INTERNATIONAL DESIGN CONFERENCE - DESIGN 2018
https://doi.org/10.21278/idc.2018.0466

EMPIRICAL STUDY OF REQUIREMENTS ENGINEERING IN CROSS DOMAIN DEVELOPMENT

S. Nilsson, L. Buffoni, K. Sandahl, H. Johansson and B. Tahir Sheikh

Abstract

Shortened time-to-market cycles and increasingly complex systems are just some of the challenges faced by industry. The requirement engineering process needs to adapt to these challenges in order to guarantee that the end product fulfils the customer expectations as well as the necessary safety norms. The goal of this paper is to investigate the way engineers work in practice with the requirement engeneering processes at different stages of the development, with a particular focus on the use of requirements in cross domain development and to compare this to the existing theory in the domain.

Keywords: systems engineering (SE), complex systems, requirements management

1. Introduction

Several challenges exist for companies in today's industry e.g. to keep up with rapid technological changes (Chang et al., 2013), or time to market that has been an issue for several years as stated by e.g. Smith and Reinertsen (1998) even before the 21th century and is still no less important. Another major challenge for companies is to identify both quality and cost effective solutions with an appropriate balance (Kossiakoff et al., 2011). One consequence of these challenges is that companies do not longer offer their customers products but rather complete solutions. These solutions are commonly integrated systems consisting of a combination of different sub-systems and/or individual elements e.g. software, services or products where the elements have been designed to be integrated and optimized from a life cycle perspective in relation to customer value (Meier et al., 2010). Instead of only selling individual products these solutions provide functions, service and performance (Sundin et al., 2006).

Designing these types of solutions creates a complexity with requirements in regard to several different aspects and actors that need to be considered (Vasantha et al., 2012). The integration and collaboration with relevant actors is, according to e.g. Vasantha et al. (2012) and Mont (2002), seen as an important aspect of creating a successful offering. On the same note, understanding the underlying needs throughout the life cycle is key for successful systems engineering (Sage and Rouse, 1999). Therefore, it is essential to have a suitable requirements process that can capture all relevant actors' needs and related requirements. A challenge here is the process of integrating different domains perspective into a common view. In order to ensure consistency and traceability in elicitation and management of requirements a continual interaction between requirement engineering and the elements to be developed is inevitable (Hull et al., 2011).

Building on the described background, an objective and research questions have been developed to set the frame for this paper. The main objective is *to explore how companies internally work with requirements that go across engineering-specific domains and departments from the perspective of engineers and their managers*. The context is four large companies in multi-domain engineering and the

study will provide insights regarding how requirements that stretch across engineering domains are worked with currently in the industry.

2. Theoretical framework

Requirements engineering is defined in ISO/IEC/IEEE 24765 (2010) as *"the science and discipline concerned with analyzing and documenting requirements".* Wiesner et al. (2015) suggest that requirements engineering can be divided into two parts: *requirements development* and *requirements management.* Requirements development includes activities to set the foundation for what functions the offering that is about to be designed is supposed to fulfil, e.g. elicitation and documentation. Requirements management, on the other hand, implies activities to maintain the requirements during the other design phases of the offering, e.g. change management and verification of the requirements. Figure 1 illustrate that requirements development is a step in the product design process, while requirements management is an on-going activity that continues during the sequential phases in the product design process. Sommerville and Sawyer (1997) argue that there are many different requirements processes, and that these processes do not transfer well between organizations. According to Sutcliffe (2002, p. 45) *"there is no one 'cook-book' method for requirements".*

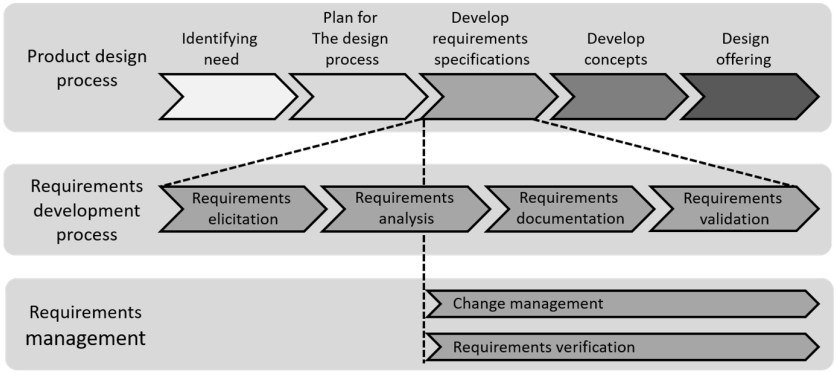

Figure 1. Requirements engineering activities in relation to the product design process, based on Wiesner et al. (2015), Kossiakoff et al. (2011), Kotonya and Sommerville (1998), and Ullman (2002)

Requirements elicitation is an early activity in the requirements development process, which is mainly about acquisition of knowledge. The first step for effective knowledge acquisition is to identify the correct actors (Lamsweerde, 2009). Sutcliffe (2002) highlights the difficulties in the identification of requirements, as it requires an understanding of people and good communication. Requirements engineering is more than understanding the user, understanding the limitations and implications of the domain is also important (Sutcliffe, 2002).

Requirements analysis should be performed as soon as initial requirements have been collected (Sommerville and Sawyer, 1997). Analysis should look for conflicts, overlaps, omissions, and inconsistencies (Sommerville and Sawyer, 1997). The decision of which feasible requirements will be included in the specification is also an important part of the requirements analysis (Hood, 2008).

If it is concluded that not all requirements will be able to be met, there needs to be criteria evaluating which requirements that should be prioritised (Hood, 2008). A complex system will have many actors, and these actors will at some point have disagreements about the system requirements, and will prioritise the requirements differently depending on their background (Kotonya and Sommerville, 1998). This implies that requirements negotiation always will be necessary. Requirement conflicts and overlaps are resolved through the interchange of information, discussions, and resolution of disagreements (Kotonya and Sommerville, 1998).

Requirements documentation is desired as the number of requirements can easily reach to a high number, Jiang et al. (2005) highlight the need for tools handling requirements as they are documented. Jiang et al. (2005) claim that a smaller project might make do with a simple word processor, but makes suggestions for different software solutions for more complex collections of requirements. 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.

Requirements validation is the final activity of requirements development (Kotonya and Sommerville, 1998). The purpose of validation is to control the requirements, certifying that the requirements represent the, by actors, sought after system (Kotonya and Sommerville, 1998; Cheng and Atlee, 2007; Stjepandić et al., 2015). Validation aims to ensure that you develop the right thing (Chemuturi and Gilb, 2013). Kotonya and Sommerville (1998) state that validation involves several different actors, including, for example, requirement engineers, and system designers. Cheng and Atlee (2007) and Stjepandić et al. (2015) similarly state that validation requires direct involvement of actors for reviewing the requirement.

Change management, concerns the need for change during, and after the requirements development process (Lamsweerde, 2009). Hull et al. (2011) points out how difficult it is to maintain the originally agreed upon requirements, and the need for having change management processes in place. If the process is well structured and designed for modifiability, changes can usually be made locally (Lamsweerde, 2009). Traceability contributes to the ease of localizing which changes need to be made (Lamsweerde, 2009).

Requirements verification is an important step and already when the requirements are defined, knowing how a product should be tested to control that it fulfils each defined requirement (Hood, 2008). A system needs to be tested for its fulfilment of requirements in different ways depending on the characteristics of the system (Hull et al., 2011). Documentation can help linking requirements to tests and test results, allowing for a correlation between test results and requirement fulfilment, but Hood (2008) also warns of the questionable validity of this type of linkage when changes occur in the requirements.

3. Research methodology

This research is based on an interview study and the authors' theoretical knowledge from earlier studies, for example, Nilsson and Lindahl (2016), Wiesner et al. (2017), and Nilsson (2017).

Four companies from different industrial domains were interviewed, in total eleven interviewees. The interviewees at the companies had roles, or experience, covering mechanical, electronic, software, and service engineering. Some companies wished to remain anonymous and therefore all companies have been left unnamed in this paper. For one of the companies (Company A), software is their primary product, whereas for the other three the hardware is the focus (Company B, C, and D), but all four companies are involved both in hardware and software development. The companies are large to moderate sized and develop complex products involving different expertise in engineering domains. All of them have Sweden as their country of origin but sell to markets all over the world. These specific companies were chosen as they show an interest in improving how they develop requirements that stretch across several engineering domains. As mentioned earlier the interviewees was chosen to represent a wide set of experiences with roles covering generalists to specialists, who show an interest in the subject and ought to have reflected upon it in their day to day work, which would give sufficient and interesting insights for us as researchers.

The interviews were performed to get insight in how companies in different domains work with requirements in practice and compare this to the theoretical research in the area. Based on the literature several topics of interest were identified and used for designing the interview and categorizing the findings. The interviews were performed and recorded by two of the authors. The analysis was thereafter performed by one of the authors that had not been a part of the interviews. As a third step the two remaining authors draw conclusions based on the findings. This has prevented biased results by letting several individual partners contribute with distinctive parts (Bell and Nilsson, 2006). More details about the execution of the interviews (e.g. interview questions) can be read in Johansson (2017). Results from the study have been presented and discussed with the included companies to verify the findings.

Although this selection of companies cannot be seen as representative for the industry as a whole, several similarities in the practices were highlighted during the interviews and can be used as representative examples. Some of the quotes and examples has been slightly altered to preserve the anonymity of the companies and to correct minor grammar issues, without changing the intention of the statements.

4. Results

In this section, the compilation of the data from the interviews is presented. The more domain specific terminology has been generalized to protect the anonymity of the companies.

4.1. Workflow

All the companies interviewed have mentioned some agility in their work process, with the software-oriented company being the most agile. More detailed representations of the requirement handling process for each company can be found in Johansson (2017).

Although there is an overall interest in increasing agility in the workflow, this is also seen as coming at the expense of in depth knowledge, and hard to achieve because people tend to have different competences and experience with different tools.

The length of the development cycle is mentioned as a challenge by all companies, with Company A mentioning 7 years delay between the research and product development stages. For the development cycle itself, time-frames from 5 years to continuous delivery are mentioned, with the software-oriented company being on the short end of the spectrum and the larger hardware-oriented companies having development cycles of 2 to 5 years.

There is trade-off between more formal processes with guaranteed quality and more flexible and efficient processes. Depending on the area and the safety norms, this balance varies across companies and across departments of a single company.

4.2. Requirement elicitation, categorization

There are a lot of common traits in the elicitation process across the companies interviewed. For all companies, the actors are known from the beginning. First the top-level requirements are collected by the product managers through a combination of the following sources:

- sales and after sales departments
- senior management that lays out the large-scale goals and road maps to follow
- a selected group of customers, customer visits
- focus or analytic groups that study market trends, buzz words that are often vague concepts identified by use-cases (e.g Internet of Things)
- previous products or projects, experience

At this top-level some companies consider these requirements external, as influenced by customers or external business opportunities, whereas others consider them internal because they collect the information about the state of the market and filter this out into internal requirements set by the company. Top-level requirements are often very vague e.g. *"better performance"*, *"support a thousand more users"* or very long-term perspectives and are then broken down into more formal specifications (called the Functional Requirements Specification or System Requirements Specification).

All companies mention product managers as being key people in the requirement elicitation process as they are in charge requirement prioritisation.

Categorization at system or functional specification level is different for different companies, with some overlap, but the following types of categories were outlined:

- Component or part of system specific requirements, each set of such requirements is assigned to a specific team or department and has a single responsible
- Trust mark or quality level requirements which are system wide requirements meant to enforce a level of quality across all company products
- Cross function requirements, involving the whole system or behavioural requirements that specify how a system should behave but not how it should be implemented
- User experience requirements, which are high-level requirements defining user expectations of the product, using language such as comfortable, responsive, faster (than previous product) etc. and can be hard to transform into a quantifiable specification. These requirements are:
- Performance requirements (number of faults allowed, lifespan, etc.)

Once the requirements are in the specification they are treated in the same way at functional level no matter which stakeholder they originated from. The distance between the customer and the engineers developing the product is cited as a major difficulty in interpreting requirements and meeting customer expectations and companies are making efforts to fill this gap by occasionally sending engineers to attend some customer meetings but this is not a standard practice at Company A and B.
Requirements can be e.g. specific to the region as well as to the product or family of products, and another difficulty cited is the definition of requirements for foreign markets as the customer needs are more difficult to predict.

4.3. Analysis

4.3.1. Formulation

As mentioned in the previous section, most requirements at top-level start out as informal or vague statements and need to be quantified and specified further. This is often an iterative process at Company A, C, and D in which first the feasibility of a requirement is analysed, then the requirement is aligned with the capabilities and a subset of the original actors is consulted again to clarify the specification or asked to approve the formalized requirements.
For hardware-oriented companies, previous product specifications are an important source at According to Company B and C, which allows quantifying requirements such as "faster" in terms of performance of past products. For new features, high-level requirements are used to determine which pre-studies should be done at Company D.
Company C mentioned moving towards Stated function disposition for formalisation of the system model as a way of better connecting cross domain requirements, but all companies strive for some form of formulation. Interviewees use terms such as quantifiable, "not fuzzy", verifiable, measurable, testable to describe the final product requirements.
It is important to mention that even at companies with detailed requirement elicitation processes, full coverage is still not achieved. Company A mentions that there is still a set of "unspoken" requirements that although never formalized are known to the people working on a particular sub-system based on their previous experience. An example of such requirements are cross-domain physical requirements where the underlying complex dependencies make the formalization process complicated beyond knowing that *"we need to place this component at this angle otherwise it does not work"*.

4.3.2. Decomposition

Companies A, C and D detect requirement conflicts during requirement decomposition by working in cross-functional teams. A common example of such conflict mentioned by several companies is performance versus cost. Another source of conflict is reconciling new features with existing software and hardware. These conflicts are then discussed and negotiated. At Company B no specific process is in place to identify conflicts, but if one is detected it goes through the negotiation process.
The decomposition generates several product specification documents covering different areas of the product. At this level, each document is assigned to a single team or responsible person (product manager), which should reduce the likelihood of local conflicts. Company B mention further decomposition of requirements but this is not formalized into documents and is done internally to clarify the requirements for the developers.
Parallelization of work on different specifications is cited as one of the sources of conflicts because there is no good idea of how this will affect other parts of the system. This makes it hard to set requirements as *"requirements that are too strict can take too much time to fulfil but too lax requirements can result in problems when putting the whole system together (e.g. on time use)"*.
Company B talks about having a *"concept CUG Gate, a tollgate with 2-4 different concepts filled into a concept paper trying to understand the requirements, a sheet to show the management issues with different types of requirements (e.g.: shorter lead time vs a specific function)"*. What this means is that they develop several variants of the system that will be prototyped to decide on the variant, which will be put into further development.

4.3.3. Prioritization

Prioritisation of requirements is done by managers but can be adjusted locally when it gets to individual teams. Company A mentions that prioritization uses up a lot of resources and Company D says incorrect prioritisation is the source of a lot of issues later in the project. More generally prioritisation is linked to the size of the company and the project. Larger companies or higher budget projects invest more resources in prioritisation, while in smaller teams and short-term projects prioritisation issues are solved through direct communication.

4.4. Norms and standards

All companies use internal standards to some degree for formulating requirement specifications, however no external standards were mentioned. When it comes to norms and standards for the products themselves, the hardware-oriented companies (Companies B, C and D) are in the process of adopting more safety related standards, which according to Company B should increase the traceability and documentation of some of the requirements. Company A points out that although standards are well defined and common place in their domain, some of the guidelines are quite open to interpretation and for others they set internal stricter requirements to be on par with the competitors.

Responsibility for legal norms and standards is often placed on a dedicated person or department within the company. Given the duration of the development cycles it is important to take into consideration not only current legislation but also legislation changes planned for the next 2-5 years.

4.5. Documentation and tool support

None of the companies use open standards for requirement specification and handling, two of the four Companies A and C use an internal relational database global to the company. Examples of such tools are Enovia and Teamcenter, a PLM tool to manage several artefacts and processes of a cross-domain development. It can be regarded as a platform where users plug-in specialized software or import data on different formats (Siemens, 2017). Company D uses a textual database. Company B has no common database, but rather uses Excel to store requirements and the product requirement specification *"is not standardised across departments, each department e.g. market product planning, has a different format, but there is an agreement across departments on how each department operates so that departments can communicate and documentation is shared"*.

The degree of documentation varies across companies and inside the companies themselves, Company D has a process in place for managing the requirements, but due to time and resource limitations it is not always used in smaller projects.

Company D mentions the House of Quality and CTQ (Akao, 1990; Chan and Wu, 2002), to define and prioritise requirements. For team-level management, Hansoft a tool for managing agile teams with backlog handling and project monitoring (Hansoft, 2017) is mentioned by one of the companies and SCRUM and Kanban mentioned by 2 companies. Two of the companies mention that DOORS, a scalable tool for managing requirements and their traceability to implementation and test (IBM, 2017), has been considered for requirement handling and deemed too cumbersome and requiring dedicated resources to handle this.

Company D wishes for a better system for requirement handling and prioritisation as they have found themselves in several times in situation where they wished *"they had put some more work in earlier"*. They also mention that there are no tools in place for international collaboration only the *"expectation of collaboration"*.

4.6. Traceability of requirements

Traceability of requirement from the high-level general requirements through the domain specific requirements to the final implementation and testing is a priority at all the four companies interviewed although it is implemented to varying degrees across companies and even across departments in Company B.

It is important to distinguish between traceability and transparency across requirements. The traceability information is there but can only be accessed on a need to be basis. Company A points out that for

example that they have traceability from stakeholders to lower-level requirements, however this information in not visible at lower-levels for confidentiality purposes, and conversely exact specifics of hardware are not known higher up in development. Access to other departments' specifications is also on a need to be basis in all companies. This is also valid for cross department requirements, they can be traced but access to them is on a need to be basis for all companies.
In the Companies A and C, the requirement identification codes allow to trace their origins, in Company B there is no ambition to trace requirements right now at company level, but the interviewees recognize it is necessary and will become even more so with upcoming security standards. They also point out that without a structured breakdown process-assigning ownership of requirements is a problem.
All companies agree that the traceability process can be improved. In particular it is quite easy to trace the requirements from bottom to the top. However, at top-level it is hard to know into which requirements a top-level requirement is decomposed into across the systems and therefore hard to detect all the places, which will be affected when changing a top-level requirement.

4.7. Interdisciplinary requirements and collaboration

All companies agree that *"communication is key for successful cooperation and the people factor is very important"* (Company A). This is in particular relevant to requirement elicitation as understanding and interpreting the requirements correctly at the different levels of the product development process is not easy. Company C underlines the difficulties related to cross domain engineering: *"You have to consider all the requirements all the time, their interdependencies, variations of system design variables, to achieve a robust solution you have to extract the knowledge needed in your design experience to choose the right solution".*
At all the companies, the different departments are well defined and separated, and each team has its own managers and prioritisation of requirements at two levels local and global. Communication between these departments is ensured by cross-function teams (Companies A and D) or people (Company B) whose job is to track the interdependencies between requirement sets from different departments.
Company A uses moderators to handle priority conflicts for cross-department requirements. At Company D, for larger projects all the different units of the company are involved, with representatives of each unit sitting in on meetings and negotiations between the different teams and *"dependency checks between departments are included in the product development processes"*. Company B has no specific process for detecting conflicts or early stage analysis.
Company A mentions that due to size *"at a certain level the overall dependencies are lost because otherwise it would be impossible to handle"*. All companies agree that some separation across departments is necessary for successful collaboration. Prioritisation is also an issue when it comes to handling interdisciplinary requirements. *"Requests from other departments are internally prioritised lower by a particular department, which is a problem when there is a time pressure"* (Company A). Companies B and D mention the same issue. Budget also limits the time spent on analysing requirements at Company D.
There is a distinction made between flexible people who move across projects, and those who are responsible for a particular, often hardware related area of the product, *"so long-term there is a mix with most people functioning within their subsystems, with open interfaces between teams, departments"* (Company A).
For all the companies interviewed physical proximity is an enabler and distance is an issue, especially due to lack of appropriate collaboration frameworks. Companies A, C, and D bring up problems in international collaboration. One such problem is the size of the different organisations. Much smaller branches for instance will not have managers for certain aspects of a project whereas much larger branches will have more resources and higher process standards that they expect to be upheld. Other issues in collaboration across departments involve differences in department unit sizes (a factor of 10 is mentioned by Company D), tools used, mentality and work processes. As collaboration increases these problems become more obvious.
When asked what should be improved in cross-domain requirement management, interviewees mentioned the need for a more collaborative approach: *"Thinking of other departments ... a more integrated mind set is something that should be aimed for"* (Company D). Companies A, B, and D agree

that to improve collaboration the first step is to do organisation reviews. At the moment, this is actively enforced only at Company A where *"the way of working is constantly reviewed, … tweaking so that overly complicated parts of the process are changed"*. There is a wish for such review processes at Companies B and D, but they mention the time and resources needed to do this as an obstacle. Company C has a positive outlook on their company-wide communication and states that although there are a lot of interdependencies between departments it is easy to identify where and what kind of collaborations are needed.

4.8. Modification of requirement specifications

All four companies have an iterative process for clarifying the requirements at the start of the development process. Three out of four mentions check points and the fourth mentions tollgates. Once the specification is approved and the development has started if changes need to be made, they are lifted one level up to the gate group (Company D) or to the systems management responsible (Company A). The development teams cannot modify requirements by themselves. Market changes can also result in modifications of the requirement specification.

How late in the process requirements can be changed varies across companies and depends on the criticality of the issue. For Company B changes are *"no problem during concept phase, harder to do at development phase, at verification phase the cost becomes high, but it is always possible if it is really important"* (i.e. for a critical safety issue). Similarly, for Company C *"changes are avoided after product sign-off but if necessary they are discussed"* and a decision is taken on whether there will be no change, a temporary change or a permanent change, however a way for *"expressing timing frames for different types of changes"* is currently lacking. For software-oriented Company A delivery cycles are much shorter depending on the source of the change and the complexity it can result in a feature being delayed until the next delivery.

4.9. Verification and validation

All four companies use the requirement specification for verification and Companies A, C, and D agree that verification needs to be done continuously. Company B considers that verification of the system cannot start until the design is complete as the requirements are not fully broken down until that point, however breaking down requirements and selecting the right components that ensure feasibility can be considered as part of verification. In Company A verification criteria are part of the requirement specification. Company C is moving to a model-based requirement representation to use requirements in calculations.

In terms of terminology, Company C distinguishes between short-term verification of functional requirements that is performed continuously and long-term verification of whether the implementation fulfils the functional behaviour that requires the presence of test rigs. At Company D they speak of validation for short-term requirements and verification for long-term requirements.

A distinction between short-term and direct requirements (e.g. requirements pertaining to low-level product specification) and long-term (e.g. user feeling) is made in terms of how they are verified.

Validation of long-term requirements on product lifespan and customer satisfaction can be complicated to get, but it is possible to obtain some insight by:

- Analysing customer satisfaction (quality index), monitoring existing fleets and using net promotor score (NPS) - ask customers about what they think about the delivery, have it as a key performance indicator (KPI) – *"but this is hard to link to the requirement specification"* (Company D)
- Using the notion of an internal customer (Companies A and C)
- Using a focus person or group that makes a judgement based on experience (Company B)

It may however take *"1 to 10 years to get feedback"* (Company D) and such reports can be quite hard to analyse. For instance, an unexpected alarm being triggered by the system with a weekly frequency was linked to repetitive patterns of unforeseen user behaviour – i.e. two employees standing on the equipment to talk to each other (Company B). Table 1 shows what verification methods were mentioned quantifiable requirements.

Table 1. Verification methods mentioned for quantifiable requirements

Verification methods Company:	A	B	C	D
Comparison to previous versions of the product		X		
NDS NETWORK description specification - use case diagrams, flows, use case realisation documents, classes Data structures	X			
Prototypes				X
Providing third party suppliers with rigorous testing protocols			X	
Quality test stack - about 10 levels of testing		X		
Root cause analysis for unpredicted issues				X
Simulation		X		
Test laboratories or centres		X	X	
Test rigs			X	X
Test specifications	X			
Tests on actual hardware		X	X	
Verification standard for tests unique to each type of product		X		
Virtual verification of the system, up to a year before the release of the product, currently parts still need to be built physically, but the aim is to be fully virtual			X	

At Company B *"test results are documented based on the FRS [functional requirement specification]"* and used to check that the products fulfil the required norms and standards.
The following issues were brought up during the interviews:

- Not all the requirements in the specification are tested, due to some requirements still being too vague, not quantifiable (user feeling) or too expensive to test. Company B for instance tests only the PRS requirements. Inversely, testing does not detect missing requirements, so engineers often have to rely on experience to see whether anything is missing.
- Better tools for tracing fault causes at sub-system level are necessary, rather than just observing issues at system level. *"It is necessary to better understand the reasons why requirements are not met by the system."* (Company D)
- Company C points out that just checking against a requirement specification is insufficient as several solutions might fulfil the requirement criteria but some might be more robust than others. Therefore "*verification needs to be complemented by robustness studies"*.

5. Discussion

Assembling the findings from the interviewees gives a picture of a common three-step requirements process:

1. Requirements from many different sources are collected, interpreted, and prioritized by a product manager, who acts internally as the voice of the customer. These requirements are typically well known in the companies, but often vaguely specified.
2. Requirements are analysed and classified in terms of whether they belong to a specific sub-system or if they are cross-functional requirements. The quality requirements are classified into guaranteed quality requirements (e.g. safety), user experience, and performance requirements.
3. Further decompositions are made until appropriate level is reached, and the resulting set of requirements can be transferred to the appropriate developer(s). Specifications that are even more detailed can be made, but only for internal purposes. The decomposition process is instrumental in detecting conflicting requirements, such as, quality-cost or new versus old functions.

The purpose of the specification process is to reduce the fuzziness of the requirements; most companies are satisfied with some kind of modelling combined with natural language. No companies in the interviews use formal specification or proofing techniques in validation. All companies acknowledge the fact that a complete specification is never possible; there are always unspoken, domain-dependent requirements.
An interesting observation is that the interviewees report an increased focus on safety requirements the recent years. In the companies, these are often handled by specialized departments or experts. This need

of guaranteed quality requirements can be hard to combine with agile methods that optimize the flow in development. Not surprisingly, companies have an interest in increased use of agile methods; the more software in the product, the more agile methods. The problem of marrying agile methods and development of safety-critical software is also a focus for ongoing research, for a recent summary see (Hanssen et al., 2017).

Prioritization and traceability are mentioned as key capabilities in requirements management in all companies. The larger product, the more necessary they become. However, both capabilities eat resources from the projects, and companies have a challenge finding the right level of cost-benefit. Most companies use some kind of database to manage requirements, and they seem generally happy with support for traceability, but would like more support for the human-intensive prioritization.

A noteworthy observation is that at a high-level prioritization is a powerful tool to direct the focus of the projects, but it can also become an impediment. Three of the companies witnessed that it is very hard for one department to request changes in priorities in another department, thus leading to a risk of sub-optimizing the development.

Traceability can be achieved with different means from direct links to human expertise. Human knowledge is mostly used in tracing over department borders and if there is a large physical distance between the organizational units. There is also a border between high-level and low-level requirements that impedes traceability, especially for hardware where there are relatively more implicit requirements. Since engineers know these requirements well, it's not necessary to trace them to high-level requirements. This notion of unspoken or "implicit" requirements is recognized in literature (Onyeka, 2013) but has been addressed mostly from the high level and customer perspective. Unexpectedly, one company regarded upwards traceability as a confidentiality issue due to market competition; it was considered sensitive to trace low-level requirements to high-level requirements.

Dependencies among interdisciplinary requirements are handled with human communication and processes. Good practices comprise using moderators to handle conflicts, regular dependency checks, cross-functional teams, and constant review of the organization. The latter is very expensive and is used only by one company. Proximity is an enabler for communication, but the size of products often makes international collaboration a necessity. It was observed that it is relative easy to communicate with a small, specialized branch compared to communicating with a large branch with more complex internal communication and unspoken practices. Most of these issues are consistent with the challenges identified in a recent literature review by Schmid (2014) with the notable exceptions of the need for confidentiality and lack of transparency and of the issues related to unspoken or implicit physical requirements. Distance as such, is a multi-faceted concept, Bjarnason (2013) introduces the term *RE distance* to describe the challenge in communication of requirements. The situation described by the companies in this study corresponds to geographical distance, socio-cultural distance, and organizational distance. Bjarnason's mapping study also includes distances in terms of time, opinion, psychology, cognition, and syntax and semantics of artefacts. It might be a way forward to make a more complete analysis of distances to guide the work of improving communication. There is a general lack of tools for communication, especially internationally.

For interdisciplinary requirements, there is still a gap between hardware and software development processes, and in particular due to development time frames (e.g. hardware decisions taken much earlier than software is developed, longer hardware development cycles). Companies A and B mention that lack of coordination between the software needs and the hardware choices results in having to adjust the software in sub-optimal ways. Modelling and simulation of the entire systems and deferring the hardware/software delimitations after a thorough analysis have been suggested as a way forward. For instance, Myers et al. (2011) describes a *comodelling* approach that combines behavioural engineering and acausal models of physical systems as a powerful and flexible means towards this end. For large-scale applications there are of course limitations of how detailed the behavioural models need to be to still give important information to the designer.

6. Conclusion and future work

Empirical studies in cross-domain systems engineering usually focus on top-level requirements engineering processes. In this paper we documented the view of domain-specific engineers in the cross-

domain product development context. Companies in this study recognize that requirements that stretch across engineering domains are clearly challenging to work with but are an essential part of a product's design process. There seem to be attempts of tackling this challenge but also some profound growing pains. The common challenge seems to lie with the desire to develop products as fast as possible without risking quality, so everyone wants to do better but are not very eager to change their way of working. By putting more efforts in requirements engineering activities in early stages of the product development, costly late changes can be avoided (Lindahl and Sundin, 2012). Exactly how much effort that is needed to pay-off is still an open question for future research.

The companies in this study all have a similar working process as far as breaking down requirements in a suitable way but later in the process struggle with requirements that stretch across several departments or engineering domains. We have found several practices from the different companies in how this can be addressed, which can be the seed for continued research into a more coherent method. It is noteworthy that deployment of practices differs largely between companies, see for instance Table 1 on verification. This indicates that future research must be prepared to document a large amount of practices and their dependencies to be practically useful for a large number of companies. Our findings also contributed a classification of findings into nine different knowledge areas, which can be the starting point in identifying more specialized studies in future.

References

Akao, Y. (1990), *Quality Function Deployment,* Productivity Press, US.

Bell, J. and Nilsson, B. (2006), *Introduktion till forskningsmetodik*, Studentlitteratur, Sweden.

Bjarnason, E. (2013), "Distances between Requirements Engineering and Later Software Development Activities: A Systematic Map", In: Doerr, J. and Opdahl, A.L. (Eds.), *Requirements Engineering: Foundation for Software Quality. REFSQ 2013. Lecture Notes in Computer Science, Vol. 7830,* Springer, Berlin, Heidelberg, pp. 292-307. https://doi.org/10.1007/978-3-642-37422-7_21

Chan, L.-K. and Wu, M.-L. (2002), "Invited Review: Quality function deployment: A literature review", *European Journal of Operational Research*, Vol. 143 No. 3, pp. 463-497. https://doi.org/10.1016/S0377-2217(02)00178-9

Chang, W., Yan, W. and Chen, C.H. (2013), "Customer requirements elicitation and management for product conceptualization". In: Stjepandic J., Rock G. and Bil, C. (Eds.), *Concurrent Engineering Approaches for Sustainable Product Development in a Multi-Disciplinary Environment*, Springer, England, pp. 957-968. https://doi.org/10.1007/978-1-4471-4426-7_81

Chemuturi, M. and Gilb, T. (2013), *Requirements engineering and management for software development projects,* Springer, US. https://doi.org/10.1007/978-1-4614-5377-2

Cheng, B.H.C. and Atlee, J.M. (2007), "Research directions in requirements engineering", *Future of Software Engineering, FOSE '07, Minneapolis, MN, USA, May 23-25, 2007,* pp. 285-303. https://doi.org/10.1109/FOSE.2007.17

Hansoft (2017), *Hansoft - The Agile Project Management Tool for Enterprise Teams.* [Online] Available at: http://www.hansoft.com (accessed 25.10.2017).

Hanssen, G.K., Myklebust, T., Johnsen, S.O. and Doss, O. (2017), "Trends in agile development of safety-critical software: A summary of the 3d international workshop on agile development of safety-critical software (ASCS 2017)", *Proceedings of the XP2017 Scientific Workshops, Cologne, Germany, May 22-26, 2017,* Association for Computing Machinery. https://doi.org/ 10.1145/3120459.3120480

Hood, C. (2008), *Requirements management: the interface between requirements development and all other systems engineering processes*, Springer, England. https://doi.org/10.1007/978-3-540-68476-3

Hull, E., Jackson, K. and Dick, J. (2011), *Requirements engineering*, Springer, England. https://doi.org/10.1007/978-1-84996-405-0

IBM (2017), *IBM Rational DOORS family.* [Online] Available at: http://www-03.ibm.com/software/products/en/ratidoor (accessed 25.10.2017).

ISO/IEC/IEEE 24765 (2010), *Systems and software engineering – Vocabulary,* ISO/IEC/IEEE, Switzerland.

Jiang, L., Eberlein, A. and Far, B.H. (2005), "Combining requirements engineering techniques - theory and case study", *12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05), Greenbelt, MD, USA, April 4-7, 2005,* IEEE, pp. 105-112. https://doi.org/10.1109/ECBS.2005.25

Johansson, H. (2017), Interdisciplinary Requirement Engineering for Hardware and Software Development - from a Hardware Development Perspective, Master Thesis, Linköping University.

Kossiakoff, A., Sweet, W.N., Seymour, S.J. and Biemer, S.M. (2011), *Systems Engineering Principles and Practice*, John Wiley and Sons, USA. https://doi.org/10.1002/9781118001028

Kotonya, G. and Sommerville, I. (1998), *Requirements engineering. processes and techniques*, John Wiley, England.
Lamsweerde, A.V. (2009), *Requirements engineering: from system goals to UML models to software specifications*, John Wiley, England.
Lindahl, M. and Sundin, E. (2012), "Product Design Considerations for Improved Integrated Product/Service Offerings", In: Kauffman, J. and Lee, K.M. (Eds.), *Handbook of Sustainable Engineering*, Springer, Dordrecht, Netherlands. https://doi.org/10.1007/978-1-4020-8939-8_62
Meier, H., Roy, R. and Seliger, G. (2010), "Industrial Product-Service Systems - IPS²", *CIRP Annals*, Vol. 59 No. 2, pp. 607-627. https://doi.org/10.1016/j.cirp.2010.05.004
Mont, O.K. (2002), "Clarifying the concept of product–service system", *Journal of Cleaner Production*, Vol. 10 No. 3, pp. 237-245. https://doi.org/10.1016/S0959-6526(01)00039-7
Myers, T., Dromey, G. and Fritzon, P. (2011), "Comodeling: From Requirements to an Integrated Software/Hardware Model", *Computer*, Vol. 44 No. 4, pp. 62-70. https://doi.org/10.1109/MC.2010.270
Nilsson, S. (2017), How Requirements Development Could Support Design of Effective and Resource-Efficient Offerings, Licentiate Thesis, Linköping University. https://doi.org/10.3384/lic.diva-143011
Nilsson, S. and Lindahl, M. (2016), "A Literature Review to Understand the Requirements Specification's Role when Developing Integrated Product Service Offerings", *Procedia CIRP*, Vol. 47, pp. 150-155. https://doi.org/10.1016/j.procir.2016.03.225
Onyeka, E. (2013), "A process framework for managing implicit requirements using analogy-based reasoning: Doctoral consortium paper", *IEEE 7th International Conference on Research Challenges in Information Science (RCIS), Paris, France, May 29-31, 2013,* pp. 1-5. https://doi.org/10.1109/RCIS.2013.6577726
Sage, A.P. and Rouse, W.B. (1999), *Handbook of systems engineering and management*, Wiley, USA.
Schmid, K. (2014), "Challenges and Solutions in Global Requirements Engineering – A Literature Survey", In: Winkler D., Biffl S. and Bergsmann J. (Eds.), *Software Quality. Model-Based Approaches for Advanced Software and Systems Engineering (SWQD 2014) Lecture Notes in Business Information Processing, Vol. 166,* Springer, Cham. https://doi.org/10.1007/978-3-319-03602-1_6
Siemens (2017), *Explore Teamcenter and discover your solution.* [Online] Available at: https://www.plm.automation.siemens.com/en/products/teamcenter/ (accessed 25.10.2017).
Smith, P.G. and Reinertsen, D.G. (1998), *Developing products in half the time: new rules, new tools*, Wiley, USA.
Sommerville, I. and Sawyer, P. (1997), *Requirements engineering: a good practice guide*, Wiley, England.
Stjepandić, J., Wognum, N. and Verhagen, W.J.C. (2015), *Concurrent Engineering in the 21st Century. Foundations, Developments and Challenges*, Springer, Switzerland. https://doi.org/10.1007/978-3-319-13776-6
Sundin, E., Lindahl M., Öhrwall Rönnbäck A., Ölundh Sandström G. and Östlin J. (2006), "Integrated Product and Service Engineering Methodology", *Proceedings of 11th International Conference of Sustainable Innovation, Chicago, USA, October 23-24, 2006.*
Sutcliffe, A. (2002), *User-centred requirements engineering*, Springer, England. https://doi.org/10.1007/978-1-4471-0217-5
Ullman, D.G. (2002), *The Mechanical Design Process*, McGraw-Hill Higher Education, USA.
Vasantha, G.V.A., Roy, R., Lelah, A. and Brissaud, D. (2012), "A review of product–service systems design methodologies", *Journal of Engineering Design*, Vol. 23 No. 9, pp. 635-659. https://doi.org/10.1080/09544828.2011.639712
Wiesner, S., Hauge, J.B., Thoben, K.D. and Peruzzini, M. (2015), "Requirements engineering", In: Stjepandic, J., Wognum, N. and J.C. Verhagen, W. (Eds.), *Concurrent Engineering in the 21st Century - Foundations, Developments and Challenges,* Springer, Switzerland, pp. 103-132. https://doi.org/10.1007/978-3-319-13776-6_5
Wiesner, S., Nilsson, S. and Thoben, K.-D. (2017), "Integrating requirements engineering for different domains in system development – lessons learnt from industrial SME cases", *Procedia CIRP*, Vol. 64, pp. 351-356. https://doi.org/10.1016/j.procir.2017.03.013

Sara Katrina Nilsson, Tech. Lic.
Linköpings Universitet, IEI
58183 Linköping, Sweden
Email: sara.k.nilsson@liu.se