Interdisciplinary Requirement Engineering for Hardware and Software Development

from a Hardware Development Perspective

by

Hanna Johansson

Supervisor(s): Sara Nilsson & Lena Buffoni
Examiner: Mattias Lindahl
Abstract

Complexity in products is increasing, and still there is lack of a shared design language in interdisciplinary development projects. The research questions of the thesis concern differences and similarities in requirement handling, and integration, current and future. Future integration is given more focus with a pair of research questions highlighting obstacles and enablers for increased integration. Interviews were performed at four different companies with complex development environments whose products originated from different fields; hardware, software, and service. Main conclusions of the thesis are:

- Time-frames in different development processes are very different and hard to unite.
- Internal standards exist for overall processes, documentation, and modification handling.
- Traceability is poorly covered in theory whilst being a big issue in companies.
- Companies understand that balancing and compromising of requirements is critical for a successful final product.
- The view on future increased interdisciplinary development is that there are more obstacles to overcome than enablers supporting it. Dependency is seen as an obstacle in this regard and certain companies strive to decrease it.

The thesis has resulted in general conclusions and further studies is suggested into more specific areas such as requirement handling tools, requirement types, and traceability.

**Keywords:** Requirement handling, interdisciplinary development, hardware, development, process, interview, traceability
Acknowledgments

This thesis started out as a joint thesis with Sheikh Bilal Tahir. To write a good interdisciplinary report, a challenger from a different field is necessary, and Bilal provided me with this opposing side of knowledge. It would also have been impossible to perform interviews in the same format if working alone. It grieves me that we were not able to complete the thesis as a joint venture, but I am happy that we were able to analyze results and come to conclusions while still working together. This allowed for a critical view on one-another’s thoughts and ideas and also for me to get a deeper understanding of software engineering.

Much of the conclusions in this report are possible thanks to the companies which were interviewed. As they are anonymous, so is the thanks to them, but without them, this thesis would not be possible.

Thanks goes out to supervisor Sara Nilsson, for providing this opportunity, allowing me to work with going into depth into a topic which really interests me. Thanks also for the feedback and support during the process of writing the thesis.

Thanks goes also to Lena Buffoni, who saw hardware development from a external perspective, and spent much time on questioning both content and format of different revisions of the thesis.

During the process I got plenty of feedback from my opponents Erika Wiberg and Mattias Nilsson. This feedback helped me make the report as good as it possibly could be.

Thanks to Lysator, the computer association at Linköping for providing me with access to ~, a room other than my office, laughter when I needed it, and to hx, knase, catears, arrwzy, and herj for support in the to me, recently discovered world of LaTeX and git. Thanks also to arrwzy, kempe, and hx for feedback on different parts of the thesis.

Lastly I want to thank my parents, and my friends, for listening to me, and giving support, in times of need. Extra thanks, and lots of love, goes to my brother Ola Johansson for the hours he put in helping me complete the final revision of the thesis.

Five years have come to an end with this master thesis. Thank you Linköping University, for five amazing years of higher education!
Abbreviations and definitions

Abbreviations

CRS  Characteristics requirements specification
FRS  Functional requirement specification
HoQ  House of Quality
HW   Hardware
IRS  Interface requirements specification
MRS  Market requirements specification
PRS  Product requirements specification
QFD  Quality function deployment
RQ   Research question
SW   Software

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 departments contribution for completion.

Gate: Formal point of agreement in frequently occurring in HW development processes. A specifically appointed group of managers and possibly experts decide at different points of time in a process, known as gates, if the project will proceed. 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.

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.

Requirement breakdown: The act of dividing general, high level requirements, into smaller more detailed requirements which work towards fulfilling this general requirement.

Traceability: Connection between different levels of requirements. High traceability means it is easy to see connection between different breakdown levels.
Contents

Abstract i
Acknowledgments iii
Abbreviations and definitions v
Contents vii
List of Figures ix
List of Tables ix

1 Introduction 1
1.1 Motivation 1
1.2 Aim 1
1.3 Research questions 2
1.4 Delimitations 2
1.5 Discussion on typology 3
1.6 Guide to the report 3

2 Method 5
2.1 Choice of literature 5
2.2 Description of companies interviewed 6
2.3 Creating and performing interviews 6

3 Theoretical Framework 13
3.1 Hardware Engineering 13
3.2 Requirement handling in Hardware Engineering 14
3.3 Software Engineering 19
3.4 Systems Engineering 19
3.5 Requirements Engineering 20
3.6 Requirement handling in requirement engineering 21

4 Results 29
4.1 RQ 1 & RQ 2 - Differences and similarities between different development processes’ requirement handling 29
4.2 RQ 3 & RQ 4 - Integration state, current and target 35
4.3 RQ 5 & RQ 6 - Obstacles and enablers, simultaneous hardware & software engineering 38

5 Discussion 41
5.1 Discussion of methodology 41
5.2 Link between different theoretical domains, interview questions, and answers 42
5.3 Discussion of results for RQ 1 and RQ 2 46
List of Figures

1.1 Reasons why this thesis exists .................................................. 2
1.2 Graphical representation of thesis ............................................. 3
2.1 Graphic representation of the approach of the thesis ...................... 5
2.2 Process of interview methodology .............................................. 6
2.3 Comparison induction and deduction ......................................... 8
2.4 Development of the interview guide ........................................... 8
3.1 Connection between theoretical fields ...................................... 13
3.2 The development process ....................................................... 14
3.3 Collected properties of requirements and requirement specifications ... 16
3.4 The House of Quality ............................................................. 17
3.5 Trade-offs between specifications ............................................. 18
3.6 The Three Dimensions of Requirements Engineering .................... 20
3.7 Requirement engineering process spiral ..................................... 21
4.1 Process of company W, mainly developing SW ......................... 30
4.2 Process of company X, mainly developing HW and electronics ....... 31
4.3 Process of company Y, mainly developing HW and services .......... 31
4.4 Process of company Z, developing both HW and SW ................... 32
5.1 Comparison for HW, SW, requirement engineering and company answers for requirement handling steps: elicitation, analysis, and documentation ........................................... 45
5.2 Comparison for HW, SW, requirement engineering and company answers for requirement handling steps: modification, validation, and verification ........................................... 46
G.1 Detailed picture of a HoQ matrix ............................................. XX

List of Tables

3.1 Methods for identifying requirements/needs along with sources ....... 23
3.2 Suggested parts of requirement documentation ............................. 26
4.1 Table of actors mentioned by companies W-Z ............................. 30
4.2 Different requirement types and characteristics according to companies W-Z ........................................... 34
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 as many actors as possible, looking from the entire life cycle perspective. This thesis focuses on requirement engineering, and the involvement of all actors. Hardware (HW) is the second focus of this thesis, and it is one of two theses with interdisciplinary focus, both which use the same interview data as material. Tahir (2017) has written the thesis with software (SW) focus. Therefore, integration of HW and SW is also covered in the thesis. Services are covered only in the results of thesis as very few of the companies studied mentioned services and none had it as their main offer.

1.1 Motivation

According to Tomiyama et al. (2007) modern systems are becoming more complex due to their multi-disciplinary nature and this complexity is making the product development process more difficult. Kotonya and Sommerville (1998) start their book by stating that the delivery of complex systems is often delayed and that the delivered systems do not meet the real needs of the user and/or buyer of the system. Kotonya and Sommerville (1998) continue to state that requirements for the system, and requirement handling are root causes for this problem. Sutcliffe (2002) claims that requirements are present in all aspects of our lives, this means that requirements should be relevant for all fields, HW as well as SW. Michael Jackson of the open university and Newcastle university writes in the foreword of Lamsweerde's (2009) book on requirement engineering, that to work with requirements means to be familiar with both formal and non-formal worlds, and being able to combine these worlds into efficient systems. According to Tomiyama et al. (2007) there are some frequently occurring issues in multi-disciplinary design; no common inter-disciplinary design language, issues handling many actors in the process, and problems generated by the mere fact that the development is multi-disciplinary. According to Tomiyama et al. (2007) it is necessary to include experts from several domains in cross-functional teams to develop multi-disciplinary products. All these factors together makes it logical to focus this thesis on investigating the requirement engineering process, looking from an inter-disciplinary perspective. This reasoning is illustrated in figure 1.1.

1.2 Aim

The aim of the thesis is to explore and identify key aspects of requirement handling from HW engineering, from both theory and from practice. This is to see if these key aspects are applicable in a combined HW and SW development environment. Further the thesis aims to explore the current state of integration between different development departments in the industry, and the mindset, enablers, and obstacles, for increased integration in the future.
1.3 Research questions

For the thesis to be considered complete it needs to answer the research questions (RQ) listed below. The RQs were identified, and clarified during the research. This resulted in pairs of RQs, 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 state of integration for companies which develop complex systems. The third pair concerns current obstacles and enablers for working towards this wanted future state.

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

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

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

RQ 4 What are companies working with complex systems doing to support an interdisciplinary development environment in the future?

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

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

1.4 Delimitations

This thesis has a SW counterpart. Within the theoretical framework focus has been upon the parts of HW development which work with requirements. Software is mentioned only briefly as it is covered by the other thesis, and requirement engineering in many ways covers similar concepts as requirement handling for SW. 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. The amount of interviews was set to 2–4 per company, and 4 companies were visited for interviews. Product service systems are not included in this thesis as none of the companies interviewed had this as their primary contribution.
1.5 Discussion on typology

As the thesis covers requirement handling, there is some terminology which can be confusing due to different use in different sources.

**Requirements vs Specifications**

What are specifications and what are requirements? What is the difference? According to Pugh (1990), it is important for all members of a team to take part in a concept evaluation because this will grant them greater insight into requirements in the specification. This phrasing makes it sound as if requirements are something a specification can have. But further, a specification is defined by Ulrich and Eppinger (2012) as a collection of individual specifications which each need to have a value and a metric. It is also in Ulrich and Eppinger (2012)’s book on Product Design and Development a clarification is attempted to be made on terminology. According to Ulrich and Eppinger (2012), a product specification is equal to product requirements, but these also in their turn are equal to engineering characteristics. Phrases like technical specifications or simply put specifications can be used defining the same thing. It all depends on the company.

1.6 Guide to the report

There is no single best way of presenting information, and in this thesis is structured to ease reading. Therefore the method presents a general approach and then specifically the execution of qualitative interviews within the thesis. The theoretical framework divides the overview of topics, and the in depth presentation of requirement handling for hardware and requirement engineering. The overview of different technical fields is needed for all RQs, and similarly the interviews contribute to all RQs. The end of the thesis is structured by RQs, to clarify the relations between results, discussion, and conclusion. The final chapter of future work and recommendations concerns several areas and RQs. The thesis structure is illustrated in figure 1.2.

---

**Figure 1.2: Graphical representation of thesis**

2. **Method**

   The methodology presents the approach of the thesis, including reasoning for the choice of theory for the theoretical framework. Qualitative interviewing and details of the method which has been applied in this thesis are lifted along with theoretical sources.

3. **Theoretical framework**

   The report continues with a theoretical framework, which introduces HW and requirement engineering, and briefly touches SW and system engineering which is covered further in a thesis by Tahir (2017). Requirement handling as described by HW engineering, is described further. An in depth description of requirement handling according to requirement engineering provides the background needed for interviews and analysis.
4. Results & Analysis
Results from the interviews are presented sorted under the relevant RQs. A comparative summary is presented for RQ 1 and RQ 2, including theory, interview questions, and interview data.

5. Discussion
The discussion covers general structure, theory, methodology, and then proceeds to discuss different topics for RQs which are originally presented in the results chapter. The discussion chapter ends with an ethics and environmental discussion.

6. Conclusion
Conclusions are given in the format of answers to the six RQs. Conclusions have a clear connection to the results presented in the thesis through the discussion chapter.

7. Future work and recommendations
In the final chapter future work and recommendations are stated, based on discussion and key points which occur after analyzing the collected interview data.
2 | Method

The aim of the thesis is to explore and identify key aspects of requirement handling, particularly from a HW engineering point of view. To achieve this the following approach which is illustrated in Figure 2.1 was taken. Theory concerning the areas HW development, and requirement engineering is covered in this report. SW development, and system engineering is covered in the parallel thesis by Tahir (2017). The theoretical framework provides an introduction to the different engineering fields, and the areas are further explored with the focus requirement handling. This theory was base for the requirement handling part of a questionnaire used during qualitative, semi-structured, interviews at 4 companies developing complex products. Interview answers, and the in-depth study of requirement handling in the different fields, made it possible to answer RQ 1 and 2 sufficiently. The aim to explore integration is more targeted at the companies resulting in the use of RQ3-RQ6 as base for interview questions concerning integration. The results of all interviews were summarized and in the result chapter answers relevant to answering the RQs are organized according to different subtopics within the RQs.

2.1 Choice of literature

In this thesis, a theoretical framework for hardware engineering, and requirement engineering is presented. In the thesis by Tahir (2017), software, and system engineering is covered. The theory presented in this thesis describes general, often established, theoretical processes. This choice of theoretical frameworks was made as little or no insight existed before interviewing the companies, concerning which types of processes they were using. It was assumed that a general background was better than a very specific, possibly irrelevant, deeper insight into a process. If the processes of the companies were known beforehand, it would have been possible to create a more specific theoretical framework. If more time existed for iteration, it would have been possible to extend the theoretical frameworks further after the interviews took place.
2. Method

2.2 Description of companies interviewed

Many of the research questions include either interdisciplinary requirement handling or development. Therefore 4 companies, working with different main focuses in complex products, were interviewed. Having being chosen by company representatives, the interviewees at the companies had different roles, or experience, covering mechanical, electronical, SW, and/or service development. Some companies wished to remain anonymous and therefore all companies have been left unnamed in this thesis. A short description follows to give slightly further insight into the different companies.

- Company W - Interviewees work mainly with SW development. The company has a HW branch as well but this is perceived as a completely separate organization.
- Company X - HW developing company. Interviewees could give insight into SW and electronical aspects as well as HW development.
- Company Y - HW developing company. Also has a service branch from which both interviewees had experience.
- Company Z - Company which develops HW-based products. Both HW and SW developers were represented among the interviewees.

A common factor for the companies is that their size is large; with at least 50 employees and shares on a regulated market (Bolagsverket, 2015) and that the product offered is complex: which means it requires input from several different branches of development. Interviews were performed to provide examples from the industry, but the small selection of companies can not be seen as representative for the industry as a whole, but only as illustrating examples which may, or may not, be omnipresent in the industry.

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.

2.3 Creating and performing interviews

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 2.3.1-2.3.6 and can be seen in Figure 2.2.

![Figure 2.2: Process of interview methodology](image)

Following these steps, the theme of a qualitative interview was set, companies contacted, and a guide created. The interview was semi-structured, supported by Flick (2007) who recommends this structure for extracting subjective data, vital in qualitative research. In line with recommendations by van Boeijen et al.’s (2014), a pilot interview was performed to test
2.3. Creating and performing interviews

The phrasing of the interview questions. This pilot 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 the 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. van Boeijen et al. (2014) recommends 3–8 interviews but does not state if this is per company or for the study in total. Kvale and Brinkmann (2009) recommend 5–25 subjects, presumably for the study as a whole. 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. Audio recording is a recommended complement by van Boeijen et al. (2014). 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 company, and selected data is presented in the results chapter of this thesis. Further analysis of the data can be seen in the discussion chapter.

2.3.1 Thematization of thesis interviews

The perception at the beginning of this thesis was that the topic, integrated/interdisciplinary requirement engineering, was not frequently mentioned in any of the theoretical frameworks. Even requirement engineering, which covers interdisciplinary requirements, failed to lift interdisciplinary aspects. Therefore what Rubin and Rubin (1995) refers to as ‘exploratory’ interviews were performed. 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. A graphic representation of the two logics can be seen in Figure 2.3. 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. The interviews were performed as were they inductive, but as can be read in the discussion, the realization was made after collecting all data that the results were rather deductive. In accordance to methodology presented by Rubin and Rubin (1995) the theme of integrated/interdisciplinary requirement engineering was set at the beginning of the project. Focus of questions was set to requirements/specifications and integration between different development departments (for example HW and SW).

2.3.2 Design of thesis interviews

4 different companies were approached for interviews. Each company was asked to provide at least 2 interviewees. It was decided to use the same guide in all interviews, simply excluding parts based on time limits and experience of the interviewees. This decision was made to ease creation of summaries, but also to enable execution of a comparative study as it is defined by Flick (2007), which requires as systematically and similar repetition of analysis as possible on different sets of collected data. The interviewees were selected by contact persons 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 con-
cerned 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.

*Interview guide for thesis interviews*

Making use of both the thesis RQs, and the themes determined during thematization of the thesis interview, the guide outline was drafted using the following Figure 2.4. Kvale and Brinkmann (2009) argues that a more structured interview enables quicker processing of answers. By keeping track of the relation between RQs and interview questions as is done in Figure 2.4 it is easy to return from interview questions to the main aim of the thesis which is fulfilled by answering the research questions.

![Figure 2.3: Comparison induction and deduction](image)

![Figure 2.4: Development of the interview guide](image)
After revision the following themes were decided upon to be covered in the interview:

<table>
<thead>
<tr>
<th>Requirement engineering</th>
<th>Integration process</th>
</tr>
</thead>
<tbody>
<tr>
<td>requirements elicitation</td>
<td>interdisciplinary development handling</td>
</tr>
<tr>
<td>requirements categorization</td>
<td>dependency between different development teams</td>
</tr>
<tr>
<td>requirements documentation</td>
<td>modification handling in interdisciplinary teams</td>
</tr>
<tr>
<td>requirements verification</td>
<td>system reliability</td>
</tr>
<tr>
<td>requirements modification</td>
<td>difficulties</td>
</tr>
<tr>
<td></td>
<td>future expectations</td>
</tr>
</tbody>
</table>

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.

2.3.3 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 set based on the pilot interview and revised as the questionnaire changed design and was rebalanced. 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. Rubin and Rubin (1995) claims that the design of qualitative interviews needs to be flexible, iterative, and continuous and not set in stone. Aligned with this, as the interviews processed on, some questions were removed as they gave little contribution towards answering the RQ and adding too much time. 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. All interviews but one were performed in private conference rooms, aligning with a recommendation of van Boeijen et al. (2014). Lindahl (2017) suggests the use of a matrix to see the connection between RQs and interview questions. This type of matrix for the thesis interviews can be seen in appendix D. In this matrix the purpose of the questions and the expected answer type is also presented. This matrix further supports the structure recommended by Kvale and Brinkmann (2009) earlier. 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 RQs about similarities and differences. Similarly most integration interview questions work
2. Method

towards answering RQs were believed to cover integration. A more detailed instruction for reading the table can be found along side it in the appendix.

2.3.4 Transcription of thesis interviews

No direct transcript was created as this would be very time consuming, the thorough notes taken during the interviews were seen as adequate option. The audio and video-recordings were used to strengthen notes, aiding the accuracy of the summaries.

2.3.5 Meaning analysis of thesis interviews

In the thesis, a comparative study was 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 RQs, each of the topic areas defined in the guide section, and the different theoretical frameworks. This is similar to [DeCuir-Gunby et al. (2011)] description of theory-driven codes which represent concepts discovered in the research literature or topics in the interview schedule. 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. Theory-driven and data-driven codes are the most regularly used according to [DeCuir-Gunby et al. (2011)] so the use of these two types of coding is relevant. 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. Once coding had taken place, obsolete code-words were removed and data driven code words were added to the list. This list with the codewords which were used in the final iteration of coding can be found in Appendix F.

All interviews were coded and summarized into company summaries. The code word summaries can be found in Appendices H, I, J, and K. These code summaries are referenced further in the result chapter, as sources of the presented results. 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 draft to ensure that all, to the thesis relevant, themes possible were covered. This is supported by both [Rubin and Rubin (1995)] and [Flick (2007)] who speak of working iteratively with interviews. 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 types of evaluation took place during the process of the interviews.

2.3.6 Verification of thesis interviews

In the thesis, there was a continuous verification of the design of the interviews by regular feedback from the supervisors. This type of continuous verification was prompted by [Kvale and Brinkmann (2009)]. 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 RQs. At the end of the interview interviewees were asked for feedback on the interview,
2.3. Creating and performing interviews

also as recommended [Kvale and Brinkmann (2009)]. 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.
Theoretical Framework

HW and requirement engineering are covered in the theoretical framework of this thesis. To be able to answer the pair of comparative RQs, RQ1 & RQ2, which aim to compare differences and similarities in HW and SW development both these processes need to be explored. Further as previously defined, the systems which the thesis analyzes are complex, which requires covering more than one development process in the theoretical framework. System engineering usually looks to larger, and therefore more complex, systems, and integration of different development processes. Requirement engineering which is the focus of the thesis is a sub-area of system engineering [Lamsweerde (2009)] but is treated as a standalone topic-area in this theoretical framework. The relations between the different theoretical fields can be seen in Figure 3.1.

3.1 Hardware Engineering

HW engineering in this thesis is not limited to the engineering of computer components, but includes all types of physical products. Therefore, similarly HW development refers to the development of the parts of a product or system which is not SW. This means that all physical attributes of a product are assumed to be developed by HW development. Development processes and specifications described below were initially intended for HW development, but they may be applicable also in other fields.
3.1.1 HW development models

To get a general understanding of what HW development is, several different sources have been visited to get an overview. Ulrich and Eppinger (2012) and Ullman (2002) both present models with steps of development to describe the HW development process. Ulrich and Eppinger's (2012) model starts at establishing that a problem needs solving while Ullman's (2002) model start at Ullman's second step: Planning. The planning step includes, or is followed by, the identification of specifications - depending on model - which is the area of focus for this report. Ullman (2002) along with Douglas et al. (1978) include market screening or identification of existing products in this step. Once an understanding of the situation exists, comes the step of developing concepts or solutions. Ulrich and Eppinger (2012) divide this into several steps, such as concept development, system-level design, and detail design, while Ullman (2002) simply refers to it as generation of alternative solutions. Once designs are generated there needs to be a comparison and final selection of design. This final design is put through testing and refinement before production ramp-up takes place. The HW development process contain the following steps which are further illustrated in Figure 3.2.

Figure 3.2: The development process

Source: Own merge of models by Ulrich and Eppinger (2012), Ullman (2002) and Douglas et al. (1978)

It is not unusual according to theory to have ‘gates’ between these steps Ulrich and Eppinger (2012). A gate is used to confirm completion of the previous step, and to decide if the project will proceed into the following step Ulrich and Eppinger (2012). According to Hallin and Karrbom Gustavsson (2013), a gate- or ‘tollgate’- process is only used in sequential projects, and it decides both in what order steps happen, but also in what time frames. The time frames depend on the organization and are usually based on prior experience Hallin and Karrbom Gustavsson (2013). Product development according to Fraser et al. (2003) is a collaborative activity which requires the involvement of both internal and external actors. Fraser et al. (2003) continues to state that it is acknowledged that collaboration is difficult but that successful collaboration leads to competitive advantage.

3.2 Requirement handling in Hardware Engineering

According to Ulrich and Eppinger (2012), requirements come into the HW development process in the concept development stage. Ullman (2002) instead has introduces them prior to the conceptual design phase. In this chapter, HW requirements, often known as product specifications, are defined. After this it is explained how these requirements can be identified, generated, confirmed, and used.

3.2.1 Product specifications

What Ulrich and Eppinger (2012) define as ‘product specifications’, they say can also be known as ‘product requirements’ or ‘engineering characteristics’. When making an attempt at answering the question: “What are specifications?” Ulrich and Eppinger (2012) explain
3.2. Requirement handling in Hardware Engineering

how product specifications start as the needs of the customer, which are expressed in what
they refer to as ‘the language of the customer’. To collect product specifications, Ullman
(2002) refers to a technique called Quality Function Deployment (QFD), which starts with
identifying the customer, and then the customer requirements. Ulrich and Eppinger (2012)
say that the customer specifications need translating before use in technical projects. Ullman
(2002) refers to this as a translation from customer requirements to engineering specifications.
According to both Ulrich and Eppinger (2012) and Johannesson et al. (2013) the ‘translated’
specifications clearly state what products have to do, without implying in anyway how they
should fulfill the specification.

According to Ulrich and Eppinger (2012) specifications would ideally be defined early on in
the process but this is not usually the case for technology-intensive products. For these prod-
ucts specifications, according to Ulrich and Eppinger (2012), are defined twice. First, target
specifications, based on user needs, and then final specifications are set, based on technical con-
straints and production costs. According to Ulrich and Eppinger (2012) there is always a need
to refine the requirements and make trade-offs as the process goes forward.

Johannesson et al. (2013) describes the process of specifications going from target to final ones
as a natural part of the process. As the knowledge of the designed HW increases, the specifi-
cations are updated. Ullman (2002) uses different definitions, referring to target specifications
as customer requirements, and final specifications as engineering specifications.

Both Ullman (2002) and Roozenburg and Eekels (1995) mention requirements of require-
ments, or sought after characteristics of the requirements. These have been collected in Fig-
ure 3.3 Validity covers the relevance of requirements and their connection to the goals set.
Completeness ensures that all different aspects are covered by the specification. Operationality
includes many factors, measurability being the most mentioned one. There are wanted at-
tributes for the measures and values as well; they too must be complete. Further they need
to be practical - testable, state what, not how. It must also be clear how they are to be evalu-
ated, and popular criteria should be included. Returning to the properties of the requirement
specification, requirements should be orthogonal, also known as non-redundant, no character-
istic should be controlled twice. The specification needs to be concise - this can be achieved by
several different actions. Making sure to only include the external requirements, merging
similar requirements from different actors, and removing requirements that are fulfilled by
all solutions or concepts, are all actions that reduce the amount of requirement included in
the specification. Last but not least the requirement specification needs to be universal. There
is no point in measuring a characteristic if it is not offered by all solutions or concepts.
3. THEORETICAL FRAMEWORK

### Properties: requirement specification

<table>
<thead>
<tr>
<th>Valid</th>
<th>Relevant and connected to goal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete</td>
<td>Covers all requirements</td>
</tr>
<tr>
<td>Operational</td>
<td>Specifications and trade-offs are clear</td>
</tr>
<tr>
<td>Orthogonal, Non-redundant</td>
<td>No single characteristic is controlled twice</td>
</tr>
<tr>
<td>Concise</td>
<td>Specification as concise as possible</td>
</tr>
<tr>
<td>Universal</td>
<td>All solutions contain the measured characteristic</td>
</tr>
</tbody>
</table>

![Figure 3.3: Collected properties of requirements and requirement specifications](source: Combined information from Ullman (2002), Roozenburg and Eekels (1995), Ulrich and Eppinger (2012)]

According to Ulrich and Eppinger (2012) a specification is made up by two parts - a metric and a value. This connects well to Ullman’s (2002) requirement of a requirement to be measurable, and Roozenburg and Eekels’s (1995) desirable property of operationality. Ulrich and Eppinger (2012) have some requirements on these metrics - they should be complete, be dependent, not independent variables, be practical, be clear if they need to be evaluated subjectively, and include the popular criteria for comparison in the marketplace. The subjective requirements mentioned by Ulrich and Eppinger (2012) can not be measured in the same way as other requirements, as they need a group of people for control. Both **ideal** and **marginally acceptable** values need to be set according to Ulrich and Eppinger (2012). Zhang et al. (2015) speaks of the acceptable measures as constraints as they must be achieved, and the ideal measures as goals which the final product may fail to achieve and still be acceptable. These types of targets are used to aid the process of eliminating unacceptable designs.

Johannesson et al. (2013) speaks of a different type of division of specifications for HW - that of separating them into **requirements** - ‘must’s, and **requests** - ‘should’s. Roozenburg and Eekels (1995) similarly mentions a separation between **requirements** and **wishes**. Khan et al. (2015) includes “must” and “should” into their description of the four priority groups of the MoScoW model which can also be used in SW development. The four groups are: ‘Must’, ‘Should’, ‘Could’, and ‘Won’t’. The different titles speak for themselves, a must-requirement must be implemented, a should requirement would be good for the product. Could is similar to should as it would be good for the product, but it is lower ranked. Won’t-requirements are considered not possible in the current iteration and given low priority (Khan et al., 2015).

### 3.2.2 Generation of engineering specifications described using Quality Function Deployment

Ullman (2002) mentions QFD - Quality Function Deployment as the currently (authors’ note: 2002) most popular technique for generating technical specifications. QFD consists of four steps. The first step of QFD, which results in technical requirements, includes walking through the “rooms” of the House of Quality (HoQ) which was first described by Hauser and Clausing (1988) and later clarified by Ullman (2002) in the rooms that can be seen in the list below and Figure 3.4. Also, Ulrich and Eppinger (2012) make use of the HoQ in their steps for processing requirements.
3.2. Requirement handling in Hardware Engineering

The House of Quality:
1. Who are the customers?
2. What do the customers want?
3. Determine relative importance of requirements
4. Identify and evaluate the competition
5. Generate engineering specifications
6. Relate customers’ requirements to engineering specifications
7. Set engineering targets
8. Identify relationships between engineering requirements

Figure 3.4: The House of Quality

A detailed account on the HoQ can be found in Appendix C. The first steps of identifying actors (HoQ 1) and their needs (HoQ 2) are the most vital for this thesis. Actors is any person, real or legal which has an interest in, or can be affected by, the project (Ullman, 2002). In the HoQ model, requirements are collected using a questionnaire, and data is collected and reduced. Several different needs can be collected, functional requirements being one noteworthy. Making note of the source of the requirement is also of certain relevance (Ullman, 2002). After these two first steps, relative importance is determined (HoQ 3), making use of actors to create a weighting that is also relevant later during verification. Competition is looked at (HoQ 4) as it needs to be exceeded to succeed (Hauser and Clausing, 1988). Targets are set (HoQ 7) according to this goal. Needs are translated into a engineering specification (HoQ 5) making the requirements formal, assuring that the requirements have all attributes necessary (Ullman, 2002). Once this formal engineering specification is made, validation is
3. Theoretical Framework

necessary (Ullman, 2002, Hauser and Clausing, 1988), checking if the needs of the actors are represented (HoQ 6). Lastly the relationship between different requirements (HoQ 8) need to be understood as a highly complex product results in changes have implications across the products, and that certain changes can not be done at all (Hauser and Clausing, 1988).

3.2.3 Hardware development: The confirmation of requirements and final specifications

The final specifications take technical limitations into consideration. According to Ulrich and Eppinger (2012), the final specifications are set when the concept selection is in its final phase. When settling on final specifications, some trade-offs will have to be made. Ulrich and Eppinger (2012) suggest the drawing of a competitive map, see Figure 3.5, where the two dimensions represent selected metrics from the specification. Competitors are drawn out along with a box for marginal and ideal values.

![Figure 3.5: Trade-offs between specifications](image-url)

Source: Inspired by Ulrich and Eppinger (2012)

Douglas et al. (1978) emphasize on the need to go to the consumer for feedback early on, so that one does not spend too much money on a design or solution in which the consumer has no interest. Ulrich and Eppinger (2012) instead seem to emphasize on keeping the customer needs in mind when working with the trade-offs, looking to see if the final products actually belong in a different market than the intended one, rather than questioning the set specifications themselves earlier.

3.2.4 Hardware development: Using requirements during the development process

The initial requirements together with customer needs are used as a base for the concept development (Ulrich and Eppinger, 2012). Later, requirements form the base of a decision matrix, along with their importance based on an prioritization made earlier in the process. The requirements evaluate different alternatives during the concept stage of the development process (Ullman, 2002). The product can only truly be evaluated towards the engineering requirements once it is refined to such a degree that numerical measures can be made. The evaluation according to Ullman (2002) is to be used to identify which features of the product need tweaking and should take variations into account.
3.2.5 Summary of requirement handling in the hardware development process as described in this thesis

In the HW process the collection of requirements comes early, in the beginning of the process. Specifications are usually used as origin for brain-storming and problem solving, and later for evaluation of concepts and the final product. Actors' needs, as well as the current competition, are collected and translated into an engineering specification. There are many desirable characteristics for requirements to possess. When it comes to measurability, requirements are given an ideal value, and an acceptable value which solutions must achieve. There are two different specifications. One initial specification, based on the needs, and one final specification at the end of the development process. The final specification takes technical limitations into account. Reaching this final specification requires trade-offs and the requirements in the specifications need to be prioritized, determining which must, should, could and won’t be fulfilled.

3.3 Software Engineering

According to [Whitson (2016)] there are different SW development models: waterfall model, iterated waterfall and spiral model, and object-oriented model. The waterfall model is sequential, this evolved to the iterated waterfall model, which then was turned into the spiral model by Barry Boehm, read more in [Tahir (2017)]. The object-oriented model uses the iterated waterfall model as base but focuses on the object, properties, and methods. Dysfunctional SW development according to [Codington-Lacerte (2016)] laid the ground for agile SW development which strives towards increased flexibility and shorter time frames. Agile development works with releases and customer involvement is an important part of the agile SW development as they give feedback on the different releases. Read more about SW development models, and requirement handling in SW development in [Tahir (2017)].

3.4 Systems Engineering

According to [Hagan (2017)], system engineering deals with the organization, design, and management of complex and interdisciplinary engineering projects. According to [Hagan (2017)] system engineering has to connect and balance technical (various engineering disciplines) and human-centered (business, organizational, management) disciplines. [Hagan (2017)] explains that a systems engineer views problems holistically, seeing how different parts fit together and can be divided as chunks to create an even workload. Further system engineers according to [Hagan (2017)] also are responsible for analyze projects in terms of dependencies, and creating a ‘critical path’ where these dependent parts are appropriately prioritized. [Elfving (2007)] speaks of increased internal collaboration being needed in a company creating a multidisciplinary products. This collaboration is intended to aid development can also increase its costs. The collaboration is affected by many different things; what activities, which actors, and how many actors are part of the collaboration, are all examples of things which affect collaboration. Collaboration requires reconciliation between different ways to work in different departments and for collaborating departments to have a shared goal if looking to difficulties which are described by [Elfving (2007)]. According to [Küster et al. (2016)] ideally all actors would share a single process model to ease collaboration and communication. [Küster et al. (2016)] continue to state that this is difficult if different actors use different tools and/or meta-models. The key functions of a shared process model according to [Küster et al. (2016)] is to maintain consistency between different participating actors’ views. The thesis by [Tahir (2017)] goes more into depth concerning system engineering.
3.5 Requirements Engineering

As mentioned in the introduction of the theoretical framework, requirement engineering originally stems from system engineering according to Lamsweerde (2009), and was known as system analysis before becoming its own field of study (Sommerville and Sawyer, 1997). According to Sutcliffe (2002), there are many definitions of requirement engineering, but there is also an agreement that requirements concern what people want from a system, and how their needs relate to the design. Many books concerning requirement engineering, relate the requirement handling to SW development. (examples of books: Lamsweerde, 2009, Sommerville and Sawyer, 1997, Kotonya and Sommerville, 1998, Sutcliffe, 2002) According to Hood (2008), the goal of requirement engineering is to formulate the visions of actors, in whatever language suitable. In an established Figure 3.6 inspired by Pohl (2009), major problems for requirement engineering to solve are illustrated: specification, agreement, and representation.

![Figure 3.6: The Three Dimensions of Requirements Engineering](source: Inspired by Pohl (2009))

According to Pohl (2009) the dimension of specification describes the understanding of requirements at a given time. Initially, when there is only a vague idea of what the system should look like, the specification is opaque. Sutcliffe (2002) speaks of this type of input as coarse-grained and ambiguous. According to Sutcliffe (2002), one strives to end up with a ‘complete’ requirement specification which according to Pohl (2009) is defined by several different standards and guidelines. According to Pohl (2009) the representation dimension handles the level of formality of the information. According to Sutcliffe (2002) the original input is informal. This informal information is user-friendly, while the later semi-formal information provides a more structured overview. Examples of semi-formal representations are; data flow diagrams and entity relationships (Sutcliffe, 2002). At the end of the process, information also needs a formal representation, but all different levels of representation need to coexist in the final specification according to Sutcliffe (2002). Agreement according to Pohl (2009) concerns different views of the same thing. According to Pohl (2009), at the beginning everyone involved in the process has an own personal view of the system. At the end of the process this view of the system needs to be a common one, upon which all actors have agreed (Sutcliffe, 2002). These three dimensions need to cooperate to move from the initial output to the desired output. Sutcliffe (2002) argues that requirement analysis is difficult, seeing as the world is ever changing. The requirements will depend on the point of time and situation in which they were captured, and how well the future is anticipated. It is important to include
flexibility in the design so that a product can adapt to change. Jiang et al. (2005) argue that there is no universal technique which solves all requirement engineering problems. They suggest that researchers and developers combine several appropriate techniques for each project.

3.6 Requirement handling in requirement engineering

According to Sutcliffe (2002, p. 45) "there is no one ‘cook-book’ method for requirements". Sommerville and Sawyer (1997) argue that there are many different processes, and that these processes do not transfer well between organizations. With this said, Sommerville and Sawyer (1997) go on to state that elicitation, analysis and negotiation, and validation are steps that should be included in a ‘good requirements engineering process’ (Sommerville and Sawyer, 1997, p. 10). Similarly Sutcliffe (2002) goes on to include elicitation, analysis, modeling, validation, negotiation, functional allocation, but also processes for discovering and refining requirements, and reflecting over what type of requirement engineering should be used for different target products. The steps of requirement engineering according to Lamsweerde (2009) and Kotonya and Sommerville (1998) can be described by a spiral-shaped process (see Figure 3.7).

Figure 3.7: Requirement engineering process spiral


Sommerville and Sawyer (1997) also describes their recommended steps of elicitation, analysis, and negotiation using a spiral shaped process representation. The steps according to Lamsweerde (2009) and Kotonya and Sommerville (1998) are:

1. Domain understanding and elicitation
2. Evaluation and negotiation
3. Specification and documentation
4. Quality assurance

This spiral shaped process puts emphasis on the iterative aspects of requirement engineering. According to Lamsweerde (2009) and Kotonya and Sommerville (1998) the activities of requirement engineering need to be repeated until it is decided that an acceptable requirement document has been created. When requirements have been elicited, they together form
3. Theoretical Framework

an informal statement of requirements. After this the requirements are analyzed, evaluated, and negotiated, resulting in agreed requirements. These requirements are documented in a draft requirements document. Once documented, the requirements can be validated, resulting either in consolidated requirements or repetition of the process. This iteration can be triggered by for example a need to revise, adapt, or extend the requirements. These types of edits can be achieved through addition, removal, or modification of statements. Requirements, assumptions, and domain properties can all be statements in a requirement document (Lamsweerde 2009, Kotonya and Sommerville 1998).

3.6.1 Comparison of different process models

Both Sutcliffe (2002) and Sommerville and Sawyer (1997) avoid stating a specific order of the process steps. The steps mentioned by Sommerville and Sawyer (1997) are included in the spiral model described by Lamsweerde (2009) and Kotonya and Sommerville (1998), while some of Sutcliffe's (2002) recommended activities are not mentioned by other sources when speaking of process models. For example a step for modeling the requirements is missing and the iterative spiral process does not move past requirement documentation to start the functional allocation suggested by Sutcliffe (2002). The activity of discovering and refining requirements can be interpreted to be included in the iterative nature of the spiral process as it strives to refine the requirements until there is a common agreement among actors. The activity of stepping outside the process and reflecting over if the correct requirement engineering is used for the target product is also an activity completely unique to Sutcliffe (2002) in the process descriptions. Some of these steps might be healthy additions or complements to the relatively simplified spiral model presented by Lamsweerde (2009) and Kotonya and Sommerville (1998).

3.6.2 The collection of requirements

The collection of requirements is part of an early phase of the requirement engineering process which is mainly about acquisition of knowledge. The first step to effective knowledge acquisition is to identify the correct actors (Lamsweerde 2009).

Actors

According to Kotonya and Sommerville (1998), actors in the requirement engineering process include many different people. Among them are different types of engineers and experts, end users, managers, but also health and safety regulators (Kotonya and Sommerville 1998, Hull et al., 2005). All those who may be affected by the existence of the system can be seen as actors, and they are sometimes referred to as stakeholders (Lamsweerde 2009). Lamsweerde (2009) mentions how important it is to address the correct actors, and have a representative sample based on the actor’s respective roles, stakes, interests, and types of contributable knowledge. Hull et al. (2005) provide a list as a guide to start from when collecting possible stakeholders:

- Managers
- Investors
- System users
- Maintenance and service staff
- Product disposers
- Training personnel
- System buyers
- Sales and marketing
- Usability and efficiency experts
- Operational environment experts
- Government
- Standards bodies
- Public opinion and opinion leaders
- Regulatory authorities
Requirement elicitation

There are many different methods for collecting requirements which can be seen in Table 3.1. Sutcliffe (2002) states that a mixture of the different methods for gathering facts is needed, but that interviews is the most popular technique. Looking at the table which shows which elicitation methods are recommended by different sources, it would seem that further questionnaires and analysis of previously existing documentation also are established methodologies for collection of requirements.

Table 3.1: Methods for identifying requirements/needs along with sources

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Interviews</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Questionnaires</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Observation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Surveys</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Focus groups</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Documentation analysis</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Market analysis, competitive system assessment</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Simulation, prototyping, modeling</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Benchmarking</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Organizational analysis techniques</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Problem and change suggestions</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Opportunities from new technology</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scenarios</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sutcliffe (2002) highlights the difficulties in the identification of requirements, as it requires an understanding of people and communication. Along this line, Lamsweerde (2009) highlights the need for communication skills when collecting requirements. Both Sutcliffe (2002) and Lamsweerde (2009) agree that when a skill has been learned it is no longer actively reflected how it is performed, it has become tacit knowledge and a user will not describe it to an analyst even if these activities are important to the understanding. Sutcliffe (2002) recommends different ways of requirement elicitation to get around the issues of tacit knowledge. Participation, according to Sutcliffe (2002), is a good way for extracting tacit knowledge as it gives the researcher insight into the process details, opening up for new questions to be asked. Further Sutcliffe (2002) mentions people’s inability to accurately explain what they do, ambiguity, as a barrier. When expressing themselves, the use of language, which in itself is flexible in a positive regard, lends itself to misinterpretations. This problem according to Lamsweerde (2009) can also occur when people come from different backgrounds, or cultures, using different types of terminology. Sutcliffe (2002) also mentions people’s attitudes and opinions as unreliable, as they can be picked up from others without ever being thought through by the person later repeating it. Both Sutcliffe (2002) and Lamsweerde (2009) argue that people may be affected by social factors, making them voice requirements differently or not at all.

According to Sutcliffe (2002) requirement engineering is more than understanding the user, understanding the limitations, and implications, of the domain is also important.

3.6.3 Requirement analysis

According to Sommerville and Sawyer (1997) analysis should be performed as soon as initial requirements have been collected. Analysis should according to Sommerville and Sawyer (1997) look for conflicts, overlaps, omissions, and inconsistencies.
Requirements for requirements

According to Lamsweerde (2009) a requirement document needs to fulfill a collection of quality factors. These quality factors in many ways answer to the sought after characteristics of requirements covered by the sub-chapter product specifications in 3.2 HW development: requirement handling.

- **Completeness**
  The requirement ensures that the system satisfies all its objectives. Incidental and malicious behavior is anticipated. The requirement defines desired output for all different inputs (Lamsweerde, 2009, Hull et al., 2005).

- **Consistency**
  Requirements, assumptions, and domain properties are compatible (Lamsweerde, 2009, Hull et al., 2005).

- **Adequacy**
  Requirements address the actual needs for the system. This includes correct translation of requirements between levels, correctly described laws in the domain properties, and realistic environmental assumptions (Lamsweerde, 2009).

- **Unambiguity**
  Requirements, assumptions, and domain properties are formulated so they can only be interpreted in one way. Terms are used consistently (Lamsweerde, 2009, Hull et al., 2005, Kotonya and Sommerville, 1998).

- **Measurability**
  Requirements allow for alternatives to be evaluated against them, and it is possible to see if they are satisfied or not. Assumptions are observable in the environment (Lamsweerde, 2009, Kotonya and Sommerville, 1998).

- **Pertinence**
  The requirements work towards objectives for the system (Lamsweerde, 2009).

- **Feasibility**
  Requirements are realizable in terms of budget, schedule, and technology (Lamsweerde, 2009, Kotonya and Sommerville, 1998).

- **Comprehensibility**
  The formulation of requirements, assumptions, and domain properties are comprehensible for their users (Lamsweerde, 2009).

- **Good structuring**

- **Modifiability**
  It is possible to revise, adapt, extend, or contract the requirements document (Lamsweerde, 2009).

- **Traceability**
  It is easy to retrieve the context of the creation, modification, or use of items in the requirement document. The impact of item creation, modification, or deletion is easily assessable (Lamsweerde, 2009, Hull et al., 2005).
According to Hood (2008) the decision of which feasible requirements will be included in the specification is also an important part of the requirement analysis. Lamsweerde (2009) similarly to the sought after characteristics, lists unwanted defects that occur in the requirement engineering process. Some are simply opposites to the sought after characteristics, while others are completely independent. Lamsweerde (2009) categorizes these defects as errors and flaws. Examples of errors are: omissions, contradictions, inadequacies, and ambiguity and immeasurability. Examples of flaws are: noise, unfeasibility, unintelligibility, and poor modifiability.

### 3.6.4 Requirement negotiation

If it is concluded that not all requirements will be able to be met, there according to Hood (2008) needs to be some type of criteria evaluating which requirements should come first. According to Kotonya and Sommerville (1998) a complex system will have many actors, and these actors will at some point have disagreements about the system requirements, and will prioritize the requirements differently depending on their background. This results in requirements negotiation always being necessary. Requirement conflicts and overlaps are resolved through the interchange of information, discussions, and resolution of disagreements (Kotonya and Sommerville, 1998). Kotonya and Sommerville (1998) continue to state, that in principle requirement negotiation should be an objective process, basing judgments on technical and organizational needs, but they claim that this is not often reality. Kotonya and Sommerville (1998) also say that it cannot be assumed that decisions for one requirement can be applied to related requirements. According to Lamsweerde (2009), the steps of requirement negotiation are: identify actors including their objectives, identify differences in the different actors’ objectives, and reconcile differences through negotiation. An alternative negotiation process by Lamsweerde (2009) is:

1. Identify overlapping statements
2. Detect conflicts among them & document them
3. Generate conflict resolutions
4. Evaluate resolutions and select best ones

Lamsweerde (2009) speaks of how resolutions need to be correct in time. If resolutions are met too early, useful information may not be elicited yet. If resolutions are met too late, development will have started with inconsistent statements.

### 3.6.5 Documentation of specifications and requirements

Jiang et al. (2005) highlight the need for tools handling requirements as they can amount to a high number. Jiang et al. (2005) claim that a smaller project might make do with a simple word processor but makes suggestions to different SW solutions for more complex collections of requirements.

#### Contents of requirement documentation

Different authors put focus on different things to include in a requirement documentation. Table 3.2 below compiles what can be contained in a requirement document according to several different sources. Due to similar things being described in alternative terminology, liberty has been taken to reformulate certain posts and include data from different sources which originally had different names but similar or same definitions under the same list items in the table. The low amount shared check marks show how differently the document can be viewed and used in different companies and situations.
Table 3.2: Suggested parts of requirement documentation

<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Development process constraints</td>
</tr>
<tr>
<td>Integration (other systems)</td>
</tr>
<tr>
<td>Operational constraints</td>
</tr>
<tr>
<td>Origin/Source</td>
</tr>
<tr>
<td>Relation (between requirements)</td>
</tr>
<tr>
<td>Design constraints</td>
</tr>
<tr>
<td>Likely variants &amp; revisions</td>
</tr>
<tr>
<td>Motivation</td>
</tr>
<tr>
<td>Acceptance test data</td>
</tr>
<tr>
<td>Background information</td>
</tr>
<tr>
<td>Conflicts</td>
</tr>
<tr>
<td>Cost figures</td>
</tr>
<tr>
<td>Customer dissatisfaction</td>
</tr>
<tr>
<td>Customer satisfaction</td>
</tr>
<tr>
<td>Definition of the scope (include/exclude)</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Descriptive text bridging different sections of the document</td>
</tr>
<tr>
<td>Domain and environmental information</td>
</tr>
<tr>
<td>Function, action performed</td>
</tr>
<tr>
<td>Functionality, capability</td>
</tr>
<tr>
<td>General constraints</td>
</tr>
<tr>
<td>Glossary</td>
</tr>
<tr>
<td>History</td>
</tr>
<tr>
<td>Interfaces</td>
</tr>
<tr>
<td>Objectives</td>
</tr>
<tr>
<td>Overall system properties and attributes</td>
</tr>
<tr>
<td>Product perspective</td>
</tr>
<tr>
<td>Quality characteristics</td>
</tr>
<tr>
<td>Rationale</td>
</tr>
<tr>
<td>References</td>
</tr>
<tr>
<td>Requirement number</td>
</tr>
<tr>
<td>Requirement type</td>
</tr>
<tr>
<td>Requirements (sorted according to domain)</td>
</tr>
<tr>
<td>Responsibilities</td>
</tr>
<tr>
<td>Service</td>
</tr>
<tr>
<td>Stakeholder description</td>
</tr>
<tr>
<td>Status</td>
</tr>
<tr>
<td>Summary of models (for deriving requirements)</td>
</tr>
</tbody>
</table>
3.6. Requirement handling in requirement engineering

Johannesson et al. (2013) mention the importance of being able to track requirements, including information such as origin, motivation, relations, and status of requirement. Kotonya and Sommerville (1998) states that requirement documentation should describe: what services and functions the system should provide, the operational constraints of the system, overall properties of the system, definitions of systems with which the system will integrate, the domain in which the system will be applied, and constraints on the development process of the system.

According to Sommerville and Sawyer (1997) the documentation needs to be flexible, allowing for the user to omit parts and add sections as needed. They suggest having stable and variant parts, as some parts such as introduction and glossary always should be included. Hull et al. (2005) calls for a clear version system so it is possible to differentiate between old and new requirements. According to Hull et al. (2005) a correct structure of requirements assists the users of the requirement document in many ways: minimizing the number of requirements by detecting omissions and duplications, and rejecting poor requirements. It also helps the reader understand large amounts of information by organizing requirements in sets for different topics. Good structure also helps the reader evaluate the requirements, allowing them to eliminate conflicts between requirements and reuse requirements across projects.

The handling of rejected requirements

Hood (2008) illuminates the issue of documentation of rejected requirements. If these requirements are not documented along with their reason for rejection they may come up again in the future and have to be re-evaluated. Hood (2008) also speaks of the issues of managing changed requirements. It needs to be decided if a new requirement can simply replace an old one, or if the development of the requirement should be traceable. One also should consider if reasoning behind decisions should be documented.

3.6.6 Requirement validation

According to Kotonya and Sommerville (1998), validation is the final stage of requirement engineering. The purpose of validation is to check the requirements, certifying that the requirements represent the, by actors, sought after system (Kotonya and Sommerville, 1998, Cheng and Atlee, 2007, Stjepandić). Validation, according to Chemuturi and Gilb (2013), ensures that 'you built the right thing'. Kotonya and Sommerville (1998) state that the process involves several different actors, including requirement engineers, and system designers. Cheng and Atlee (2007), Stjepandić similarly state that validation requires direct involvement of actors for reviewing the requirement. The job of the actors is to analyze the requirements looking for problems, omissions, and ambiguities (Kotonya and Sommerville, 1998, Chemuturi and Gilb, 2013). According to Kotonya and Sommerville (1998), the requirement document, organizational standards, and organizational knowledge are inputs to the requirement engineering validation process. A list of problems, and agreed actions are outputs. Chemuturi and Gilb (2013) mention several different techniques for validating requirements such as brainstorming, story-boarding, prototyping, and different types of reviews either from experts or end users. Kotonya and Sommerville (1998) point out that the requirement validation process is a lengthy one. Kotonya and Sommerville (1998) mention that there is a tendency towards rushing through it which can, and most probably will, result in rework much more costly than if the validation would have been given the appropriate time frame. According to Kotonya and Sommerville (1998) problems which can be discovered during requirement validation are unclear, conflicting, or unrealistic requirements, and missing information. Resolving problems in requirement engineering requires discussions and negotiation.
3.6.7 Modification handling in requirements

Requirement modification, along with requirement evolution, concerns the need for change during, and after the requirement engineering process (Lamsweerde 2009, Hull et al. 2005) points out how difficult it is to maintain the originally agreed upon requirements, and the need for having modification handling processes in place. As a project proceeds, different levels of formality are needed in the change process. Early on, changes should be easy to perform, but later on, when formal agreements exist, the change process must also be formal - taking into consideration the impact of changes on the project as a whole. Lamsweerde (2009) discusses the natural evolution of requirements, and gives several examples of causes; a need for change can occur from requirement evaluation and analysis, or it might occur once development has started. It might also occur after the system has been deployed as this gives a new understanding. According to Lamsweerde (2009), if the process is well structured and designed for modifiability, changes can usually be made locally. Traceability contributes to the ease of localizing which changes need to be made (Lamsweerde 2009).

3.6.8 Verification using requirements

The use of requirements for verification is generally not included in the area of requirement engineering. To be able to make a comparison towards steps after the requirement process in other theoretical areas however this chapter was included also in the requirement engineering section. Hood (2008) states the importance of knowing already when the requirements are defined, how a product could be tested to see if it fulfills them. It is also important to state when tests should be carried out, along with who is responsible for carrying out the tests. If there is not enough budget to test all requirements, it needs to be decided which requirements should be tested and why. According to Hull et al. (2005), a system needs to be tested for its fulfillment of requirements in different ways depending on the characteristics of the system. The assessment according to Hull et al. (2005) should consist of a mixture of tests and trials. Documentation can help linking requirements to tests and test results - allowing for a correlation between test results and requirement fulfillment, but Hood (2008) also warns of the questionable validity of this type of linkage when changes occur in the requirements. The next question in verification according to Hood (2008) is what happens if a test fails. This is why requirement prioritization is important. It decides which requirements must be met.

3.6.9 Summary of the requirement engineering process

There are three dimensions in requirement engineering; the specification, agreement, and representations. There is a consensus that requirement engineering requires an iterative process to work. Requirements are collected from different actors using different types of elicitation techniques; interviews, questionnaires, and documentation being the most usual ones. Once collected, requirements are analyzed through evaluation and control towards requirements for requirements. Since many actors are involved in the development of complex products, there is a need for requirement negotiation to enable a final agreement of what the system needs to fulfill. There also is a need for having a modification handling process in place. Requirements are documented to ensure traceability but also understand why some requirements were not chosen, the documentation adds towards a complete, and formal, specification. The requirements need to be validated so that the needs of the actors are correctly represented in the requirement specification. The requirements are used for verifying the system during and after development.
4 Results

In this chapter, interview results are presented, highlighting differences and similarities in the different companies’ requirement handling processes. After this interview results concerning current and future integration, and future obstacles and enablers for increasingly integrated development are presented.

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

In this chapter data from the interviews conducted at four different companies regarding their requirement handling process is presented.

4.1.1 Actors

Below, a comparison of actors mentioned by the different companies are presented in Table 4.1. 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. This means 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 H.1, I.1, J.1, and K.1.

Reuse of knowledge in creating requirements

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 H.5). At company X they use a platform way of thinking, building things on a shared base, and the reused knowledge is also established (Appendix I.3). At company Y, internal requirements originate either from experience, or newly developed technology (Appendix J.5). They also use the concept of reuseability in their requirement collection methodology (Appendix J.5). In company Z, old requirements can be reused in new projects (Appendix K.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.

4.1.2 Overall processes at interviewed 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 H.2). At company X instead there are gates during the process.
4. RESULTS

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>

At company X they also work with ‘handshake’, and only things previously ‘OK:ed’ should be ‘handshaked upon’ (Appendix I.2). Also company Y works with gates, and it is a gate-group which make official decisions of changes at these gates (Appendix J.10). In company Z, gates are used during the early development to approve changes. The gate ‘committee’ here consists of the technical manager, and project manager (Appendix K.10).

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

Figure 4.1: Process of company W, mainly developing SW
4.1. RQ 1 & RQ 2 - Differences and similarities between different development processes’ requirement handling

Customer interaction

The view on who the customer is 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 K.5).
4. RESULTS

Company W sees the potential benefits of increasing collaboration between customers and developers (Appendix H.5).

Generalization of requirements

Company X and company Z mention how their customers are very different, or in different regions, making generalizations concerning their needs difficult to make (Appendix I.17,K.5).

Breakdown of requirements

In three of four of the 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. This is similar to the breakdown described by the other figures.

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.
4.1.3 Time-frame

According to interviewees at company W, in SW; time-frames and customer needs are framed by the project time limitation, so the project determines how much time can be given to customer needs and doing separate projects within the project (Appendix H.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 K.16). According to company Y, in big HW development projects, time can be spent on categorization and prioritization of requirements. This is compared to service development projects, which have less time, where time is given only for basic identification of key requirements (Appendix J.7). In this time comparison company X agrees, claiming that service development usually has shorter time frames than HW development (Appendix I.2). Company X proceeds to claim that it takes a long time before everything can be implemented in HW development, requiring prioritization (Appendix I.18).

4.1.4 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 H.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 H.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 J.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 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 K.6). In the interviews at company X, general ‘must’ requirements - or ‘trust marks’ do exists, but there is no other mention of any prioritizations or rankings (I.6).

4.1.5 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 H.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 I.8). In company Y, the requirement specification base is shared globally within the company (Appendix J.8). In company Z, the company has an internal standard for documenting requirements. All requirement utilize the same type of document regardless from their origin (Appendix K.8). The understanding between departments is inadequate, and is seen as difficult by interviewees at company Z (Appendix K.4). Company X comments that requirements are not currently universally understandable (Appendix I.16).

4.1.6 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 L.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 company 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>

#### 4.1.7 Modifications

Modification handling is different in the different companies. In company W modifications require change requests. Change requests are processed regardless if a change takes place or not (Appendix [H.10]). In company X, design review forums/meetings are used, here the concepts regarding cross functional requirements are discussed and design concepts are reviewed. It is at these forums it is decided if a change of function is acceptable or not (Appendix [I.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 [J.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 [K.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 [J.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 [K.10]).
4.1.8 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 H.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 I.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 J.12). In company Z there is not 100% traceability between the requirement givers and the specification but all actors within the company have a personal contact (Appendix K.9). Some parts have better traceability, in company Z for example, the link between the fuzzy MRS and the PRS is very clear (Appendix K.9). In company Y the interviewees perceived the traceability as good (Appendix J.9). Prestudies allow them to map and describe the underlying reasons for requirements (Appendix J.8). There however were also weaknesses which came to light concerning traceability. In company W the is a lack of overview of the connection between requirements on different levels was mentioned (Appendix H.9). Interviewees at company X explained that traceability is more clear 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 J.9).

4.1.9 Validation

Validation, or the control of technical requirements toward 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 there is an annual measure of user reaction which is partially indexed. 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 I.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 (Appendix J.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 have focus persons/groups who can make a judgment, but the case can also be that the design team makes the judgment based on experience (Appendix K.12).

4.2 RQ 3 & RQ 4 - Integration state, current and target

In this chapter, results which concern integration of development departments, both current and future interests, is presented.

4.2.1 Department specific process adaption

According to company W there is no all-around process, this means that it is hard to pick one correct process for the current development (Appendix H.3). In company X all processes originate from the same general process, implying that the processes are adapted (Appendix I.2). At company Y similarly the different departments start from the same development process (Appendix J.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 the specific project in which it is being applied (Appendix L.3). Further the requirements of company X 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 L.8). At company Z the time
frame of a project depends on its size. If the project concerns a completely new product, the concept phase is 6 months and the design phase 1 year (Appendix K.2).

4.2.2 Product definitions

According to company X, today, a product includes more than its physical aspects, such as services and software (Appendix I.15). At company Y, competition has forced the focus of service to change to offer service-agreements instead of spare parts (Appendix J.16). At company Z the idea is that it is functions being delivered, not articles, and that the focus of the company should be on the functions (Appendix K.18).

4.2.3 Balancing of requirements and compromising between departments

According to company W it is challenging to keep a balance between several different actors and their requirements. All different actors need to cooperate and be adaptive to each other (Appendix H.14). Similarly, company X explains that with a complex system there are a lot of requirements, and there is always a need for balancing of requirements (Appendix I.6). According to company X, it is hard to manage the high amount of requirements since it requires balancing and compromising (I.8). ‘Attributes’ (as mentioned by company X) need to be balanced, and so does performances (Appendix I.16). In company X there will be several viable solutions, and only a balance of requirements can show which is the optimal compromise (Appendix I.6). In company Z, there is always a need to balance the requirements. All requirements need to be considered, but not all requirements can be implemented (Appendix K.6). They claim that stakeholders understand the need for compromise (Appendix K.4). In company Y balancing and compromising is not mentioned explicitly, but it is important to understand why requirements should be included (Appendix J.4).

4.2.4 Goals for dependency in development and requirement handling

The opinions regarding dependency differ among the interviewed companies. In company W there is a long term goal to cut the costs of development, and they are achieving this goal by decreasing the amount of dependencies (Appendix H.18). At company X, the interviewees have different point of view regarding dependency. They emphasized on requirements related to HW and SW need to be separate, not dependent (Appendix I.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 I.15). At company Y, checklists are used in an attempt to map which departments affect which other departments (Appendix J.15). At company Z the perception is that the dependency between different development departments is very high during the actual development (Appendix K.15), and that to create a system, contribution from several departments is needed (Appendix K.15).

4.2.5 Collaboration

The view of 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 that include acting as interfaces between sub-systems, or different departments, will be more important (Appendix H.18). Due to this separate mindset the interviewees at company W believe that product development teams should have more interfaces to improve their understanding of context (Appendix H.18). At company X it is almost the opposite. Between some sub-functions there is very close interaction (Appendix I.14). In such a complex product, company makes use of, and sees multi-objective design optimization as necessary. Multi-objective design optimization is described as having to consider all requirements and their interactions (Appendix I.14). Company X currently has close collaboration and a high amount of integration between all
relevant departments according to the interviewees (Appendix I.18). At company Y the interviewees believe 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 J.18). In company Z the separation between departments is clear even though they work together very closely together (Appendix K.14). In interdisciplinary development, one group will own the requirement or problem, and will need to ask other groups for help with solving it (Appendix K.14).

### 4.2.6 Organizational systems and traceability issues

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 H.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 of sub-departments, they do not currently have a good way for tracking this back and highlighting requirements for change in other sub-departments’ branches (Appendix H.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 I.17). They believe that what they need is a meta-model\(^1\) to see correlation between objects (Appendix I.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 J.8). Similarly, company Z states 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 K.18). Today the connection between requirements and software code is not clear, or non-existent at company Z (Appendix K.11). If they had a structure in a database, it would be easier to understand the connection all the way down to the programmer. Company Z wants an increased traceability (Appendix K.18).

\(^1\) A meta-model according to the interviewees at company X is a more general model explaining connections without going to far into detail. It focuses more on conveying general understanding than on exact accuracy of detailed information.
4.3 RQ 5 & RQ 6 - Obstacles and enablers, simultaneous hardware & software engineering

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

4.3.1 Division and separation

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

4.3.2 Problems concerning change of process

At company W, the perception is that it is hard to convince others of the existence of problems (Appendix H.17). At company Y the perceived issue is that it is hard to convince, and also involve, the senior management in new ideas (Appendix J.17). At company Z there is opposition to change. It is stated in an interview that even when new requirement handling processes have been tried out, and lessons are available, the will to actually learn is lacking from the organization and management (Appendix K.17).

4.3.3 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 have too much work they become a bottleneck for the entire process (Appendix J.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 H.15). Company Z want more system engineers and function responsible people to solve similar issues as these roles would relieve currently overloaded positions and departments (Appendix K.18).

4.3.4 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 thorough categorization and prioritization of requirements. In service, time is limited and within requirement handling only key requirements can be identified. No categorization or prioritization takes place because there simply is not time (Appendix J.7). Further company Y mention how collaboration is added on top of the existing workload, and no additional time is budgeted for this task (Appendix J.18). Company X simply mentioned that service usually has shorter time-frames without putting it it positive or negative light (Appendix I.2). Time limitation was mentioned as the main
4.3. RQ 5 & RQ 6 - Obstacles and enablers, simultaneous hardware & software engineering

problem in requirement engineering for company Z. They feel that they have too little time and few staff for the task (Appendix K.7).

4.3.5 Increased traceability in requirements process

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

The core of this thesis is the RQs. Having the ‘correct’ RQs therefore is very important. Initially the thesis had only three RQs, covering differences, similarities, and obstacles towards future increased integration. The final RQs 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 was added. For the RQs concerning obstacles and enablers to be relevant, the question of current and future integration needed answering. Once these RQs were established, the choice was made to use them as an outline to much of the thesis content. They played a key role in the creation of the interview guide and therefore also the interview questionnaire. They also had a part in the creation of ‘theory-driven’ code words. Results are grouped according to them, as is the result discussion in this chapter. First in this discussion chapter however, comes a section which attempts to show links between theoretical claims, interview questions, and the answers from the companies concerning six steps of requirement handling. As the theory is based on more general theories, and the interview data is from more specific processes, it is not always possible to make accurate comparisons, and the lack of common factors does not necessarily mean that the companies do not make use of researched processes.

5.1 Discussion of methodology

As mentioned in the method, chapter 2 interviews was the only reasonable data collection methodology, even if other methodologies might have been better choices looking to the detailed nature of the RQs. 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.

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 (van Boeijen et al. 2014; Kvale and Brinkmann, 2009) 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.
5.2 Link between different theoretical domains, interview questions, and answers

After exploring the different theoretical frameworks for HW development, SW development, and requirement engineering it was possible to make an initial comparison between models, processes, and concepts. Once interview data was collected, this could be added to the analysis. 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. The different theoretical fields have slightly different views on how requirements are used in development. HW and SW agree that requirements are used starting early on in the process (Ulrich and Eppinger 2012, Tahir 2017). Requirement engineering has more focus on the link between requirement and testing, resulting in the requirements being used more towards the end of the process (Hood 2008). HW uses requirements continuously, as base for initial ideas as well as the decision matrix, and verification template. The iterative nature of SW development allows for stating requirements, testing, and then reviewing requirements, without much use of requirements in between in the development. This explains the lack of connection between requirement and code explained by software developers in interviews. The companies seem to be formulating requirements and then use them in tests (tests are covered further in the chapter concerning test and verification below). Companies developing HW should be able to make use of requirements further as described by HW theory, as inspiration and base for decision matrices. It is possible that this already takes place, but it is not widely mentioned in the interview data.

5.2.1 Requirement elicitation

The first step in the requirement handling process is to collect requirement, an act referred to as requirement elicitation. Different theoretical frameworks say that it is appropriate to use questionnaires, interviews, or in the SW’s case, observations, to collect requirements (Hauser and Clausing 1988, Ullman 2002, Sutcliffe 2002, ISO 2011, Hull et al. 2005, Lamsweerde 2009). In HW and requirement engineering, all affected actors should be included (Lamsweerde 2009), and in SW the development team is given focus (Tahir 2017). In practice, requirements are collected internally, company Z makes use of questionnaires as suggested by HW and requirement engineering theory. This elicitation from internal departments shows signs of integration in company Z’s current structure. Company W and Y give development team much freedom with requirements as suggested by SW theory, this even if company Y does not have a SW focus. Placing the requirements in the individual development departments in this way, opposing to company Z speaks of little integration currently in company W and Y. Something echoed by company W in the interviews were the HW department was referred to as a completely separate organization. Requirements coming from customers or other external actors are usually handled by market, service, or customer unit, and the interviewees have little to say of these departments’ specific processes. To get a more detailed understanding of how external requirements are collected, interviews with employees from these departments would be necessary. Theory puts emphasis on involving all affected actors, and companies currently focus on internal actors. If external actors are included at all, there still is no direct connection between them and the developers. How does a company identify success factors in its products?

5.2.2 Requirement analysis

After collecting requirements, they are analyzed. A part of requirement analysis for both HW and SW is prioritization. In HW, this is done through weighting (Ullman 2002) and SW through rankings (Tahir 2017). Prioritization issues being lifted means that the companies
are applying prioritization processes, a common factor for both HW and SW development. Worth mentioning is that the companies are suggesting review as a solution for solving conflicts, and once issues are resolved, having a shared prioritization process might be possible. Prioritization and its issues are discussed more thorough in the ‘prioritization’ chapter. The theoretical framework concerning HW engineering shares requiring certain traits from the requirements with the requirement engineering framework (Lamsweerde 2009, Ullman 2002, Roozenburg and Eekels 1995). From the interviews very different requirement types were given, see section 4.1.6, but the traits mentioned by HW and requirement engineering only occurred in one of the sets of interview data. More discussion on this is presented in the chapter ‘Requirement types and characteristics’. For the analysis HW specifically looks to compare with the existing market, using market screening as mentioned by Douglas et al. (1978), Ullman (2002). SW looks to differentiate and clarify requirements Tahir (2017). The breakdown which is done by 3/4 of the companies could be seen as a type of requirement clarification. Breakdown being a common factor in companies with very different focuses supports increased integration and the RQ concerning similarities. Comparison with competition is not mentioned explicitly by the companies in the interview data, this however does not mean it does not happen, as it may take place at a more strategic level in the company, possibly earlier in the process.

5.2.3 Requirement documentation

Once requirements are collected and have been analyzed, they are documented. HW, SW and requirement engineering have very little in common when it comes to requirement documentation. In the theoretical framework covering requirement engineering 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 (Kotonya and Sommerville 1998, Lamsweerde 2009, Sommerville and Sawyer 1997, Hull et al. 2005, Sutcliffe 2002). HW speak of measures and target values being included in the requirement documentation (Ulrich and Eppinger 2012), SW of functions and services (Tahir 2017). This lack of commonalities does not necessarily mean that the fields have nothing in common, but rather that the terminology is very wide. This is further supported by the interview data on what a requirement documentation contains, everything from requirement number to related drawings and motivations for the existence of the requirement (Appendices I.8, J.8). All companies however do sport an internal documentation standard which in many ways supports a interdisciplinary environment and is a similarity looking to the RQs. This standard document usually had certain parts which should be included, similar to the requirement characteristics mentioned in the requirement analysis chapter. The IRS documentation which exists in company Z is a sign of their current state of integration, it however gets mixed reviews from the interviewees. Requirements in the IRS may not be relevant to the receiving party, or be at a too abstract level, making the documentation an obstacle instead of an enabler for integration. If the users see this type of document as cumbersome to use and understand, they will be opposed to using it. The interviewees also mention cases in which the documentation works and provides requirements very accurately. It would seem this is a case where increased understanding is needed between departments, once this understanding exists, IRS may very well be welcome aids for increased integration.

5.2.4 Requirement modification

After documenting requirements, it will become necessary at some point to modify them. As stated by company X, plans are means to move forward and should be flexible, and easy to change (Appendix I.2). Modifications which are initiated by external factors, such as an actor requesting an additional requirement to be fulfilled late in the process, do not have any mention in the theoretical framework considering HW. HW theory only mentions evolutionary
change (Johannesson et al., 2013). This is peculiar when all companies use formal modification handling processes and many of the companies interviewed have HW in their focus. In SW and requirement engineering there are mentions of formal modification handling processes (Tahir, 2017; Hull et al., 2005). It is probably not that the HW companies are innovative, rather the theoretical framework is flawed. If no theory exists concerning HW and modification methodologies handling external changes, this needs to be researched. SW also has the ability to absorb changes in its iterative models (Tahir, 2017) and this is the methodology which seems to be used by company W who mention ‘checkpoints’ instead of gates. This type of iterative modification methodology is more reasonable to not be mentioned by HW theory.

As stated in company X, there are physical factors stopping HW and SW from sharing certain processes. A clear difference between the two fields. The gate process has its advantages, time, cost, and other effects of the change are given a formal review in the companies, this is exactly what Hull et al. (2005) suggests in the theoretical framework of requirement handling. Having to implement a formal process in HW companies is not all good however. There were mentions from the companies that it is difficult to get confirmation from both managing positions which are over encumbered in work, but also from external actors. A more informal, iterative modification process would perhaps save both time and money. When interviewing company Z, they had a quite clear time-frame, even mentioning specifically at which gate a more formal modification methodology became necessary. It would be interesting to see when this point of time occurs in the other companies, and if the companies were not anonymous it would also be possible to compare when this point of time is depending on the product.

5.2.5 Requirement validation

Validation aims to confirm that the needs of the original requirement givers are correctly represented in the final requirements. One company failed to mention validation, and another claims that once the translation from fuzzy requirements to technical requirements has taken place no further validation has to be performed. This does not align with validation as described in the theoretical framework. Validation means to check the requirements towards needs, company Z here claims that checking refined requirements towards early requirements is enough validation. The companies which are following the recommendations of theory, are doing this only partially, making use of actor validation which is recommended by requirement engineering (Kotonya and Sommerville, 1998; Stjepandić, Chemuturi and Gilb, 2013). These companies involve only internal actors, while the theory recommends involving all relevant actors, external included. The reasons for this could be many, one being that it is difficult in a company which develops a complex product to motivate one of the few actors to take their time for validation. Making use of internal actors makes the process more streamlined as these actors already know the process. HW recommends using competitive mapping to ensure that requirements are at a suitable level compared to competitors, this is not mentioned at all by the companies in the interviews. It is possible that this, as previously mentioned, that this comparison with competition takes place at a strategic department, not with the developers which was the role of many of the interviewees.

5.2.6 Use of requirements in test and verification

Having gone through all the prior steps of requirement handling, eventually a product will be produced. Once there is a product, it can be tested, and the requirements verified. HW needs to verify as they go, as described in theory where HW uses requirements in decision matrices and as base for verification templates (Ulrich and Eppinger, 2012). SW and requirement engineering focuses on tests; which tests, when to tests, how to test, how to relate tests to requirements (Hull et al., 2005; Hood, 2008). The SW has a completely different set of characteristics which allows for it to be completely rebuilt from one day to another. This highlights
5.2. Link between different theoretical domains, interview questions, and answers

There is a clear difference between the two fields of HW and SW. It is not possible for HW to test in the same way. However, at the interviewed companies, ‘testing’ was the recurring answer for questions concerning verification. HW has solved this issue of not being able to test until a final product exists by testing all parts possible as soon as the individual parts are done. It would seem that HW theory needs additions concerning this type of testing.

5.2.7 Summary of comparisons

In the Figures 5.1 and 5.2, a graphical summary of the comparisons can be found. Elicitation and analysis have many agreements between the different theoretical fields, but not much consensus between theory and company data. In documentation there is little in common both between theoretical frameworks and the theory and company data. In modification, SW’s and requirement engineering’s formal modification methodology is found in the company data. In validation the actor validation from requirement engineering is applied also in companies but only with internal actors. In verification, testing which occurs both in SW and requirement engineering, is mentioned by all companies, also those who work with HW.

<table>
<thead>
<tr>
<th></th>
<th>Elicitation</th>
<th>Analysis</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware</strong></td>
<td>questionnaire</td>
<td>weighting</td>
<td>measurement</td>
</tr>
<tr>
<td><strong>Engineering</strong></td>
<td></td>
<td>customers &amp; characteristics</td>
<td>engineering</td>
</tr>
<tr>
<td></td>
<td></td>
<td>competitors</td>
<td>&amp; target</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>specification</td>
</tr>
<tr>
<td><strong>Software</strong></td>
<td>interviews</td>
<td>rankings</td>
<td>software</td>
</tr>
<tr>
<td><strong>Engineering</strong></td>
<td>observations</td>
<td>differentiation</td>
<td>specification</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&amp; functions</td>
</tr>
<tr>
<td><strong>Requirement</strong></td>
<td>questionnaire</td>
<td>caracteristics</td>
<td>source</td>
</tr>
<tr>
<td><strong>Engineering</strong></td>
<td>interviews</td>
<td></td>
<td>constraints</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>relations</td>
</tr>
<tr>
<td><strong>Company</strong></td>
<td>experience</td>
<td>review</td>
<td>documentation</td>
</tr>
<tr>
<td><strong>answers</strong></td>
<td>customer interaction</td>
<td>prioritization</td>
<td>inter-department</td>
</tr>
<tr>
<td></td>
<td></td>
<td>breakdown</td>
<td>standard</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>documentation</td>
</tr>
</tbody>
</table>

Figure 5.1: Comparison for HW, SW, requirement engineering and company answers for requirement handling steps: elicitation, analysis, and documentation.
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

At first glance it is possible to believe actors is a part of answering RQ 1, differences. The amount of different actors which occurred at the four companies (section 4.1.1) however hints at the difficulty to make general conclusions concerning actors in the process. The companies have different focuses and this alone makes it natural to assume that different actors are relevant to the companies. The actors would require further study to be able to make a more accurate inter-company comparison, if it is possible at all. In theory concerning hardware, more specifically the house of quality (Hauser and Clausing, 1988, Ullman, 2002), no particular actors are pinpointed, there is only the statement that everyone affected should be involved.

As different actors will be affected in different companies, it is possible that the companies have no actors in common. In requirement engineering a list of examples is provided (Hull et al., 2005), but it is also stated that this list is a base, and that it should be modified. Requirement handling is unique for every company. Interviewing a couple of more companies, or more companies in one specific field, might result in reaching a ‘max’ amount of actors for the different fields, in the shape of a list similar to that of Hull et al. (2005) where a comparison can be made. If deeper descriptions of these actors were made it might be that the actors in the processes having more in common than is currently visible. Additionally with other questions in the questionnaire, completely different actors may have been revealed.
5.3.2 Overall processes at companies

Processes including gates, which was mentioned in the HW theory by Ulrich and Eppinger (2012), are used at all companies but one, the more SW focused company W uses 'checkpoints'. An explanation for this is the more agile process applied in SW development Codington-Lacerte (2016), Tahir (2017). This shows that there is a difference between the level of finality at the points of agreement in the HW and SW processes, where SW is more flexible, which needs to be taken into consideration in answering RQ 1.

On the other hand, all of the companies have some type of formalized process. So there is a shared understanding of the need to generalize the process used into something schematic. According to interviewees working with SW this schematic needs to be more agile to enable more efficient SW development. In company Z an increased agility was requested, both specifically into the SW department, but also generally into the development (Appendix K.18). This would lead towards the conclusion that SW requires a more agile process to optimize development, and to be able to coordinate with HW, HW needs to move towards a more agile process as well. There is very little that points to this being the direction HW is moving at the companies interviewed. The theory concerning basic HW development, which describes a sequential model, has also remained similar for years, making it possible to merge models which were 10 years apart into one Figure 3.2. It will be difficult to create one shared process which is promoted by Küster et al. (2016). Agile is, as described by Codington-Lacerte (2016), sprung from dysfunctional, in this case SW, development. If issues addressed by agile development do not at all exist in HW development there is no point in applying agile development processes there. Ulrich and Eppinger (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. It is possible that sources exist which speak of iterative and agile HW development models, but as none of the interviewed companies seemed to implement non-sequential processes, no such theory was added to the theoretical framework during thesis iterations. It is probably for the better to figure out what can be shared as is the aim of RQ 2 and implement a partially shared process and otherwise aim for an increased understanding of differences between the departments.

5.3.3 Requirement handling processes

Within the overall processes, there are requirement handling processes. Three of four of the companies apply requirement breakdown, and the fourth company has different levels of requirement documentation. Increasing the amount of requirements to break down for a more complex product should not be any problem for companies which already have a structured breakdown process. This similar methodology is relevant to answering RQ 2, as it should allow for an easy transition into more interdisciplinary requirement handling. The requirement breakdown, as explained in chapter 5.2.2 confirms the use of the requirement characteristic good structure (Lamsweerde 2009, Ullman 2002, Roozenburg and Eekels 1995), and differentiation explained by SW development.

5.3.4 Time-frame

Time-frames is a topic typically relevant for answering RQ 1. Naturally, projects which differ in size will also differ in time-frame given by management, as do time-frames in different organizations (Hallin and Karbom Gustavsson 2013). These different time-frames will continue to occur in more interdisciplinary projects. The division of time is different in the different development fields. HW generally has longer time frame due to longer lead-times, and SW spends much of its time on tests and verification. Time frames mentioned for HW and SW vary between a couple of months and several years, making comparisons between them difficult. The development processes simply put are different, and to be able to have a
more integrated interdisciplinary process these differences need to be considered. When HW
and SW need to cooperate today, HW processes can only be shortened to a certain extent due
to physical limitations. SW which is more iterative should be able to complete several iter-
ations, including tests, but the results need to be output in a different way to be understood
by HW. At company Z, HW feels SW is less complete (Appendix K.6) while SW developers
in several companies state that it is rare that other departments wait for them, which means
they often feel as if they are done first. It is not only SW which needs change, also HW needs
to understand SW’s different time-frame. Instead of having a mindset of different procedure
being equal to incorrect procedure it would seem both HW and SW need to learn respect for
the limitations and special requirements of the opposing field of development.

5.3.5 Prioritization

Several companies mention the issue of different prioritization in different departments. In-
creasing the amount of departments involved through a more interdisciplinary process could
add further to this issue as the interviewees seemed to think that for every level or step in
the process a new prioritization was introduced. This means that the issue of prioritization
reasonably needs to be resolved to achieve an efficient development. If the different depart-
ments had further understanding of each other’s processes, they might be able to prioritize
more objectively as they would understand the impact their own prioritization had on other
departments. There however is no guarantee that this is the case, nothing in theory or the
interview data from this thesis supports it. Further studies in companies with more trans-
parency would be needed.

A difference, as sought by RQ 1 was identified when it comes to prioritization. It is the
company Z which has a process which includes both HW and SW development which did
not lift complaints concerning prioritizations as the others. There, prioritization officially is
removed altogether after requirements have entered the requirement specification. This or-
der of steps follows that of requirement handling according to requirement engineering with
requirement negotiation as described by [Lamsweerde 2009]. The negotiation is already fin-
ished when requirements enter the specification. In requirement engineering prioritization is
necessary to decide which requirements must be fulfilled [Hood 2008], but with company Z’s
logic, all requirements in the requirement specification must be fulfilled. This statement; that
all requirements, once in the requirement specification, need to be fulfilled for the product to
be finished, makes all requirements equal for following departments.

It is this equal value of requirements which contrasts the prioritization issues mentioned
by the other companies. The complaint that other companies have, suggest that they do not
follow this theoretical model by [Lamsweerde 2009] in their requirement handling process.
In HW theory, prioritization is used as basis for values in decision matrices [Ullman 2002],
which means that the original prioritization, which occurs before requirements are put into
a requirement specification at company Z, needs to be archived so that a correct solution
can be selected at the end of the development process. Company W lifts an important issue
springing from the different internal priorities; something with a low priority might effect the
system as a whole, requiring it to be given higher priority (Appendix H.15).

The prioritization, or lack thereof, in company Z, combined with internal prioritizations
which do not support the whole solution in company W show that for a more interdisci-
plinary 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 as described by [Lamsweerde 2009], and in this case, more system engineers
will be needed to enable increased integration, as it is they, according to [Hagan 2017], who
have an understanding of how different pieces of a complex system fit together. Revision of
prioritizations should be possible to include in the already established modification method-
ologies. Prioritizations will then be made with the whole product in mind, and weighting
solutions in decision matrices based off these prioritizations as described by Ullman (2002), should result in selecting the, overall, best solutions.

5.3.6 Documentation standards

Documentation contains some similarities as sought by RQ 2. All companies use an internal documentation standard, the extension of unity however varies. Even if not specifically speaking of a documentation standard, Sommerville and Sawyer (1997) say that documentation should have stable and variant parts which hints at at least reusing documentation. Already having a shared documentation standard within the company should therefore support an increasingly interdisciplinary development. In this way working cross-departmental is not equally difficult as certain understanding already exists between departments in the shared parts of the requirement documentation. Further the shared documentation encourages collaboration, avoids delays in productivity, and reduces misunderstandings in general. The need for certain department specific factors in the documentation is understandable, but it is equally important that the documentation can be understood across department borders, something not currently true for the companies visited as stated by company X & Z. A more detailed study into the requirement specifications would be necessary to understand exactly why it is difficult to understand the requirement specification from different departments, if the documentation truly is unified.

5.3.7 Requirement types and characteristics

Similar to the actors, the requirement types mentioned by the companies were very different; usability, and robustness as mentioned by company Z, were actually requirement characteristics or requirements on requirements similar to those mentioned by Hull et al. (2005), Lamsweerde (2009), Roozenburg and Eekels (1995), Ullman (2002), Ulrich and Eppinger (2012) and Kotonya and Sommerville (1998) in the requirement handling chapters for HW development and requirement engineering respectively. At first glance it would seem requirement types and characteristics concern RQ 2, and contain difference which need to be overcome. However this conclusion can not be drawn, to be able to further compare requirement types a study specifically targeted towards this would be needed. This study would support the understanding of which requirement types are used in what development area.

5.3.8 Modifications

Theory claims modifications should be easy early on, and that later changes should be analyzed in a formal change process (Hull et al., 2005). The mindset towards modifications seems to be similar at the companies, especially concerning late additions. There are however also differences, as sought by RQ 1. For example only two of the companies speak of evaluating modifications in terms of time, costs, and effects, as described by Lamsweerde (2009). Looking to similarities, RQ 2, there is an agreement within the companies, both from HW and SW development departments, that a formal modification methodology should be used late in the process, at least when considering changes at a more general level. Further late modifications are only made if they are absolutely necessary. Having a formal modification methodology has certain advantages; an agreement is sought and the formal process makes sure that several actors’ opinions are covered. Further Hull et al. (2005) supports it. Company W (Appendix H.10) sees the possibility to avoid administration and this mentioned formality also late in the process if changes are performed locally. This ‘solution’ is mentioned by Lamsweerde (2009) according to whom, local changes are possible if the process is structured and designed for modifiability. This might be more common in SW development focused companies, so this being a statement from the only SW-focused company makes sense. A possible advantage of SW in modification needs to be confirmed in further studies.
5.3.9 Validation

In validation a unusual difference was found. It would seem it has little to do with differences between SW and HW, however it concerns the one company interviewed which has a strong focus on both, and therefore should already apply certain integration. Company Z claims, that no further validation is necessary once requirements are translated into technical requirements. This is a very bold claim. How can it be said with such certainty that the original interpretation was the correct one? After a lengthy process, is there any guarantee for that the interpretation has not changed internally? It is difficult to believe that the translation is viable as validation for the complete process, and companies should, aligned with requirement engineering theory as mentioned by Kotonya and Sommerville (1998), Stjepandić try to include the customers more in their validation process. Even if the other companies often make use of so-called internal customers, they are at least involving the customer later in the process. The purpose of validation is also specifically defined as check if the requirements represent the system sought by the actors, for example in their originally inputed needs (Kotonya and Sommerville 1998, Cheng and Atlee 2007, Stjepandić). It would be reasonable to believe, that as both theory suggest it, and other companies apply it, that also company Z should aim to involve more actors in their validation.

5.4 Discussion of results for RQ 3 and RQ 4

RQ 3 and RQ 4 concern the current and future approach to integration in companies. Many companies already have some aspects which support integration, but the view on the future of integration is very different, which is to be expected for something which is unpredictable. This thesis works from the hypothesis that integration will increase so in this chapter factors working for or against this are highlighted.

5.4.1 Roles and approaches in requirement handling

The roles of departments differ between the companies. There seems to always be one department which provides the initial requirements, a coordinating actor can be a good start for more interdisciplinary requirement handling as the target of requirement handling according to Sutcliffe (2002) is to come to a common agreement. The coordinating actor should take responsibility for reaching a common agreement. In companies W and X other departments than the one providing the initial requirements, input requirements important to them. Looking to RQ 3, these companies might be more accustomed to interdisciplinary processes as these companies’ departments already have to represent their field of development’s interests in the process of developing complex products. 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. These aspects which normally are handled by system engineering according to Hagan (2017). If independent departments which are not system engineers also have this understanding, this is very beneficial for cooperation. Having to deliver something to other departments or a shared requirement document is likely a supporting factor for more interdisciplinary development.

5.4.2 Department specific process adaption

The processes described in the theoretical framework, both generally for HW as well as requirement handling, are generic and vague, and the application of processes in companies naturally is company-specific. Küster et al. (2016) mention the concept of a shared process model. Looking to the current state of integration which is covered by RQ 3, some companies (X & Y) have a standard process from which department specific processes origin. The shared process should increase cross-department understanding, Küster et al. (2016) suggest
that the shared model helps to maintain consistency between different actors’ processes. At the same time the fact that the processes are adapted to fit the different department implies that the companies understand that different departments have different needs in their processes. Depending how the standard process is designed, it will be more or less appropriate to continue using in a more interdisciplinary development process. This means that the future integration, as covered by RQ 4 might require additional steps to be included in the core process, which encourage interaction between departments. Company W, which currently do not have a shared core process, claims that a suitable process needs to be chosen for the current development (Appendix [H.3]). This should mean that a core process which is more suited for one development branch, needs to be rebalanced so that all departments feel the core process is relevant to their process. However, Küster et al. (2016) mentions that if an identical process model is used by all actors, it cannot satisfy all of the individual actors needs, so adaption is still necessary.

### 5.4.3 Product definitions

More than one company states that the definition of a product is changing. This aligns with the initial statement by Tomiyama et al. (2007), that systems are becoming more complex and have a more multidisciplinary nature. Describing current integration, RQ 3, products now include services, or products which were traditionally HW-only which now also include SW. Already having this understanding shows a certain step towards integration, and also enables increased interdisciplinary development. The change towards more integrated products has taken place, but however, management and structures are not always in step. In company Y they speak of the difficulty of convincing management of new ideas (Appendix [J.17]) and in company Z focus is on ‘traditional’ HW development (Appendix [K.18]). This is further discussed in the next chapter as this concerns RQ 5, obstacles for future integration.

### 5.4.4 Balancing of requirements and compromising between departments

There is already an understanding that balancing is necessary in company X & Z (Appendix [I.6,K.6]), and that in a complex product development a common goal needs to be reached. Departments being willing to work towards a common, shared optimization is a good sign. Increasing interdisciplinary development will mean more compromises and more negotiation, according to Kotonya and Sommerville (1998) negotiation is always necessary. This should not be an issue as the companies seem ready, looking to their current mindset towards requirement balancing (Appendix [I.8,I.6,K.6]). Companies W and Y make no mention of balancing and compromising. This does not mean that there is no such thing in these companies.

### 5.4.5 Goals for dependency in development and requirement handling

According to Hagan (2017) it is the responsibility of system engineers to work with dependencies, and the analysis of these. The term system engineer was not used as described by theory in any of the company, instead alternative ways of handling dependencies existed. The approach of company Y with checklists and interdependence mapping (Appendix [J.15]) implies an understanding of the dependency. Company Y is building models to support dependency, instead of working against it. Company W’s long term goal to decrease dependency (Appendix [H.18]) might not be all bad though. As many dependencies as possible should be removed to allow for an as independent progress as possible in separate departments, which is something you want according to company X (Appendix [I.19]). Having many dependencies further increases the need for administration. Having many dependencies might also limit the amount of liberties taken by the individual departments in testing new ideas. Dependencies however can not disappear altogether as then the end result will
not be one product but several separate parts of a product. Both company X and Z have an understanding that there are close interactions between certain departments and contribution from several departments is needed to create a system (Appendix I.15, K.15). To achieve an optimal, well-functioning, complex system, companies need to embrace and understand when dependencies are needed. As Ullman (2002) spoke of in the eighth room of the House of Quality, an awareness of relationships, and therefore dependencies, need to be created, because dependencies will almost always exist.

5.4.6 Collaboration

Fraser et al. (2003) claims that collaboration is difficult but that successful collaboration leads to competitive advantage. Looking to the current state of integration, RQ 3, company X has close interaction and a high amount of integration (Appendix I.18). RQ 4 concerning the future is mentioned by company Y which sees an increased collaboration with other international sites in the future (Appendix J.18). Company W has a complete opposite view, wanting that people still work in separate sub-systems in the future (Appendix H.18). Fraser et al. (2003) speaks of collaboration being a natural part of product development, so to hope for it to be completely non-existent is unrealistic. Between these two views lies company Z. Company Z’s concept of letting different departments own requirements (Appendix K.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. This type of meetings could be redundant and result in the increased cost of collaboration as mentioned by Elfving (2007). Focus should be upon 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 benefit from collaboration with other departments in these aspects. However, with an increasingly interdisciplinary development, naturally an increased collaboration can be expected, as stated by Elfving (2007), the amount of actors involved effect the collaboration. The increase in collaboration company Y predicts, and company X already has, might be a result of increased interdisciplinary development.

5.4.7 Traceability

All companies speak of some traceability in their requirement handling. Traceability is only covered briefly in this thesis by the requirements ‘good structuring’ and traceability for requirements in the requirement analysis chapter and shortly in the chapter covering modification handling. For future studies on requirement handling, traceability needs to be covered further, also theoretically. Looking to the current integration, RQ 3, company W mentions that the traceability towards original requirement givers is limited due to integrity (Appendix H.9) and company Z mentions how traceability is lost in the requirement breakdown (Appendix K.9). Also company X mentions this loss of traceability, but adds that this is not necessarily a bad thing. At some point, the responsibility of fulfilling requirements needs to be entrusted to the owner of the requirement.

Looking to future integration, RQ 4, as a company grows and complexity of product increases, including more development departments, traceability naturally will need to have a certain basic level for a shared understanding of why requirements need to be fulfilled. In a complex product, several functions work together towards the fulfillment of the same requirement, and at the same time one function work towards several requirements simultaneously. According to Lamsweerde (2009) traceability assists in localizing additional changes needed when modifications take place. What is needed is an overview to provide this traceability. Once traceability is documented in an overview, revisions will be needed in case of
change. If traceability is drawn too far, the revision of traceability documents would become close to impossible to keep up to date, and the developers would spend more time with administrative tasks than actually developing.

Further the verification process is in place to control that requirements are fulfilled, and maybe what is needed as a complement to the perceived ‘lacking’ traceability is an increased verification, or a closer link and increased traceability between the requirement process and the verification. This is mentioned by Hood (2008), as the importance of knowing, already when stating requirements, how they can be tested. Similarly the requirement characteristic ‘measurability’ also supports this conclusion.

5.4.8 Organizational systems and traceability

There seems to be a shared opinion among the companies that they want to increase their understanding of other departments, and traceability of requirements. The current state of integration as should be answered in RQ 3, is that two of the companies make use of requirement handling systems which is good for integration. The companies who are not currently using organizational systems are considering, or at least have considered, getting one (Appendix J.8, K.11, K.18). This is implied by having done trials with organizational systems, and mentioning benefits of having one. The use of requirement handling tools is recommended by Jiang et al. (2005) for complex collections of requirements. When it comes to tools, company X speaks of creating a meta-model to visualize links between requirements and increase traceability. There is definitely already a movement within the companies to improve structure and increase and simplify traceability, corresponding to RQ 4. The by company X suggested visual model, could possibly be generated by the organizational system, if the correct data is entered into this system.

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 data given.

5.5.1 Division and separation

Company W mentioned SW and HW being like two different organizations (Appendix H.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 K.17) does not only point to understanding, but also a ‘us and them’ mentality. This type of mindset will make a change towards more interdisciplinary development difficult, pointing out obstacles, as sought by RQ 5. Relating to difficulties mentioned by Elfving (2007), collaboration requires departments to work together towards a common goal. The mindset in company X, acknowledging that the understanding of other departments and relationships is not complete (Appendix L.17) is much more open to improvement in case of increased integration, answering RQ 6, an enabler to 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 the employees towards other departments in a company, needs to be addressed before increased integration, so that it does not become an unnecessary obstacle, RQ 5.
5.5.2 Problems concerning change in process

Companies W, Y, and Z all have complaints concerning convincing others that change is needed. This is an obstacle in increased integration, RQ 5, as this also requires change. Only in company X is this not addressed, and maybe this is the only company where management support for the requirement handling process is appropriate, possible answer to RQ 6. Change is hard, and convincing others, especially managers, that it 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. System engineering gets little mention in the interviews, but it could be relevant to look into if a system engineer could promote certain changes as they, according to Hagan (2017), work with the balancing of technical - engineering, and human-centered - management among others - disciplines. In this case the technical discipline seems to need some support in convincing the management. There was mention of attempts to change and increase integration, but support was mentioned as the main obstacle in company Z.

5.5.3 Dependency issues

In company Y and W, the issue of having too many depend on one person or department was shared, this overload is an obstacle, RQ 5. Company Z already sees addition of more actors in coordinating roles as a solution to this, an enabler, RQ 6, as the role which has to many depend on them in their case is different types of managers. A system engineer would, according to Hagan (2017), analyze the project in terms of dependencies, and find a ‘critical path’ of correct prioritizations. Regardless of which roles have too much incoming dependency, it would seem reasonable to first decrease dependencies as much as possible. Further the workload should be rebalanced before, if there is still a need, additional actors are introduced.

5.5.4 Time limitations

In company Y it was mentioned that the different development processes had different amounts of dedicated time for requirement handling (Appendix J.7), and at company Z it was felt that too little time and staff exist for the task (Appendix K.7). Both these situations are obstacles to integration, RQ 5. Company Y also had issues with not getting time scheduled for expected increased collaboration (Appendix J.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. Kotonya and Sommerville (1998) mentions how there is a tendency towards rushing through requirement validation, but this might be true for the requirement process as a whole. That 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 Reuse of knowledge

Looking to the future the reuse of knowledge might actually become an obstacle, RQ 5. Naturally time is saved when processes and decisions can be based on prior knowledge and experience. But the prior knowledge and experience also runs the risk of being adapted to the way of working which was valid at the time the knowledge first was collected. Experience is not mentioned as a source of requirements in the theoretical framework of HW or requirement engineering, and it would be reasonable to believe, if not mentioned by a total of at least 5 sources concerning requirement handling, it might be an inappropriate method. If a company wants to increase interdisciplinary development, new data needs to be collected with this in mind. Using sources such as interviews, and questionnaires, actually interacting with external actors as recommended by theory ISO (2011), Hull et al (2005), Lamsweerde...
If this is done, interactions which had never occurred previously might occur. In this aspect, reusing knowledge, with all its benefits, can be a danger to the future of integrated development.

### 5.5.6 Increased traceability

As covered in the prior discussion on traceability, some companies see issues with traceability, but they also see potential solutions to these issues. Companies want increased traceability (Appendix I.18, K.18), and this increased traceability should support more complex, and integrated systems and products, especially in tracing dependencies in modifications as explained by Lamsweerde (2009). The fact that companies are striving towards increasing traceability also means they indirectly are getting prepared for increased integration, RQ 4 and RQ 6. There is a potential danger in company Y who feel that their traceability is already good enough (Appendix I.9), if it turns out this is not true in case of an increase of integration. 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. This so 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 Relations between Dependency, Traceability, and Collaboration

Generally, the companies want to decrease dependency, increase traceability, and have very mixed opinions on collaboration. How do these relate to each other? If dependency is decreased, the need for traceability should also be decreased. This as Lamsweerde (2009) stated that the use of traceability is to modification dependencies. Decreased dependency could result in decreased collaboration as the departments then would work independently, simply relying on timely delivery from other departments. If traceability is increased, increased dependency should not be an issue and collaboration between teams should be easier, so increased traceability is an enabler for increased interdisciplinary development RQ 6. So the combination of decreased dependency and increased traceability makes little sense. Companies should focus on one of the two, and adapt collaboration accordingly. More complex products will require more collaboration, therefore more dependency which will be supported by increased traceability.

### 5.6 Discussion of environment and ethics

At the beginning of the interviews, an assumption was made that since the information that had been sent out included the statement that recording devices would be used, and that interviewees had agreed and booked times, that they also agreed to the recording device. It became very clear that very few, if any, of the interviewees had read the information sent out. At the same time as we wanted to allow the interviewees to say no to recording, we knew the importance of having recordings so that the collected data could be reviewed. In the end, only one company denied us the permission of video recording due to their internal regulations. Permission to record audio was retained at this company.

The anonymity of the companies in this thesis was requested by one of the companies. After this company requested anonymity, it was decided to anonymize all of the companies, regardless of whether it had been requested. As the thesis concerns interdisciplinary requirement handling for complex products, it was difficult to draw the line, what had to be left in to retain understanding, and what had to be left out in respect of anonymity. In some cases it was very easy to say, this is too product specific and needs to be left out, but in most cases it needed to be considered if the terms used by the company in question are general terms of the industry or if the use of the terms also gives the company away. By asking the companies...
5. Discussion

for approval/proof-reading, we hope that these types of misses in anonymization will have been avoided in the thesis.

For the interviews audio recordings and video recordings were made. These were kept for use within the project group during the project. After the end of the project, video recordings were deleted to ensure the anonymity of the interviewees. In the questions, it was specifically asked for both strengths and weaknesses in the processes of the company. When presenting weaknesses, this must be done in a respectful way, and exaggerations be avoided. Even if the data is anonymized, the companies are aware of the participation of their interviewees, and therefore it is important to present the data collected as it was originally stated, so that no conflict occurs at the workplace for the interviewee.

The thesis set out with the hypothesis that companies are moving towards an increased amount of integration, and as it was discovered in some of the interviews that this was indeed true, it was also discovered that not all of the companies provided their employees with additional time needed for the task of increased coordination and integration. Even if it would seem that the end result is improved by increased coordination, it is important to keep the humans psychological health in mind and provide them with separate time dedicated to the task. Otherwise a coordination meant to result in increased productivity and improved result, may completely backfire with employees falling ill or even leaving the company due to unreasonably high stress levels.

Assuming that the thesis is read by companies, and an increased interest is taken in requirement handling and interdisciplinary requirement engineering. The implementation of one common, shared, central requirement organizational system would reduce the amount of different data storage needed, both digitally and in paper. Further if work is done to improve the collaboration and integration of different departments, developers will work with the correct requirements earlier and less redundant work will be done. This is less wasteful in many different ways, environmentally being one of them. The impact of developers traveling between different geographical locations might also be possible to decrease if coordination is done right. However and increased collaboration might require more travel, which effects the environment negatively.
6 | Conclusions

The aim of the thesis is reached with regards to the answers to the RQs. Requirement handling processes are covered through answering RQs concerning differences and similarities. The current integration state is covered, and opinions of future integration are covered briefly. For future increased integration mostly obstacles were lifted, leaving the RQ concerning enablers for future increased integration poorly answered.

RQ 1 What differences in HW and SW engineering need to be considered during interdisciplinary requirements handling?
Requirement elicitation is performed differently in different development fields. Further different development processes have individual perspectives on time, where software has a short time-cycle, and hardware has a longer, which are difficult to unite. An interdisciplinary development will require parallel time-frames. Internally in a company departments have different prioritizations. For further integration, coordinating actors of priorities are needed. Priorities should be set before documentation, and revised through an official modification methodology.

RQ 2 What do the companies have in common which might support interdisciplinary requirements handling?
Both HW and SW development have established generalized processes. These processes contain points of decision such as ‘gates’ or ‘handshakes’. This is something SW developers already want more of in their processes. SW and HW processes already share requirement break-down, and the use of internal documentation standards, so it is not impossible for HW and SW processes to have things in common. Companies practice formal modification methodology late in the process for changes which affect several departments. They also share mindset when it comes to late additions of requirements, namely that they should only be implemented if absolutely necessary. Traceability, that is a big topic that was uncovered in the thesis, is coupled with overviews which exist in some format in all of the companies. The companies, however, need to accept that traceability is only possible to a individual level because after this traceability becomes administratively cumbersome. Personal responsibility needs to be taken onwards from this point for requirements being fulfilled. Issues exist in the understanding of what happens in other departments, but increased traceability of requirements, which is a solution proposed by the companies, might not be the actual solution to this. Increasing traceability between verification and requirements could also solve the issue of lacking interdisciplinary understanding.

RQ 3 What is the current state of integration in companies which develop complex systems?
There is an insight that a product includes more today than the one previously delivered which most commonly was hardware focused. Additions comes in the shape of added services or additional SW. The companies have sub-processes where department deliver requirements to either other departments or a coordinating actor. If the company applies standard processes, the process is altered to fit the project or department.
Most who work with requirements understand that balancing, prioritization, and compromise of requirements are necessary. Some want to decrease dependency between departments and requirements, and some companies believe that dependency is necessary. From the observations made, dependency could be decreased to a certain extent, but key dependencies must remain to produce an optimal complex product.

**RQ 4 Are companies that work with complex systems striving towards a more interdisciplinary development environment?**

The acceptance that exists, that a product entitles more than it used to, might be a correct first step towards other interdisciplinary improvements. The companies are mentally ready for an environment with more actors, more requirements, and more integration. If a standard process exists in a company, it must be applicable for all departments to support increased interdisciplinary development. There are very different opinions concerning interdisciplinary development. Some of the companies studied believe that processes need to remain in sub-system divisions due to the complexity being too high. Other companies say that everything needs to contribute to the final product, resulting in increased collaborations in the future. Collaboration needs to be balanced so that different department specializations still are exploited. Companies see the issue of not understanding interactions, and want to create visual models of requirements and their relations, which would help in complex, interdisciplinary, projects.

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

There seems to be a belief that the two disciplines are too different to be able to benefit from each others’ processes. The mindset of the employees needs to be that ‘we are one company, producing one product, together’ before increased integration, so that a ‘we and them’ mindset does not become an additional obstacle to integration. Change is hard, and to convince others, specifically management, of its benefits is difficult. For example, time is not dedicated to the requirement process. Management does not fully appreciate how much time is required for a good, thorough requirement process. There is also a need to decrease dependencies, and re-balance workloads, to fix existing unbalance between actors. Reusing old requirements, or basing new requirements purely on experience, can be an obstacle for future integration if integration has not previously been a part of the product.

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

Some companies are looking to increasing their amount of coordinating actors, or giving existing such actors more power. This will support simultaneous HW and SW engineering. Companies are striving towards an increased traceability in their requirements, especially between different internal levels and departments. This type of traceability would support more interdisciplinary projects.

From all conclusions mentioned above, requirement handling methodology needs to be more unified before it is spread as a common methodology in an interdisciplinary company. It is also possible that this unification of methodology can take place as a part of the integration process.
7 Future work and Recommendations

Performing additional interviews, combined with a modified interview questionnaire would allow for a further understanding of different departments’ requirement specifications and allow for a more detailed comparison of these within the company. A modified questionnaire could also shift focus towards the tools used in requirement handling, which play a big part in the success or failure of said process. Additional interviews with SW companies is needed to determine if the SW development process more adapted to modifications. If how the management effects the requirement handling positively is interesting in future studies, company X should be further questioned as this was the only company where no issues were linked directly to the management.

Looking towards more extensive projects which could be performed based on the results of this thesis, requirement types need more looking into to enable a comparison between HW and SW. Next to requirement types, further study would also be needed to be able to compare actors and their roles between different companies. A topic which came up and was not very well covered but a big issue for the companies was traceability. For this an increased understanding is needed, but also a look into proposed solutions, for example the relationship model suggested in the results of this thesis.


Appendix A. Interview invitation, original interview guide

Linkoping, 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

Note on changes: 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 RQs, 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. 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?

   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?

   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

VI
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?
(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. Discover if process is continuous.</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>Identity 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>R-18</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-1</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-2</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-3</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-4</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-5</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Reflection upon other’s process</td>
<td>Positive from other, possible negatives from own area</td>
</tr>
<tr>
<td>I-6</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Allow for interviewee to add missing information on difficulties</td>
<td>Difficulties and the handling of them</td>
</tr>
<tr>
<td>I-7</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-8</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

<table>
<thead>
<tr>
<th>Differences and similarities</th>
<th>“our” process</th>
<th>commonalities</th>
<th>comparison</th>
<th>constraints</th>
<th>relations</th>
<th>inspiration from limitations</th>
<th>special features</th>
<th>unique principles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research questions</td>
<td>collab</td>
<td>cooperation</td>
<td>dependency</td>
<td>difficulties</td>
<td>diversity</td>
<td>integration ideas</td>
<td>interdisciplinary modification methodology</td>
<td>interdisciplinary projects</td>
</tr>
<tr>
<td>Integration state</td>
<td>directives (from management)</td>
<td>driving force</td>
<td>forecasting methods</td>
<td>involved actors</td>
<td>key factors</td>
<td>previous missteps</td>
<td>trend</td>
<td>validity time</td>
</tr>
<tr>
<td>Future obstacles and possibilities</td>
<td>House of Quality (HoQ)</td>
<td>prioritization</td>
<td>proceeding criteria (steps)</td>
<td>Quality function deployment (Qfd)</td>
<td>relationship between requirements</td>
<td>requirement of requirements (RoR)</td>
<td>requirements specification (criteria)</td>
<td>selection criteria (model)</td>
</tr>
<tr>
<td>Hardware development</td>
<td>domain</td>
<td>functional</td>
<td>service (PSS)</td>
<td>steps (process)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software development</td>
<td>approach</td>
<td>components</td>
<td>element</td>
<td>needs</td>
<td>phases</td>
<td>product development life cycle</td>
<td>user</td>
<td></td>
</tr>
<tr>
<td>System engineering</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guide topic</td>
<td>Requirement engineering</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>actors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>elicitation methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(interviews, questionnaires, documentation analysis)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>iterative</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>requirement characteristics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>requirement documentation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>requirement validation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Requirement elicitation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>actors</td>
</tr>
<tr>
<td></td>
<td>elicitation technique</td>
</tr>
<tr>
<td></td>
<td>execution</td>
</tr>
<tr>
<td></td>
<td>norm</td>
</tr>
<tr>
<td></td>
<td>standards</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Requirement categorization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>categorization</td>
</tr>
<tr>
<td></td>
<td>external interface</td>
</tr>
<tr>
<td></td>
<td>Non-functional</td>
</tr>
<tr>
<td></td>
<td>Other</td>
</tr>
<tr>
<td></td>
<td>system features</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Requirement documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>department specific</td>
</tr>
<tr>
<td></td>
<td>documentation methodology</td>
</tr>
<tr>
<td></td>
<td>documentation standard</td>
</tr>
<tr>
<td></td>
<td>unified</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Requirement verification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>consistency</td>
</tr>
<tr>
<td></td>
<td>continuous verification</td>
</tr>
<tr>
<td></td>
<td>validation</td>
</tr>
<tr>
<td></td>
<td>verification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Requirement modification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>modification feasibility</td>
</tr>
<tr>
<td></td>
<td>modification feature</td>
</tr>
<tr>
<td></td>
<td>modification methodology</td>
</tr>
<tr>
<td></td>
<td>modification misspecification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Interdisciplinary development handling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hw dev</td>
</tr>
<tr>
<td></td>
<td>interdisciplinary actors</td>
</tr>
<tr>
<td></td>
<td>structure</td>
</tr>
<tr>
<td></td>
<td>sw dev</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>dependency</td>
</tr>
<tr>
<td></td>
<td>dependency issues</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>Modification handling in interdisciplinary teams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>communication</td>
</tr>
<tr>
<td></td>
<td>compromise</td>
</tr>
<tr>
<td></td>
<td>interdisciplinary modification methodology</td>
</tr>
<tr>
<td></td>
<td>time frame</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>System reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>comparison</td>
</tr>
<tr>
<td></td>
<td>obstacles reliability</td>
</tr>
<tr>
<td></td>
<td>reconnection</td>
</tr>
<tr>
<td></td>
<td>reliability techniques</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>difficulties</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>difficulties</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Guide topic</th>
<th>future expectations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>enablers future</td>
</tr>
<tr>
<td></td>
<td>long term</td>
</tr>
<tr>
<td></td>
<td>obstacles future</td>
</tr>
<tr>
<td></td>
<td>responsibility</td>
</tr>
<tr>
<td></td>
<td>short term</td>
</tr>
</tbody>
</table>
Appendix F. Codeword after coding

Codes from RQs:
- 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
Appendix G.  Detailed description of House of Quality

In the theoretical framework a brief description of HoQ (House of Quality) is given, in this appendix the more extensive version can be found.

G.1  HoQ 1: Identifying customers/actors

Ullman (2002) states, that in most design situations there is more than one customer. Ullman’s (2002) definition of customer is similar to what is referred to as ‘actor’ in this thesis. An actor can also be known as a stakeholder - which is any person (real or legal) which has interest in, or can be affected by, the project. Hauser and Clausing (1988) agree that a ‘customers’ in QFD includes more than the end-consumer. Ullman (2002) however states that the consumer is usually considered the most important of these customers or actors, unless the product is very complex, such as a space rocket or an oil rig.

G.2  HoQ 2: Identifying customer/actor attributes

The customer needs are referred to customer attributes by Hauser and Clausing (1988). Hauser and Clausing (1988) claim that a typical application will have 30–100 customer attributes (Note: 1988). Ullman (2002) mentions the following steps for collecting information: First the information needed is specified. After this type of data collection method is determined. In the case of a questionnaire or interview questions need to be designed and then ordered. Once questions are designed, data is collected and then reduced. Ullman (2002) defines different types of customers attributes which can be collected as data, these are listed below:

- Functional requirements
- Human factors
- Physical requirements
- Reliability
- Life-cycle concerns
- Resource concerns
- Manufacturing requirements

Ullman (2002) further shows a tendency in the described process towards sorting requirements according to which customer/stakeholder who is making it, relating to the requirement traceability mentioned by Johannesson et al. (2013). This could be one possible way to create the ‘bundles’ which are described by Hauser and Clausing (1988) and can be seen to the left in the HoQ in Figure G.1.
Figure G.1: Detailed picture of a HoQ matrix

Source: Inspired by Hauser and Clausing (1988)
G.3 HoQ 3: Determining the relative importance

Ullman (2002) lifts the questions "To whom is the requirement important?" and "How is a measure of importance developed for this diverse group of requirements?". To answer these questions Ullman (2002) recommends a weighting to be done by asking customers to divide 100 points among requirements counteracting the option of rating each requirement on a scale from 1 to 10, which results in many seemingly evenly weighted requirements. Hauser and Clausing (1988) speaks of several alternative methods of weighting the different requirements, but always include the customer in the process. After weighting has been done, the different customer needs' relative importance has been determined, and is inserted into its column in the HoQ, see HoQ 3 in Figure G.1.

G.4 HoQ 4: Looking to the competition

This step is important according to Ullman (2002), in order to identify the current level of satisfaction of the customer. Existing designs are rated on a 5-point scale revealing possible improvements. This 'benchmarking', according to Ullman (2002), gives an insight into the customers' perception of the competition as well. Hauser and Clausing (1988) speak of the need to exceed the competition, and how this is only possible if the competition is known. In the HoQ model by Hauser and Clausing (1988) objective measures are added in the 'basement' of the HoQ. See both the customer perception and the objective measures which are marked with 'HoQ 4' in Figure G.1.

G.5 HoQ 5: Generation of engineering specifications

The next step in the HoQ is to translate the so far collected data into engineering specifications. The engineering characteristics are placed above the relationship area (HoQ 6) in the HoQ seen in Figure G.1. To qualify, Ullman (2002) states that measurability is mandatory for all engineering parameters. It is important to clarify, as mentioned earlier in the section concerning requirement characteristics, how each requirement will be measured. Measurement units are together with the objective measurements in the basement of the HoQ in Figure G.1. Hauser and Clausing (1988) state that if there is no relation to a customer need, an engineering requirement might be redundant. Customer needs which are not yet represented in engineering requirements can, on the other hand, point to opportunities.

G.6 HoQ 6: Relationship between customer requirements and engineering requirements

It is important to identify which engineering requirements answer to which customer needs and naturally there will be overlaps where engineering parameters measure several customer requirements. Ullman (2002) and Hauser and Clausing (1988) mention how the relationships are sometimes scientifically proven and at other times only the intuition of the developers. According to both Ullman (2002) and Hauser and Clausing (1988) it is important in this step both to make clear how strong the connection is between the different types of requirements, and also how the relationship was established. HoQ 6 can be found in the middle of the house in Figure G.1 and in this area, relations between engineering characteristics and customer needs are indicated with check marks or crosses of different intensity.

G.7 HoQ 7: Setting engineering targets

Ullman (2002) states that it is important to set targets early on in the project, since targets set close to the end of the process are adapted to project circumstances and therefore easy to meet, and meeting these has little or no meaning. At the same time Ullman (2002) puts emphasis on how important setting the right target is, and how setting this target can be difficult early on in the process. One should aim to offer...
a better design than the competition, but still not differentiate unnecessarily. Hauser and Clausing (1988) claim that if one does a thorough job when correlating engineering requirements to customer attributes, establishing target values should be easy. Having too high targets, according to Ullman (2002), might imply that the specification is not market accurate. To include the target customers, Ullman (2002) recommends using delighted and disgusted targets. At what fulfillment of the requirement will customers be disgusted, alternatively delighted?

G.8 HoQ 8: Identifying relationships between engineering requirements

Dependency can, and most often will, exist between different engineering requirements (Ullman 2002). Thus, work to meet one specification might have a positive or negative effect on the achievement of others. According to Hauser and Clausing (1988), there can be features with so many relations that they are left alone rather than changed due to the resulting, possibly negative, influences on others features. The point of this room in the HoQ is to create an awareness of the relationships so they can be handled in an analytical manner (Ullman 2002). In the detailed Figure G.1 HoQ 8 can be found at the top, as a roof of the house. In here the same markers for relationships are used as in HoQ 6.
Appendix H. 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.
H.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

H.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 make 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 can not take place several times.

H.2.1 Time frame

No interview data responding to this codeword.
H.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.

H.3.1  Iterative and agile methods

No interview data responding to this codeword.

H.4  Understanding

No interview data responding to this codeword.

H.5  Elicitation methods

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

H.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.

H.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

H.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.
H.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.

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
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.

H.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.

H.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.

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

H.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.
H.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.

H.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.

H.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.

H.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.

H.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.

H.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.

H.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.
H.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.

H.11.2 Continuous verification

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

H.11.3 Relation between requirements and the product

No interview data responding to this codeword.

H.12 Validation techniques

No interview data responding to this codeword.

H.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.

- 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.

H.14 Interdisciplinary projects & communication

It is hard/untrivial to see the connectivity between hardware and software, the impact/influence they have on eachother. 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".
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
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.

H.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.

H.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.

H.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.

H.17.1 Inter-cultural work
Difficulties due to work across boarders include language barriers, time differences, and cultural differences.
H.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).

H.18.1 Obstacles future

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

H.18.2 Enablers future

Better communication and interfaces between departments are important.

H.19 In a perfect world

No interview data responding to this codeword.
Appendix I.  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.

I.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

I.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.

I.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 therefor 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.
I.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.

I.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.

I.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.

I.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.

I.5.1 Customer interaction

No interview data responding to this codeword.

I.5.2 Strengths & weaknesses

1. Strengths
   The elicitation of requirements is done in structured way.

2. Weaknesses
I.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.

I.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.

I.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.

I.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.

I.7 Limitations & obstacles

No interview data responding to this codeword.

I.8 Requirement documentation and characteristics

The main theme which they follow while writing about requirements is how requirements are described in the 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. 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.

I.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.

XXXIV
I.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.

I.8.3 Requirement types

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

I.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.

I.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.

I.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 a easy task. At this point however, someone has the responsibility of fulfilling the requirement, so a more detailed traceability is not needed.

I.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.

I.10.1 Interdisciplinary modification methodology

I.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.

I.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.

I.11.2 Continuous verification

No interview data responding to this codeword.

I.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.

I.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)

I.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.
I.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.

I.14.1  Interactions

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

I.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.

I.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.

I.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.

I.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.

I.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.

I.17.1  Inter-cultural work

No interview data responding to this codeword.
I.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 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.

I.18.1  Obstacles future

I.18.2  Enablers future

Interaction and collaboration between departments,

I.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 J. 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.
J.1 Actors

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

J.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.

J.2.1 Time frame

No interview data responding to this codeword.

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

J.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.

J.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.

J.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.

J.6 Requirement analysis
In service, conflicting requirements are avoided through a review process.

J.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.

J.6.2 Balancing requirements
No interview data responding to this codeword.

J.6.3 Setting targets
No interview data responding to this codeword.

J.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.

J.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.
J.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.

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

J.8.3 Requirement types
- Technical requirements
- Market related requirements (world/region)

J.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.

J.8.5 Impact of legal factors/legislation
No interview data responding to this codeword.

J.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.

J.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.

J.10.1 Interdisciplinary modification methodology
No interview data responding to this codeword.

J.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.

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

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

J.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.

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

J.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.

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

J.14.2 Relations
No interview data responding to this codeword.

J.14.3 Interdisciplinary actors
No interview data responding to this codeword.

J.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.

J.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.

J.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.
J.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.

J.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.

J.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.

J.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.

J.19 In a perfect world

No interview data responding to this codeword.
Appendix K.  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.
K.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.

K.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.
K.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.

K.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.

K.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 PRS as well. The approach is however far from purely agile, and has more characteristics of a waterfall model.

K.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.

K.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.

K.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.

K.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.

K.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.

K.6.2 Balancing requirements

No interview data responding to this codeword.

K.6.3 Setting targets

Once the group leader understands the requirement they create a test tolerance, or a completion criteria.

K.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.

K.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 an 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.

K.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.

K.8.2 Relationship between requirements

Which concepts are critical to the functionality of other concepts?

K.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

K.8.4 Impact of standards
Safety norms affect related functions. New standards emerge effecting more components. (still high focus on safety)

K.8.5 Impact of legal factors/legislation
There is a main legislation which covers all products, but then certain product have a special legislation.

K.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.

K.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.

K.10.1 Interdisciplinary modification methodology
No interview data responding to this codeword.
K.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.

K.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.

K.11.2 Continuous verification

No interview data responding to this codeword.

K.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.
K.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.

K.13 Roles & responsibilities

Roles:
- Project manager
- Supervisor
- Product owner
- Machine responsible
- Group managers/leaders
- Technical leader
- Sales manager
- System engineer

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.

K.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.

K.14.1 Interactions

No interview data responding to this codeword.
K.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.

K.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.

K.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.

K.17.1 Intercultural work
Different countries/regions have different norms. This is information which is needed when specifying requirements.

K.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.

K.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.

K.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.

K.19 In a perfect world
No interview data responding to this codeword.
Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extraordinära 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/