, HW engineering is generally referred as manufacturing of physical parts of computer. The thing which is noticeable in [Johansson (2017)] thesis is that HW engineering is related with development of physical parts of a system or a product. The detailed discussion about HW engineering and requirements handling in HW engineering is given in [Johansson (2017)] thesis.

The systems which the thesis concern are complex, which requires covering more than one development process in the theoretical framework. System engineering usually looks to larger, and therefore more complex. The organization of this chapter can be seen in Figure 2.1. The arrows in the Figure 2.1 are used to indicate the flow and organization of chapters in this report.

![Figure 2.1: Connection between theoretical fields](Own figure)

2.1 SW engineering

There are several authors who defined SW engineering in different ways but the definition of SW engineering which is given by IEEE is probably the most accepted one. According to IEEE, it is an application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of SW; that is, the application of engineering to SW, this definition is further summarized by [Sommerville (2016)] who states that, the SW engineering is an engineering discipline concerns with all aspects related to the production of SW right from scratch (conceptual design), development (coding), operation, and ends at maintenance.

After defining SW engineering, rest of the sub-sections are organized in following way; First the discussion has been made about what is SW engineering mechanism and how it works in form of standard phases/steps. Next, the introduction is given about SW engineering process. The purpose behind this introduction is to get the understanding about
how different types of SW engineering processes which influence the overall SW engineering mechanism. Finally the process models of each process type are also mentioned as sub section of each process to give an overview about how SW engineering could be done.

2.1.1 SW engineering mechanism

The SW engineering mechanism is a combination of phases or steps which are acted in sequential manner to produce SW. These steps are considered as a guideline regarding how SW development is done in an organized way Sommerville (2016). The role of requirements in SW engineering mechanism is much vital because the whole SW is based on requirements specification. The requirements are basically needs or expectations of different stakeholders from specific SW which is going to be developed. The detailed discussion about requirements handling in SW engineering is provided in later sections of the report.

How SW engineering mechanism works?

The basic idea about steps/phases which are commonly involved in SW engineering mechanism are originated from literature of authors Sommerville (2016), Braude (2003) and Humphrey (1990). The description related to verification and validation is explored from Bell (2005).

- SW specification/requirements analysis (answers: what)
  The SW specification or requirements analysis acts as a platform where specific stakeholders (customers, developers, testers, users etc.) sit together in order to define the characteristics along with specific constraints related to the proposed SW. These characteristics are known as requirements which is the subjected area of this report. The SW specification and analysis is the phase where requirements engineering is performed. As a result, the standard SRS (SW requirement specification) document is produced, which is all about answering of ‘what’ is going to be built.

- Design (answers: how)
  The second step is known as design phase, where methodology is defined regarding ‘how’ the specific proposed SW is going to be built. This methodology contains group of elements in the form of figures (flowcharts, etc). The design document is produced on the basis of requirements which are provided earlier.

- SW development/coding
  The phase where developers perform their activity in terms of writing code/program in order to develop SW.

- SW Testing (verification & validation)
  The testing is a set of activities which is performed in order to ensure that the proposed SW is designed according to the relevant SRS (SW requirements specification) document. These activities are carried out right from the beginning, middle, or at the end of SW development process. It also depends upon which type of development process model is followed. It has noticed that there are different types of testing involved in SW engineering. The types which are specifically related with testing of proposed SW requirements are *system testing* and *acceptance testing*. The system testing acts as a mean to check whether the proposed SW fulfills its overall requirements (objectives) Humphrey (1990). The acceptance testing ensures that the designed SW system is tested to make sure that it fulfills requirements and expectations related to the customers Sommerville (2016).
  The verification & validation techniques are used to verify and validate proposed and
2.1. SW engineering

designed SW. Barry Boehm (known as pioneer in SW engineering) explained the difference between both techniques - verification is the process to ensure that the SW which is going to be built is the ‘right’ SW, but on the other end - validation is a technique ensures that the designed/built SW fulfilled all of the requirements of the customers Pfleeger and Atlee (2006). The fact need to remember is that if SRS does not capture the needs, a verified system might not be valid. The verification is considered as developer’s view (development team prospective) of implementation of SW, while validation is considered as external view (end-user prospective) of the system through the eyes of testers after development. The unit and system testing are sub-sets of verification technique and acceptance testing is a major sub-set of validation Bell (2005).

- SW maintenance

The SW maintenance is the phase which is performed to meet the changes whether they are initiated from the customers after the development has started or the requirements which are raised due to current market demand right after the delivery of designed SW. The maintenance is the phase which is helpful for removing defects which are mismatched with actual requirements specification. The maintenance phase also provides an opportunity to add advance features to meet the latest challenges of competitive market Braude (2003).

2.1.2 SW engineering processes

The SW engineering processes are differentiated into two types, which are proposed by Sommerville (2016). The first type is plan driven while the other is agile development process. The five steps which are mentioned in SW engineering mechanism are commonly used in all development processes but the way of utilization of these steps depends upon which type of SW development process model currently follows.

Plan driven/structured process

In plan driven development process, all activities which contribute towards development of SW are planned on advance basis. These advance plans are used as tools to measure the overall performance of development process Sommerville (2016). The plan driven process is mainly focused on doing estimation in advance to deal with the major factors like cost, time, actors, and deliverables. These factors are considered as major characteristics of any proposed SW which is going to be built Wilson (2007). The traditional waterfall and V-shaped development models are major examples of plan driven process.
1. The Waterfall model

The traditional waterfall model was first presented in a paper written by [Winston W. Royce](1970). The waterfall model is considered as first model where concept of stages was introduced for production of SW [Sommerville](2016). The waterfall model is also known as the first model where the term ‘requirements’ was used as a step along with other steps in SW development model. The detailed overview regarding ‘requirements’ is given later in this chapter.

<table>
<thead>
<tr>
<th>Stages</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements definition</td>
<td>The journey of waterfall development model starts from the step where requirements related to proposed SW are defined and analyzed. The requirements definition stage is responsible for dealing with requirements in terms of defining services, constraints, and goals related to specific SW (functions) and HW (physical components of the system) <a href="2010">Mohammed et al.</a>. The tool which is used in this stage to achieve mentioned features of requirements is meetings with relevant stakeholders stated by <a href="2016">Sommerville</a></td>
</tr>
<tr>
<td>System &amp; SW design</td>
<td>The quota of requirements assigned to proposed HW/SW systems is carried out at system/SW design stage. The quota of assignment is truly based on the type of requirements <a href="2016">Sommerville</a>. System/SW design stage is responsible for the selection of appropriate framework and the designing of whole system architecture <a href="2010">Mohammed et al.</a>.</td>
</tr>
<tr>
<td>Implementation &amp; Unit testing</td>
<td>The deployment of SW design and activity of testing on unit basis are performed during this stage. The term deployment means transformation of selected design architecture into group of programs or program units <a href="2016">Sommerville</a>. So implementation and unit testing is the stage where actual coding is done and testing is performed to validate whether the implemented SW meets its requirements or not <a href="2010">Mohammed et al.</a>.</td>
</tr>
<tr>
<td>Integration &amp; System testing</td>
<td>The operation of integration on group of programs or individual programming units is carried out during integration and testing stage. The major reason behind testing of whole system is to ensure that the implemented system is developed according to its requirements specification. Finally, the produced SW is handed over to concerned stakeholders <a href="2016">Sommerville</a>.</td>
</tr>
<tr>
<td>Operation &amp; Maintenance</td>
<td>The operation and maintenance is the extensive stage of waterfall model. The actors who are involved in operation and maintenance stage are the people from SW support department. The activities at this stage are started from the installation of system/SW into working environment. The role of maintenance comes into play to rectify errors which were not found during earlier stages of SW development process. This stage is also responsible for doing up-gradation in implemented SW in terms of adding new features in it <a href="2016">Sommerville</a>. The companies with large setups usually have change control board (CCB), which is responsible to keep check and balance on every change which is done at maintenance phase <a href="2010">Mohammed et al.</a>. The role of CCB is discussed in later section of the report.</td>
</tr>
</tbody>
</table>
2.1. SW engineering

The list of stages along with their tasks is provided in Table 2.1. The theory related to waterfall model is explored from Sommerville (2016) and Mohammed et al. (2010). The formulation of waterfall model is presented in Figure 2.2.

2. The Enhanced waterfall model (V-model)

The enhanced waterfall or V-model is updated version of traditional waterfall model. The V-model basically exhibit testing activities which are related to analysis and design phases of SW engineering [Pfleeger and Atlee (2006)]. The V-model depicts SW validation activities which are carried out during each stage of traditional waterfall model [Sommerville (2016)].

The difference between both models is highlighted in Figure 2.3. The key factors which create difference between both models are mentioned in the Table 2.2.
2. Theoretical Framework

Figure 2.3: Comparison between Waterfall & V-model
Inspired by Pfleeger and Atlee (2006)

Table 2.2: Differentiation between V-model & traditional waterfall model

<table>
<thead>
<tr>
<th>Waterfall Model</th>
<th>V-Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>In waterfall model, the SW validation rely on development stages where there is no facilitation given by system testing to validate requirements Pfleeger and Atlee (2006).</td>
<td>The key factor of V-model which differentiates it from the waterfall model is SW validation process. The SW validation is performed at each stage of V-model Sommerville (2016) that is not followed in waterfall model.</td>
</tr>
<tr>
<td>The testing activity is performed at the stage when development is done in waterfall model Unnati S. Shah (2016).</td>
<td>The probability of success is higher in V-model than the waterfall model due to the presence of test plans earlier in SW development life cycle (SDLC) Mohammed et al. (2010). So The activity of testing is performed at each stage during SW development process in the V-model.</td>
</tr>
<tr>
<td>In waterfall model, we do not have combination of phases feature where high and low level phases do emphasize on different vital activities related to SW development Mohammed et al. (2010). The combination of phases feature work effectively in V-model.</td>
<td>The V-model contains the combination of high and low level phase, where the high level design phase emphasizes on system architecture and design. The output of this phase is an integration test plan. The lower level design phase is used to design SW components and tests are generated for unit testing Mohammed et al. (2010).</td>
</tr>
<tr>
<td>When we look towards time and cost factors in waterfall model, the focus is placed on documents and artifacts Pfleeger and Atlee (2006).</td>
<td>In case of V-model, the error in software development life cycle are found earlier as compared it with waterfall model. This factor saves time and reduce cost in overall SW development process Unnati S. Shah (2016).</td>
</tr>
</tbody>
</table>
Boehm’s Spiral Model - hybrid model

The spiral model was first introduced by Barry W. Boehm in an article [Boehm (1988)], which is taken as a major source to explain this model. The main difference between previously discussed models and Boehm’s spiral model is risk analysis and prototyping. We can say that risk analysis factor differentiates Boehm’s spiral model from previously discussed SW process models. The above mentioned models were based on plan driven (document/code driven) approach [Boehm (1988)]. The other prominent feature of spiral model is its hybrid (iterative + incremental) nature. The spiral model is considered as a forerunner of agile development, that is why it has worth to discuss before move to the agile development process. The contents which are written to explain this model are based on how this model works and how risk management is performed in form of spirals. The invention of spiral model was opened a new horizon in SW engineering field at its time with its key features like prototyping and risk evaluation as mentioned by Sommerville (2016) and Peters, James F, Pedrycz (2000).

How spiral model works

The other difference between Boehm’s spiral model with other traditional models is the execution of phases in SW development process. As shown in Figure [2.4] spiral model, the execution of phases is done in form of rounds rather than sequential order of stages as seen in previously discussed models. Here each round represents a specific phase which plays its part to make SW process successful Sommerville (2016). The whole spiral model is distributed into four iterations or rounds, the detail of activities which are carried out in each round is as follow.

- Round-1: The first sector is the phase where requirements and development plan of proposed SW are handled in terms of budgeting, specific constraints and alternatives like staffing, design, and etc. After that, the role of risk evaluation comes into play along with prototyping options Sommerville (2016). This is all happened right before producing of concept of operation document which contains high level description about how the system should proceed Pfleeger and Atlee (2006). The role of concept of opera-
tion document is vital here, because with the help of it consistency and completeness in requirements are get ensured through inspection \cite{Pfleeger2006}. This document is treated as an output of this first iteration.

- Round-2: After first phase now it’s time to analyze alternatives to overcome risks. The one of the alternative is known as prototype \cite{Sommerville2016}, which gives an opportunity to reduce risk in terms of finding unsuitable requirements which could disturb the whole development process of proposed SW \cite{Peters2000}. The output of this iteration is suitable/authentic requirements \cite{Pfleeger2006}.

- Round-3: Now it is time to do a vital task in shape of selecting right model for SW/system development \cite{Mohammed2010}. There are couple of scenarios given by \cite{Sommerville2016} in this regard, for example in case of user interface has greater level of risk than prototyping (a throw away piece of code) is the right approach to follow. In case of other risk e.g. safety factor, than formal transformation is the best solution \cite{Sommerville2016}. In short, it is all about which kind of risk is encountered and select best approach to tackle it. This iteration of development produces design as an output \cite{Pfleeger2006}.

- Round-4: The final phase is known as execution. In execution phase, the whole development exercise is examined through execution of whole project to decide either to go further with another round or not. If the decision is yes then next phase of proposed SW is carried out \cite{Mohammed2010, Sommerville2016}. The fourth and final stage is related to testing activities in form of unit, system, and acceptance tests.

**Agile development process**

The SW development models of plan driven process were based on static approach for SW development. The revolution came in this approach in late 1990s, when a group of developers proposed a unique way of development SW, which was based on flexibility and efficiency. The flexibility and efficiency features lead SW development process towards agility \cite{Pfleeger2006}. That is how agile development came into existence in light of agile manifesto by \cite{Cockburn2001}. The agile SW development process is based upon the concepts \textit{iterations and continuity}, which are involved in planning and carried out during whole SW development process \cite{Sommerville2016}. The iterative development brings whole system as early as possible and then implements the changes to the functionality of each sub system after every new release or iteration \cite{Pfleeger2006}. That is also considered a ultimate goal of agile development is to get the satisfaction of the customer by \textit{early and continuous delivery} of proposed system \cite{Pfleeger2006}.

**Comparison between agile & plan-driven development**

The given scale in Figure 2.5 has been sketched to highlight the core features of agile SW development process, which dominate over plan-driven development process. The idea about Figure 2.5 has taken from \cite{Cockburn2001} who explained about agile manifesto. The detail about comparison between both development processes is as follow.

1. **Individual & interaction over processes & tools**

   In plan-driven development environment, the processes and tools which are used to carry out activities for development of SW are selected in advance and documented as well. There is no room to make changes in selection after that. In agile SW development environment, the initiative is given to individuals who are involved in SW development process. The feature of communication in this regard is very important which has elaborated by \cite{Boehm2003} - says that the focused on implicit and shared
knowledge between involved people could become helping hand to get maximum performance through *individual and interaction* [Cockburn et al. (2001)]. The collective team strategy, team meetings, pair programming, planning games, and SCRUM are considered as enablers to increase collaboration and communication between team members [Boehm and Turner (2003)].

2. Extensive documentation Vs working SW

In plan-driven development, all activities related to planning and implementation are stated in form of documents. But in agile SW development test cases and code-source are key factors despite of doing in detail documentation as mentioned by [Boehm and Turner (2003)]. As mentioned by [Pfleeger and Atlee (2006)], the agile development puts time investment on producing working SW rather than producing bulk of documents. [Martin (2003)] has spoken in favor of less documentation says that, to have much documentation requires much time and resources for management and synchronization of documentation with the code.

3. Agreement compromise Vs customer collaboration

In plan-driven development, if something has not achieved according to the plan (e.g., some specified requirements are not achieved) than negotiation is the only source to convince stakeholders at the end. These negotiations are done to get approval from stakeholders on un-achieved tasks. The following. In case of agile development, the close collaboration with customers during whole development process of SW is the key of success [Pfleeger and Atlee (2006)]. As the result, the stakeholders are aware in time about the current status of their proposed product rather than surprised them at the end of whole SW development process.

Figure 2.5: A comparison b/w Agile & plan-driven development processes

[Cockburn et al. (2001)]
Agile development methodologies

The methodologies which are used in agile SW development process utilize agile manifesto principles which are explained earlier. These methodologies are known as SCRUM and XP (extreme programming). The SCRUM is very common in complex development environment of large companies.

Extreme programming (XP)

The XP facilitates developers to increase their productivity and lessen the burden in terms of administrative overload [Wilson 2007]. There are several practices exist under XP but the practices which are common now a days to facilitate development team are as follow.

- **User stories**: [Sommerville 2016] defined the role of user stories in context of requirements. The user stories or scenarios are used to represent requirements which are implemented as list of tasks in sequential manner. [Martin 2003] also defined user story as a hint or suggestion of a requirement related to proposed SW. This hint is used as a tool by actor (customer) and utilizes it as a mean for implementation of requirements on the basis of cost estimation and prioritization. [Sommerville 2016] also described user stories as a base of planning game which is nothing but a set of user stories that fulfills the functional requirements of proposed system or SW.

  To elaborate user stories, [Ambler 2002] has given an example of a company named SWA to elaborate how to write user stories against relevant use cases. The company SWA does business of selling goods online. The list of use cases or actions that a customer can perform on company’s website are: placement of order, request for help, read product reviews, write review about a purchased or used product, and return product as well. Here, two major use cases 'Place order' and 'Search for item(s)' are picked up as example to write their user stories. The detail about use cases with their relevant user stories is given in Table 2.3.

- **Test driven development (TDD)**: In TDD context, developers work in a pair to write test cases of each requirement (task) before the actual coding starts. These tasks need to be executed successfully when the latest code is joined by the system [Sommerville 2016]. The concept of TDD is discussed by [Bell 2005] in context of user stories, where the client is the one who specifies acceptance tests for each user story. In XP, acceptance test acts as a trigger of SW project and written before development and implementation of use cases. At the beginning, the tests will obviously fail but as soon as actual coding begins these test cases will be successfully passed one by one. So at the end implementation is finish [Bell 2005]. A test case is used to test the system’s component. The input data and conditions are two basic elements of a test case as mentioned by [Pfleeger and Atlee 2006]. Basically, the input data is chosen to show the output that depicts the performance of the code. The selection of test cases and writing the tests are important which could satisfy developers and customers about the correct execution of the program on all input. To achieve these mentioned goals. [Pfleeger and Atlee 2006] described following steps. First, the test objectives are determined. After that, the selection of right test cases is performed and described a test designed to meet a desired objectives. Here the role of objectives is to verify how to organize input to pick appropriate test cases. The development of series of tests that correspond to user requirements before the start of actual coding is the core of TDD [Sommerville 2016].

Scrum

The role of Scrum is defined by [Pfleeger and Atlee 2006] says that, *scrum* is an iterative methodology of agile development where SW is produced in specific time slot known as
### Use Cases

<table>
<thead>
<tr>
<th>Use Cases</th>
<th>User Stories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Placement of Order</td>
<td>The customer can add an item to current order.</td>
</tr>
<tr>
<td></td>
<td>The customer can delete an item from current order.</td>
</tr>
<tr>
<td></td>
<td>Against each single order, the system give complete information of that order e.g. item name, item ID, item catalog’s code, per unit price, number of ordered items, and total amount before deduction of taxes &amp; discounts.</td>
</tr>
<tr>
<td></td>
<td>The customer can enter their shipment information for an order.</td>
</tr>
<tr>
<td></td>
<td>The customer can also enter alternative address for billing purpose of their order.</td>
</tr>
<tr>
<td></td>
<td>The system performs calculation and show subtotal before addition of taxes, discounts, and shipping &amp; handling cost for an order.</td>
</tr>
<tr>
<td></td>
<td>The system perform calculation and shows a total amount of an order.</td>
</tr>
<tr>
<td></td>
<td>The system can calculates and show taxes, discounts, and shipment &amp; handling cost if applicable.</td>
</tr>
<tr>
<td></td>
<td>The customer can look the overview of an order for verification purpose before scheduling it for execution (fulfillment).</td>
</tr>
<tr>
<td></td>
<td>The system give opportunity to the customer to update the order before scheduling it for execution (fulfillment). The system creates a summary of an order and sends it by an email to the respective customer.</td>
</tr>
<tr>
<td>Search for Item(s)</td>
<td>The customer search for item(s) from item(s) backlog.</td>
</tr>
</tbody>
</table>

Table 2.3: Elaboration of user stories through use cases

Inspired by [Ambler (2002)]

### sprint

A sprint normally consists of 30-days time span which is used as a mean to implement the backlog of uncompromisable requirements. SW development teams are the people who utilize this methodology most effectively.

**Scrum - as an agile project management tool**

Sommerville (2016) describes the term Scrum as an agile project management framework to manage the agile development activities in an organized way. The Scrum usually shows the current progress of activities related to proposed SW development. The traditional SW development process required specific persons like managers (project managers) to monitor on going activities and how proposed SW objectives are achieved in time. The SW development team of agile development environment introduced the role of Scrum Master, The ‘Scrum Master’ is the person who administrates overall activities which are carried out during Scrum process Sommerville (2016). The role of a team and good interaction & participation of team members have vital impact in development of effective SW projects Pfleeger and Atlee (2006). Pfleeger and Atlee (2006) also pointed out the importance of a team size, says that the teams with smaller size are most effective than the larger ones. The reason is that when the size of team increases, the communication paths are also expended as well.

**Scrum work flow**

The explanation about Scrum work flow process is extracted from Schwaber (2004) and Sommerville (2016), who defined the Scrum work flow precisely. The Figure 2.6 is used to present how Scrum work flow is executed and which elements are involved to carried out the whole activity. The entire work flow and the role of each element are as follow. The Scrum work flow begins from the conceptual goal that the SW is to be produced Schwaber (2004). The role
2. Theoretical Framework

![Scrum Process Diagram](image)

Figure 2.6: Scrum Process
Inspired by [Schwaber 2004]

of project owner or product manager comes into play here. The responsibilities of project owner starts from identification product features (requirements), prioritization of requirements, and perform continuous monitoring of product backlog to ensure that the project proceeds according to the defined goals [Sommerville 2016]. The product backlog initially contains list of both functional and non-functional requirements, which then scrutinized on the basis of prioritization and finally organized with only functional requirements [Schwaber 2004]. The backlog acts as to-do list [Sommerville 2016], which then prioritized to keep those items on top priority which will generate most values. This prioritized product backlog is considered as a beginning point where contents and priority list are changed most probably at the beginning of project [Schwaber 2004]. The duration of each sprint is normally 2-4 weeks and organized with a sprint planning meeting. The sprint planning meeting gives an opportunity to the product owner and development teams to meet each other to get directions for the next sprint. Here product owner shares the thoughts in terms of expectations with other teams. The expectations are obtained from highest priority list of product backlog. After that, development teams give the estimation about what will be achieved till next sprint [Schwaber 2004]. Daily meetings or Scrums are held where team members share what they have done since from last meeting and what they are going to do till upcoming meeting. They also highlight obstacles which disrupt them to fulfill their targets [Schwaber 2004]. This short meeting is all about getting information regarding current progress of each individual team member [Sommerville 2016]. The last and final event of sprint considers as sprint review meeting, which is described by [Sommerville 2016] who says that sprint review meeting serves for two purposes. First, it helps to improve process through revisits the way which was followed by the development team to perform their tasks. The team also give their feedback on how the development process could be improved further. The second purpose of this meeting is to provide input regarding the product along with its current state which contributes in product backlog review. This review further uses for next sprint or iteration as well.
2.2 Requirements handling in SW engineering

The term *requirements* can be described as deliverables (services, constraints, etc.,) which proposed SW need to fulfill user’s needs Sommerville (2016). These requirements are initially in the form of expectation or desires, which customers or other stakeholders want from the proposed SW. The process which is used to transform these needs or desires into formal or standardized requirements is known as *requirements engineering process*. The requirements engineering process gives an opportunity to SW development team to deal with requirements in an organized way and lead them towards successful SW development.

2.2.1 Requirements engineering mechanism

The requirements engineering mechanism acts as a process of discovering needs related to the proposed SW from relevant stakeholders (elicitation and analysis). It converts their needs into standard form of document (requirements specification) and ends at inspection (validation) of specified requirements Sommerville (2016). The general representation of requirements engineering mechanism is given in Figure 2.7. The whole engineering process consists of four phases - Feasibility study, requirements elicitation & analysis, requirements specification, and ends at requirements validation. The output of each phase is shown inside the attached white circle part with each phase as given in Figure 2.7. The description about each phase is as follow.

![Figure 2.7: Requirements engineering process](image)

*Inspired by Sommerville (2016)*

**Phases involved in requirements engineering**

1. **Feasibility study**

The journey of requirements engineering process starts from feasibility study which is considered as pre-study phase. The role of the pre-study phase is to make sure that the desires of customers could be fulfilled through existing technologies (HW, SW). Feasibility study also provides an opportunity to SW development team to carry out advance estimation about cost factor, related to the the proposed system’s allocated budget Sommerville (2011), Bell (2005). The feasibility report is considered as output document of feasibility study. There are two general ways described by Bell (2005) to
2. Theoretical Framework

perform feasibility study. The first way is to do it in a detailed manner and the second way is to do it on ad-hoc basis.

2. Requirements elicitation & analysis

The requirements elicitation and analysis is the phase where specific stakeholders are chosen and their needs are gathered through different ways. These ways are known as observation and interviews. The observation is used to observe the work-environment of specific client and can also be utilized to study the already existed product of same kind Pandey et al. (2010). Interviews are used to collect information from different stakeholders Sommerville (2011).

This phase consist of 2-steps, where requirements are collected in the form of needs through elicitation and analyzed these raw requirements in business prospective. The analysis of requirements is helpful for handling conflicting and immature requirements. The techniques like negotiation, agreement, communication, and prioritization of requirements are helpful in this regard as mentioned by Pandey et al. (2010). The SW description is considered as output of this phase.

3. Requirements specification

The requirements specification is the output of three activities - listening (elicitation), thinking (analysis) and writing (defining) requirements Bell (2005). Requirement specification is used to transform those requirements received from analysis phase into documented format. This document contains requirements which are categorized in two types - User requirements and SW requirements Sommerville (2011). These two type of requirements are considered as output of this phase.

4. Requirements Validation

The role of requirements validation is to make sure that the requirements are realistic, complete, and consistent. The requirements validation makes sure that the true needs of the customer are met. The rectification of unavoidable errors also done by validation which could reside in the requirements documentation and than modified to avoid further errors Sommerville (2011). The SRS (SW requirements specification) document is produced as an output of requirements validation Pandey et al. (2010).

2.2.2 Requirements engineering - Spiral model

The role of the spiral model is to describe how three major activities (elicitation & analysis, specification, and validation) are carried in requirements engineering process Sommerville (2016). The spiral model of requirements engineering process is considered as conceptual model otherwise these activities are carried out in an interleaved way.

Spiral model - As representation of requirements engineering

How requirements engineering process performed in spiral format is presented in Figure 2.8. Agile development approach utilizes requirements engineering spiral model and evolutionary prototyping. The whole spiral model is divided into three sectors (requirements gathering, requirements standardization, and requirements inspection) as shown in Figure 2.8. The mentioned activities are carried out in iterative way. When last sector finished its task, the output of whole process is achieved in form of system requirements document. The description of each sector is as follows.

1. Requirements gathering (elicitation and analysis)

Requirement gathering is a set of activities where members of SW development team find and collaborate with users (customers, users, stakeholders etc.) to know their expectations (services, functions, features etc.) from proposed SW Sommerville (2016).
2.2. Requirements handling in SW engineering

The goal of requirements elicitation is to find out thoughts of different stakeholders, constraints, and facts etc. The iterative way of working in spiral model Figure 2.8 gives an opportunity to exit from spiral any time when development team feels that elicitation of requirements has been done successfully.

a) The loop of elicitation and analysis

The process of requirements elicitation and analysis is presented in form of loop in Figure 2.9. The idea about loop is originated from Sommerville (2016). The basic theme behind the given process model in Figure 2.9 is to give in depth understanding about the activities which are carried out inside requirements elicitation.
& analysis sector of main Spiral model. Here in this loop of elicitation & analysis, each activity is carried out in form of sequential steps (S1, S2, S3, and S4) as shown in Figure 2.9. The loop of elicitation and analysis (S.1) starts from identification and understanding of requirements where documentation is performed after true understanding about requirements. The second step (S.2) is related to the activities which lead development team towards organizing requirements. The requirements which are gathered during the first step are in scattered form (without classification). The process of differentiation and classification of these scattered requirements is the task of this step. The process of prioritization and negotiation are core activities followed during this step (S.3). The need of these activities is a demand for the SW development process due to the presence of different stakeholders. The requirements could get inconsistent, so step-3 is responsible to tackle inconsistency. The last and final step (S.4) of this loop is to keep the record of requirements in the form of documentation draft. These steps form in a loop so these requirement are considered as input into the next spiral round. There are different enablers e.g. wikis, white boards, and shared environments etc., are helpful to carry out these activities Sommerville (2016).

b) Approaches for requirements elicitation and analysis

The two basic techniques to perform requirements elicitation as highlighted by Sommerville (2016) are as follow:

- Interviews - The exercise of doing interview is considered as basic source of getting feedback from different users to know their thoughts regarding proposed system Sommerville (2016). This approach is mentioned by Nuseibeh and Easterbrook (2000) under the category of traditional technique for requirements elicitation.
- Observation - The second approach which is used for requirement elicitation is to observe different users behavior to understand how they performed their jobs in an operational environment through different tools Sommerville (2016). This approach is mentioned by Nuseibeh and Easterbrook (2000) under the category of contextual technique for requirements elicitation.

Bell (2005) is summed up list of activities which are carried out in requirements elicitation and analysis phase.

- Elicitation and clarification of user requirements.
- Differentiate requirements into specific types.
- Make requirements ready in form of document (for use of next sector).
- Perform negotiation and do agreement for approval of requirements specification.

2. Requirements standardization (Specification)

Kotonya and Sommerville (1998) and Sommerville (2016) assigned different terms to recognize this sector of requirements engineering, but the most common term which is used in this regard is requirements specification or standardization of requirements as mentioned in Figure 2.8. The requirements standardization is the procedure of creating document which contain system and user requirements. The main purpose to write this standardized document is to keep requirements free from any confusion, ambiguity, and insufficient contents etc. Although in practical scenario, it is hard to get this type of flawless document mentioned by Sommerville (2016), Bell (2005) mentioned about the features which need to have in well organized requirements specification document are as follow.
• The requirements specification document should mainly concern to specify properties of the intended system rather than implementation.
• The requirements are written in a way through which they are easily testable.
• The requirements should be clearly defined.

As described earlier, this sector of requirements engineering is recognized by different names so the document which is produced as output of this activity is also recognized by different names - the functional specification (FRS), software requirements specification (SRS) are major terms to recognize these documents as highlighted by Kotonya and Sommerville (1998).

Use cases
A use-case is a scenario of events. The use cases are commonly used as a approach to write requirements specification document. The use cases are written in form of text to represent users points of view for a specific system. The UML (unified modeling language) is used as a major tool to represent these test cases in form of figures Bell (2005). Actor acts as a separate model element that use to represent a user entity or system that play the role in whole scenario. The use case is also considered as scenario which is created by requirements engineer or the person who is responsible to carry out the tasks to get understanding about the whole proposed system Nuseibeh and Easterbrook (2000).

Classification of requirements

The requirements are classified into two major types as given in its relevant sector of requirements engineering spiral model as shown in Figure 2.8. Sommerville (2016) pointed out a vital issue which could rise during requirement engineering process due to not having proper isolation between different levels of requirements descriptions. The following issue is resolved through classification of requirements into two categories which are as follow.

a) User requirements
The user requirements are the requirements which are described in common/natural language. The user requirements are usually represented in form of figures which describe about what outcome system will give and constraints which it must be considered. User requirements are abstractions which are initiated by customers or end users rights at the start of requirements engineering Sommerville (2007).

b) System requirements
The system requirements are detailed and thorough explanation of what the proposed design system expect to deliver in terms of functions, services and operational constraints. The system specification document is a precise document which depicts true functionality of a designed system. Sommerville (2007) highlighted that this document may be considered as a part of contract between system buyers/customers and developers. Basically system requirements are considered as explanation of user requirements which are purely deal by SW engineers.

3. Requirements inspection (validation)

The third and final sector of requirements engineering spiral model is known as Requirements inspection (validation) as shown in Figure 2.8. The process of requirements inspection or validation is used to examine the requirements on which both system and
customers could not be compromised. In requirements documentation, the role of requirements validation is important because if something wrongly entered, it could lead system failure during its development phase or after deployment in operational environment. To ensure entirety and firmness in requirements, the validation is the best tool to utilize [Sommerville (2016), Kotonya and Sommerville (1998)].

The difference between requirements analysis and requirement validation process as highlighted by [Kotonya and Sommerville (1998)]. The requirements analysis is used to deal with ambiguous and unfinished requirements but on the other hand requirement validation is used to deal with firm and complete set of requirements right from the beginning.

Requirements inspection techniques

The techniques which are used for inspection of requirements are described by [Sommerville (2016)] are as follow.

a) Requirements reviews
The activity of requirements reviews is performed by group of reviewers in SW development team whose role is to check deficiencies and improper requirements [Sommerville (2016)].

b) Requirements testing/test-case generation
This activity is hinted towards the concept of TDD (test driven development). The test cases are written and executed already on basis of given requirements before actual coding is performed [Sommerville (2016)].

2.2.3 Requirements modification management

The modification or change in requirements happens in almost every system. The stakeholders (specially customers) are keen to have requirements modification management in their systems. This change has impact on almost all aspects of interdisciplinary environment (HW, SW, and company environment). The change control management is a way to manage these changes and see the level of difference created due to implementation of these changes in newly build system [Sommerville (2016)].

The article written by [Nuseibeh and Easterbrook (2000)] also highlighted about the need and features of requirements modification management in whole development environment of SW. The requirements modification management role comes into play when these changes need to be adjusted in requirements specification document. The tools and techniques which are used in this for configuration and version control purposes.

Traceability

The role of traceability is important to create links to examine and handle changes in requirements at different places of requirements documentation [Nuseibeh and Easterbrook (2000)]. The role of traceability as defined by [Sommerville (2016)] says that the traceability is set of policies which are used to create relationship between each requirements and the requirements. [Pfleeger and Atlee (2006)] defined characteristics of traceable requirements. According to their definition, the requirements are said to be traceable when they are organized and uniquely identified for reference purpose. The other feature traceable requirements is to make sure that all the entries from requirements definition are consistent with their connected entries in the requirements specification. [Pfleeger and Atlee (2006)] also pointed out the resemblance between verification and traceability say that; Verification is a kind of traceability process where the correlation between requirements specification document and defined requirements document is verified.
2.2. Requirements handling in SW engineering

In this way, the traceability or detectability between requirements specification and defined requirements documents is get ensured.

Gotel and Finkelstein (1994) who explored requirements traceability (RT) thoroughly sated that "Requirements traceability refers to the ability to describe and follow the life of a requirement, in both a forwards and backwards direction (i.e., from its origins, through its development and specification, to its subsequent deployment and use, and through all periods of on-going refinement and iteration in any of these phases)."

Pfleeger and Atlee (2006) further stated that traceability provides links between SW development entities like requirements, specification, design, implementation, and verification. As mentioned earlier, that the correlation between defined list of requirements (according with customer's view) and requirements specification document (accordance with developer's view) must need to be established. In this regard, the process management need to be implemented right from the beginning to until completion of the life cycle. Basically, The role of process management as mentioned by Pfleeger and Atlee (2006) is important in this regard which issues links. The links are acted as binders to connect the system development entities. According to the citePfleeger2006, if these links are not established then there is no other alternative to create test cases to check whether code implemented the requirements properly or not. A specific point which need to be remembered about traceability is mentioned by Nuseibeh and Easterbrook (2000), says that traceability links are only helpful to see the possible influence of change rather than automated reasoning about change. The reason which has been mentioned by Nuseibeh and Easterbrook (2000) is that links are carried small amount of significant information.

Requirements modification management process

The requirements modification management process is presented by Sommerville (2016). Basically, it is a 3-steps process which is useful for change control board (CCB) to plan and manage requirements modification policy in an organized way. The importance of requirements modification management process is highlighted by Nuseibeh and Easterbrook (2000) says that, this process not only concerns with managing documentation but also involved from the beginning of SW development project when requirements elicitation and risk estimation is performed. The modification process also contributes in system analysis while it is in operational mode. The CCB is the group of people where each member represents one department (development, documentation, testing, maintenance, and release). These departments are involved in whole development process. There are two main roles of CCB. First, it ensures that the major change is observed by all concerned stakeholders and second every change is approved before implementation starts Humphrey (1990).

How modification management process works?

The basic idea behind the representation of Figure 2.10 is to show how requirements modification process is carried out. The modification management starts right after requirements documentation has been accepted by stakeholders. The major benefit of using standard process to perform this modification management process in an organized and concise way. The
2. **Theoretical Framework**

The author of this modification management process [Sommerville (2016)] has defined the steps, which are involved in requirements modification management process. The rest of the discussion in this regard is originated from [Sommerville (2016)].

The whole process as given in Figure 2.10 starts from identification of problem (specific change in this case). The role of *problem analysis and change specification* is to analyzed the change proposal to verify its validity. The analysis report is returned back to change originator to get more specific proposal of change in return. If the request is accepted than it is forwarded to next phase for further analysis and costing otherwise it is pulled out.

The *change analysis and costing* is the way to examine the impact of the proposed change through traceability and knowledge of system requirements. The cost estimation of modification is performed in all steps (documentation, design, implementation, etc.), when this task is completed, the CCB can take final decision whether to accept or reject the proposed changes in requirements.

The final stage is responsible for modification implementation where it is required, the modification is usually done in requirements documentation, system design, and implementation phases.

### 2.2.4 Summary of requirement handling in SW engineering process

After the study of both development fields (HW by [Johansson (2017)], SW), it has noticed that SW engineering has a vast field in terms of development processes. That is the reason why brief overview has been given regarding how SW engineering is done. The SW development processes (plan driven, Agile) are discussed to give an overview about their working methodologies. The SW development models (the waterfall model, V-model, and Boehm’s spiral model) are discussed to elaborate how SW development is performed under different process models. The agile development process has been discussed in detail due to its large domain of users. The requirements handling in SW engineering is discussed under four standard steps - feasibility study, requirements elicitation (gathering) & analysis, requirements specification, and ends at validation of requirements. A requirements engineering spiral model is presented to help SW development teams to produce SW in a successful and organized way.

### 2.3 System engineering

The system engineering is a collaborative approach between two different disciplines or interdisciplinary environment, it combines right technologies and use principles for management in a coordinated way [Blanchard and Fabrycky (2005)]. The system engineering is not similar with other traditional disciplines like electrical, mechanical, and civil engineering where organization and implementation of engineering process require much resources. [Stevens et al. (1998)] defined system engineering as a way to design solid solution to tackle complexity of product development in an organized way.

[Blanchard and Fabrycky (2005)] discussed principles and objectives of system engineering, which are treated as general guidelines. The way of implementation of these principals can be differ from one proposed system to another.

### 2.3.1 Role of engineers and focused areas in system engineering

The role of system engineers and the areas which need to be focused during system engineering process is highlighted by [Stevens et al. (1998)]. The system engineers provide framework to other interdisciplinary field without compromising on their independent existence. The another factor which system engineers need to be aware of are technical aspects of a system, user requirements and good negotiation skills. [Blanchard and Fabrycky (2005)] emphasized
that the focus of system engineers need to be put on early design and development stages. These early stages is the phase when effective utilization of output from each development step can lead engineers towards development of functionally sound and reliable system.

2.3.2 Focused areas in system engineering process

The four areas on which system engineering process is mainly focused are mentioned by Blanchard and Fabrycky (2005) are as follow.

1. Top down approach
   The top down approach takes the system as a whole. The whole system is a combination of elements which play their roles during different phases of system development. In top-down approach, system engineers look towards overall performance of the system’s components to make ensure that their produced result is according to the plan. The other main focus of top-down approach is the utilization of system components in the right direction.

2. The paradigm of system engineering life cycle
   The role of life-cycle paradigm in system engineering is vital. The life cycle paradigm consists of all steps which are followed right from start till end of the system engineering process. These phases are known as design and development, production/construction, distribution, operation, support & maintenance, retirement, phase-out and finally ends at disposal. These steps are mainly focused upon the actors who are involved in whole system engineering process.

3. How to define system requirements
   There are two key factors emphasized by Blanchard and Fabrycky (2005) that need to be considered regarding system requirements. The first factor does emphasize that requirements need to be well defined and specified accordingly. The second factor emphasizes on having traceability between requirements. The traceability need to be implemented in a way that the requirements at system level are easily visible towards both direction top to bottom & vice versa, These factors can be achieved through creation of effective relationship between requirements and design criteria. The keeping track of efficient analysis leads towards getting capability of taking decision in time during design process.

   Pugh (1990) supplies his readers with a set of guidelines on the product design specifications - PDS: they are to be used as a control, need to be clear and brief, user-friendly and sparse, leave room for edits, and have at least an estimated quantified parameter. According to Pugh (1990) each specification should start from a different viewpoint to increase flexibility, the document should be dated and additions should be clearly documented. Pugh (1990) states in his book on total design that if a good reason occurs, even if it is during the design process, the basic PDS should be changed. He further goes on to claim that the product design specifications is a living document, which evolves with the project, so that upon completion it matches the final product.

4. Interdisciplinary teamwork/collaboration
   Blanchard and Fabrycky (2005) defined collaboration in interdisciplinary working environment as a approach, which is helpful to provide reliable platform. The platform is helpful to keep check on design requirements throughout the whole design and development process. This exercise requires in depth knowledge about other multiple design fields and mutual understanding. It also provides all facilitation through selection of right methods, techniques and tools which can become helpful for the implementation of system engineering process effectively.
2.3.3 System engineering life cycle activities

The activities which are involved in system engineering life cycle paradigm are as follow.

**Figure 2.11: Activities of system acquisition process and interaction with life cycle**

*Inspired by Blanchard and Fabrycky (2005)*

**Formulation of life-cycle**

The theme of the Figure 2.11 is originated from Blanchard and Fabrycky (2005) book. Blanchard and Fabrycky (2005) presented the whole work flow to describe how system engineering process is carried out into different phases of system engineering life cycle. The Figure 2.11 is acted as a conceptual model to understand whole system engineering process. It can be taken as general guideline about the right formation of steps which are need to be followed in system engineering life cycle.

The Figure 2.11 is divided into levels and sub-levels. The main level consists of four basic phases where the rectangles at the top with underlined text are contained sub-levels (highlighted by gray color). The phases which are mentioned in main level are considered as major building blocks to represent how product life cycle generally works. The sub-levels are used to describe which and how activities are carried out inside each phase.

Pugh (1990) has a different alternative process, for creating the design core in total design. Starting with collecting information from the market a specification is created. From this concept design is developed and then detailed. Once the detailed design is created the product is manufactured and sold.
2.3. System engineering

Classification of phases

Blanchard and Fabrycky (2005) divided the life cycle phases into two parts to clarify that which phase is used for what purpose.

1. Acquisition part

The first three phases conceptual/preliminary design, in-depth design & development, and production/construction of system engineering life cycle are considered as acquisition part.

2. Utilization part

The last phase (usage, support, phase-out, and disposal of product) is considered as utilization part.

As the research of this thesis is based on requirement engineering i.e. why the discussion has been made about the role of acquisition part. The acquisition part is the phase where requirements handling is initiated and processed according to the need of each phase. The detailed information which has been given inside the description of below given phases is extracted through the book of Blanchard and Fabrycky (2005).

Phase-1: Conceptual/ Preliminary design

The first phase is a combination of two sub-phases which have different modes of operation. The conceptual design phase deals with system level activities and preliminary design phase deals with sub-system level activities.

1. Conceptual Design

The initial identification of requirements regarding the system are treated as input in conceptual design phase. The role of concept stage is to explore new opportunities in terms of business and develop introductory system requirements and acceptable design solution Haskins (2011).

Pugh’s (1990) described about total design strives to create a PDS, starting with a brief, followed by a literature search which then branches into several subareas such as patents, reports, and statistical data, which includes questionnaires, along with many others. Similarly as mentioned in the general information on definition of specifications and requirements Pugh (1990) includes the needs of the user. According to Pugh (1990) identifying and understanding user needs requires exploration of several avenues related to the product area, and that there are long term gains in improving how user needs are dealt with as it will grant a competitive advantage.

The steps which are used during both phases of acquisition are performed in a repetitive manner. The detail of each step described by Blanchard and Fabrycky (2005) is listed below.

a) Requirements analysis

In requirement analysis - the needs or expectations related to the proposed system are identified and translated into organized requirements. The activities are performed during this phase are performed in predefined way.

b) Advanced system planning

The requirements which are explored on the basis of customers needs during first step, are further analyzed for advanced system planning during this phase.

c) Feasibility analysis

The feasibility analysis phase is used to perform feasibility study with ultimate goals. The feasibility analysis is also used to define a technical road map in terms of requirements to design the system.
2. Theoretical Framework

d) Operational Requirements
   The requirements which are purely related with operational environment are defined during operational requirements phase. These requirements are considered as functions, which system must need to provide without any compromise on its operational environment.

e) Maintenance and support
   The framework in terms of providing support and maintenance facility regarding proposed design system and planned system life cycle afterward is built here.

f) Performance measures
   The performance measures are taken in terms of performing identification and prioritization of goals along with defining their respective criteria.

g) System functional analysis and requirements allocation
   The analysis of functional level requirements of the system and allotment of these requirements to sub-systems level are done during this phase.

h) System analysis and trade-off
   The overall system analysis and study related to interdisciplinary environment is done by interdisciplinary team. This study is performed on the solutions, which are not highlighted or used in their development environment.

i) System specification and conceptual design
   The system specification is developed and proposed design is reviewed during system specification and conceptual design phase.

According to [Pugh, 1990] a long list of elements can be included in a PDS, 32 elements in total. In Table 2.4, 21 elements are listed, and these are the elements [Pugh, 1990] recommend including in a partial PDS. As seen in the Table 2.4, some elements go into detail on HW development such as material, size, and weight, whilst SW needs to be interpreted into these elements - ergonomics can of course also include the user interface of a SW, and safety can also be the safety of SW. But seeing as [Pugh, 1990] wrote this book in 1990, it is not certain that these factors were implied in the elements. [Pugh, 1990] says that, ideally, interviews should be structured around elements of a PDS - a Product Design Specification - making the analysis easier. If questions are derived from existing products instead of fundamental needs there will be fundamental gaps in the collected data.

<table>
<thead>
<tr>
<th>Aesthetics</th>
<th>Legal</th>
<th>Product cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company constraints</td>
<td>Life in service</td>
<td>Safety</td>
</tr>
<tr>
<td>Disposal</td>
<td>Maintenance</td>
<td>Shelf life storage</td>
</tr>
<tr>
<td>Documentation</td>
<td>Manufacturing facility</td>
<td>Shipping</td>
</tr>
<tr>
<td>Environment</td>
<td>Materials</td>
<td>Size</td>
</tr>
<tr>
<td>Ergonomics</td>
<td>Performance</td>
<td>Weight</td>
</tr>
<tr>
<td>Installation</td>
<td>Politics</td>
<td></td>
</tr>
</tbody>
</table>

2. Preliminary design

The initially defined system requirements which are originated from conceptual phase are handled by preliminary phase for detail evaluation at sub-system level. According to [Pugh, 1990] the PDS is to be used during idea generation and creation of initial solutions. This ensures that all solutions are attempting to solve the same problem and have
the same conditions. [Pugh (1990)] states that criteria used during evaluation must be based on detailed requirements of the PDS. This in turn means that the criteria need to be established before solution generation. [Pugh (1990)] also makes suggestions towards how the PDS can be used during individual idea generation, seeing as it is the basis for the criteria, and therefore inspiration in what problems need solving. The purpose of this phase is to present that the chosen conceptual system comply with performance and design specifications, It also confirm that the system can be developed with current methods under the defined cost and time constraints [Blanchard and Fabrycky (2005)].

The steps which are highlighted by [Blanchard and Fabrycky (2005)] as follow.

a) Design Requirements(sub-system level)

The development of design requirements which are originated from system-level to sub-system level is performed during this step. [Pugh (1990)] speaks of the Component Design Specification, or CDS, which comes after the Product Design Specification (PDS). In a complex product, detailed design is needed for the separate components included. The CDS aims to give context and specify performance of a component [Pugh (1990)]. A table of CDS elements can be found below (Table 2.5). According to Pugh (1990) Overlap is accepted in both the PDS and the

<table>
<thead>
<tr>
<th>Table 2.5: Component Design Specification elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aesthetics</td>
</tr>
<tr>
<td>Component cost</td>
</tr>
<tr>
<td>Environment</td>
</tr>
<tr>
<td>Local constraints</td>
</tr>
<tr>
<td>Maintenance</td>
</tr>
<tr>
<td>Manufacturing facility</td>
</tr>
<tr>
<td>Materials</td>
</tr>
<tr>
<td>Performance</td>
</tr>
</tbody>
</table>

CDS, omission however is not. A CDS is developed from the PDS, but it should still be checked so that it puts emphasis on local performance, environment, and constraints,

b) Preparation of Specification

The preparation of specifications in terms of development, product, process, and material which are appropriate to sub-system level are treated here.

c) Functional Analysis and Allocation

The main task in terms of performing functional analysis and allocation of requirements at sub-system level is achieved during functional analysis and allocation step.

d) Designed Requirements and Development Plans

The initiation of in-depth design requirements and development of plans in regard of allocation these plans to specific engineering tasks.

e) Identification and Utilization

The process of identification and utilization of proper engineering tools and technology for design is carried during identification and utilization phase.

f) Trade-off studies and Review of Design

The in depth trade-off studies in terms of providing effectiveness is done during this step. The design review also performed here at specified time slot.

This is all what preliminary design phase achieved on sub-system engineering level.
Phase-2: In-depth Design and Development

The detailed design and development part in the system life cycle is the continuity of iteration during system development process as presented in Figure 2.11. The role of design and development phase is to develop a system that fulfills the customer requirements and can be developed, verified, evaluated, operated, supported, and finally got retired as described by Haskins (2011).

The two baselines which are mentioned by Blanchard and Fabrycky (2005) are derived as output from above discussed phases. These two baselines are known as functional baseline and allocated baseline. These lines are originated from conceptual and preliminary system design phase respectively Blanchard and Fabrycky (2005). The steps which are involved during this phase are as follow.

1. Design Requirements
   The design requirements are used to describe subsystem, units, lower level components, and SW modules which are developed in this step.

2. Technical Activities
   The implementation regarding core technical activities, which are need to be achieved the design goals are carried while performing technical activities.

3. Integration
   The process of integration is performed on elements and activities during integration step.

4. Utilization of tools/aids
   The effective use of design tools and other means which can be helpful to ease the development tasks are exercised here.

5. Design and Documentation
   The process of making data regarding design and documentation is done during design and documentation step.

6. Physical Engineering and Prototyping
   The development of engineering and related prototyping models are need to be performed during this step.

7. Miscellaneous Tasks
   The tasks which are related with implementation of design reviews, evaluation, and providing feedback feature along with combining design changes when needed are performed during this step.

Phase-3: Production/ Construction

The production/ construction is the last part of acquisition phase which completes the acquisition process of the system. There are two major roles performed by production/ construction phase. First, it does system assessment through analysis & evaluation which also gives assurance to the development engineers that modification is done according the plan or not. Second, it provides facility of modification which gives opportunity to take right steps towards improvement of product. These modification have different types which are highlighted by Blanchard and Fabrycky (2005) in Figure 2.11.

The whole journey of modification for improvement consists of 6-stages or steps which are as followed.
• The handling of modification starts from *proposed design modification(s)*, which are purely related with design constraints.

• In next step, the modifications which are proposed at first step are got combined during *synthesis of modifications* step.

• The modifications which are related to *prototyping* of proposed production model are performed here.

• After finishing the prototyping modification, the process of *testing and evaluation* is performed on selected production model.

• This step has core importance in production/ construction phase because all the modifications which have been done in previous steps are verified through *test & evaluation* here.

• The last step of this journey is used to perform *configuration* of reviewed list of items. The output of this configuration process is treated as feedback information for first step (proposed design modifications) of this phase.

This is what we achieved through acquisition part during system engineering life cycle paradigm which as explained by Blanchard and Fabrycky (2005).

### 2.3.4 System engineering in context of requirements handling

The article written by Grabler and Yang (2016) regarding system engineering explains how requirements handling is done during system engineering life cycle. The life cycle model in this regard, acted as a platform which can be used as a major development tool. It is used to confirm that the proposed system is developed according to the given requirements.

**Requirements driven system engineering approach**

The requirements are need to be purposeful in all stages of system engineering life cycle. So the requirements need to be result oriented at any stage where the control is present on specific point of time. This approach is valid specially during conceptual and development phases. It gives an opportunity to take right decisions to fulfill the requirements of later phases in efficient and cost effective way Grabler and Yang (2016).

The idea about the Figure 2.12 is originated from the article Grabler and Yang (2016), where requirements driven system engineering approach is presented on the basis of system development life cycle. The journey of this approach begins from first stage *concept and development phase*, which is responsible for developing conceptual design of proposed system. The output of this phase has a great influence on next stages which are connected with this first phase. The arrows from bottom to top direction are used to drive requirements which are gathered at first stage (concept and development).

• **Concept and development**

  The requirements of all kinds are handled at *concept and development* stage. The requirements which are handled at concept and development stage are gone through analysis, evaluation, and specification at every next stage Grabler and Yang (2016).

• **Production stage**

  In production stage, the requirements and their constraints (availability, technology, and resources) are used to get ensure that requirements are fulfilled or not. The role of system engineers here is to make a balance between solutions to organized requirements in terms of business, time, and budget constraints. These four are core elements to achieve general requirements for production stage during system development Grabler and Yang (2016).
2. Theoretical Framework

Figure 2.12: Requirements driven system engineering approach
Inspired by [Grabler and Yang (2016)]

- Operation

The role of operation step is to strengthen system’s services through system operational engineers. These engineers perform several tasks like operate the system, monitor performance of system operator and overall system itself [Haskins (2011)]. The operation step is a kind of management stage where system production is done by a set of requirements. The activities at operation stage have influence on design and planning. The role of operation step in system engineering is like a facilitator, who gives support in process of production of a system and its services [Grabler and Yang (2016)].

- Disposal

The basic theme behind disposal/recycle is described by [Haskins (2011)] says that it defines the way how the system will exclude from its operational environment and retired. [Blanchard and Fabrycky (2005)] mentioned two reasons why the need of disposability is required in system engineering. First, the ever growing absence of insufficient resources in selected areas requires much intention on conservation. The second reason is that to protect surroundings (environment). In context of system life cycle, there is some quantity of waste is present in production and some amount of items related to product, component, etc., need to be disposed off. That is the reason why some items are disposed during operational use of the system. Finally at the end, there are lot of items will not remain in use when the system is retired from inventory [Benjamin S. Blanchard (2016)].
2.3.5 Summary of requirement handling in systems engineering

The literature regarding system engineering and specifically in context of requirements engineering is limited, so the effort has been made to present valuable contents from limited literature sources about the topic. The major concept which has been tried to present precisely in system engineering life cycle is to discuss how system engineering is performed in complex development environment. In context of requirements handling, the requirements-driven system engineering approach is presented. The main theme behind the description of requirements-driven approach, is to highlight how system engineering played its role right from the point where requirements are defined and how it carried out till the final stage. The final stage is the stage, when designed product is handed over to the customer.
3 | Method

The method chapter mainly concerns the research methodology of the thesis, more specifically the methodology of qualitative interviewing. A theoretical framework was previously provided as introduction to the different engineering fields, and the areas were further explored with the focus requirement handling as can be read in the previous chapter. This further exploration was needed to answer the two first RQs sufficiently. ‘Qualitative interviewing’ was selected as main data collection methodology to provide the opportunity to collect qualitative information about specific topic. This happens in a real time environment, where both entities sit together and can share their thoughts and feelings effectively leading towards correct learning and understanding. This chapter is shared with Johansson (2017) thesis due to the use of same research methodology to evaluate the RQs. In this thesis interviews with professionals were carried out with experts from different technical backgrounds, working in different fields which all come in contact with requirement handling in some way. The interviews were conducted using a previously defined interview guide which contained questions created based on the main research questions, as answering these RQs was the ultimate goal of the thesis. As the interviews took place in different companies, it was possible to compare the collected data between companies before comparison with the theoretical framework. This was efficient for answering RQ1 & RQ2. RQ3-RQ6 were more directly linked to the interview questions and the intention was to find consensus in the statements from the different companies concerning these RQs.

3.1 Interview methodology

According to Kvale and Brinkmann (2009), an ‘interview’ is a type of conversation between two or more persons (interviewer and interviewee), on a specific or common agenda. A big part of the qualitative interviews is the qualitative analysis following the data collection. Gibbs (2007) speaks of the functions of qualitative analysis and begins by bringing up the finding of patterns and explanations. Gibbs (2007) speaks of induction and deduction as two contrasting logics of explanation. Induction is the justification of a general explanation based on the collection of big amounts of data from particular, but similar, circumstances. Deduction is the justification of a particular situation, based on general statements of the circumstances.

Alternative data collection methodologies

van Boeijen et al. (2014) uses a figure, see Figure 3.2 inspired by Sleeswijk Visser et al., 2005, to speak of interviews as a way to collect surface knowledge, i.e. what people say and think. To collect deeper knowledge such as what people actually do and what tools they use, observations are needed. To get the deepest understanding of what people know, feel, and dream, generative sessions should be used. Further Boijen et al. mentions four different types of knowledge spanning from explicit at surface level, observable and tacit when going deeper, and latent as the deepest type of knowledge.
Due to the time limitation and the fact that the thesis work was conducted from Linköping University and not in one of the companies, it was decided to perform interviews to collect information. Observations would have been interesting, but this was not possible to perform at this time.

**Creating and performing interviews**

[Rubin and Rubin](1995) claim that the design of qualitative interviews needs to be flexible, iterative, and continuous and not set in stone. Concerning flexibility as interviews are performed, new information might surface which requires further questioning. Being flexible also allows the interviewer to ask different interviewees different questions depending on their background and knowledge.

[van Boeijen et al.](2014) outlines a possible procedure for creating the interviews, starting
3.1. Interview methodology

with the creation of an interview guide. The interview guide should include a list of topics originating from the research questions in the project and the interview guide should be tested in a pilot interview. van Boeijen et al.’s (2014) second step is to interview the right people, for this van Boeijen et al. (2014) recommends interviewers to perform three to eight interviews. When the interview is performed it should take roughly an hour and preferably be recorded. Once interviews have been made, summaries or transcripts should be created and analyzed. Kvale and Brinkmann (2009) speaks of seven stages of an interview inquiry which have been further explained by Lindahl (2017) in a lecture given at Linköping’s University 31st January 2017. The stages, except for the final stage concerning reporting, will be explained in chapters 3.1.1-3.1.6, and can be seen in Figure 3.3.

Figure 3.3: Process of interview methodology
Shared figure with Johansson (2017)

Following the steps described in chapters 3.1.1 to 3.1.3, the theme of the interview was set, companies contacted, and a guide created. A pilot interview was performed to test the phrasing of interview questions. It also allowed the interviewers to make time estimations for the different sections of the interviews. These time estimates were used as guide during the conduction of the actual interviews. The pilot-interviewee had some experience from SW development in a company creating complex systems, but was not at the same level as experts which were encountered at companies later. This resulted in the time approximation being inaccurate, and too short for the actual interviews. In the thesis a total of twelve interviews were performed at four different companies, with a minimum of two interviews per visited company. Two of the interviews were spontaneous, interviewees being invited by other interviewees on the spot, two other of the interviews were performed simultaneously on the interviewees’ request. All interviews except one were audio recorded, and when possible the interviews were also recorded on video. After the interviews summaries were made. Audio and (video) recordings provided a way to check the notes taken during the interviews. Each interview was intended to take 1 hour but depending on the time provided by the company representatives the duration varied between 45 minutes and almost 2 hours (for the interview with two simultaneous interviewees). After the data had been collected, notes were clarified making use of recordings, and the collected data was analyzed using both theory-driven and data-driven codes. The coded data was summarized per each company, and a selected data is presented in the results chapter of this thesis. Further analysis of the data takes place in the discussion chapter.

3.1.1 Thematizing
Thematizing answers the questions "why?", "what?", and "how?". These answers clarify the purpose, which can be either exploratory or hypothesis-testing. It also clarifies the theme, which needs to be known so that relevant questions are asked in relation to it. According to Rubin and Rubin (1995) the theme is something set at the beginning of the project to help the researchers keep on target towards the intended learnings. What Rubin and Rubin (1995) refer to as designing seems to have changed name to "thematizing" over time. During thematizing, one needs to know which theories will be applied in the analysis so that the correct
data is collected in the interviews \cite{Kvale2009}. \cite{Flick2007} speaks of different research perspectives, placing semi-structured interviews in the "approach to subjective viewpoints". \cite{Flick2007} recommends theoretical coding and content analysis (among others) as interpretative method for this research perspective. \cite{Rubin1995} claims that the interviewer chooses the initial subject of research but this subject is changed by the interviewees input.

Thematization of thesis interviews

The interviews performed in this thesis were believed to be exploratory, as the perception was that the topic, integrated/interdisciplinary requirement engineering, was not frequently mentioned in any of the theoretical frameworks. Even requirement engineering, which it would be reasonable to believe covers interdisciplinary requirements, failed to lift interdisciplinary aspects. The theme was set to integrated/interdisciplinary requirement engineering and the focus of questions was set to requirements/specifications and integration between different development departments (for example HW and SW).

3.1.2 Designing

According to \cite{Kvale2009} the designing of the interview study goes deeper into the "how?". Before starting to interview, an overview is needed including an understanding of the interdependence between the different stages as well as the extent of the end result of the study. In the design stage it is determined how many interviews will be performed, where the amount according to Kvale and Brinkmann can vary between 5 and 25 subjects depending on different circumstances. \cite{Kvale2009} and \cite{vanBoeijen2014} recommend to stop making interviews when it is felt that yet another interview would not yield any further information additional to the one already collected. \cite{Rubin1995} refer this to a phenomena called theoretical saturation but they limit its applicability to when the interviewer has chosen an area of focus.

According to \cite{Rubin1995}, interviews should be designed for depth, detail, vividness, and nuance. An important thing that needs to be considered already during the design of the interviews according to \cite{Rubin1995} is how the different pieces of information are collected from different interviews as well as different sites, which can be put together after the interviews.

\cite{Rubin1995} emphasize the importance of the selection of interviewees, asking the question "With whom do you talk and why?". \cite{Rubin1995} mention three requirements for interviewees to fulfill: knowledgeable, willing to talk, and representative of all perspectives. On the selection of interviewees \cite{Flick2007} speaks of sampling stating that those who are of interest to qualitative researchers are the people who are genuinely concerned and also experienced with the topic of the study. He also mentions how the process is iterative and one may start with the interviewee one thinks of as most knowledgeable but then the sampling criteria might change and new interesting potential interviewees appear. According to \cite{vanBoeijen2014} the environment in which the interview is performed plays an important role. The atmosphere should be relaxed, and the interview should take place somewhere where there are no distractions.

Design of thesis interviews

4 different companies were approached for interviews. Each company was asked to provide at least 2 interviewees. To make it easier to summarize collected data from different sites it was decided to use the same guide in all interviews, simply excluding parts based on time limits and experience of the interviewees. The interviewees were selected by contact persons.
3.1. Interview methodology

at the different companies, after they had received a short description of the knowledge relevant to the project. The contact persons’ expertise was trusted in the selection of suitable interviewees for the research project. As Flick (2007) argues that interviewees should be concerned and experienced with the issue at hand, and the researchers’ perception was that most of the interviewees were, this way of selecting interviewees was considered as acceptable.

Rubin and Rubin (1995) put emphasis on the connection between interviewee and reason behind selection of interviewee. Even if there was no possibility for the interviewers to affect who were interviewed in this thesis, a list of roles was composed before the interviews took place:

SW Developers - The reason for speaking with SW developers is their connection to actors, including end customers, and their involvement in collecting needs, and planning a development environment.

HW Developers - The reason for speaking with HW developers is similar to that of the SW developers. In many companies they are responsible for developing HW upon which SW is run, connecting the two disciplines. As the research questions aimed to discover differences and similarities between fields it was important to speak with actors from both development departments.

Liaison Developers/members - Because of their role of middle man, bridging between SW and HW development teams. With their facilitating tasks the liaison developers were believed to be aware of obstacles as well as enablers for the interdisciplinary work at companies.

All interviews but one were performed in private conference rooms, aligning with the recommendation of van Boeijen et al. (2014).

3.1.3 Interviewing

According to Kvale and Brinkmann (2009), the beginning of the interview is crucial, being the time where the trust of the interviewee is earned through attentive listening, authentic interest, and respect for the interviewees’ answers. It is also important that the intentions of the interviewer are clear. The interview starts with a briefing, defining the situation and purpose of the interview and taking care of formalia such as the use of a sound recorder and if the subject has any initial questions. After the interview, a debriefing takes place, a possibility to get feedback on the interview’s structure and contents. Rubin and Rubin (1995) argue that interviews in many ways are simply guided conversations that share traits with regular conversations. Just as abrupt changes of topic are not always taken well in regular conversation, they should also be avoided during interviews.

**Creation of thesis interview script/guide**

In preparation of interviews an interview script or guide is created. This script provides the structure of the interview, which can be more or less tight. According to Kvale and Brinkmann (2009) the guide can have very different characteristics, it could either simply contain some topics, or contain a detailed questionnaire.

According to van Boeijen et al. (2014), the interview should start with general topics such as experience but it is also important to time the interview in such a way that all topics can be covered since the final topics usually are the most important ones. A more structured interview will result in easier conceptual structuring of the analysis. The analysis should be taken into consideration whilst formulating the interview questions, enabling quicker processing of answers, for example coding can be done during interviews allowing for clarification of answers so that they can be correctly categorized. Kvale and Brinkmann (2009).
Interview guide for thesis interviews

Making use of both the thesis research questions, and the themes determined during thematization of the thesis interview, the guide outline was drafted using the following Figure 3.4. After revision the following themes were decided upon to be covered in the interview:

**Requirement engineering**
- requirements elicitation
- requirements categorization
- requirements documentation
- requirements verification
- requirements modification

**Integration process**
- interdisciplinary development handling
- dependency between different development teams
- modification handling in interdisciplinary teams
- system reliability
- difficulties
- future expectations

These themes were the base for the initial guide for the interview, and were included in the letter sent out to companies along with the interview invitation. The letter can be found in appendix A.
Interview questions

According to Kvale and Brinkmann (2009) when formulating interview questions academic language should be avoided so that the interview questions remain easy to understand. Further they should also be kept short. Research questions, usually formulated using theoretical language, can be the base for several interview questions - preferably formulated using everyday language. The use of a more easygoing tone encourages for spontaneous answers from the interviewees. According to Kvale and Brinkmann (2009) to receive the fullest answers one must ask open questions. Standard questions do not necessary result in standard answers. Different subjects interpret the same question differently and answer vary as well. According to Kvale and Brinkmann (2009) there are several types of interview questions:

1. Introductory Questions
   For spontaneity
2. Follow-up Questions
   For further elaboration
3. Probing Questions
   Pursuit of answers
4. Specifying Questions
   Gain precise answers among general statements
5. Direct Questions
   Introduction of topics after spontaneous answers
6. Indirect Questions
   Asking about others’ or interviewee’s hidden opinions
7. Structuring Questions
   For keeping interview on track and within relevant topics
8. Silence
   Allow reflection
9. Interpreting Questions
   Rephrasing back to interviewee for clarification

When looking to Rubin and Rubin (1995), naturally the categorization of types of questions is slightly different. Rubin and Rubin (1995) specifies three types of questions: main questions, probes, and follow-up questions. The main questions are the questions which begin and guide conversation, probes are used to complete, clarify, or receive examples and evidence from the interviewee. Follow-up questions are used to get the interviewee to elaborate further on their given answers to main questions.

According to Kvale and Brinkmann (2009) there is no such thing as a correct follow-up question. Once the first question has been asked, it is up to the interviewer to be an attentive listener and make use of the interviewee’s answers to ask suitable and yielding follow-up questions during the interview. It is easier to ask good second questions if one is familiar with the topic Kvale and Brinkmann (2009).
**Interview questions of thesis interview**

An interview questionnaire was created early on and as the project carried on some of the questions were rephrased, and others removed completely as they did not result in constructive answers. During the interview, interviewers took turns asking questions, alternating based on the (A) and (B) indexation ahead of the different sections. This methodology was established after a pilot interview, to get a better presence from both interviewers in the interview. The time approximation for each section was based on the pilot interview and revised as the questionnaire changed design and was rebalanced. Looking to the different question types described by Kvale and Brinkmann (2009) in the previous section, A were asked before recording, when meeting interviewees in reception areas. Due to time constraint mostly G questions were asked, following the guide. Naturally also B, C, and D questions occurred spontaneously, to clarify answers. This was to create understanding when unclear answers were given. E & F were not used. Silence was often due to the interviewee taking a natural break or due to note-taking by the interviewers, and was not silent reflection as described for question type H. When phrasing questions, attempts were made to keep questions open for some interpretation, especially with the use of the word 'your'. This gave the interviewee the possibility to give both general, company-wide descriptions, as well as mention individual approaches if they felt this was relevant for the question. Even if open ended questions were recommended by Kvale and Brinkmann (2009), it was established early that time was of essence during the interview. This resulted in opening questions on topics being of 'yes/no' characteristic, and the follow-up questions (a, b, c, etc.) being open ended. This allowed interviewees to easily identify if questions or even sections of the questionnaire should be skipped based on the experience of the interviewee. The final questionnaire which was used in a majority of the interviews can be found in appendix B.

The original questionnaire can be seen in appendix C. As the interviews processed on questions R-3, R-10, and R-17 were removed. R-3 and R-10 as they did not contribute towards answering the research questions, and R-17 as it and R-18 were similar and R-18 was felt to be the better question of the two. Several of the follow-up questions were scraped including R-1 (a,b,d), R-6 (a), R-8 (b), R-13 (a,b), R-18 (a) in an attempt to shorten the interview time. The order of questions R-5, R-6, and R-7 was changed to help keep the interview on topic; Promoting subquestion R-7 (a) and R-5 (a) to become the main question for respective questions. Answers given to questions which were later removed, were included in the analysis only if they were relevant. Many questions however were removed on the basis that the early interviewees misunderstood them completely, rendering these answers of little use in the thesis.

Lindahl (2017) suggests the use of a matrix to see the connection between research questions and interview questions. This type of matrix for the thesis interviews can be seen in appendix D. As differences and similarities were believed to be discovered by asking representatives of different departments the same question, most of the interview questions covering requirement engineering were assumed to work towards answering the research questions about similarities and differences. Similarly most integration interview questions work towards answering research questions were believed to cover integration. A more detailed instruction for reading the table can be found along side it in the appendix.

### 3.1.4 Transcribing

According to Kvale and Brinkmann (2009) the action of transcription is to transform information from one form into another. When interviews are transcribed the data is seen as a "fixed truth" even if it actually is an interpretation by the transcriber. A transcription should be used to continuously unfold the meanings of what was said during the interview. According to Lindahl (2017) things can get lost in transcription, such as tone of voice and pauses which might imply attitudes or the need to take time and think before answering. According
3.1 Interview methodology

To Kvale and Brinkmann (2009) an audio recording compared to video recording misses body language, postures and gestures which can help in the evaluation of the truthfulness of the answer or if the topic is sensitive to the interviewee.

**Transcription of thesis interviews**

Due to the wish of several of the interviewed companies to remain anonymous, no direct transcript was created as this would be very time consuming. Instead the audio and video recordings were used to strengthen notes taken during interviews and aided the accuracy of the summaries.

### 3.1.5 Analyzing

Kvale and Brinkmann (2009) mention how important it is to think of the analysis already before the data is collected. That way the analysis process influences both preparations and execution of the interviews as well as the transcription. According to Kvale and Brinkmann (2009), in a dream scenario, the interview would be fully analyzed at the end of the interview itself. Depending on the goal and purpose of the study different types of analysis should be performed. If the purpose is explorative, analysis should be done with depth for each individual dataset, whilst comparative purposes requires repeating the analysis as systematically and similar as possible on different sets of collected data. Flick (2007) speaks of different types of comparison during analysis: the comparison within categories, within cases, and between cases. For the category, data can be compared between interviews. For the case, data is compared within the interview. Comparison between cases allows for comparison of interviews as wholes to each other, seeing how similar or different they are from one another. Flick (2007) mentions how it is important to step away from the individual cases and make general statements based on the collected data.

**Analysis of meaning**

According to Kvale and Brinkmann (2009) extensive sets of data from interviews can be structured using coding and condensation, providing researchers with an overview.

1. **Meaning coding**
   
   Coding involves assigning a text segment keywords to simplify later identification. Categorization on the other hand conceptualizes statements making them quantifiable. Codes can either be determined beforehand by the research or developed during reading of the collected data. Both Gibbs (2007) and Flick (2007) mention coding as a process which takes place after data has already been collected. DeCuir-Gunby et al. (2011) speak of three different types of coding: theory-driven, data-driven, or structural which means they are allowed to grow from a project’s research goals and questions. Gibbs (2007) refers to the theory-driven coding as “concept-driven” and makes no mention of structural coding. This can be connected to DeCuir-Gunby et al.’s (2011) statement, that theory-, and data-driven coding is the most common way of deriving codes. If using theory-driven coding, codes represent concepts discovered in the research literature or topics in the interview schedule. According to DeCuir-Gunby et al. (2011) theory-driven coding has three steps: Generating the code, reviewing and revising the code in relation to the data, and determining the reliability of both the coders and the code. Similarly DeCuir-Gunby et al. (2011) has steps for data-driven coding, starting with reducing the raw data and identifying subsample themes. Once this is done, themes are compared across subsamples. Then codes are created and their reliability is determined. Kvale and Brinkmann (2009) refers to Gibbs and claims that “In principle, anything can be coded” and proceeds to list examples including events, activities, norms, conditions or constraints, and settings.
3. Method

2. Meaning condensation
   Whilst keeping the main sense of interview responses, meaning condensation compresses collected data into briefer statements. To perform meaning condensation, the interview first needs to be read-through in full, and "meaning units" be determined by the researcher. After this, dominating or "central" themes for different parts of the interview should be put as simple as possible. Finally the "meaning units" need to be looked upon according to the study purpose, ensuring that all essential, non-redundant themes are included in a conclusive descriptive statement (Kvale and Brinkmann, 2009).

3. Meaning interpretation
   According to Kvale and Brinkmann (2009) meaning interpretation is about going beyond what the interviewee has said and managing to work out structures, and also relations, which are not understandable from the text alone. The same data can be interpreted in multiple ways without any of the interpretations being incorrect. New interpretations might occur over time when new insights are gained and one re-evaluates already evaluated data.

Meaning analysis of thesis interviews
In the thesis, a comparative study is made, comparing different companies approach with each other, and the theoretical framework. To analyze the data, coding was used, along with meaning condensation to discover central themes. The meaning coding along with condensation was a key input to summaries. Theory-driven code words were established in preparation of the interviews. These were based on the research questions, each of the topic areas defined in the guide section, and the different theoretical frameworks. Even if theory-driven codes were prepared, none of the interviews were coded live during the interview as neither of the interviewers had any experience in this. These codes were used retro-active together with data-driven codes which occurred during analysis. It was decided that additional data-driven code words were allowed to emerge once the data had been collected as tendencies not mentioned in theory could occur. The complete collection theory-driven codewords can be found in appendix E.
3.1. Interview methodology

Codes from research questions:
- Differences and similarities: limitations, special feature, relations
- Integration state: interdisciplinary modification methodology, dependency, interdisciplinary projects
- Future obstacles and possibilities: actors, directives (from management)

Codes from theoretical framework:
- HW development: steps involved (model), requirement handling techniques, relationship between requirements, prioritization
- SW development: steps (process)
- System engineering: No remaining code words after coding
- Requirement engineering: iterative, elicitation methods (interviews, questionnaires, documentation analysis), actors, requirement characteristics, requirement documentation, requirement validation

Codes from guide topic: Requirement engineering
- requirements elicitation: actors, elicitation technique, standards
- requirements categorization: categorization
- requirements documentation: documentation standard
- requirements verification: validation, continuous verification, verification methodology
- requirements modification: modification methodology

Codes from guide Topic: Integration process
- interdisciplinary development handling: interdisciplinary actors
- dependency between different development teams: dependency
- modification handling in interdisciplinary teams: communication, time frame, interdisciplinary modification methodology
- system reliability: obstacles
- difficulties: difficulties
- future expectations: obstacles future, enablers future, responsibility

Data-driven code words
- Verification: Relation between requirements & product, testing
- Requirement analysis: requirement analysis, balancing requirements, target setting
- Requirement characteristics: requirement types, legal factors/legislation
- Interdisciplinary: interactions, roles, traceability, understanding
- Other: customer interaction, future approaches, intercultural work, methodology, optimum (in a perfect world), strengths and weaknesses
Once coding had taken place, obsolete code-words were removed and data driven code words were added to the list which can be found above. All interviews were coded and summarized into company summaries. The code word summaries can be found in Appendices F, G, H, and I. These code summaries are referenced further in the result chapter, as sources of the results presented. The results which are in the thesis can be seen as an attempt at meaning condensation. The meaning units are represented by the different headings in the results chapter of this thesis, and the meaning condensation was performed from the code summaries instead of from each individual interview. The headings were reviewed after the first drafting to ensure that all, to the thesis relevant, themes possible were covered. Meaning interpretation can be seen in the discussion of this thesis, which is the only, and final evaluation of all collected data performed concerning meaning interpretation specifically. Other evaluation has taken place during the process of the interviews.

3.1.6 Validation of interviews

Validation should, according to Kvale and Brinkmann (2009), take place not only once the interviews have been performed and the data analyzed, but during the entire process from thematization to reporting. For each of the seven stages Kvale and Brinkmann (2009) have suggestions of how validation can be performed. During the thematizing, it is the theoretical presuppositions that need to be validated. During the design, validation should be made to ensure that the right methods relative the purpose of the study are being used. During interviewing, validation should be performed live with the interviewee, questioning that the correct meaning is understood as the data is being collected. During transcribing it is the linguistic style chosen that needs validation.

Verification during thesis work

In the thesis, there was a continuous verification of the design of the interviews by regular feedback from the supervisors. The theme and also choice of interview as data collection method was verified this way. The interview design and codewords were verified using a pilot interview. Making use of interviewees and interviews for verification the interview questions were continuously reformulated if other answer types than the expected were received and some interview questions were scratched when they did not add enough towards answering the research questions. At the end of the interview interviewees were asked for feedback on the interview. If any unclarities occurred during analysis, interviewees were reconnected with per email for clarification. Before publication, the companies involved have been allowed to review the thesis.
4 Results & Analysis

The results chapter begins to present interview results highlighting differences and similarities in the different companies’ requirement handling processes. A short intermission provides an overview of the separate theories on requirement handling presented in the theoretical framework, the interview questions which were presented in the previous chapter, and the interview answers concerning the companies’ current requirement handling processes. The results chapter then continues to present interview results concerning current and future integration, and future obstacles and enablers for increasingly integrated development.

4.1 RQ 1 & RQ 2 - Differences and similarities between different development processes’ requirement handling

In this chapter, the interview data from interviewees at four different companies regarding their requirement handling process is presented. Following this chapter comes a comparative chapter which shows similarities and differences in interview data and the previously presented theoretical framework.

4.1.1 Actors

In comparative Table 4.1, the actors mentioned by the different companies can be seen. Some actors have different names but share the same function in the different companies. The customer unit at company W has a similar function as the product manager at company Y, and the market department at company Z. Meaning that in some companies, one single person takes on the job done by an entire department in other companies. At company X, research and development is represented by product and function owners. At company Y R&D was mentioned as a combined function, but in the table it has been registered as research and product development as to avoid adding an additional item. For the full lists of actors, see appendices [F.1] [G.1] [H.1] and [I.1].

4.1.2 Techniques for elicitation

The interviewed companies have different actors who belong to different departments, involved in, or responsible for, requirements collection. The scope of project plays a key role for requirements collection in company Z (Appendix [I.2]). In company W, on the other hand, they used communication as major tool to collect requirements from different stakeholders and actors (Appendix [F.5]). Customer investigation is the major tool used by company X to collect requirements (Appendix [G.5]). The internal requirements originate from experience or newly developed technology in company Y. The actors (product owners and service developers) contribute to requirements collection, and interviews are key tool to the collection of requirements from customers. Company Y also uses the concept of reuseability in their requirement collection methodology (Appendix [H.5]).
4. RESULTS & ANALYSIS

Table 4.1: Table of actors mentioned by companies W-Z

<table>
<thead>
<tr>
<th>Actor</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brand</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business development</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Customer unit</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Function owner</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Legal department</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Market</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Researchers</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quality</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parts</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Product developer</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product management team</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product manager</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product owner</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product planning</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Production</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sales</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Special product</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Supply</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>System manager</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Testers</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reuse of knowledge**

The concept of reuse of knowledge is a recurring concept. At company W requirements are collected through communication but they are mostly based on experience (Appendix F.5). At company X they use a platform way of thinking, building things on a shared base, and the reuse knowledge is also established (Appendix G.3). At company Y, internal requirements originate either from experience, or newly developed technology (Appendix H.5). They also use the concept of reuseability in their requirement collection methodology (Appendix H.5). In company Z, old requirements can be reused in new projects (Appendix I.5). A common factor for all companies is that several products can be similar, or related to one-another, allowing for this type of reuse.

4.1.3 Overall processes at companies

It is possible to see similarities between the different companies’ processes, even if the companies in question work with very different main areas for their respective complex products. At company W, during the requirement process there are certain checkpoints, including a preliminary review (Appendix F.2). At company X instead there are gates during the process. At company X they also work with “handshake”, and only things previously "OK:ed" should be "handshaked upon" (Appendix G.2). Also company Y works with gates, and it is a gate-group who makes official decisions of changes at these gates (Appendix H.10). In company Z, tollgates are used during the early development to approve changes. The gate “committee” here consists of the technical manager, and project manager (Appendix I.10).
Agile development

There is only one company W which follows an agile way of development (Appendix F). The other three companies (X,Y,Z) follow a partially agile approach in their development processes Appendices G, H, I). The one thing which has been found mostly common in all interviewed companies’ development environments is that, they are keen to adopt agile development methodology in their development processes.

4.1.4 Requirement handling processes

Below, in Figures 4.1-4.4 the requirement processes for the four interviewed companies are presented in a much simplified manner. These processes figures are based on steps of the different companies’ requirement handling processes which can be found in Appendices F.2, G.2, H.2 and I.2 respectively.

Figure 4.1: Process of company W, mainly developing SW
Shared with Johansson (2017)

Figure 4.2: Process of company X, mainly developing HW and electronics
Shared with Johansson (2017)
Customer interaction

The view on who the customer is different in the different companies. All companies acknowledge that the end-user is one type of customer, but in company Y it is differentiated between the end-user, and suppliers as customers. Similarly customers of company W might not always mean end-users. Looking to the process, in Figure 4.1 which represents company W’s process, customer interaction goes solely through the customer unit. In the company X’s process, illustrated by Figure 4.2 customers act as input to general requirements but the
development department does not have any direct interaction with them. In company Y’s process, showed in Figure 4.3, customers might be included in the prestudy, but there is no guarantee. If looking to company Z’s process, described in Figure 4.4, the market department is meant to represent the customer, as they speak to the sales organization and visit the customer to collect requirements (Appendix I.5). Company W sees the possible benefits of having a closer collaboration between customers and developers (Appendix F.5). Company X and company Z mention how their customers are very different, or in different regions, making generalizations concerning their needs so difficult (Appendices G.17, I.5).

Breakdown of requirements

In the three out of the four processes, a clear breakdown of requirements takes place. Usually this breakdown is from main requirements, or general requirements, to more detailed requirements. In the process of company Y, Figure 4.3, the levels of requirements change between project proposal, PRS, and product statement.

Roles of departments

Different departments have different roles in the process. One department usually is in charge of giving more general requirements, or guidelines. This is product planning in company W & X (Figures 4.1 and 4.2), senior management and business development in company Y (Figure 4.3), and the market department in company Z (Figure 4.4). Remaining departments are allowed to contribute requirements relevant to them in the processes of companies X and Z (as described in Figure 4.2 and 4.4). In company Y’s process (described in Figure 4.3), R&D are left to refine the requirements. In company W there is no mention of the role of the remaining departments in the requirement handling process.

4.1.5 Time-frame

According to the interviewees at company W, in SW; time lines and customer needs are framed by the project time limitation, so the project determines how much time can be given to customer needs and do separate projects within the project (Appendix F.2). Company Z explains that SW development spends a lot of its time verifying, and has more verification steps and efforts than HW development (Appendix I.16). According to the company Y, in big HW development projects, time can be spent on categorization and prioritization of requirements but on the other hand in service development projects, which have less time and the time is given only for basic identification of key requirements (Appendix H.7). In this time comparison company X agrees, claiming that service development usually has shorter time frames than HW development (Appendix G.2). Company X proceeds to claim that it takes a long time before everything can be implemented in HW development, requiring prioritization (Appendix G.18).

4.1.6 Prioritization

As the requirements go through different levels, they are prioritized differently. An example of this is that product managers prioritize among requirements, and then hand them over to system managers who have other priorities (Appendix F.6). When working between departments and levels in this way, it is necessary to have a common understanding in order to make a correct prioritization (Appendix F.6). Company Y agrees with company W, and adds the layer of different nationalities’ ethics in a multicultural company to the prioritization dilemma. Prioritization is different depending not only on department but also depending on country (Appendix H.6). In company Z prioritization is only made in the form of a 1-3 ranking when collecting requirements, where 1 equals a “must” requirement, and a 3 is
4. Results & Analysis

a requirement which would be good to fulfill. The ranking is only relevant in the step from market/customer. After requirements are accepted into the PRS, they no longer have any ranks (Appendix I.6).

4.1.7 Documentation standards

When asked about documentation standards, none of the companies referred to external standards, instead all companies turned out to have internal standardization of requirement documentation. In company W, the requirement documents are internally standardized, and the organizational system/tool is used across the whole company (Appendix F.8). In company X, the requirements are written in the same way, which means the requirement documentation is unified. It is however stored and used in different ways depending on department (Appendix G.8). In company Y, the requirement specification base is shared globally within the company (Appendix H.8). In company Z, the company has an internal standard for documenting requirements. All requirements utilize the same type of document regardless from their origin (Appendix I.8). Company X comments that requirements specification are not currently universally understandable (Appendix G.16).

Different documents used for different purposes

There are different standards used for requirements documentation in interviewed companies:

- CRS (characteristics requirements specification) - is used as internal standard by company W as a container for requirements in a specific area (Appendix F.6).
- Unified documentation standard - is used by company X as their base document (Appendix G.8).
- PRS (product requirements specification) - is used by companies Y & Z as their standard requirements specification document on top level (Appendices H.8, I.8).
- MRS (market requirements specification) - is used by company Z, it is used as base document for storage of other types of requirements (Appendix I.8).
- FRS (functional requirements specification) - is used by company Z in their development and testing departments at SW side (Appendix I.2).
- IRS (Interface requirements specification) - is used by company Z for making relation strong between two departments where one department acts as sender of requirements and the other is at receiving end.

4.1.8 Requirement types and characteristics

When asked about requirement characteristics and types, the companies provided very different types of answers. Therefore, as can be seen in Table 4.2 very few of the characteristics and types are shared between companies. At company W, very general characteristics were presented, such as external, internal, functional and non-functional. Here, performance was mentioned as an example of non-functional requirements. Company Y managed to generalize their requirements to be either technical or market related. Both companies X and Z went into detail, focusing both on departments and characteristics as described in the theoretical results. Company Z also included the level of formality in requirement characteristics.
### Table 4.2: Different requirement types and characteristics according to companies W-Z

<table>
<thead>
<tr>
<th>Characteristic or type</th>
<th>W</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>External</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internal</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functional</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-functional</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>HW</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SW</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reliability</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Technical</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Market related</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Fuzzy</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Formalized</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Performance</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dimensions</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Service</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Feeling of product</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Norm testing</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Usability</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Robustness</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

#### 4.1.9 Interdepartmental dependency

The level of dependency inside the interviewed companies solely depends upon the role of each department inside product development process. There is always some dependency between different departments in all of the interviewed companies. The types of requirements which are handled by different departments are also define the level of dependency required at each department. The dependency exists in company W between different development teams. Once development is started, they are located in different geographical locations, so the interaction between each department defined how they tackle dependency between each other (Appendix F.15). The different locations of production departments also matters in context of interdepartmental dependency at interviewed companies. In company X, all different development departments are dependent on each other (Appendix G.15). At company Z, the perception is that the dependency between different development departments is very high during the actual development. In design department at company Z, they find more dependency than others (Appendix I.15).

#### 4.1.10 Input requirements

Requirements must begin somewhere, and for the companies studied, there is one thing in common for the input requirements, and that is the abstraction level. When requirements arrive at product management in company W, they are at a high level, formulated as a statement, which is generally very broad (Appendix F.2). At company X, the initial general requirements for technology are set based on input from customers (Appendix G.2). At company Y, that is the senior management which contributes towards initial high level requirements (Appendix H.2). At company Z, the market department investigates what the customers want and output a set of fuzzy requirements in the market requirement specifi-
4. Results & Analysis

Additionally at company Z, requirements at a high level are collected using a form which is sent out to each stakeholder to fill in (Appendix I.2).

4.1.11 Modifications

Modification handling is different in interviewed companies. In company W, modifications require change requests. Change requests are processed regardless if change takes place or not (Appendix F.10). In company X, design review forums/meetings are used, here the concepts regarding cross functional requirements are discussed and design concepts are reviewed. At these forums, it is decided if a change of function is acceptable or not (Appendix C.10). In company Y, the new requirement is taken into the project and evaluated in terms of time, cost, and effects if added by a gate-team who then make an official decision about change (Appendix H.10). At company Z, the reason for the change request needs to be clearly stated and documented. Similarly as in company Y, time, cost, and technological restrictions are evaluated when deciding whether to perform changes or not, and the decision is made at gates (Appendix I.10).

Looking to late modifications, according to company Y, some modifications are necessary. This can be caused due to incorrect conclusions in the prestudy. If this is the case, modifications need to be made to avoid ending up with a product nobody wants (Appendix H.10). Similarly company Z claims that some changes are necessary, as some problems simply must be solved. Changes late in the project however are more restricted, meant only to solve problems, not to add functionality (Appendix I.10).

4.1.12 Traceability

The linkage to the original requirement givers is mixed over the companies. Company W is openly against it being possible to trace this (Appendix F.9), they refer to the integrity of the original requirement giver. Company X claims that the original requirement givers are stated in the requirement specification (Appendix G.9). Company Y implies by saying that there is a discussion with the actors who originally stated the requirements, that these actors are available (Appendix H.12). In company Z there is no 100% traceability between the requirement givers and the specification but all actors within the company have a personal contact (Appendix I.9). Some parts have better traceability, in company Z for example, the link between the fuzzy MRS and the PRS is very clear (Appendix I.9). In company Y the interviewees perceived the traceability as good (Appendix I.9). Prestudies allow them to map and describe the underlying reasons for requirements (Appendix H.8). There however were also weaknesses which came to light concerning traceability. In company W, there is lack of overview of the connection between requirements on different levels (Appendix F.9) was mentioned. For company X, it was explained that traceability is easier looking bottom-up because when looking down from the top level, traceability is lost at some point. They continued to stress however, that at this point, someone has the responsibility of fulfilling the requirement, so a more detailed traceability is not needed (Appendix G.9).

Differentiation in traceability

The levels of traceability is fluctuated from company to company, which solely depends upon their working criteria in development environment. The company W is not allowed every actor to get involved in traceability due to keep secrecy as much as possible. The reason mentioned by them is that the involvement of lot of several persons means lot of risk (Appendix F.9). The company X has traceability from bottom to top during whole development process but from top to bottom they are lacked of it (Appendix C.9). The company Y has traceability which is created solid link between business or customer needs and gets final product design as a result where they have clear visibility (Appendix H.9).
4.1. Validation

Validation, or the control of technical requirements towards the originally stated needs by actors is seen differently at the different companies. At company W, no interview data could be coded as validation. At company X, they collect the perception of the product at different times. Company X also has a so-called internal customer, which tries to validate the results of development. The internal customer represents the (external) customer, and does not work directly with requirements but with the feeling and the perceived product (Appendix G.12). In company Y there is a discussion with the actors who originally stated the requirements. Company Y also uses prototypes and limited releases to validate the product. Validation over time is performed to find unanticipated occurrences of requirements (Appendix H.12). At company Z, the opinion is that all requirements in the PRS are verifiable, so if the PRS is tested, it is assumed that the customer need is correctly reflected. For customer opinion, company Z has focus persons/groups who can make a judgment, but the case can also be that the design team made the judgment based on experience (Appendix I.12).

4.1.14 Safety and regulations

In two of the companies, a clear connection to laws and regulations in the requirement process was presented. At company X, the solution is that mandatory standards are handled by standard management. The standards are usually focused on implementation and safety of systems (Appendix G.8). On the topic of safety at company Z, it was outspoken that safety norms affect related functions (Appendix I.8).

4.1.15 Testing and verification

The verification and testing of requirements are done through different methodologies in interviewed companies. The company W is performed verification by system verification department who used CRS to verify robustness and stability features in requirements. They mentioned that they use TDD (test driven development) approach in their testing phase (Appendix F.11), but as they have separate testing phase than the use of TDD looks vague. In company X, there is a person in charge to do verification which is performed in form of virtual and physical verification. At company X, it depends upon different stages of product development process. At the end, the verified prototype is created as output of that process (Appendix G.11). The testing activity is performed at company X by system responsible actors, their testing techniques are based on types of requirements (Appendix G.11). In company Y, product design is used to specify what can be delivered to end customer which is compared through verification with actual requirements to inspect whether they achieved actual requirements related to proposed product or not. The simulation is used as a tool to check the design in company Y (Appendix H.11). In company Z, the verification is performed for same purpose like in company Y to verify functional requirements of their product. They are also used this methodology for detecting conflicting requirements. The PRS and FRS documents are used as their main sources for verification. The testing is performed here only on high valued requirements through their internal testing techniques (Appendix I.11).

Continuous testing

The testing process is performed on continuous basis in majority of the interviewed companies W, Y, & Z (Appendices F.11-H.11-I.11). Except company X where they are performed testing right at the time when they feel that the appropriate functions are offered by the designed product (Appendix G.11). The companies which are focused on continuous testing are performed or wished to be performed testing right from the beginning of project development activities.
4.2 Link between different theoretical domains, interview questions, and answers

After exploring the different theoretical frameworks for HW development (presented in Johansson (2017) thesis), SW development, and requirement engineering in SW development. It was possible to make an initial comparison between models, processes, and concepts. Interview questions were linked to the collected data and finally also interview data could be added. As the thesis concerns requirement engineering it was decided to make the comparison based on the key steps described; elicitation, analysis, documentation, modification, validation, and verification. Use of requirements in development was added as this was relevant in the different development fields.

4.2.1 Requirement elicitation

In SW engineering prospective, the commonalities are found while performing requirements engineering process during exploration of actors as described by Sommerville (2016) and Pandey et al. (2010). The requirements gathering (elicitation & analysis) process which is in detail described by Bell (2005) and explained in form of loop with steps in Figure 2.9 in theoretical framework chapter. The tools which are used to gather requirements in interviewed companies were interviews, observation which were highlighted by Nuseibeh and Easterbrook (2000) and Sommerville (2016) as well. According to the Johansson (2017) thesis, in HW development similarities with requirement engineering can be found in the identification of actors and their needs in the quality function deployment’s house of quality which is described by Hauser and Clausing (1988), Ullman (2002). According to Johansson (2017) thesis, requirement engineering elaborates further on different elicitation methods while HW, even if including "selection of data collection method" in its steps, then proceeds to imply that some type of questionnaire or interview is used for the data collection. In the interview, questions concerning important actors and elicitation methods were asked. Different companies include different actors in their requirement process which can be read in section 4.1.1. Requirements are collected, partly internally from different departments, often based on experience. Requirements are also collected from the market, service, or customer unit, who interacts with both existing users and potential customers (Appendices F.5, G.5, H.5, I.5). The Table 4.3 gives the information about how requirement elicitation is performed under the consideration of HW and SW engineering theoretical framework, requirement engineering, interview topics and questions, and their answers.

Table 4.3: Comparison of requirement elicitation

<table>
<thead>
<tr>
<th>Requirement elicitation</th>
<th>HW engineering</th>
<th>SW engineering</th>
<th>Requirement engineering</th>
<th>Interview topics and questions</th>
<th>Interview answers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Actors (mainly consumers), data collection methods (questionnaire)</td>
<td>Actors (multiple stakeholders, development team), collection methods (interviews, observations)</td>
<td>Actors, requirement elicitation methods</td>
<td>Important actors</td>
<td>Actors are listed in Table 4.1 mixed elicitation methods; ranging from own experience to requirements based on customer interaction.</td>
</tr>
</tbody>
</table>
4.2. Link between different theoretical domains, interview questions, and answers

4.2.2 Requirement analysis

In SW engineering, the analysis of requirements is helpful for handling conflicting requirements. There are different techniques like negotiation, agreement, communication, and prioritization of requirements as mentioned by Pandey et al. (2010). According to HW engineering as mentioned in Johansson (2017) thesis, where different types and ranking of requirements are defined. In the thesis of Johansson (2017), the analysis of HW requirements specifically looks to compare with the existing market, using market screening as mentioned by Douglas et al. (1978), Ullman (2002). HW engineering shares the use of requirement characteristics with requirement engineering Lamsweerde (2009), Roozenburg and Eekels (1995), Ullman (2002) as mentioned in Johansson (2017) thesis. From the interviews very different requirement types were given, see section 4.1.4. Some of the companies spoke of prioritization, several companies mentioned some type of review for solving conflicts. A requirement breakdown takes place in 3/4 of the interviewed companies, dividing requirements among departments. The techniques and approaches which are followed in HW, SW, and requirements engineering along with interview questions & answers are mentioned in Table 4.4. The basic reason behind this given Table 4.4 is to highlight the comparison of different requirement analysis approaches which are followed in both (HW, SW) engineering fields.

Table 4.4: Comparison of requirement analysis

<table>
<thead>
<tr>
<th>Requirement analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW engineering</td>
</tr>
<tr>
<td>SW engineering</td>
</tr>
<tr>
<td>Requirement engineering</td>
</tr>
<tr>
<td>Interview topics and questions</td>
</tr>
<tr>
<td>Interview answers</td>
</tr>
</tbody>
</table>

4.2.3 Requirements documentation

The requirements documentation is an important phase during requirements engineering process in SW engineering. The validation of requirements is required to keep requirements documentation complete, consistence, and realistic Sommerville (2011). The SRS (SW requirements specification) considers as standard document of requirements in this regard. In context of HW Johansson (2017) thesis mentioned about that the extracted theory covering HW development lacks information on preferred style for documentation of requirements. The only thing mentioned is that specifications are documented once they have been translated into engineering specifications, as they are called by Ullman (2002), and then they need to have a measure and a target value Ulrich and Eppinger (2012). In the theoretical framework covering requirement engineering in Johansson (2017) thesis, the requirement document can contain an abundance of posts, with constraints, relations, and the original source of requirements being the most frequently recurring in the literature explored from Hull et al. (2005), Kotonya and Sommerville (1998), Lamsweerde (2009), Sommerville and Sawyer (1997), Sutcliffe (2002). In the interview questions covered when to document, which standards the companies used in their requirement documentation and how requirements from other departments are handled. All companies answered that they apply a company internal documen-
The comparison of different requirements documentation or specification process steps which are followed in the different (HW, SW) disciplines can be seen in Table 4.5. The given Table 4.5 is also refereed as summary which gives information about requirement documentation and specification done under both (HW, SW) engineering fields.

**Table 4.5: Comparison of requirement documentation and specification**

<table>
<thead>
<tr>
<th>Requirement documentation/specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW engineering</td>
</tr>
<tr>
<td>Engineering specification with measure and target values</td>
</tr>
<tr>
<td>SW engineering</td>
</tr>
<tr>
<td>Defines functions &amp; services related to the proposed SW using SRS &amp; FRS, concise and realistic</td>
</tr>
<tr>
<td>Requirement engineering</td>
</tr>
<tr>
<td>Highlights constraints, relations, and the source of requirements</td>
</tr>
<tr>
<td>Interview topics and questions</td>
</tr>
<tr>
<td>When to document &amp; which standard used. Relation to the requirements of other departments</td>
</tr>
<tr>
<td>Interview answers</td>
</tr>
<tr>
<td>Internal documentation standard, some include special inter-department documentation. The understanding of other departments’ requirements depends on own knowledge and communication.</td>
</tr>
</tbody>
</table>

### 4.2.4 Requirement modification

In SW engineering context, the modification handling is truly based on selected development model. The interviewed companies development processes are followed traditional waterfall development model, which is based on plan-driven development process with lack of agility in modification handling. In context of HW engineering, [Johansson (2017)](Johansson, 2017) thesis mentioned that there is only type of modification men-

---

58
tioned for the requirements, which consists of natural evolution from target specifications to final specifications explained by [Johannesson et al.] (2013). The thesis by [Johannesson] (2017) further highlighted about requirement engineering focuses on formality of modification handling, emphasizing that early modifications should be easy to make, while later modifications need to be formal, questioning their implications on the project’s budget (money and time) [Hull et al.] (2005). Interview questions were asked concerning modification methodology and its handling, depends on at which time a modification was requested. Requirement modifications are handled in gate-processes in many of the companies. These companies evaluate time, cost, and other effects in a formal review. Late modifications are performed only if they are deemed critical, that is, if the product will fail to sell or similar, unless the change is made (see detail in section 4.1.11). The comparison of different requirement modification processes which are followed in HW, SW, and requirements engineering along with respective interview questions & answers are summarized in Table 4.6.

4.2.5 Requirement validation

In SW engineering, requirements validation is performed to ensure firmness and completeness in requirements [Kotonya and Sommerville] (1998). Requirements reviews, prototyping, and TDD (test driven development) are the techniques which are mentioned by [Sommerville] (2016) to perform inspection/validation of requirements. According to the [Johannesson] (2017) thesis, HW development makes use of customer and competitive mapping, as mentioned by [Ulrich and Eppinger] (2012), to validate that the correct requirements are used. Requirement engineering as highlighted by [Johannesson] (2017) thesis says that it brings in the original actors, those who stated needs at the beginning of the process, to confirm that the right things are specified. Questions were asked concerning how the connection between technical requirements and needs was controlled. Questions were also asked concerning the continuity of the control. Companies performed validation with more or less actual interaction with original requirement givers, such as internal customers, and focus groups. One company is failed to mention validation and another is claimed that once the translation from fuzzy requirements to technical requirements has taken place no further validation has to be performed. Continuous validation takes place at two of the interviewed companies. The comparison of different approaches and techniques which are followed in HW & SW engineering for requirement validation is given in Table 4.7.

<table>
<thead>
<tr>
<th>Requirement validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW engineering</td>
</tr>
<tr>
<td>SW engineering</td>
</tr>
<tr>
<td>Requirement engineering</td>
</tr>
<tr>
<td>Interview topics and questions</td>
</tr>
<tr>
<td>Interview answers</td>
</tr>
</tbody>
</table>
4.2.6 Use of requirements in development

In SW engineering, the requirements are used right from the beginning of the project, in contest of needs or desires of stakeholders. Their needs or desires depict what customers actually want. The implementation of these requirements is done with the help of development methodologies. The role of requirements ends at the final stage where validation is performed to ensure that requirements are successfully achieved from developed SW [Humphrey 1990]. In HW engineering according to Johansson (2017) thesis, requirements are used already in the beginning, as a base for ideas, and then again in the end for verification [Ulrich and Eppinger 2012]. The thesis by Johansson (2017) also stated that requirement engineering attempts to link test and test results to the requirement specification already during development. Here the limitation is that changes in the requirements might result in unreliable linkages [Hood 2008]. The interview question on the topic was how the requirement specification is used during development? The answer of this question is given in section 4.1.7, where different documents inside the separate internal documentation standards have different uses. Their much of the use is storage, but some of the use is communication between departments, and also different stages of development. The summary about how requirements are handled during HW & SW development, with interview topics questions and answers can be seen in Table 4.8.

Table 4.8: Comparison of uses of requirements during development

<table>
<thead>
<tr>
<th>Requirements use during development</th>
</tr>
</thead>
<tbody>
<tr>
<td>HW engineering</td>
</tr>
<tr>
<td>Used as constraints and targets to achieve product goals, and with which methods and tools</td>
</tr>
<tr>
<td>SW engineering</td>
</tr>
<tr>
<td>Used as constraints and targets to achieve product goals, and with which methods and tools</td>
</tr>
<tr>
<td>Requirement engineering</td>
</tr>
<tr>
<td>Possible to link tests and test results to the requirement specification</td>
</tr>
<tr>
<td>Interview topics and questions</td>
</tr>
<tr>
<td>How is the requirement specification used during development</td>
</tr>
<tr>
<td>Interview answers</td>
</tr>
<tr>
<td>Different documents have different use</td>
</tr>
</tbody>
</table>

4.2.7 Use of requirements in test and verification

The role of requirements in test and verification is to ensure that the product which is going to be built is right and tested through series of test cases [Humphrey 1990]. In context of HW engineering, the thesis by Johansson (2017) described that it shows how requirements can be put into decision matrices, towards which different concepts or designs are evaluated [Ulrich and Eppinger 2012]. Furthermore, in the thesis by Johansson (2017) mentioned about requirement engineering focuses on which and how tests will be performed, already at the time of the definition of the requirements [Hood 2008, Hull et al. 2005]. The interview questions concerned the control of requirement fulfillment, including if this control was continuous. In many companies the responsibility for verification lies with either a person or a department. Verification is performed through tests, which aim to verify functionality. All companies but one apply continuous testing, starting as soon as there is something that can be tested. The summary regarding comparison of use of requirements in test and verification in different development processes (HW, SW) can be found in Table 4.9.
Table 4.9: Comparison of how requirements are used for testing and verification in different processes

<table>
<thead>
<tr>
<th>Requirements in testing and verification</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HW engineering</strong></td>
</tr>
<tr>
<td>Decision matrix</td>
</tr>
<tr>
<td><strong>SW engineering</strong></td>
</tr>
<tr>
<td>Plans, Requirements &amp; design Specification, TDD, Use Cases</td>
</tr>
<tr>
<td><strong>Requirement engineering</strong></td>
</tr>
<tr>
<td>When to test, what to test</td>
</tr>
<tr>
<td><strong>Interview topics and questions</strong></td>
</tr>
<tr>
<td>How is fulfillment of requirement controlled. Is this control continuous</td>
</tr>
<tr>
<td><strong>Interview answers</strong></td>
</tr>
<tr>
<td>Either departmentally or person in charge, verified through tests, which verify functional requirements. Testing mostly continuous, and starts right from the start.</td>
</tr>
</tbody>
</table>

4.3 RQ 3 & RQ 4 - integration state, current and target

In this section, results which concern integration, both current and future ambitions, is presented.

4.3.1 Process adaption - common with [Johansson](2017) thesis

According to company W, there is no all-around process which means that a suitable process needs to be chosen for the current development (Appendix F.3). In company X all processes originate from the same general process, implying that the processes are adapted (Appendix G.2). At company Y similarly the different departments start from the same development process (Appendix H.2). Company X takes this one step further and has a general time plan from which all projects originate. The time plan is customized according to project (Appendix G.3). Further the requirements of company X are written in the same way, which means that requirement documentation is unified. It is however stored and used in different ways depending on department (Appendix G.8). At company Z, the time frame of a project depends on its size. If the project concerns to completely new product, the concept phase duration is 6-months and for design phase it is 1-year (Appendix I.2).

Mixed development models in processes

The interviewed companies (X,Y, and Z) do not have pure agile development processes in their development environment. These mentioned companies are currently using waterfall model but keen to move from this traditional model to hybrid development model which contains both iterative and incremental properties together. Company W mentioned about Scrum as their working methodology in their development process (Appendix F.3) but on the other hand they are still utilizing waterfall process model as well. The product development process (PDP) in company Y provides a structured way of continuous checking of system towards requirements. This approach gives them an opportunity to monitor the whole system continuously (Appendix H.11). In company Z, the HW in the loop (HIL) uses as a tool to test requirements. They apply continuous testing through daily builds (Appendix I.11).
4. RESULTS & ANALYSIS

### 4.3.2 Balancing of requirements & compromising between departments

According to company W, it is challenging to keep a balance between several different actors. All different actors need to be cooperative and adaptive (Appendix F.14). Similarly, company X explains that with a complex system there are many requirements which always need balance environment (Appendix G.6). It is hard to maintain a high amount of requirements, since it requires balancing and compromising. ‘Attributes’ (as mentioned by company X) need to be balanced (Appendix G.16). In company X, there are several viable solutions and only a balance of requirements can show which is the optimal compromise (Appendix G.6). In company Z, there is always need to balance the requirements. All requirements need to be considered, but not all requirements can be implemented (Appendix I.6). They claim that stakeholders understand the need for compromise (Appendix I.4). In company Y balancing and compromising are not mentioned explicitly, but they accept that it is important to understand why requirements should be included (Appendix H.4).

### 4.3.3 Dependency goals

The opinions regarding dependency between requirements differ among the interviewed companies. In company W there is a long term goal to cut the cost of development, and they are achieving this goal by decreasing the amount of dependencies (Appendix F.18). At company X, the interviewees have different points of view regarding dependency. They emphasized on requirements related to HW and SW need to be separated, not dependent (Appendix G.8), but company X still lifts the importance of interdependence between departments as there are some very close interactions and relations which is quite contradictory (Appendix G.15). At company Y, checklists occur that attempt to map which departments affect with other departments (Appendix H.15). At company Z, the perception is that the dependency between different development departments is very high during the actual development (Appendix I.15), and that to create a system, contribution from several departments is needed (Appendix I.15).

### 4.3.4 Collaboration

The view about the future collaboration is very different in the companies. Company W believes that people will still work in separate sub-systems, and the people with roles of interfacing between sub-systems, and different departments, will increase importance (Appendix F.18). Due to this separate mindset product development teams should have more interfaces for improved context (Appendix F.18). At company X it is almost the opposite. Between some sub-functions there is very close interaction (Appendix G.14). In such a complex product, multi-objective design optimization is necessary. One has to consider all requirements and their interactions (Appendix G.14). Company X currently has close collaboration and a high amount of integration (Appendix G.18). At company Y, the interviewees believe that there will be more collaborations with other international sites. This will result in an increased demand on collaboration skills. Best practices will be shared between sites (Appendix H.18). In company Z, the separation between departments is clear even though they work together very closely (Appendix I.15). In interdisciplinary development, one group will own the requirement or problem, and will need to ask other groups for help to solve it (Appendix I.14).

### 4.3.5 Organizational systems and traceability

The companies lift some issues in their requirement processes. Company W mentions that there is a lack of overview of the connection between requirements on different levels (Appendix F.9). As fuzzy requirements are decomposed, both into sub-requirements and spread in the organization, it is hard to keep track of dependencies in the system. If a change is done
in one branch, they do not currently have a good way for tracking this back and highlighting requirements for change in other branches (Appendix F.17). At company X there is a need to understand interactions and relations, where the best solution will be found, despite variation in results (Appendix G.17). They believe that what they need is a meta-model to see correlation between objects (Appendix G.18). At company Y it is mentioned that they do not use any project requirement database, but the use of such a database would enable visually seeing connections between requirements (Appendix H.18). Similarly, company Z stated that if they would use a database, there would be a possibility to link requirements to tests, or parts, and split requirements to different requirement owners (Appendix I.18). Today, the connection between requirements and code not clear, or non-existent at company Z (Appendix I.11). If they had a structure in a database, it would be easier to understood the connection all the way down to the designer. Company Z wants to increase traceability (Appendix I.18).

4.4 RQ 5 & RQ 6 Obstacles and enablers, simultaneous HW & SW engineering

In this section, interview data which supports or deters increased integration between different development departments is presented.

4.4.1 Division and separation

In the different companies, the separation between departments varies. In company W the separation between different development departments is extremely well defined (Appendix F.14). HW is treated as completely different organization according to the SW developers (Appendix F.17). In company X, they acknowledged that sometimes there is lack of full understanding of other departments’ requirements or the relationships which exist between requirements. They realized that the SW and HW people work in isolated ways (Appendix G.17). At company X, all departments being able to work completely separate from each other. With processes running in parallel is described as the ideal but impossible situation (Appendix G.19). In company Y, separation is described only by a statement concerning different departments have separate road-maps (Appendix H.15), how much separation, or collaboration, there is in company Y is not really clear. In company Z, the interviewees intended to point out what separates the HW development from the SW development. They really put emphasis on how different both their process and deliverables were (Appendix I.16).

4.4.2 Problems concerning change of process

At company W, it can be hard to convince others of the existence of problems (Appendix F.17). At company Y the issue lies in that it is hard to convince, and involve, the senior management in new ideas (Appendix H.17). At company Z there is opposition to change, even when things have been tried out, and lessons are available, the will to actually learn is lacking from the organization and management (Appendix I.17).

4.4.3 Cooperation required from higher authorities

The actors who are involved in decision making on top level management in interviewed companies (W,X,Y,Z) are need to be more cooperative, in terms of understanding their people needs as mentioned by interviewees. In company W, they mentioned about close collaboration between product management, departments, and customers is beneficial to improve overall working environment (Appendix F.5). The company Y pointed out that if several people are involved in decision making it becomes more difficult to get approval from every one and it is hard to convince senior management to implement new ideas due to their own...
priorities and plans (Appendix H.13). The interviewees at company Y also mentioned that management can play a vital role to improve overall performance in product development through effective decision making capability (Appendix H.18).

4.4.4 Dependency issues

When working with complex products, dependencies will occur. At company Y, a problem was described as following: when a small group, or a single person has too much work that become a bottleneck for the entire process (Appendix H.17). A similar issue was brought up by company W; generally parts upon which many other parts build become bottlenecks, due to a chain reaction of dependency (Appendix F.15). The company Z wants more system engineers and function responsible people to solve similar issues (Appendix I.18).

4.4.5 Time limitations in different development processes

In company Y, a comparison was made between HW and service development. In big HW development projects, time can be spent on categorization and prioritization of requirements. In service, the time might have to identify which requirements are key requirements (Appendix H.7). Further company Y mentioned how collaboration is added on top of the existing workload, and no additional time is budgeted for this task (Appendix H.18). Company X simply mentioned that service usually has shorter time-frames without putting it positive or negative light (Appendix G.2). Time limitation was mentioned as the main problem in requirement engineering for company Z. The company Z feels that they have too little time and few staff for the tasks (Appendix L.7).

4.4.6 Increased traceability in requirements process

Some companies want improvement in traceability (Appendices G.18,I.18). Company Y has a perception that the link between business or customer needs, and final product design can already be seen (Appendix H.9). At company W it falls upon a small amount of people to have an overview, and see to understand the connection between several different ‘boxes’ (Appendix F.14).
5 | Discussion

The core of this thesis has been the research questions. Having the "correct" research questions therefor is very important. Initially the thesis had only three research questions which were covering differences, similarities, and obstacles towards future increased integration. The final research questions were formulated to create a balance between different focuses, positives and negatives, or current and future, becoming balanced pairs. Differences and similarities were already a balanced pair. To balance obstacles, enablers were added. This modification had done before the execution of pilot interview, the reason of this modification is to make a balance and comparison between interview questionnaire in a right way. For the research questions concerning obstacles and enablers to be relevant, the question of current and future integration needed answering. Once these research questions were established, the choice was made to use them as an outline to much of the thesis content. They play a key role in the creation of the interview guide and therefore also the interview questionnaire. They also have a part in the "theory-driven" code words. Results are grouped according to them, as is the result discussion in this chapter. Lastly, the conclusion strives to answer the six RQs.

5.1 Discussion of theoretical framework

Concerning the balance between the different chapters in the theoretical frameworks, it was continuously reflected and found that SW engineering includes more detail in terms of different types of developments processes along with their models. In SW engineering, the risk estimation methodology is also introduced in shape of Boehm spiral model for successful completion of SW engineering process. The requirements engineering is done with further explanation to provide the reader with same understanding of its basic concepts than the other fields. In HW as mentioned in Johansson (2017) thesis for example, many models are similar, and have remained the same for several years. In SW, new models keep occurring and gaining ground and need describing. The theoretical framework presented in this thesis describes general processes, which leaves the comparison between theory and very specific processes presented in interviews questionable. Is it even possible to make a direct comparison between theory and data collected, or are there too many sources of misunderstanding? It is true, that the theory presented is general, but the anonymous data from the interviews has taken a step towards generalization. Even if it is impossible to say if the data is at the same abstraction level or not, similarities found support the comparison. The difference between the theory and the interview data which need to be questioned. It should be possible to draw conclusions based on similarities, while conclusions based on differences need further study. In some cases, when specific models or methodologies are mentioned by the companies for example, these could have been included in a revised theoretical framework. No extensive time was given to this type of revision. There is always the question of whether other, or additional theory could have been collected. In this case, certain topic areas came to light, for example testing and traceability. The testing is further discussed in context of verification and validation techniques and traceability is further discussed in the discussion of results. How-
ever due to time limitations no additional theory was added to the theoretical framework, resulting in full analysis concerning this not being possible in this thesis.

5.2 Discussion of methodology

In the method, chapter two different types of studies were described briefly, inductive and deductive. In this thesis interviews, it was initially believed that interviews were performed applying induction, speaking to experts in their separate fields to conclude to some type of general knowledge on requirement engineering and integrated/interdisciplinary development. The final result however leans more towards deduction as only particular situations could be confirmed as common between the different data sets. Different interviewees, even if perceived as very different, managed to provide similar data. Instead of providing completely unique data, which would be necessary to transform individual situations to general knowledge as described for induction, deduction was possible after all interviews were performed.

If this outcome had been known going into the study, maybe a different approach would have been taken, and the questionnaire would have been designed differently. With the results of this thesis as base, a purely deductive study could be performed, focusing on one of the topics which was shared among several companies, and absent in the theory studied. Now however interviews were chosen, designed, and executed as described in the method chapter. This discussion aims to discuss the applied method as well as the impact of the choice.

5.2.1 Impact of chosen method on results

All the decisions which were made concerning method, which can be read about in the method chapter, had consequences on the study. Some on the results.

Choosing interview as data collection method

As mentioned in the method, chapter interviews were only reasonable data collection methodology, even if other methodologies might have been better choices looking to the detailed nature of the research questions. Observations for example, and part-taking in a requirement handling process would give a more detailed, but also company specific understanding. The chosen method allows for the comparison which is possible in this thesis, but lacks certain depth as the questionnaires as stated by van Boeijen et al. (2014) only scratch the surface of knowledge.

Amount of interviews

The amount of interviews suggested by theory varies, and is not specifically defined for a company-related situation as the one in this thesis. van Boeijen et al. (2014) recommended three to eight interviewees and Kvale and Brinkmann (2009) said that circumstances dictate the amount which can be between five and twenty-five. In this thesis, twelve interviews were performed, at four different companies. If the recommendation from van Boeijen et al. (2014) is for each separate company, two of the companies would require additional interviews to reach van Boeijen et al.’s (2014) minimum of three interviews. Following Kvale and Brinkmann’s (2009) recommendation further interviews would have to be performed at all companies. If, however, the recommendation is for the study as a whole, needless interviews have been performed according to van Boeijen et al. (2014), and the amount is within the span mentioned by Kvale and Brinkmann (2009). As the perception was that the knowledge saturated level was not reached, it can be assumed that performing further interviews, at least at the companies where only two interviews were performed, could possibly have provided
more conclusive results. At the same time, the amount of interviewees provided by the companies is reasonable, as they had to take an hour each out of their day without any direct gain for them, except possibly the results of this thesis.

Design of interviews

As the interviewees within the different companies were either all from the same department, or scattered from very separate fields and levels, the output from interviews varied. In some of the summaries even contradictory data occurs, while other summaries feel unified. This depends on the span of interviewees. A wide variety of interviewees was positive as it gave many perspectives, having several interviewees with similar background however allowed for confirmation of statements between interviews, instead of having single statements on many different topics.

When scheduling the interviews with the companies some difficulties arose. As it had been decided to let the companies decide interviewees, this required initial information from the companies’ side before information could be sent out concerning the interviews. Once contacts were given, these were not always informed by the person who had provided the contacts about the interview so when they received the information, it was the first time they heard of it. At one company the mistake was made to, when provided with several possible interviewees, consider who could be most relevant for the study, as suggested in theory by Rubin and Rubin (1995) and Flick (2007). It turned out that those for whom the interviewers had interest, had little or no interest to become interviewees, resulting in further delays in scheduling of any interviews at all at said company. As the process of getting contact information was lengthy, and the potential interviewees had mixed response times. It took longer than originally expected to perform all interviews. This allowed for a lot of reflection to take place between the first and last interviews. This reflection is visible in the method chapter where edits and changes of the questionnaire are described. In the interview invitation, an hour was requested for each individual interview. In some of the companies, the interviewees were very generous with their time, allowing also for interviews which were longer than one hour. At one company, interview times were stacked together in such a manner that only 45 minutes could be used, resulting in only half of the interviews being possible to perform in a good way, briefly touching upon the other half. After this company, attempts were made to schedule interviews more in detail with the companies through email, adding buffers and scheduling more than one hour per interview.

Analysis of interviews

The coding of the interviews was difficult as many statements were very detailed and specific for the companies at which the interviewees worked. For the summary, certain generalizations had to be made to allow the companies to remain anonymous.

- Theory-driven code words: The theory-driven code words were extracted in hopes of creating a common understanding of the theory collected, before data was collected from interviews. The idea was to use these as a base for the coding of the interviews. These code words supported the general approach needed to allow the companies to remain anonymous. The codes did, as described by DeCuir-Gunby et al. (2011), mostly represent concepts discovered in the theoretical framework.

- Data-driven code words: When coding took place, many data-driven code words occurred to complement the somewhat limited theory-driven code words, supporting Gibbs’s (2007) and Flick’s (2007) claims of the coding process taking place after interviews. Many of these data-driven code words however, could be added to theory-driven code words in the summary, resulting in very few reaching the final report. In hindsight, the approach worked surprisingly well, but many of the theory-driven code
words originally generated ended up being superfluous. It can be seen if the list presented in the method chapter is compared with the given table of codewords.

- Code word summaries of interviews: The code word summaries were created using the same headings, which represent code words, to improve the overview. When summarizing the data in the results chapter it can however be seen that data in the same chapter of results (theme) comes from different chapters in the code word summaries. If more time was available, the summaries could have been organized further to ease comparison between companies. As earlier stated however, the companies covered very different topics and certain information needs to be grouped together in the summary to retain understanding.

**Execution of interviews**

The interviewees ended up having very different backgrounds. The choice to use the same questionnaire for all interviews was based on the ease of analysis, creating the possibility to easily compare interviews, in a "between case analysis" as mentioned by Flick (2007). Naturally one question rendered very different answers depending on the interviewee’s background, and the different interviewees were able to elaborate to different extents on the different topics, making the time approximation created based on the pilot interview inaccurate and of little use. The questionnaire questions were too numerous, and covered an excessively wide spread of topics. This is another possible reason for the number of interviews (12 in recommended spans of 3-8 and 5-25) not rendering saturated data. Future studies should probably focus on a single, more specific topic. Even interviewees, at the feedback session of the interviews, commented on how general the questionnaire was, and how it would be necessary to ask more specific questions for them to be able to provide in-depth answers. The interview performed with two interviewees at the same time naturally resulted in more aligned answers, and focus was on similarities and not on differences. One person was the main speaker, leaving space to wonder what the other person would answer to questions where the interviewee chose to simply agree in this case. The summary also became shorter than the others due to less diverse answers. So as this type of interview was efficient from the interviewers point of view. This approach took longer for the two involved interviewees and left questions unanswered for one of the two interviewees, and for a comparative study such as this, it is probably preferred to have all interviews separate, as they were performed at other companies.

**Notes, recordings, and transcripts**

During the interviews notes were taken, and audio and/or video recordings were made. The audio and video recordings were meant to support the collection of interview data. The note-taking technique of the two interviewers was very different, with one taking very extensive notes, and the other noting down only key-words. The extensive notes required only small confirmations from the recordings when cleaning up the interview notes. The use of key-words required a complete listen-through of audio recordings to create the complete notes for each interview. The audio recording equipment should allow for a more relaxed note-taking, and an increased interaction with the interviewee, however, the later required "transcription" is very time consuming. Also taking notes can help encourage the interviewee to keep talking, that what is currently being covered is interesting.

5.3 **Discussion of results for RQ 1 and RQ 2**

RQ 1 and RQ 2 concern differences and similarities in different requirement handling processes. Specifically HW and SW development, and how differences and similarities affect
an increased integration. In this chapter, discussions are conducted attempting to lift how differences and similarities support or counter an increased interdisciplinary development.

5.3.1 Actors

The number of actors (stakeholders) are varied from company to company as mentioned in section 4.1.1, which creates difficulty to make a conclusion about actors. The actor is an important entity who does interact with the system. In use case diagrams, the actor is considered as one of the major elements. In requirements handling process during SW engineering, the identification of actors plays a vital role in requirements elicitation and analysis. The number of actors is based on companies business strategies. In theory concerning hardware mentioned in Johansson (2017) thesis, more specifically the house of quality, no particular actors are pinpointed, everyone affected should be involved. In requirement engineering a list of examples is provided. Interviewing a couple of more companies might result in reaching a “max” amount of actors, in the shape of a list similar to that of Hull et al. (2005), and deeper descriptions of these actors might point at the actors in the processes having more in common than is currently visible. The additional actors can be explored through adding additional questions in the interview questionnaire.

5.3.2 Educational & professional background

The educational background and professional work experiences of actors, who are specifically involved in interdisciplinary requirements handling during development process in large companies have impact in overall working environment. This factor has explored from the conducted interviewed companies W,X,Y, and Z. The reason of it is that they belong from different disciplines and have more or less past work experiences. The HW people are one who are still relying on tradition ways of performing tasks. The past experiences of HW people are emphasized on systematic or plan driven development approach. The SW people are focused on agile way of working where continuous production and integration are their major goals.

5.3.3 Overall processes at companies

The similarity in terms of processes has found through gates, which are used at all companies except one. It is the more SW focused company W which uses “checkpoints” to manage their projects. The checkpoint is similar to backlog items list which is used in project planning in an agile sprint highlighted by Schwaber (2004) and Sommerville (2016). All companies however have some type of formalized process. So there is a shared understanding of the need to generalize the process used into something schematic. However this schematic needs to be more agile to enable more efficient SW development, that is the feature of agile manifesto as well. In company Z, increase agility was pointed by interviewees, both specifically into the SW department, but also generally into the development (Appendix I.18). This topic is further discussed in Johansson (2017) thesis, says that it would lead towards the conclusion that HW requires a more agile process to optimize development, and be able to coordinate with SW development team. There is very little point to this in the direction of HW is going at the companies interviewed. The basic HW development theory has remained similar for years as Ulman’s (2002) and Ulrich and Eppinger’s (2012) models were possible to merge into one figure of HW development process as mentioned in Johansson (2017) thesis. Similarly Ulrich and Eppinger’s (2012) referenced to Hauser and Clausing (1988) for steps of requirement handling even if the source was 24 years old at the publishing of Ulrich and Eppinger’s book. HW developers at company Z have acknowledged the need of more testing in SW development to reach a complete product (Appendix I.16), but do not consider applying...
a more test-intense development process themselves. SW development at (Appendix I.18) say that management still has a focus on "traditional" development and it is not adaptive to SW development processes. The transition from developing a product which solely consisted of HW, to a product including SW, services, and more, might be difficult.

5.3.4 Requirement handling processes

The way to do interaction with stakeholders is similar in companies interviewed. In the different requirement handling processes, the stakeholders are often included early in the process which is also a core benefit of agile development process in SW engineering (Pfleeger and Atlee 2006). This approach is found common in all interviewed companies. It has noticed that interaction between the stakeholders (specifically customers) inside the companies need to be improved. Company W spoken about having trace back to the actors who collected requirements from the customer (Appendix F.9) is important to protect the privacy of the customer. Company Z lifted the issue of getting changes acknowledged by the customer (Appendix I.5), hinting at issues occurring due to lack of communication between developers and customers. The mentioned issue is also discussed in theoretical framework regarding requirements handling under SW development. The requirements elicitation and analysis is the phase where requirements are gathered and analyzed through classification of requirements Sommerville (2016). The three out of four companies already applied requirement breakdown structure, and the fourth company has different levels of requirement documentation. The breakdown of requirements among different development fields in a more complex product should not be any problem. This shared methodology is needed to allow an easy transition into more interdisciplinary requirement handling. The roles of departments differ in the companies. There seems one department which provides the initial requirements. A coordinating actor can be a good start for more interdisciplinary requirement handling. The two companies (W & X) where remaining departments input requirements, which are important for them might to be more accustomed for a more interdisciplinary process. These companies’ departments already have to represent their field in a complex product. At company Z, the approach using interface requirement specifications also shows an understanding of the roles of other departments, and how to best work together developing a complex product. Having to deliver something to other departments or a shared requirement document can be a supporting factor for more interdisciplinary development.

5.3.5 Techniques for elicitation

The process of collecting requirements is varied from company to company which hugely depend upon the nature of product which they are produced. The role of actors who are responsible to carried out this activity, and tools which are used in this regard are different in interviewed companies. The company X used investigation of customers as their major technique to collect requirements from different stakeholders (Appendix C.5). The feasibility study can be a good option to identify and investigate customers but it needs to be performed before requirements elicitation and analysis process as given in Figure 2.7 In company(Y), the internal requirements are originated from previous work experience or newly developed technology of same type. Interviews are considered as their main technique to perform requirements elicitation/collection process (Appendix H.5). The requirements engineering spiral model in context of SW development as shown in Figure 2.8 can be used to adopt approaches for development, where requirements are discovered at different stages Sommerville (2016).
5.3. Discussion of results for RQ 1 and RQ 2

5.3.6 Time frame

The size of the project matters a lot for overall planning under the time frame to produce final product. The following factor does impact on requirements handling in complex product development. Why?, because shorter time frame gives less time to perform in depth analysis of requirements. The size of the project is varied in interviewed companies and has impact in their time frame as well. The time frame management is done by higher authorities (decision makers). SW engineering - specifically in agile SW development, continuous testing and quick release of SW is the key to shorter the overall time span of production rather than investing much time on process and tools as followed in development during plan-driven development in HW [Cockburn et al., 2001]. The agile approach of development produces more output in less time. In case of HW engineering as mentioned in [Johansson, 2017] thesis, requires longer time frame for production due to complex development process and longer lead times. Service has a shorter time frame according to both companies X & Y. Company Y stated that economy as the reason for this shorter time frame (Appendix H.16). The development processes simply put are different, and to be able to have a more integrated interdisciplinary process these differences need to be considered. In the thesis of [Johansson, 2017], it is highlighted that HW processes can only be shortened to a certain extent. Time frames mentioned vary between a couple of months to several years, which makes further comparison difficult.

5.3.7 Prioritization

The interviewed companies mentioned about the issue of having different prioritization level in different departments to handle requirements. Increasing the amount of departments involved through a more interdisciplinary process would add further to this issue which reasonably needs to be resolved to achieve an efficient development environment. The company Z which did not lift these types of complaints. The prioritization at company Z is officially removed altogether after requirements have entered into requirement specification. The concept of prioritization of requirements is mentioned under the Scrum process in SW engineering theoretical framework, where project owner is the person who performs this task as described by [Schwaber, 2004]. In interviewed companies, there is lack of such a process which could be a one reason why they are suffering. In context of HW engineering, the order of steps which follows requirement handling where requirement negotiation technique by [Lamsweerde, 2009] is discussed in [Johansson, 2017] thesis. The same point is also described in comparison between plan driven and agile development in theoretical framework of SW development through [Cockburn et al., 2001]. Company W lifts an important issue that springs from this, namely that something with a low priority might effect the system as a whole, requiring it to be given higher priority (Appendix F.15). This factor leads us to believe that for a more interdisciplinary development there is a need for a person, or department who makes priorities for the product as a whole. This prioritization should take place before the formal documentation of requirements. It is possible that the system engineers who exist at some of the companies already fill that role. Revision of prioritization should be possible to include in the already established modification methodologies.

5.3.8 Documentation standards

The role of documentation is important to organize and standardize requirements. The importance of keeping standardized documentation is also realized by interviewed companies. All companies (W,X,Y,Z) followed an internal documentation standard, where the extension of unity however varies. Already having a shared documentation standard within the company should support an increasingly interdisciplinary development. The need for specific department during documentation is understandable. A more detailed study into the requirement specifications would be necessary to understand exactly why it is difficult to un-
5. Discussion

understand the requirement specification from different departments if the documentation is truly unified. The requirements specification document is considered as must need document, it states that which requirements are required to be fulfilled by [Bell 2005].

5.3.9 Input requirements

For future increasingly interdisciplinary development, having a common starting point should be positive. All of the companies (W,X,Y,Z) start with high level requirements, but their origin differs. It would not seem however that this difference is related to HW and SW so no difficulty for an increased integration, as long as the company as a whole agrees from where requirements should originate.

5.3.10 Modifications

In SW engineering, requirements modification management process is presented to plan and manage requirements modification in an organized way [Sommerville 2016]. Basically, it is a three steps process starting from problem analysis and change specification, followed by change analysis and costing, and ends at implementation of change. The role of CCB (change control board) here is to verify that the major changes are observed by all concerned stakeholders and check that every change is approved before implementation [Humphrey 1990]. SW maintenance in SW engineering is also beneficial to modify system with additional requirements to meet the latest changes of competitive SW market [Braude 2003]. In HW prospective according to [Johansson 2017] thesis, modifications should be easy early on, and include a formal change process later in the process [Hull et al. 2005]. Only at company X is the possibility seen to avoid administration and formality late in the process if changes are performed locally as mentioned by [Lamsweerde 2009]. According to [Lamsweerde 2009], local changes are possible if the process is structured and designed for modifiability which might be more common in SW development focused companies. However, this needs to be confirmed in further studies. The mindset towards modifications otherwise seems to be similar at the interviewed companies, especially concerning late additions. Modifications are evaluated in terms of time, costs, and it effects at two of the companies. The late modifications are only made if they are absolutely necessary. There is an agreement in HW and SW development that a formal modification methodology should be used late in the process, at least when considering changes at a more general level.

5.3.11 Traceability

All interviewed companies mentioned more or less about traceability in their requirement handling process. The fact that needs to be remembered about traceability as described by [Nuseibeh and Easterbrook 2000] is that, traceability is only beneficial to look the possible influence of change but not in itself reasoning about the change. After analyzing the feedback from interviewed companies, it has found that the level of traceability is fluctuated from company to company. The reason is that when productivity of a company increases the complexity in its production environment also increases. For example, as mentioned in previous chapter, one interviewed company do not allow every actor in their working environment to get involve in traceability environment to keep their level of secrecy high. On the other hand, another interviewed company has clear form of traceability between their business and customer needs, which gives them clear visibility in their final product.

5.3.12 Validation

A factor which was found common at interviewed companies regarding validation process, is the lack of clear understanding about the difference between validation and verification of
requirements. The difference between both techniques is presented in theoretical framework of SW engineering. According to the literature, the role of validation is to get ensure that the product which has built is the right product in terms of giving right functionality [Bell (2005)]. The one interviewed company mentioned that transformation from fuzzy to technical requirements is implemented so there is no further validation is required but not given any concrete reason in this regard. In reality, it is hard to accept because their transformation could be wrong. The validation is good for creating a sense that requirements and models which are elicited give a right representation of requirements as stated by stakeholders [Nu-seibeh and Easterbrook (2000)].

5.3.13 Testing and verification

The process of testing requirements which are followed by development team can be performed on continuous or after specific time interval, is based on which development process model has been chosen for SW development [Cockburn et al. (2001)]. The interviewed companies need to realize the importance of different development process models as provided in theoretical framework of SW engineering. The feedback given by interviewed companies is emphasized on continuous way of testing but actually all of them except one follows waterfall model where continuous testing is not possible. The company W is using TDD (test driven development) approach in their development environment, which gives them an opportunity to test their requirements and design specification before actual development (coding). The verification is the process to verify that, is the product which is going to be built is the right one? [Bell (2005)]. The interviewed companies adopted different strategies to perform verification of requirements. The companies can get benefit through practicing system and acceptance testing in their product development processes. System testing gives an opportunity to check that the designed system fulfills its overall objective or not [Humphrey (1990)] and acceptance testing gives assurance that system has fulfilled all customer requirements [Sommerville (2016)].

5.4 Discussion of results for RQ 3 and RQ 4

RQ 3 and RQ 4 concern the current and future approach for integration in companies. Many companies are already supporting integration to some extent, but the view on the future of integration is very different, which is to be expected for something which is unpredictable.

5.4.1 Process adaption

In the thesis by [Johansson (2017)], processes are described in the theoretical framework, both generally for HW as well as requirement handling are generic, and the application of processes in companies naturally is company-specific. Currently some companies (X & Y) have a standard process from which department specific processes originate. The shared process should increase cross-department understanding. At the same time, the fact that the processes are adapted to fit into different department implies that the companies understand that different departments have different needs. In fact, it depends on how the standard process is designed, it will be more or less appropriate to continue using in a more interdisciplinary development process. The steps may be required to include in the core process, encouraging interaction between departments. Company W claims that a suitable process needs to be chosen for the current development (Appendix F.3), so if the current core process is more suited for one development branch, it needs to be balanced so that all department feel the core process is relevant and appropriate for them. The type of process which is selected for development of product is played a vital role in integrated development environment. According to the feedback from interviewed companies, the current state of integration inside
interdisciplinary development environment needs improvement. The waterfall model can be suitable for development of projects in non-complex development environment where companies are more entrusted on process and tools. The reason is that the approach of stages follows in waterfall model where process moves slowly but steadily [Winston W. Royce (1970)]. On the other hand, the companies with complex development environment need agility in their development processes [Mohammed et al. (2010), Sommerville (2016)] to adopt changes in requirements frequently. A good thing which has been found in interviewed companies is that they are keen to adopt agile development approach in their integrated development environment. The data showed that only one company is utilizing agile development process in their development environment where SCRUM and KANBAN are used as well (Appendix F.3). They believe that in this way they have strong way of collaboration between working teams but in reality this argument is hard to accept because SCRUM gives good collaboration environment inside the team but inter-team communication with SCRUM is a tough task. The agile development process is best solution to handle integrated working environment, why? because agile approach of development gives an opportunity to shuffle the role of the people who work in development teams. The mentioned way of working increases the productivity of overall development process inside the companies, this approach leads towards shared knowledge and increased collaboration inside the development teams [Cockburn et al. (2001)]. The interviewed companies (X,Y, & Z) do not have pure agile development process in their development environment, they currently have hybrid development environment (waterfall & agile). Although move from traditional waterfall model to agile development or hybrid development environment in interdisciplinary development environment is a tough task, but it is required to meet todays’ challenge of handling interdisciplinary development environment.

5.4.2 Balancing and compromising

There is already an understanding that balancing between actors is necessary in companies X & Z (Appendices G.6, L6), and that in a complex product development a common goal needs to be reached. Departments being willing to work towards a common and shared optimization is a good sign. Increasing interdisciplinary development will mean more compromises and more negotiation, but the companies seem ready to look their current mindset towards requirement balancing (Appendices G.8, G.6, L6).

5.4.3 Dependency goals

The approach of company Y with checklists and interdependence mapping (Appendix H.15) implies an understanding of the dependency between requirements at their different departments. Company Y is building models to support dependency, instead of working against it. Company W’s long term goal to decrease dependency (Appendix E.15) might not be all bad though. As many dependencies should be removed to allow an independent progress as possible in separate departments, which is something you want according to company X (Appendix G.19). The companies X and Z have same understanding that there is close interactions between certain departments and contribution from several departments are needed to create a system (Appendix G.15, L15). To achieve an optimal, well-functioning, and complex system, companies need to embrace and understand when dependencies are needed. As mentioned in Johansson (2017) thesis in context of HW development through Ullman (2002), eighth room of the House of Quality, an awareness of relationships, and therefore dependencies, need to be created, because dependencies will almost alway exist. Dependencies however can not disappear all together as then the end result will not be one product but several separate parts of a product.
5.4.4 Safety and regulations

Legal requirements are handled separately as these requirements are also important. This fact is found similar in interviewed companies. Company X highlighted that standards and regulations allowed for certain prediction of future expectations from external actors (Appendix G.8). Companies X & Z also mentioned a process or even a department for handling legal requirements. Increased integration should not mean much difference, especially if the legal department is separated from the development department. If this is the case, their process probably will not change at all. Issues are rather for companies which do not currently have a process for handling legal requirements and standards. It can generally be recommended to unify before increasing integration, or putting time into unification as a part of increased integration.

5.4.5 Collaboration

Company X has close interaction and a high amount of integration (Appendix G.18). Company Y sees an increased collaboration with other international sites in the future (Appendix H.18). Company W has a complete opposite scenario in this regard, they want that people still work in separate sub-systems in the future (Appendix F.18). Between these two views lies company Z. Company Z’s concept of letting different departments own requirements (Appendix I.14) implies that company Z believes in some separation to divide responsibility, but they also understand the need for collaboration between departments. This leads to a similar reasoning as that of dependency, that collaboration needs to be balanced. A high level of collaboration could result in a high amount of meetings relative to the actual development taking place. The focus should be on where inter-department collaboration could result in relevant added value, and where collaboration is absolutely necessary to output a working product. It can be assumed that in some aspects departments are so specialized that they may not be benefited from collaboration with other departments in these aspects. However, with an increasingly interdisciplinary development, naturally an increased collaboration can be expected, and it might be this increase in collaboration company Y predicts, and company X already has. The interviewed companies are focused on close collaboration and communication as their key tools to handle integrated development environment. Some companies who feel lack of communication are keen to improve and others who have good communication and collaboration in their environment are getting benefits in their working environments. It is a challenge for companies with complex interdisciplinary environment to provide better communication and collaboration between different actors who are involved in different development environments. The role of moderator is in place, who bridges the gap between different actors in company W (Appendix F.14). In company Y all departments are involved in their interdisciplinary environment (Appendix H.14) and same answer is given by company X as well (Appendix G.14).

5.5 Discussion of results for RQ 5 and RQ 6

RQ 5 and RQ 6 concern obstacles and enablers for future interdisciplinary requirement handling. Looking to the results there is a clear unbalance, more obstacles than enablers have been mentioned. This however can be due to interviewees feeling that they already work interdisciplinary (company X), or alternatively feeling that they will not increase integration (company W), resulting in the company not working to provide enablers for this. This chapter tries to highlight different obstacles and enablers based on the feedback given by the interviewed companies.
5.5.1 Division and separation

Company W mentioned that SW and HW being like two different organizations (Appendix F.17), even if their final product requires HW and SW to work together. Company Z’s interviewees tendency to lift differences between HW and SW development (Appendix I.17) does not only point to understanding, but also a "us and them" mentality. The "us and them" type of mindset will make a change towards more interdisciplinary development difficult. The mindset in company X, depicted that the understanding of other departments and relationships is not complete (Appendix G.17) and need to be improved in case of increased integration. Company Y for which separation is not clear, might need to first map the relation of interdisciplinary departments, to enable increased collaboration in the future. The mindset of employees towards other departments in a company needs to be addressed before increased integration. So that it will not become an unnecessary obstacle in future.

5.5.2 Different ways of working cause problems to handle changes

Companies W, Y, and Z have complaints concerning convincing others that change is needed. Only in company X this was not addressed, and maybe this is the only company where management support for the requirement handling process is appropriate. Change is hard, and convincing others, especially managers, that is beneficial, is difficult. However if one company is successful, it might be a good idea to look into this company and see what they are doing successfully, and see why this is not the case at other companies. The cooperation between higher level management with their lower level staff is always required. The higher level management has power to take decisions and plan strategies, so if these people have close interaction with their lower staff, they can take productive decisions and plan strategies to achieve organizational goals. The interviewed companies have different views in this regard, like company W mentioned that closeness is required between higher management and customers (Appendix F.5). The company Y mentioned that if lot of people will get involve in decision making then it is hard to satisfy everyone (Appendix H.13).

5.5.3 Interdepartmental dependency

The level of dependency inside interviewed companies is solely depends on which type of product they build. In context of product development, the role of requirements related to specific product defines how much level of dependency is required inside different departments of a company, noticed through the feedback from interviewed companies. The company W has a long term goal to cut their development cost, in this regard they are trying to keep level of dependencies as low as possible between their departments (Appendix F.18). The followed approach is leaded them towards having independent way of working inside their development environment. At company X, the interviewees have different point of view regarding dependency. They are emphasized on requirements related to HW and SW need to be separated, not dependent (Appendix G.8).

5.5.4 Time limitations

In company Y, it was mentioned that the service development process has less dedicated time for requirement handling (Appendix H.7). At company Z, they felt that too little time is assigned and less staff exist to handle the tasks (Appendix I.7). Company Y also had issues with not getting time scheduled for expected increased collaboration (Appendix H.18). Even if only half of the companies interviewed had time-issues, it is still worth reflecting over how important appropriate time-division is for successful requirement handling. As stated by Kotonya and Sommerville (1998) how there is a tendency towards rushing through requirement validation, but this might be true for the requirement process as a whole. There is a lack
of understanding of how extensive requirement handling is needed to provide the company with improved results or products.

5.5.5 Modifications of requirements

The higher frequency of changing in requirements could degrade overall development process in an interdisciplinary development environment, if there is no solid modification handling process in place. This is what explored from requirements modification management process defined by Nuseibeh and Easterbrook (2000) and discussed in theoretical framework of SW engineering. The different issues mentioned by interviewed companies in this regard. Company W mentioned about the time when change is occurred. Company Z highlighted about cost issue which could increase due to adaptation of change in requirements after verification phase. The concept of CCB (change control board) as discussed in theoretical framework of SW development (section 2.2.6) can play a vital role in development environments of the companies to handle their requirements modification process in an organized way.

5.5.6 Increased traceability

As covered in the prior discussion on traceability, some interviewed companies see issues with traceability, but they also see potential solutions to these issues. The interviewed companies want to increase traceability to handle complex, and integrated systems or products development. The fact that companies are striving towards increasing traceability also means that they are indirectly getting prepared for increased integration. Pfleeger and Atlee (2006) encouraged traceability in context of requirements elicitation and analysis says that - collaboration is needed to develop link between requirements definition and requirements specification. This kind of interlink improves bounding between customer(s) and developer(s) views which leads towards having strong traceability. There is a potential danger in company Y who feels that their traceability is already good enough, if it turns out this is not true in case of an increase of integration needs further improvement. To make a direct correlation between defined and specified requirements is a hard job, it requires process management during whole development life cycle Pfleeger and Atlee (2006). In fact, it is all about making links between defined requirements, specification, design, implementation, and verification as stated by Pfleeger and Atlee (2006). No matter the current or future method of implementing traceability, it must be required for this traceability, in whatever form it is, to be possible to scale. It is possible to add new departments to a common system, database, or even map, for traceability, when interdisciplinary development is increased.

5.5.7 Possible collaboration and knowledge sharing between employees

The knowledge and work experience of professionals who are working in a company from several years are much beneficial for those who are newly hired and have less work experience. The interviewed companies who have blend of employees (seniors & juniors staff), are getting benefits in shape of saving time and money which could be required to train newly hired people. As a result, they are able to fill the communication gap between employees and have better working environment to solve problems or issues which could arise in their working environment.
6 | Conclusions

The aim of the thesis is almost achieved in connection to the answers of the RQs. The questions related to requirements handling process are covered with the help of answers of RQs related to differences and similarities. The current state of integration in interdisciplinary development environment is investigated. The views related to the future integration are not thoroughly covered. In future, more integration is required, that is mostly obstacles were raised. The RQs related to enablers in future aspect for increased integration were answered improperly.

What differences in HW and SW engineering need to be considered during interdisciplinary requirements handling (RQ1)?

Requirement elicitation, documentation standards & their use, and verification methodologies are done differently. Different development processes have different perspectives on time. The verification and validation are utilized differently into different development environments. The different development fields (HW, SW) is one of the major reason of time fluctuation required for development. The SW development is produced more output in short cycle of time but in case of HW development the result is opposite. This difference in time is difficult to balance. The prioritization of requirements are different between departments inside the companies. The enhancement in integration is required to make better coordination between actors who set priorities. The prioritization is required to be set before documentation, and revised through proper modification methodology. The placement of CCB (change control board) inside companies can play a vital role in this regard. The educational backgrounds and work experiences of team members in development process need to be utilized as a collective approach to achieve organizational goals.

What do the companies have in common which might support interdisciplinary requirements handling (RQ2)?

The interdisciplinary development has established processes with gates or handshakes (platforms for decision making), where companies prefer agility. The agility is suitable to support a shared process inside interdisciplinary development environment, where the shared process means requirement break-down and continuous testing. The internal documentation standards and the level of input requirements being similar. The traditional way of modification handling is followed by companies, where modification comes into play late in the process for general changes which affects several departments. The requirements related to safety and regulations are handled differently from other types of requirements in interdisciplinary development processes. In case of interdisciplinary development, this different way of handling requirements matters a lot. The traceability is an important topic which has been tried to cover according to SW engineering and linked with the feedback of interviewed companies. The One interviewed company has implemented it effectively and rest of the interviewed companies are still striving to have traceability in their development environment. The traceability needs to be increased between departments of interviewed companies, specifically in their requirements verification process.

What is the current state of integration in companies which develop complex systems (RQ3)?

The interviewed companies considered their close collaboration and communication as key tools to handle their integrated development environment. Here, the geographical loca-
tion of actors influences on how much collaboration is possible. The development process type plays a key role in handling the integrated development. If the company applies standardized processes, the process is altered to fit the project or department. The people who deal with requirements need to realize that balancing between requirements in terms of prioritization is necessary. Dependency is the topic where mixed feedback is given. Many want to decrease dependency while others believe dependency is necessary. After observation from both sides (HW and SW), dependency could be decreased to a certain extent, but the key dependencies must be presented to produce an optimal complex product.

*Are companies which work with complex systems striving towards a more interdisciplinary development environment (RQ4)?*

The companies are mentally prepared for an environment with more actors, requirements, and integration. Here, first of all, companies must need to ensure that a core process is applicable in all departments. The opinion related to the change of processes differs from one company to another. Many believe that processes need to be remained in sub-system divisions due to their complex nature. The opposite point of view is that everything needed to contribute towards the final product, resulting in increased collaborations in future. The interviewed companies who are in favor of interdisciplinary environment are trying to bridge the communication gap between individual actors in interdisciplinary projects. The hybrid development process model (combination of agile & waterfall models) is one of the solution for them. The interviewed companies are also keen to have visual models of requirements with their relations, which would be helpful for handling complexity in their interdisciplinary development environment.

*What obstacles exist for simultaneous HW and SW engineering (RQ5)?*

It has been observed that employees of two disciplines (HW, SW) belief that they could not benefit from each other’s processes. This mindset of employees should be changed in a way that they feel "we are one company, producing one product together with common goals". In this way, "we and them" culture will not become an additional obstacle in interdisciplinary development environment. The people who are decision makers in the companies need to be more cooperative in terms of understanding the needs of their development teams. For example, time is not dedicated to the requirement handling process and it is not fully appreciated from them that how much time is required for a thorough requirement handling. There is also need to balance workload between different actors involved in development process. In future prospective, the reuse of old or addition of new requirements is purely based on experience. So if the companies do not have experienced people it can be an obstacle for future integration. From HW prospective, companies have fear that continuous development could result in a higher frequency of changing requirements which could degrade overall development process in an interdisciplinary development environment.

*Having looked at the obstacles, what enablers exist for simultaneous HW and SW engineering (RQ6)?*

Companies are looking towards increase traceability between requirements which is a handy tool to support interdisciplinary projects. The concept of sharing knowledge and work experiences is already established and could contribute towards making interdisciplinary working environment more result oriented and organized. Communication and collaboration are productive to get mutual understanding between key actors in an interdisciplinary working environment. The companies are looking forward to utilize these tools more and more in future.

The conclusions which are mentioned above need to be consolidated before consider as a common methodology in an interdisciplinary environment.
7 Future work and Recommendations

The room for improvement is always present. Conducting more interviews specifically with SW companies along with redesign questionnaire would be a right choice to further explore the requirements handling process. This approach will be helpful to perform further studies in order to improve integrated development environments. The tools and processes in development environment could play important roles in requirements handling process in HW and SW engineering. The modified questionnaire need to be focused on these mentioned key points. The agile development is considered as a key of success in SW companies, which can be utilized at HW based companies to increase their productivity as well. More and more collaboration is currently needed inside the companies.

The results of the thesis could be more valuable for other related projects. The requirement types needs to be explored further to make effective comparison between HW and SW development. Additional study will be more beneficial to compare actors and their roles between different companies. Traceability between requirements in interdisciplinary development environment is a major challenge which was not well covered in this thesis. Risk management is beneficial for project management which needs to be explored further to place a solid risk management in interdisciplinary development process.
Bibliography


Appendix A. Interview invitation, original interview guide

Linköping, February 24, 2017

Dear interviewee, your company has shown interest in participation in our study of interdisciplinary requirement engineering. The study aims to find synergies and learnings from the separate fields of software and hardware development. Initial information has been collected through a theoretical review and now we wish to deepen our understanding through interviews with relevant actors from your company.

1 Topic areas for the interview

The topics we intend to cover in the interview concern both requirement engineering, and integration, along with the overlapping field of interdisciplinary requirement engineering as well.

2 Interviews

We, who will be performing the interviews, are two master students from the fields of mechanical engineering and computer science.

- Agenda
  After a brief introduction from our part we would like to collect some background information about the interviewee before proceeding to cover the above mentioned topics.

- Recording
  The interviews will be recorded (audio, video) with your prior permission. Your confidentiality naturally is our first priority during the whole project.

3 After the interviews

The results gathered from the interviews will be analyzed using prior collected theory, concluding in answers to the thesis' research questions on interdisciplinary requirement engineering.

- Feedback
  To take part of the results from the interview, interviewees are welcome to access our final thesis report once it is finished.

Contact

Feel free to contact us with any questions or suggestions.

Hanna Johansson - hanjo712@student.liu.se
Bilal Tahir Sheikh - sheta181@student.liu.se
Appendix B. Final questionnaire

The first time X minutes represents how many minutes the section would approximately be allowed to take to keep within the frame of 60 minutes total for the interview. The second number /X represents the total so far in the interview. (A) Warm-up (3 minutes/3)

1. Field of work now and previously
2. Experience

Questions covering topics concerning requirement engineering:

(A) Requirements elicitation (9 minutes/12)

R-1. Which actors are important in your requirement process?

a) How are important actors identified?

b) Are there any key actors?

c) Are original actors stated in the requirement specification? (traceability) Are they revisited during development?

d) Are these actors revisited in future development processes?

e) To what extent are your requirement influenced by norms and (the) standards?

R-2. What technique do you use to collect requirements? Why?

a) Do you work differently with requirements depending on their origin? Say external/internal requirements?

R-3. (What strengths and weaknesses do you see in your requirement collection method?)

(B) Requirement categorization (7 minutes/19)
B. Final Questionnaire

R-4. How do you organize product requirements?

R-5. (prev. R-7) Do you use same approach to all requirements?
   a) (R-5.) Does this approach also apply to requirements which concern many areas? (interdisciplinary)

R-6. (prev R-5) Which are the main requirement categories/types your development area encounters?
   a) (R-6.) Do any unique requirement types in your area effect the methodology or process?

R-7. (prev R-6) How do you manage requirements which are given (by actors) once development has already started?

   a) How do these type of requirements appear? From where?

   b) How do you handle additions depending on time and actor? (beginning, middle, end)(prioritization)

(A) Requirement documentation (7 minutes/26) (make sure to note if these questions have already been answered in previous questions)

R-8. How and when do you document requirements? What advantages can you see in your documentation? (who is responsible?)

   a) Are requirements maintained throughout the process?

   b) Is the requirement specification seen as a living document or set in stone?

R-9. What type of specification\requirements document do you use? (Standard? Why? Strict or flexible?)

   a) Is there a unified standard? Or a department specific standard? Is it an open standard which we could find through for example IEEE or an internal version?

      Examples of standards: SRS (system requirement specification) PDS (product design specification)
R-10. What advantages/disadvantages can you see in your type of documentation?

R-11. Are specifications for other departments (in other disciplines) easily accessible?
   a) If yes, is this useful? Why/Why not? Understandable? Important?
   b) If no, would you like to have better access? Why/why not?

R-12. How is the requirement specification used during development?
   a) How is the traceability during the development process, is the link between requirements and product clear?

(B) Requirement modification (2 minutes/28)

R-13. Do you have a modification handling methodology in place?
   a) If yes, could you describe it briefly?
   b) If no, do you think you could govern from a modification methodology, and what would it include?
   c) Are you satisfied with this methodology? Why/Why not?

(B) Requirement verification (6 minutes/34)

R-14. How do you make sure you don’t have any conflicting requirements in your specification? (Responsible person?)

R-15. How do you make sure that the customer needs are correctly reflected in the requirement specification?

R-16. How do check that the system fulfills the requirements?

R-17. Is it evaluated during the whole process if the system complies with the need?
R-18. Do you continuously check the system towards the requirements?

a) Do you do one test or perform continuous testing?

b) What is, in your opinion, the best time to perform verification?

c) Do you use your specification documentation during verification?

Note: Clarify change of topic

Questions covering topics concerning the integration process:

(A) Interdisciplinary development handling (4 minutes/38/4)

I-1. Is there a clear separation between different departments different development processes?

I-2. What departments/actors are involved in interdisciplinary development?

I-3. Can you explain your structure?

(A) Dependency (2 minutes/40/6)

I-4. How much dependency is there between different development departments?

a) Does one department have to await results of the other?

b) Are there any clear bottlenecks?

(A) Modification handling (2 minutes/42/8)

I-5. Are you successful in handling modifications during interdisciplinary development?

a) (Are there any methods/processes/techniques in place?)

(A) Reliability (2 minutes/44/10)

I-6. If you compare different development processes, which do you feel is more reliable? Why?

VI
a) Do you feel that you could benefit from development processes currently in use in other departments?

b) Do you have any own development processes you believe other departments could make use of? (which?)(how?)

(B) Difficulties (2 minutes/46/12)

*Note: Bring up difficulties mentioned previously in the interview*

I-7. Are there any additional difficulties you feel you are faced in interdisciplinary development projects which have not yet come up?

a) (How do you handle these difficulties?)

(B) Future expectations (7 minutes/53/19)

I-8. What are future expectations for integrated development?

a) Long-term or short-term?

b) What obstacles exist?

c) Why do you believe this? What enablers exist?

I-9. (Is there any specific person in charge to handle future possibilities of interdisciplinary development?)

a) (If yes, what do they do?)

b) (If no, do you think your company would benefit of having such a role and what do you think they should do?)
(B) Wrap up (3 minutes/56/22)
Is there anything you have been waiting for us to ask you?

Is there anything you want to ask us?

Thank you for your time and participation!
Appendix C. Original interview questionnaire

Questions:
(A) Warm-up (3 minutes)

• Field of work now and previously
• Experience

Questions covering topics concerning requirement engineering:
(A) Requirements elicitation (9 minutes)

R-1. Which actors are important in your requirement process?
   a) How are important actors identified?
   b) Are there any key actors?
   c) Are original actors stated in the requirement specification? (traceability)
   d) Are these actors revisited in future development processes? (later in the same process)
   e) To what extent are your requirement influenced by norms and standards?

R-2. What technique do you use to collect requirements? Why?
   a) Do you work differently with requirements depending on their origin? Say external/internal requirements?

R-3. What strengths and weaknesses do you see in your requirement collection method?

(B) Requirement categorization (7 minutes)

R-4. How do you organize product requirements?

R-5. Which are the main requirement categories/types your development area encounters?
   a) Do any unique requirement types in your area effect the methodology or process?

R-6. How do you manage requirements which are given by actors once development has already started?
   a) How do additional requirements appear? From where?
   b) How do you handle additions depending on time and actor? (beginning, middle, end)(prioritization)

R-7. Do you use same approach to all requirements?
   a) Does this approach also apply to requirements which concern many areas? (interdisciplinary)

(A) Requirement documentation (7 minutes)

R-8. How and when do you document requirements? (who is responsible) (make sure to note if these questions have already been answered in previous questions)
a) Are requirements maintained throughout the process?
b) Is the requirement specification seen as a living document or set in stone?


a) Is there a unified standard? Or a department-specific standard? Is it an open standard which we could find through for example IEEE or an internal version?
Examples of standards: SRS (system requirement specification) PDS (product design specification)

R-10. What advantages/disadvantages can you see in your type of documentation?

R-11. Are specifications for other departments (in other disciplines) easily accessible?

a) If yes, is this useful? Why/Why not? Understandable? Important?
b) If no, would you like to have better access? Why/why not?

R-12. How is the requirement specification used during development?

a) How is the traceability during the development process, is the link between requirements and product clear?

(B) Requirement modification (2 minutes)

R-13. Do you have a modification handling methodology in place?

a) If yes, could/should describe it briefly
b) If no, do you think you could govern from a modification methodology, and what would it include?
c) Do you believe this methodology is well suited for handling modifications? Why/Why not?

(B) Requirement verification (6 minutes)

R-14. How do you make sure you don’t have any conflicting requirements in your specification? (Responsible person?)

R-15. How do you make sure that the customer needs are correctly reflected in the requirement specification?

R-16. How do you check that the system fulfills the requirements?

R-17. Is it evaluated during the whole process if the system complies with the need?

R-18. Do you continuously check the system towards the requirements?

a) Do you do one test or perform continuous testing?
b) What is, in your opinion, the best time to perform verification?
c) In what way do you use your specification documentation during verification?

Note: Clarify change of topic

Questions covering topics concerning the integration process:

(A) Interdisciplinary development handling (4 minutes)

I-1. Is there a clear separation between different departments different development processes?

I-2. What departments/actors are involved in interdisciplinary development?

I-3. Can you explain your structure and place yourself in it?

X
(A) Dependency (2 minutes)
I-4. How much dependency is there between different development departments?
   a) Does one department have to await results of the other?
   b) Are there any clear bottlenecks?

(A) Modification handling (2 minutes)
I-5. Are you successful in handling modifications during interdisciplinary development?
   a) Are there any methods/processes/techniques in place?

(A) Reliability (2 minutes)
I-6. If you compare the different development processes, which do you feel is more reliable? Why?
   a) Do you feel that you could benefit from development processes currently in use in other departments?
   b) Do you have any own development processes you believe other departments could make use of? (which?)(how?)

(B) Difficulties (2 minutes) Note: Bring up difficulties mentioned previously in the interview
I-7. Are there any additional difficulties you feel you are faced in interdisciplinary development projects which have not yet come up?
   a) How do you handle these difficulties?

(B) Future expectations (7 minutes)
I-8. What are future expectations for integrated development?
   a) Long-term or short-term?
   b) What obstacles exist?
   c) Why do you believe this? What enablers exist?
I-9. Is there any specific person in charge to handle future possibilities of interdisciplinary development?
   a) If yes, what do they do?
   b) If no, do you think your company would benefit of having such a role and what do you think they should do?

(B) Wrap up (3 minutes)
1. Is there anything you have been waiting for us to ask you?
2. Is there anything you want to ask us?
3. Thank participant for participation!
Appendix D. Relation between research and interview questions

The questions removed by the end of the thesis interviews are represented by struck through checkmarks in the table. Checkmarks in parenthesis implies uncertainty, or dependency on chosen answer from interviewee, if the question actually contributed towards answering one of the research questions.

<table>
<thead>
<tr>
<th>RQ</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Purpose of question</th>
<th>Expected answer type</th>
</tr>
</thead>
<tbody>
<tr>
<td>R-1</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Establishment of, to the company, relevant actors</td>
<td>Actors and their individual importance</td>
</tr>
<tr>
<td>R-2</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Establishment of existing elicitation techniques</td>
<td>Methods in place and reasons behind them</td>
</tr>
<tr>
<td>R-3</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Highlight strengths &amp; weaknesses in the elicitation methodology</td>
<td>Evaluation of elicitation method</td>
</tr>
<tr>
<td>R-4</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Establishment of existing requirement categorization</td>
<td>Different categories used or no answer</td>
</tr>
<tr>
<td>R-5</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Discovery of, to the department, relevant requirements</td>
<td>Areas of requirements interviewee encounters</td>
</tr>
<tr>
<td>R-6</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Requirement modification handling, and prioritization</td>
<td>Examples of methodologies and motivations on prioritization</td>
</tr>
<tr>
<td>R-7</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>(✓)</td>
<td>(✓)</td>
<td>Discovery of treatment of cross-domain vs own domain requirements</td>
<td>Same or comparison</td>
</tr>
<tr>
<td>R-8</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Establish the how and when of requirement documentation.</td>
<td>Description of requirement documentation process</td>
</tr>
<tr>
<td>R-9</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Identify documentation standard</td>
<td>Description of used standard</td>
</tr>
<tr>
<td>R-10</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deepen understanding of documentation</td>
<td>Pros and cons of documentation</td>
</tr>
<tr>
<td>R-11</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>See current integration between departments concerning requirements</td>
<td>Yes/no, pros and cons</td>
</tr>
<tr>
<td>R-12</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine use of specification during development process</td>
<td>Description of specification vs development process</td>
</tr>
<tr>
<td>R-13</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Establish if methodology exists for modification handling</td>
<td>Yes/No, description of existing or wanted method</td>
</tr>
<tr>
<td>R-14</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Probe of responsibility for coherency in specifications</td>
<td>Method/process and responsible actor</td>
</tr>
<tr>
<td>R-15</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Determine use of specification at end of development process</td>
<td>Process description</td>
</tr>
<tr>
<td>R-16</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See if requirement is used as control for system</td>
<td>process or methodology</td>
</tr>
<tr>
<td>R-17</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Determine if process continuous</td>
<td>Yes/No why</td>
</tr>
<tr>
<td>I-1</td>
<td>✓</td>
<td>✓</td>
<td>(✓)</td>
<td>(✓)</td>
<td></td>
<td></td>
<td>Clarify division between HW and SW</td>
<td>Yes/No possible explanation</td>
</tr>
<tr>
<td>I-2</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Understand extent of involvement</td>
<td>Listing of involved parties</td>
</tr>
<tr>
<td>I-3</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Understand structure</td>
<td>Explanation of structure</td>
</tr>
<tr>
<td>I-4</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Understand dependency HW/SW</td>
<td>Description of possible overlaps and dependencies</td>
</tr>
<tr>
<td>I-5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Identify common/interdisciplinary modification handling</td>
<td>Yes/No, possible methods</td>
</tr>
<tr>
<td>I-6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Reflection upon other’s process</td>
<td>Positives from other, possible negatives from own area</td>
</tr>
<tr>
<td>I-7</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Allow interviewee to add missing information on difficulties</td>
<td>Difficulties and the handling of them</td>
</tr>
<tr>
<td>I-8</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Get a prospect from interviewees on the future</td>
<td>Expectations, obstacles and enablers</td>
</tr>
<tr>
<td>I-9</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>See if a role exists and responsibility is given for the future</td>
<td>Yes/No, why, what they do</td>
</tr>
</tbody>
</table>
## Appendix E. Codewords sorted by origin

| Differences and similarities | “our” process  
commonalities  
comparison  
constraints  
relations  
inspiration from  
limitations  
special features  
unique principles |
|-------------------------------|---------------|
| Research questions            | collab  
cooperation  
dependency  
difficulties  
diversity  
integration ideas  
interdisciplinary modification methodology  
interdisciplinary projects  
reliability |
| Integration state             | directives (from management)  
driving force  
forecasting methods  
involved actors  
key factors  
previous missteps  
trend  
validity time |
| Future obstacles and possibilities | House of Quality (HoQ)  
prioritization  
proceeding criteria (steps)  
Quality function deployment (Qfd)  
relationship between requirements  
requirement of requirements (RoR)  
requirements specification (criteria)  
selection criteria (model)  
steps involved (model)  
techniques (handling requirements) |
| Hardware development          | domain  
functional  
service (PSS)  
steps (process) |
| Software development          | approach  
components  
element  
needs  
phases  
product development life cycle  
user |
<p>| System engineering            | Theoretical framework |</p>
<table>
<thead>
<tr>
<th>Guide topic integration process</th>
<th>actors</th>
<th>elicitation methods (interviews, questionnaires, documentation analysis)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement engineering</td>
<td></td>
<td>iterative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requirement characteristics</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requirement documentation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requirement validation</td>
</tr>
<tr>
<td>Requirement elicitation</td>
<td></td>
<td>actors</td>
</tr>
<tr>
<td></td>
<td></td>
<td>elicitation technique</td>
</tr>
<tr>
<td></td>
<td></td>
<td>execution</td>
</tr>
<tr>
<td></td>
<td></td>
<td>norm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>standards</td>
</tr>
<tr>
<td>Requirement categorization</td>
<td></td>
<td>categorization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>external interface</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Non-functional</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Other</td>
</tr>
<tr>
<td></td>
<td></td>
<td>system features</td>
</tr>
<tr>
<td>Requirement documentation</td>
<td></td>
<td>department specific</td>
</tr>
<tr>
<td></td>
<td></td>
<td>documentation methodology</td>
</tr>
<tr>
<td></td>
<td></td>
<td>documentation standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td>unified</td>
</tr>
<tr>
<td>Requirement verification</td>
<td></td>
<td>consistency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>continuous verification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>validation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>verification</td>
</tr>
<tr>
<td>Requirement modification</td>
<td></td>
<td>modification feasibility</td>
</tr>
<tr>
<td></td>
<td></td>
<td>modification feature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>modification methodology</td>
</tr>
<tr>
<td></td>
<td></td>
<td>modification misspecification</td>
</tr>
<tr>
<td>Interdisciplinary development handling</td>
<td>hw dev</td>
<td>interdisciplinary actors</td>
</tr>
<tr>
<td></td>
<td></td>
<td>structure</td>
</tr>
<tr>
<td></td>
<td>sw dev</td>
<td></td>
</tr>
<tr>
<td>Dependency</td>
<td></td>
<td>dependency</td>
</tr>
<tr>
<td></td>
<td></td>
<td>dependency issues</td>
</tr>
<tr>
<td>Modification handling in interdisciplinary teams</td>
<td></td>
<td>communication</td>
</tr>
<tr>
<td></td>
<td></td>
<td>compromise</td>
</tr>
<tr>
<td></td>
<td></td>
<td>interdisciplinary modification methodology</td>
</tr>
<tr>
<td></td>
<td></td>
<td>time frame</td>
</tr>
<tr>
<td>System reliability</td>
<td></td>
<td>comparison</td>
</tr>
<tr>
<td></td>
<td></td>
<td>obstacles reliability</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reconnection</td>
</tr>
<tr>
<td></td>
<td></td>
<td>reliability techniques</td>
</tr>
<tr>
<td>difficulties</td>
<td></td>
<td>difficulties</td>
</tr>
<tr>
<td>future expectations</td>
<td></td>
<td>enablers future</td>
</tr>
<tr>
<td></td>
<td></td>
<td>long term</td>
</tr>
<tr>
<td></td>
<td></td>
<td>obstacles future</td>
</tr>
<tr>
<td></td>
<td></td>
<td>responsibility</td>
</tr>
<tr>
<td></td>
<td></td>
<td>short term</td>
</tr>
</tbody>
</table>
Appendix F. Codeword summary company W

Company/interviewee background

Summary of codewords from interviews from company where majority of interviewees were software developers.

Main take-aways from this company

- Hardware and software development very separate, as different organizations.
- Understand that process needs to be adapted to current development.
- Process at interviewed department strongly adapted to software development.
- Believe in decreased dependency.
- Believe in continued separation between different departments.
F.1 Actors

- Customers
- Customer unit
- Product managers - in charge of money/budget
- System managers
- Product developers
- System verification - testers
- Researchers
- Product management team

Key actors are defined by the process

F.2 Steps of process or model

A development process starts from a request for a new product or, if a problem is discovered, a fix of an existing product. It can also originate from the sighting of a business opportunity. Requirements originate from customers through the customer unit. The product developers themselves have little direct contact with customers. When requirements arrive at product management they are at a high level, formulated as a statement, which is generally very broad. Main requirements are put into an organizational system, an action that makes the requirements formal. After this they are then divided, broken down according to subsystem. In this step a feasibility study also is carried out on the requirements. Time lines and customer needs are frame on the basis of project time limitation. Requirements are developed into more descriptive requirements with measurements. These low level requirements are seen as a mutual understanding between all (development) departments and the customer. During the process, actors might be revisited for clarification. Once requirements are spread out in subsystems, KPIs are monitored (not the requirements themselves). During the requirement process there are certain checkpoints, including a preliminary review, before there is a freeze phase. Similarly there are checkpoints with test managers towards the end of the process, where system verification takes place. Being costly, some verification cannot take place several times.
F.3. Methodology

F.2.1 Time frame
No interview data responding to this codeword.

F.3 Methodology

- Use kanban
- Use scrum
- Work in releases
- Work in sprints
- Use user stories
- Monthly or weekly deployment
- Work with patching
- Have use cases
- Work with flows
- Work iterative, allow for cross department requirement definition, application, and revisions
- Try to look to which products and actors are impacted by a process.
- Requirements provide a blueprint, or a framework
- A requirement can not be added without prior review
- There is no process that is solely good, a suitable process needs to be chosen for the current development.
- The specific approach (for testing) is in place to check which requirements are fulfilled or not fulfilled.

F.3.1 Iterative and agile methods
No interview data responding to this codeword.

F.4 Understanding
No interview data responding to this codeword.

F.5 Elicitation methods
Requirements are collected through communication and based on experience. Those with experience can see connections, and dependencies.

F.5.1 Customer interaction
Departments could benefit from closeness to product management and the customer. Even if the customer is an important actor, developers seldom have any direct interaction.
F.5.2 Strengths & weaknesses

1. Strengths
   A good set of tools for handling requirements. Better control to see what’s going on and why.

2. Weaknesses

F.6 Requirement analysis

Reviews are performed with system managers, in these reviews duplicates and conflicts are discovered. Conflicts are resolved through merge, removal, or assignment of requirement to a particular department.

F.6.1 Categorization and prioritization of requirements

Looking to the origin of requirements, the customer always comes first. As the requirements go through different levels, they are prioritized differently. Product managers prioritize among requirements and then hand them over to system managers who have other priorities. After system managers come kanban teams and their leaders which work with priority levels such as “high” and “low”. On the lowest level are the product developers who work using example wise scrum and they prioritize based on their internal criteria. When working between departments and levels in this way, it is necessary to have a common understanding to make a correct prioritization. That the list of things which could be implemented is without end, further promotes the need of prioritization.

F.6.2 Balancing requirements

No interview data responding to this codeword.

F.6.3 Setting targets

No interview data responding to this codeword.

F.7 Limitations & obstacles

There are a lot of people involved in the process and information gets filtered and reinterpreted a lot. The communication from customer to product manager is not straight forward, and the actual product developers are far away from the customer. They rely a lot on the main requirements, but also “hunches”, which they are not certain are correct.

F.8 Requirement documentation and characteristics

The requirement specification acts as a container, or a guardian tool. The requirement documents are internally standardized, and the organizational system/tool is used across the whole company. There are the main requirements, which are input to the organizational system, and then there is the characteristics requirement specification (CRS) which serves as a container for requirements in a specific area. It can contain, among others; performance, latency, and robustness. The system managers have their own set of more detailed requirements which is used to see which requirements are being tested, how they could be implemented, and use cases for developers. There are also functional requirements, which are used directly by product development. The requirements contain a requirement ID which has a number that allows it to be identified from which area the requirement originates. Requirements also have slogans, descriptions, and pass criteria. The requirement specification document treated as living document.
F.8.1 Requirement handling techniques
All requirements are treated equally. In the organizational system they make use of tags. They also work in versions, everything is version controlled. It is up to the separate development teams to decide exactly how they work with requirements.

F.8.2 Relationship between requirements
No interview data responding to this codeword.

F.8.3 Requirement types
All requirements are equal. External requirements are separate from the internal ones. The internal requirements work together to fulfill the external ones. Functional and non-functional requirements are kept separate. It not always that non-functional requirements are used during the development. Depends on what kind of requirement it is if requirements are used for testing during development or in a later stage.

F.8.4 Impact of standards
Some requirements, such as those based on standards, are very strong requirements, so everyone is aware of them. Not all requirements are specified (value) more than that they need to fulfill the requirement set by the standard.

F.8.5 Impact of legal factors/legislation
There are lots of legal requirements. Intellectual property rights also need to be taken into consideration when developing the system.

F.8.6 Advantages & disadvantages(documentation
1. Advantages
   The administration of documentation is good. Ease of access due to segmentation.

2. Disadvantages
   Inflexible and difficult to handle in terms of change something.

F.9 Traceability
At the same time as it is not seen as acceptable to be able to identify the from which company (customer) a requirement originated, the reason is to keep secrecy because involvement of lot of stakeholders could cause risk. Although there is some traceability in the organizational system. There is however a lack of a so called birds eye view or overview of the connection between requirements on different levels.

F.10 Modification methodology
Modifications require change requests. These can originate from both the internal iterative process and from external actors. Change requests are processed regardless if change takes place or not. Most usually modifications only take place if the change request appears due to an issue. If for example an issue is discovered in development, it’s taken back up the loop (see the process description). Project managers are in charge of deciding if a modification will take place. Depending on who makes the request, different decisions might be made (important customers for example). The project managers look at the requests and see their impact, how much time is needed for implementation, if the change fits in...
the project, and the importance of the change. They also consider the effect on existing priorities. The project managers can decide to postpone a fix to a future release, making the current release limited, or to not do the change. If the release is limited, correction packages are rolled out later. System managers are in charge of tracking these packages and keeping the product up to date. In a large organization, change requires a lot of administrative tasks. If the modification takes place closer to the design however, it does not need to be an equally extensive process. The development can not however just go in and perform changes as this gets messy.

F.10.1 Interdisciplinary modification methodology
In the case of interdisciplinary modifications, a group where all characteristic knowledge is represented meets every 2 weeks. Issues outside the own domain are brought up, identified, classified, a solution agreed upon which can then be executed by the separate teams.

F.11 Verification methodology
There is a separate department (actor); system verification, which has sub-departments working with test of CRS, robustness, stability. There is a variation of verification structures, tests, verification activities etc. As soon as you have something to verify, tests can be performed.

F.11.1 Testing
All requirements are used during test, either in product development tests (small-scale tests which take place in the development process) or verification tests (test which takes place after the development process). Testing can be planned already at the stating of requirements, defining equipment and tools for the tests. If issues occur during testing, trouble reports are generated. A critical trouble report can halt the whole process, but the most common procedure is to loop back up as far as necessary in the process chain. Tests can be performed isolated or in a more realistic (complicated) environment. In the test lab test models can be created which simulate the behavior of the customer. A template of tests are run on a daily basis. Test driven development (TDD) methodology is in place to perform testing.

F.11.2 Continuous verification
Various requirements are checked at different times. The development teams create their own tests for the requirements.

F.11.3 Relation between requirements and the product
No interview data responding to this codeword.

F.12 Validation techniques
No interview data responding to this codeword.

F.13 Roles & responsibilities

- Everyone has their own part of the process. Development only sees their own part, while architects and performance engineers need to see a more complete picture of the product.

- It is the responsibility of the product manager to make sure that customers’ needs are correctly reflected in the requirement specification.

XXII
• System managers sometimes are invited to participate in studies of high level requirement to increase the level of understanding.

• System and test managers are responsible for discovering/avoiding conflicts between different requirements.

• System managers are in charge of communication with middleware and platform.

• There are "guardians" for non-functional requirements who are in charge of testing, verifying, and synchronizing these requirements.

• People responsible (for functions) are involved during review of tests.

• Requirements are added and tag by product managers, and also responsible for version controlling.

They work with more of a waterfall structure

F.14 Interdisciplinary projects & communication

It is hard/untrivial to see the connectivity between hardware and software, the impact/influence they have on each other. Even if hardware is upgraded, software might not be able to utilize the upgrade. Some people have an overview and see and understand the connection between several different "boxes", others see nothing but their own "box". It is also challenging to keep a balance between several different actors. All different actors need to cooperate and be adaptive to each other. In an interdisciplinary project, if you have no need for certain information, then access to it is obsolete. By utilizing a personal network, one gets more relevant (filtered) information. Communication is key, understanding the underlying "why".

F.14.1 Interactions

No interview data responding to this codeword.

F.14.2 Relations

No interview data responding to this codeword.

F.14.3 Interdisciplinary actors

The separation between different development departments is extremely well defined. And this is necessary, due to the size of the company. If the company was smaller, the need wouldn’t exist. Different types of hardware and software focused departments. There is also a physical separation between system managers and product developers, making their interactions similar to interdisciplinary. There is role of moderator in place who bridges the gap between different actors.

F.15 Dependency

There is one dependency between development departments at a high level in the early phases. It comes with challenges such as needing something from another department, that is not the other department’s highest priority. Something with a low priority might effect the system as a whole, requiring it to be given higher priority. Another dependency is that between different development teams once development has started. These development teams can be at different geographical locations. Due to inter-dependencies between different levels of the product, chains of dependency are formed, leaving teams which are dependent upon other teams with their own dependencies in turn, to wait more than the average development team. Generally parts upon which many other parts build become bottlenecks.
F.16 Differences

- Hardware development is less iterative due to a longer turnaround time, at a much higher cost.
- Hardware development is hard to understand for a software developer because it is a very different domain.
- In a waterfall development process, each developer knows their own product part, in agile process, in-depth knowledge is lost.
- Hardware need to be more careful in the break down of requirements, if something is lost or missed it is very expensive.

F.17 Difficulties

The breaking down of requirements is a big challenge, it is difficult to manage to cover all major things. It can be difficult to specify requirements (set targets). A too strict requirement is still possible to achieve, but at an unreasonable amount of effort. As fuzzy requirements are decomposed, both into sub-requirements and spread in the organization, it is hard to keep track of dependencies in the system. If a change is done in one branch, how can this be tracked back and correct requirements be highlighted for change in other branches? It is hard to write testable requirements. Some requirements are impossible to test. The interaction with people is seen as a difficulty by this company. Especially concerning connections outside the organization. Synchronization between fields is difficult. Systems are so big that a small number of people can not handle them (on their own). It can be hard convincing others of the existence of problems. Hardware is a completely different organization (according to SW) The shift from waterfall to agile approach in development demands expertises and efficiency in development environment, but right now they have mixed (waterfall & agile) development environment.

F.17.1 Inter-cultural work

Difficulties due to work across boarders include language barriers, time differences, and cultural differences.

F.18 Future approaches

There is a research department which looks far into the future, 7-8 years. What they research is decided by the CTO.

- There is a long term goal to cut the costs of development and this will be done by decreasing the amount of dependencies.
- Independent components
- People will still work in separate sub-systems
- People interfacing between sub-systems, and different departments, are important
- Product development teams should have more interfaces for improved context
- A person is needed to bridge the gap between software and hardware development, which currently are two very different organizations.
- See what new can be added from technology, push ideas upwards in the loop.
- Customers asking for features is one of the main drivers.
• Need a sensitive person with a broad network who can pick up on hints from other departments on for example mismatch of requirements.

• Working with the future is difficult as it is unknown.

• Another difficulty is that there are a lot of people and all have their own needs (not a straight clear path into the future).

F.18.1 Obstacles future
The technology are changing rapidly so the prediction about future is difficult.

F.18.2 Enablers future
Better communication and interfaces between departments are important.

F.19 In a perfect world
No interview data responding to this codeword.
Appendix G.  Codeword summary company X

Company/interviewee background

Summary of codewords from interviews from company where interviewees had mixed backgrounds, mechanical and electronic engineering.

Main take-aways from this company

- Existing interdependency in the shape of multi-objective design/optimization.
- The product is more than just the physical product.
- All processes originate from a core process.
- Understanding of limitations of different development departments. Physical aspects for hardware development.
- Traceability needs to be let go at some point.
Typology

In this company the words "attribute" and "functions" occurred frequently in the interviews. Customers interact with attributes, examples of attributes are responsiveness, comfort and physics of a system, also known as touch and feel. Attributes can be realized by either systems or functions. Functions and attributes however, can not be put into a hierarchy. A product will need functions to perform the wanted task, and these functions will contribute to attributes.

G.1 Actors

- Customers
- Product owners
- Function owners
- Product planning
- Legal requirements
- Security
- Safety
- Brand
- Durability
- People responsible for both attribute and function
- All stakeholders are connected to the functionality

G.2 Steps of process or model

Everything originates from the same general process. Before a project begins, requirements are defined. Based on input from customers, general requirements for technology are set, and then iterated. Every department delivers their requirements. Different departments provide different requirements. Product planning provides product attributes, other groups provide technical attributes and performance requirement. Product development confirms which requirements need to be fulfilled. Once the requirements are given to the product development, they create a development toolset and work with the requirements up until 3 months into production. Requirements are broken down to the different systems. Attributes which include target setting and realization of common activities between departments, are split into systems and functions. Functions in turn are used to make function lists, use cases and specific requirements, they use use cases to test the functional requirements. After this division of requirements the process of developing solutions begins. Product development are responsible for checking if requirements are fulfilled or not during verification. During the process there are gates. Requirement objects are exported (from organizational system) for review. Work with "handshake", and only things previously "OK:ed" (higher in hierarchy?) should be handshaked upon. Most plans are used only as a mean to move forward, and are easy to change. The process needs to be flexible. Requirements are given to suppliers, no requirements are given by suppliers. The process is set up to work in parallel, with software and hardware working separately.
G.2.1 Time frame

The concept phase is usually 6 months, if it is only an edit of an existing product it is 3 months. Implementation can take up to 1 year. Services usually have shorter time frames. Some components of the product are needed earlier, and therefore need to initiate development earlier. Early in the process, things are more unstable since facts are lacking. Later in the process less changes are possible. Requirements mature as time passes.

G.3 Methodology

There is serial collaboration and parallel collaboration. In parallel collaboration obstacles are identified as early as possible. When working with complex products, they start with processes which have long lead times and input processes with shorter lead time into schedule later. They have a general time plan from which all projects originate. The time plan is customized according to project. Use a platform way of thinking, reuse knowledge. There is always a need to re-plan, as no project goes according to plan.

G.3.1 Iterative and agile methods

Every process has to be constantly adjusted. For example when working with requirements, first state general requirements, then "deep dive", further research them, and then update the original requirements according to research. Repeat.

G.4 Understanding

Software and hardware are only part of the product. You need to understand the system, with all its complexity: functions, attributes, systems. In the process one needs not only to extract knowledge, but also understanding. One needs to understand behavior, and implement separate versions during verification.

G.5 Elicitation methods

The user comes first. For the elicitation of attributes they do customer investigation. They interview target customer, have workshops with randomly selected users from different social groups and incomes
represented. The team center(environment) is used to split attributes into functions and systems, the functions consist of functions definition. New technology can also contribute to requirements. Collected data is input into a product guide.

G.5.1 Customer interaction
No interview data responding to this codeword.

G.5.2 Strengths & weaknesses
1. Strengths
   The elicitation of requirements is done in structured way.

2. Weaknesses

G.6 Requirement analysis
There is a lot of breakdown from what to how. Can it be afforded, both looking to money and time? How does it scale? What can be changed, and what is actively not changed? It is important to get the full picture from different views. What is best for the customer? What is important to the customer? Calculations
   Need to understand different abstraction levels of functionality, and the perspective of the user. Is the requirement realizable? Requirements change depending on the level of development they are at. Also verification depends on the abstraction level of the product.

G.6.1 Categorization and prioritization of requirements
Answer the question "What requirements do you need to fulfill?". Have "trust marks" - general "must" requirements. There are mandatory requirements. They prepare separate list of cross functional requirements(CFR) to understand different abstraction levels of functionality,

G.6.2 Balancing requirements
With a complex system, there are a lot of requirements and there is always a need for balancing. The final output is always a compromise. Balancing is important, as not all things can be edited in a project. There will be several solutions, and only a balance of requirements can show which is the optimal compromise.

G.6.3 Setting targets
They test for requirement level, what target to achieve. Interviews give needs and level wanted in the product. There needs to be a gap, it can not be too easy to fulfill requirements. When plans are made, they are for support, and do not represent the final goal.

G.7 Limitations & obstacles
No interview data responding to this codeword.

G.8 Requirement documentation and characteristics
The main theme which they follow while writing about requirements is how requirements are describe in best understandable way. The requirements are divided between different organizational systems which are used for storing requirements. Some departments have their own system for requirements.

XXX
The requirements are written in the same way, which means the requirement documentation is unified. It is however stored and used in different ways depending on department. The organizational system contains base documents and drawings requirements contain the following information:

- Text
- Measure
- Method of verification
- Robustness
- Validity of reliability

Requirements are sorted by function, by system, and by relation.

G.8.1 Requirement handling techniques

It is hard to manage the high amount of requirements, it requires balancing and compromising. The balance between attributes, and balancing performance is also hard. When requirements are agreed upon they become "handshake requirements". It is important to get the requirements down to the department/supplier level. The team center is use as a base requirements document along with documents portal.

G.8.2 Relationship between requirements

A new function might affect 20 of 200 requirements. A new version of the product affects 120 of 200, which in turn affects so many relations, that it is equal to changing all requirements. Having a good view of the relationship between requirements, allows for avoidance of conflicting requirements. When stated, requirements on hardware and software need to be separate, not dependent. It is important to reflect over how requirements should work together.

G.8.3 Requirement types

- Attribute
- Function
- System
- Hardware requirements
- Software requirements
- Service requirements
- Performance requirements
- Functional requirements
- System requirements
- Reliability

G.8.4 Impact of standards

Mandatory standards are handled by standard management, which handle DIN and ISO-standards. Making use of a standard library, requirements refer to standards. The standards are usually focused on implementation and safety of systems.
G.8.5 Impact of legal factors/legislation

Legal requirements are external requirements. Legal requirements are treated differently, they are not always formulated so they fit into the existing structure of requirements but they are still distributed among part systems and functions as well as possible. Legal requirements need some extra attention, and should be checked verified already at concept level. The legal department can make use of legislation and give other development departments a heads up on requirements which will need to be fulfilled in the future.

G.9 Traceability

The Original requirement givers are stated in the requirement specification. Traceability is easier looking bottom-up, when looking down from the top level, traceability is lost at some point, i.e. why they do not consider it as an easy task. At this point however, someone has the responsibility of fulfilling the requirement, so a more detailed traceability is not needed.

G.10 Modification methodology

A process/product undergoes constant evolution. Outside changes can not be controlled. At design review forums/meetings - the concepts regarding cross functional requirements are discussed and design concepts are reviewed and it is decided if a change of function is acceptable or not. How a change is handled depends on which company or area from which it originates. It can be decided to change the product, temporarily change the requirements, or permanently change the requirements. If requirements are always too high to reach, maybe they should be changed. However they also might be necessary, and if a temporary change of requirements is not possible, the request for change needs to be turned down. If the change is big, the project needs to be dialed back to the concept stage. If a requirement is not fulfilled, the solution needs to be reworked until the requirement is fulfilled. If a change takes place, certain rework is necessary, and tests which have already been performed need to be performed again. The project scale(duration) has a key role to perform modification, they have 6-month duration in this regard otherwise they apply the concept of reuse.

G.10.1 Interdisciplinary modification methodology

G.11 Verification methodology

There is a person in charge of verification. Customer environment testing, durability testing, experience, feedback, repairs, data from workshops, old drivers are used for collecting data. The requirements are used during verification. Virtual development is verified virtually, some parts may require physical verification but the aim is to only use virtual verification at an early stage. After this a verification prototype is created. Test systems, test rigs, and standards are used for verification. Verification should take place all the time, but should be scheduled based on the time it takes to perform a change. The timing of verification also depends on the abstraction level. Model different solutions. Trade-off curves There needs to be a check if the product can be built, and what needs need to be answered by the product. The conflicting requirements are refined at sub-system level and implementation level.

G.11.1 Testing

Test methods are decided by a system responsible. For testing test systems, and test rigs are used. Perform more computer testing, find more efficient ways for testing. Certain departments need the whole product to be finished to be able to perform testing, while others can perform isolated tests. Testing also depends on what type of requirements are set for the product. Some requirements are
functionality dependent, and testing can only be performed once the appropriate functionality is offered by the product.

**G.11.2 Continuous verification**

No interview data responding to this codeword.

**G.11.3 Relation between requirements and the product**

There is a good traceability between legal requirements and the product, but in other cases, a person in charge of verification has the knowledge/understanding of the connection between requirements and the product. Different requirements result in different functions.

**G.12 Validation techniques**

There is an annual measure of user reaction which is partially indexed. They collect the perception of the product at different times. There is also an so called internal customer, which tries to validate. The internal customer represents the (external) customer, and does not work directly with requirements but with the feeling. (perceived product)

**G.13 Roles & responsibilities**

Program management work with budget and makes sure the right attributes are being built. Below program management is technical product management and mechanical processes. In the requirement specification it is stated who is responsible for the requirements. A person is responsible for verification. There are 4 people responsible for both attributes and functions and function analysis. Product managers are in charge of high level requirements. System responsible handles the overview made up by functional lists with user perspective including customer functions, and system perspective including product functions. Subsystems and PSS are responsible for delivering parts and working internally with requirements. The company is responsible for the product as a whole in the end.

**G.14 Interdisciplinary projects & communication**

Software can compensate for failures in hardware. Between some sub-functions there is very close interaction. It needs to be determined what collaboration is needed. Collaboration and communication to find consensus always takes place. Communication is an important part of the development process.

**G.14.1 Interactions**

In such a complex product, multi-objective design optimization is necessary. One has to consider all requirements and their interactions.

**G.14.2 Relations**

How can a system function if you do not know the relation between different parts? The approach to requirements is about extracting knowledge and seeing multi-objective relations. Some things can not be modified due to the amount of difficulties which occur.
G.14.3 Interdisciplinary actors

They mentioned about list of attributes here e.g. noise vibration harshness (NVH), durability, safety, design, etc., attribute leaders. Functions has function owners. Systems has system responsible. Many different departments in very detailed sub-functions.

G.15 Dependency

All different development departments are dependent on each other. Today, even the complete product (physical) is just a part of the product (including services). Interdependence between departments is very important. There are some very close interactions and relations.

G.16 Differences

Hardware development feels there is a need to learn more about software changes. How requirements are written and described is very important and is not currently universally understandable. Different timing on different levels is necessary for finding new capabilities. In hardware, physics has spin-off effects. Programming allows exactly defining functions. IT and electrics allows for application of parallel collaboration, which is robust and agile. Physic aspects make agile development difficult in hardware development. Safety requirements are different for hardware and software. Developers break down requirements in different ways to suppliers at different levels. Inter-culturally it differs how it is handled if a requirement is not met/reached.

G.17 Difficulties

A small company on a worldwide market results in it being hard to collect qualitative data from all relevant areas, and it is also not possible to develop for all different regions even if many factors in input requirement differ. It is hard to break down requirements in a correct way due to the many interactions between different parts. Need to understand interactions and relations, where the best solution will be found, despite variation in results. Once the construction of the product has begun, certain things no longer can be changed. Needs to safeguard that all combinations work for the customer (when offering several versions of a product). Outside changes can not be controlled. Some times there is not a full understanding of other departments' requirements or the relationships which exist between requirements. The software and hardware people work in isolated ways. The accuracy in models need to be implemented in early stages of product development. The design need to be more mature than present. The forums are used to find solutions, platform are used to solve certain problems, reuse of knowledge.

G.17.1 Inter-cultural work

No interview data responding to this codeword.

G.18 Future approaches

Need a product/process which is simple but robust, insensitive to disturbances, modular and possible to reuse, and evolutionary so it is possible to build upon over time. The number of functions is increasing, increased integration of IT and other industries in previously product focused processes. Close collaboration and integration. Connectivity as a mean to integrate IT. Function is stepping outside beyond the product. Increase traceability Use meta-model to see correlation between objects. Need to see the big picture/ have an overview. Want more modulized. Module interface model. It takes a long time before everything can be implemented, so they start with the key 2%, the 2% which separate you from XXXIV
your competitors. Which differentiates the product. Want to see improvements in their modification process in terms of assigning different time slots to perform changes. Need to learn more about software changes, testing, and feel need of more agility in their processes. The SW & HW people need to come close to each other to make interdisciplinary development effective and organized. Interdependencies between requirements. Need flexibility in modification.

G.19. In a perfect world

In an ideal situation, all departments would be able to work completely separate from each other. The processes would run in parallel and be migrated after a certain time.

G.18.1 Obstacles future

G.18.2 Enablers future

Interaction and collaboration between departments,

G.19 In a perfect world

In an ideal situation, all departments would be able to work completely separate from each other. The processes would run in parallel and be migrated after a certain time.
Appendix H.  Codeword summary company Y

Company/interviewee background
Summary of code words from interviews from company with interviewees working in service development, with background in hardware development.

Main take-aways from this company
- Comparisons between service and hardware development.
- International collaboration issues.
- Lack of organizational system, but understanding of benefits of one.
- Process with gates.
- Reuse of knowledge.
H.1 Actors

- Sales
- Service
- Business development
- Product owner
- Project management - in touch with customers
- R&D - Developing
- Quality

H.2 Steps of process or model

The different departments start from the same development process. A pre-study is performed to understand the requirements. Which product is covered by the requirements? Should the same service be offered for all product? Senior management contributes high level requirements, these are detailed and put into a project proposal. Get product requirement specification from business development. If the product specification seems unrealistic the project manager needs to approve of a feasibility study which is performed before it is turned into a product statement by R&D. The final document is a project specification. The project specifications includes specifications of money, time, and people. Requirements are stated, there is a design brief, and a plan is made. Plans are made, not necessarily because they can be kept, but because they are needed to have common picture of the project.

H.2.1 Time frame

No interview data responding to this codeword.
H.3 Methodology

Requirement maintenance takes place during the gate process. They perform root-cause analysis if requirements are not fulfilled, to understand issues, so that new development can take place in the future.

H.3.1 Iterative and agile methods

No interview data responding to this codeword.

H.4 Understanding

It is important to understand why requirements should be included. Plans are needed to provide a common understanding of a problem. Recently introduced a system to increase understanding from a market perspective.

H.5 Elicitation methods

They look into different markets, and consider which geographical areas should be included. Internal requirements originate from experience or newly developed technology. Both product owners and service developers are allowed to contribute requirements. They look into previous requirements, sales, talk to the business development department. Interviews also take place. They also use concept reuseability in their requirement collection methodology.

H.5.1 Customer interaction

Feedback is gotten from end customer in 1-10 years time. They have an online monitoring system to see how the product is performing at the customer. They also use net promoter score, asking the customer what they think about the product. Requirements differ depending on the customer. End users and retailers will ask for different things.

H.6 Requirement analysis

In service, conflicting requirements are avoided through a review process.

H.6.1 Prioritization of requirements

Prioritization is different depending on department. Prioritization is also different depending on country. Duration of a project also matters to handle prioritization.

H.6.2 Balancing requirements

No interview data responding to this codeword.

H.6.3 Setting targets

No interview data responding to this codeword.
H.7 Limitations & obstacles

In big projects, time can be spent on categorization and prioritization of requirements, in service, you might have time to identify which requirements are key requirements. Service need to scale their projects based on economy.

H.8 Requirement documentation and characteristics

Requirements are documented in a product requirement specification (PRS). The requirement specification base is shared globally within the company. A requirement includes the requirement giver, a requirement number, a reason for the requirement.

H.8.1 Requirement handling techniques

They do not use any project requirement database which would enable see connections between requirements due to this type of database’s high complexity. It would require too much administration. In big development projects they use quality function deployment, and the house of quality.

H.8.2 Relationship between requirements

No interview data responding to this codeword.

H.8.3 Requirement types

- Technical requirements
- Market related requirements (world/region)

H.8.4 Impact of standards

If a completely new product is being developed, it is necessary to pinpoint what the standards are concerning that product.

H.8.5 Impact of legal factors/legislation

No interview data responding to this codeword.

H.9 Traceability

The pre-studies allow them to map and describe the underlying reasons for requirements. The perception is that the link between business or customer need and final product design can be seen.

H.10 Modification methodology

When looking to hardware development, the new requirement is taken into the project and evaluated in terms of time, cost, and effects if added. A gate-group (aka steering committee, gate-team, board of project) need to make official decisions of changes at gates. Modifications which are necessary due to incorrect conclusions in the pre-study need to be made to avoid ending up with a product nobody wants. External influences, such as market changes are taken into account if necessary. The modification in requirements during development is based upon project total time period.
H.10.1 Interdisciplinary modification methodology
No interview data responding to this codeword.

H.11 Verification methodology
The product design specification specifies what can be delivered, this needs to be compared with the actual requirements. Requirements are compared, and it is inspected if they can be reached or not. Feedback is received on products already delivered, if they are working or not.

H.11.1 Testing
Simulation is used to check the design.

H.11.2 Continuous verification
The product development process (PDP) provides a structured way of continuously checking the system towards the requirements, the system is continuously monitored.

H.11.3 Relation between requirement and the product
The requirement specification acts as input for the product specification, planning starts from them.

H.12 Validation techniques
There is a discussion with the actors who originally stated the requirements. Prototypes and limited releases are made to validate the product. Validation over time is performed too find unanticipated occurrences.

H.13 Roles & responsibilities
No interview data responding to this codeword.

H.14 Interdisciplinary projects & communication
When working in an interdisciplinary environment, conflicting requirements are avoided by including all actors, and then going through all the requirements. For a new product, all departments are involved. Feedback is given by the different actors in case of conflict. It is written into the process, which actors to involve, and at which levels to involve them. If a project is started in one department, they involve expertise from other departments. In big development projects, representatives of other departments are added to the team. In smaller projects, the developing department is given a contact person in other departments. Cross-boarder work requires additional communication.

H.14.1 Interactions
No interview data responding to this codeword.

H.14.2 Relations
No interview data responding to this codeword.
H.14.3 Interdisciplinary actors
No interview data responding to this codeword.

H.15 Dependency
Checklists which occur in the PDP attempt to map which departments affect which other departments. Different departments are separate and have their own road-maps. Even if road-maps are parallel, they need to include information from other road-maps. Requests made by one department to another might be promised to be fulfilled, but the promise can be unreliable when it comes to time, making planning difficult.

H.16 Differences
Service need to scale their projects based on economy, so they can not apply the same processes and steps as hardware developers. Hardware development projects are longer than service based projects. Due to competition, focus of service has to change to offer service-agreements instead of spare parts.

H.17 Difficulties
When many people are involved in the decision making it becomes more difficult. It is hard to convince and involve the senior management in new ideas. It is hard to find project managers at smaller locations, and enough experts to send to meetings with bigger locations.

H.17.1 Intercultural work
The organization might be differently sized between different countries. Small organizations are usually fit for their purpose, but can not go the extra mile. When working cross-boarders the employees have different cultures, and different agendas. When working cross-sites the lack of real interface is a difficulty. Planning and creating the road map is possible through visiting each other on site, and through video-conferences. But doing the actual project work is hard.

H.18 Future approaches
Due to the market there is a need of changed business model. Believes there will be more collaborations with other international sites. This results in an increased demand on collaboration skills. Best practices will be shared between sites. Want to be able to understand future requirements earlier to be able to develop technology ahead of time. Want to make their processes parallel rather than sequential. Need someone who performs an organizational review, gets input, and makes the transition into the future more smooth, as currently they are rushing into things.

H.18.1 Obstacles future
Collaboration is added to the existing workload, and takes a lot of time. The collaboration will be more demanding on management, and time is needed to get involved and learn how to coach others if the location turns out to be best practice. Currently review of how to perform changes for the future is lacking.

H.18.2 Enablers future
share best practices to improve working environment. The organizational reviews will be good mean for getting inputs from different actors which can make transition smooth.
H.19 In a perfect world

No interview data responding to this codeword.
Appendix I. Codeword summary company Z

Company/interviewee background

Summary of codewords from interviews from company where interviewees had mixed backgrounds, hardware and software development, also at different levels - management and executing roles.

Main take-aways from this company

• Very structured process with gates.
• Use of official form to collect requirements from departments.
• Inter-department understanding is difficult.
• Requirements are not ranked after inclusion in requirement documentation.
• It is important for someone to own the requirement, so they feel responsible for it. It can be a person or a group.
I. Coded Word Summary Company Z

I.1 Actors

Departments:
- Market
- Design
- Supply
- Production
- Assembly
- Procurement
- Parts
- Special product
- Quality
- Maintenance
- Service market
- Technical groups
- Sales

Within each department are sub-departments.

I.2 Steps of process or model

The market department investigates what the customer wants and outputs a set of fuzzy requirements in the market requirement specification (MRS). This MRS is used as a base for other requirements, but they also collect requirements at a high level in a form which is sent out to each stakeholder to fill in. The forms are given to experts who evaluate which requirements are reasonable and correct in time. Requirements which are deemed reasonable belong to the scope. The scope contains about 600 requirements which are cataloged and inserted work breakdown structure for the product as a whole. They use a product requirement specification (PRS) at the top level, each row in PRS represents one requirement. Once a requirement is entered into the (PRS) it is decided upon (formal). Once accepted into the PRS, a requirement moves in the process towards becoming a technical requirement. The PRS is broken down in system design. Requirements go from top level to function & interface level and are tested. The 600 requirements are divided among different development functions. Functional requirements need to be split into interface requirements, connecting different modules. There are no gates inside the requirement collection/elicitation process, but there is a requirement specification gate at which the first draft of requirements is done and accepted and a requirement quality gate, where understanding is created through the use of different concepts. If 2 concepts are conflicting and of the same rank they can be kept parallel. If this is the case, a eventually favored concept will be "talked down" and an underdog concept will be "talked up". Can not say that the PRS is set until it is broken down to its smallest component. After a tollgate 2 (TG2) requirements are set. No design evaluation or design work takes place before TG2. The requirements are set in the beginning of the design phase, before verification. Once verified no changes can be made.

Functional requirements are extracted from the MRS and PRS, and input is also given from the IRS. Functional requirements end up in a functional requirement specification (FRS). There is a need for simultaneous handling of fuzzy requirements (MRS), PRS creation, and PRS breakdown (FRS & IRS).
A designer states their personal interpretation of a requirement to initiate a dialog with stakeholders. During the process there are formal checkpoints to keep the FRS up to date. During interdisciplinary development there are design reviews with a person responsible for the interaction requirement specification (IRS). At the design reviews implementations are presented and interpretations of requirements are discussed. A requirement needs a fixed measure and surrounding tolerances, when a requirement has this it can be given to the test department. Requirement givers are notified when their requested requirement is fulfilled.

### I.2.1 Time frame

The time frame of a project depends on its size. If it is a completely new product, the concept phase is 6 months and the design 1 year. There is a time schedule which needs to be followed. Each project has a time-line. Same quality is expected, but it should take shorter time to produce than it previously has.

### I.3 Methodology

It is important that engineers own their own requirements, that they are not the requirements of others. Use systematical function breakdown in software development.

#### I.3.1 Iterative and agile methods

Within sub-functions, modifications are handled in an agile way. The function developers modify their local FRS when needed, and try to push for update in the FRS as well. The approach is however far from purely agile, and has more characteristics of a waterfall model.
I.4 Understanding

A stakeholder gives a requirement, but the receiver has to understand it, and is allowed to reformulate/translate the requirement to improve the understanding of it. The stakeholder is revisited to ensure that the correct interpretation was made. It is group leaders who need to understand the requirements first. The understanding between departments is very difficult to get. There are people working with merging the collected requirements into an understandable specification. Stakeholders understand he need for compromise. The FRS is designed so that it is adapted to the needs of the designers, and is easy for users to understand. It is important to understand at which level you need to be before verification.

I.5 Elicitation methods

The market department speaks to the sales organization and visits the customer to collect requirements. They also use previous sales data, such as sales volumes to set optimal requirements. Old requirements can be reused in new projects. Personal connection with stakeholders is beneficial for them for requirements elicitation. They follow the layered approach to organized the requirements where each layer represent how and where to handle specific requirements types.

I.5.1 Customer interaction

The market division represents the customer. And it is very rare that the developers have direct customer exchange. The market sales manager and after sales management work as reference. Market sales collects information on what product the customer wants. After sales management is in touch with current users of the product. The customers however are very different, making a general picture of the customer difficult. Service market representatives are in touch with service technicians. The designer’s/developer’s interpretation of requirements is used as a basis for a dialog with stakeholders. First when there is an interpretation there will be feedback on if it is correct or not. It is easy to change the local FRS, which is owned by the designer/developer, but it is more difficult to get change acknowledged by the customer. If a requirement can not be fulfilled, developers need to reconnect with customers for negotiation and revision.

I.6 Requirement analysis

Those who work with requirement break-down can identify conflicts between requirements. If two requirements of the same rank are conflicting, one needs to be removed. When breaking down requirements, their feasibility is also determined. Requirements are controlled using simulations.

I.6.1 Categorization and prioritization of requirements

When the different departments provide requirements through the form, they need to rank them 1-3, where 1 equals a "must" requirement, and a 3 is a requirement which would be good to fulfill. There is always a need to balance the requirements. All requirements need to be considered, but not all requirements can be implemented. If two requirements conflict at the same ranking, one will have to be left out of the PRS. The ranking is only relevant in the step from market/customer. Once requirements are accepted into the PRS they do not have any ranks. There are no requirements which "might" be implemented, once a requirement is in the PRS it is supposed to be fulfilled by the product.

I.6.2 Balancing requirements

No interview data responding to this codeword.
I.6.3 Setting targets
Once the group leader understands the requirement they create a test tolerance, or a completion criteria.

I.7 Limitations & obstacles
Time limitation is the main problem in requirement engineering. Too little time/staff for the task. Low availability of persons who need to sign off/approve of requirement updates. The requirement tests do not identify if requirements are missing. There are many ideas which the different development departments would like to add which there is no time for implementing. Requirements can also be restrictions.

I.8 Requirement documentation and characteristics
The company has an internal standard for documenting requirements. All requirement utilize the same type of document regardless from their origin. Requirements are categorized according to project. Chapters in the requirement specification are based on functionality. Requirements are documented in the early phases using digital spreadsheets or word processors. Looking to the process sketch, requirements are documented between TG0 and TG2, but there is still possibility for movement up to TG5. The developers start from a template when stating requirements, so that all requirements handled by a department are treated equally. Some experiments have been made in a issue-based requirement handling software. The different types of requirement specifications are:

- MRS - Market requirement specification
- PRS - Product requirement specification
- FRS - Functional requirement specification
- IRS - Interface requirement specification

The use of requirement documentation across development departments varies, SW believe they along with testing are the only departments using FRS. The requirements do not cover all functionality, there are currently too few requirements for that. One row in the PRS represents one requirement. The developers try to keep the FRS up to date, and it is a living document. The IRS looks different depending on from which department it originates, and is more or less easy to understand for the receiving department depending both on the relation between the two departments and also on the characteristics of the giving department’s requirements. What is contained in the requirement specification:

- Rank
- Work package
- References
- Tolerances
- Measure
- Completion criteria
- Product affiliation
- Volumes
- Norms
A good requirement is a requirement which can be verified. All requirements in the PRS are verifiable. The PRS focuses on functions, and does not separate requirements based on HW or SW. The PRS does not show any technical solutions. All requirements in a specification should be at the same level of knowing what to design.

### I.8.1 Requirement handling techniques

When creating FRS, they try not to duplicate requirements which are already stated in the IRS. For software developers, the requirement specification is purely documentation, and is not (currently) used during development.

### I.8.2 Relationship between requirements

Which concepts are critical to the functionality of other concepts?

### I.8.3 Requirement types

**Different levels of refinement:**

- Fuzzy requirements
- Formalized requirements

**Requirement types:**

- Performance
- Dimensions
- Service
- Environment
- Norm testing
- Usability
- Robustness
- Feeling based requirements

**Source of requirements:**

- Internal - implication specifics, the own department
- Internal - other development departments
- External - from market

### I.8.4 Impact of standards

Safety norms affect related functions. New standards emerge effecting more components. (still high focus on safety)

### I.8.5 Impact of legal factors/legislation

There is a main legislation which covers all products, but then certain product have a special legislation.
I.9 Traceability

The developers try to keep the link between the FRS & the PRS. The link between the fuzzy MRS and the PRS is very clear, but the breakdown of the PRS can be unclear. The MRS is used by SW people to do confirm that whether they fulfill all requirements are not. There is not 100% traceability between the requirement givers and the specification but all actors within the company have a personal contact. The connection between the FRS and the final product is not clear either. When working with the collection of requirements using forms, the collected requirements are retained in their original form and a new version is created for the merged version of requirements. This way it is possible to go back to the originally stated requirements in the forms. At the top level it is possible to see which requirement is going into which sub-level of requirements.

I.10 Modification methodology

The stakeholder always has to approve changes. Changes after tollgate 2 requires approval of the technical manager, project manager, and stakeholders and is triggered by change requests. It is possible to make change requests up to tollgate 4. The perception at lower levels is that there is a change committee for the MRS and PRS, but that it is not used. The reason for the change request needs to be clearly stated and documented. Time, cost, and technological restrictions are evaluated when deciding whether to perform changes or not. However, some changes are necessary, as some problems simply must be solved. Changes late in the project naturally are more restricted. These changes usually solve problems, they do not add functionality. Changes which come late are made as rapidly as possible and documented afterwards. It is also possible to turn down requirement modifications. When changes take place in the FRS, the PRS is supposed to be updated as well, but this is not always the case. Changes at this level are communicated verbally. Requirements are never taken out of the specification, the change is symbolized by striking through the now invalid requirement. A new requirement can not have the same identifier as an old one. The revision of documentation is done manually in their excel file.

I.10.1 Interdisciplinary modification methodology

No interview data responding to this codeword.

I.11 Verification methodology

It is considered, how the stakeholder will be convinced that the function delivered by the product, complies with their given requirements. Conflicting requirements are detected by developers 9/10 times. It is also possible that conflicts are discovered by those who work with breaking requirements down. At verification, 90% of requirements are being tested. Verification should take place when the product is in, or very close to, its finished state. It is important that tests are followed through on a product made with the correct material, containing the correct software, and even in the right color. There is little point in verification prior to this product state. This also means that verification is dependent on getting a finished product before being able to get to work. For some areas requirements, such as vibrations and feeling, verification is performed by making a comparison with an old version of the product. This due to the difficulty to write clear specification within these areas. The PRS is the source for verification. The FRS is also used during verification, by the test engineers, and it is also used as a checklist for the developer to ensure that all functions have been implemented. The views of what verification is differs, some see the break own of requirements as verification, the feasibility study being one important part. This view includes simulations of the product as verification of the breakdown. The all verification activities are based on V-shaped process model as they first break down requirement than analyze them and finally test the requirements while going up as followed in V-model.
I.11.1 Testing

Only written requirements are tested, missing requirements are not identified by the testing process. It needs to be determined how many and which tests will be performed. Requirements with no or low risk are not tested. The specification helps point out which tests are needed to confirm fulfillment of a function. It further helps in deciding on type of prototype, amount of prototypes, and test duration. Different types of tests:

- Stress test
- Calculations
- HW platforms
- Rigs
- Test groups
- Mock-up test
- Simulation
- Field testing

At a functional level, 60% of requirements can be tested. Test and lab engineers in the "real verification lab" can run more complete tests on a complete product. These test engineers accept test requests from other departments. There is a test stack with 10 levels of testing. The product is CE-marked before field testing. The field testing is for operational details and handling productivity issues. Due to very different customers, it is hard to create general tests. Once testing has been performed, a report is generated. Either the test was successful or it failed. A fail can result in redesign of product, removal of requirement, or return to stakeholder for a new requirement as the current requirement is too hard to fulfill. The software department uses hardware in the loop as a method to test requirements. They apply continuous testing on their daily builds.

I.11.2 Continuous verification

No interview data responding to this codeword.

I.11.3 Relation between requirement and the product

Connection between requirements and code not clear, or non-existent. Software developers even state that the current set of requirement is not detailed enough for new developers to be able to work from.

I.12 Validation techniques

All requirements in the PRS are verifiable, so if the PRS is tested, since it originates from the fuzzy requirements, it is assumed that the customer need is correctly reflected. For customer opinion, they have focus person/group who can make a judgment, but the case can also be that the design team makes the judgment based on experience. It is important to make sure that what is designed is correct.

I.13 Roles & responsibilities

Roles:

- Project manager
- Supervisor
The product owner handles the PRS. The technical leader is responsible for making sure there are no conflicts between requirements. The system engineers take care of interdisciplinary requirements which might not be handled otherwise. They are responsible for norms and standards as it is part of their job to sign off on requirements. System engineers work with ambiguities and solving conflicts. The project scope of 600 requirements is divided among 5 groups with 150 requirements each. A work package leader is responsible for this collection of requirements, and is the group leader of design groups. It is the responsibility of the work package leader and members to make sure that the requirements get fulfilled. At a functional level, developers are responsible for their own requirements. Service market takes care of data from technicians on the field and are responsible for that the feelings of the customer are represented in the development process. The responsibility for key functions of the product are assigned to specific persons, who are then in charge of making decisions on examplewise compromises for sad key function.

I.14 Interdisciplinary projects & communication

The separation between departments is clear, but they work together very closely. All relevant actors are part of the project. Each group is expected to provide requirements. They handle the interdisciplinary development environment through assigning key task as responsibilities to specific people who can perform this job more effectively. In interdisciplinary projects, all stakeholders need to provide one person, who is the focal point, into the project. The complete team is always needed. The IRS is used as a catalyst to carry the dialog with other engineers. There are design reviews where implementations are presented and interpretations of requirements are discussed. In interdisciplinary development, one group will own the requirement or problem, and will need to ask other groups for help with solving it. All requirements are equally important, but are important at different levels of the product.

Modifications are handled through personal contacts and discussions. The process works because they sit (physically) near each other.

They work with a personal connection between developers and internal stakeholders, but the amount of interaction depends on the developers preferences. The SW side is more reliable in terms of doing verification than HW side.

I.14.1 Interactions

No interview data responding to this codeword.

I.14.2 Relations

No interview data responding to this codeword.

I.14.3 Interdisciplinary actors

- Software development
- Mechanical (Hardware) development
I. CODEWORD SUMMARY COMPANY Z

- Electronics
- Design
- Market
- Service Market
- Production

I.15 Dependency

There are some base properties of the product that need to be defined early on as they effect many other requirements. The hardware development needs some data from suppliers to be able to work. The perception is that the dependency between different development departments is very high during the actual development. To create a system, contribution from several departments is needed. Almost all functions of the product will cross modules, modules being represented by HW, SW and so on. This means that the functions depend on these modules. The fulfillment of certain requirements depend on very many different factors. In design department, they find more dependency than others.

I.16 Differences

Mechanical and electronic departments are traditionally owners of requirements. Software developers are doers, implementing functions. This also means that SW needs input from hardware and electronics to be able to do their job. SW needs to understand the full functionality. Requirements concerning for example robustness mean very different things for the SW and HW developer. (SW = reliable handling, HW = 8mm steel) Requirements from mechanical (HW dev) are easier to find out. SW are in a more raw data format and harder to understand and input in FRS and IRS. Software needs to be tested to be improved before the actual verification takes place. SW spend a lot of time verifying and has more verification steps and efforts than HW. HW and SW speak different languages when it comes to testing. HW have drawings, SW have "norm requirements" which are simpler and can be applied first in a prototype stage. HW has production facilities and tools, 1 year of preparation for production. SW can redesign within a day once they have the product to apply the SW on. SW waits a lot for others, but SW seldom is the cause of waiting. At some point they will reach the limit of mechanical differentiation.

I.17 Difficulties

People who are not involved in the whole process have difficulties understanding the total amount of work involved. It is difficult for all employees to have the same knowledge and understanding. It is also difficult to know when to input additional knowledge into a project. Due to underrepresentation of certain roles, developers need to know for example system engineering to compensate. When a small group, or a single person, have too much work, they become a bottleneck for the entire process. They have issues in requirements completeness where they encountered some issues. Certain parts of the process requires sign-offs from people who have a very low availability. Certain factors can not be tested through simulation. The change in requirements is possible in verification phase it could increase development cost as well. There is opposition to change, even when things are tried and lessons are to be had, the will to actually learn is lacking. There is a strong tradition, as a mechanical and producing company. The lack of a requirement handling system results in difficulties in handling requirements in a structured way. The process of traceability further need to be improve. A lot of things in the SW is not described in the PRS, and is only discovered during the break down of requirements, and this takes place too late. The requirements ownership is another issue which they face during traceability of requirements. They feel that SW people need to know more about system engineering process, they are
also lacking of system engineers now a days. The people from mechanical side need expertise to break down requirements in systematic way.

I.17.1 Intercultural work

Different countries/regions have different norms. This is information which is needed when specifying requirements.

I.18 Future approaches

If they use a database, there is a possibility to link requirements to tests or parts and split requirements to different requirement owners. They are going to convert their PRS into digital format. More simulations of the product, making use of computers and rigs. Structure in a database, easier to understand connection all the way down to the designer. Want an increased traceability. Make the whole development department more agile. Add more agility to the SW development. External components might benefit from working with more detailed requirements. Delivering functions, not articles, the focus of the company should be on functions. Want more system engineers and function responsible people. These people should be given more power and control in the development. It is important with people who have an overview. Method improvement, talk to experts concerning methodology. Automation is the future.

I.18.1 Obstacles future

Currently, their more focus is placed more on processes rather than methods which is not good, as they personally want to see improvements in system. The software development feels that the management is still very focused on “traditional” hardware development, and do not have understanding for the software development.

I.18.2 Enablers future

They are utilizing comparison technique to see the difference between new and old requirements which they feel very beneficial to make reflection of needs more clear in their requirements specification which they feel more beneficial for future use as well. The management can play a vital role to improve overall performance in product development.

I.19 In a perfect world

No interview data responding to this codeword.
På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/