Sanja Aaramaa

DEVELOPING A REQUIREMENTS ARCHITECTING METHOD FOR THE REQUIREMENT SCREENING PROCESS IN THE VERY LARGE-SCALE REQUIREMENTS ENGINEERING CONTEXT
SANJA AARAMAA

DEVELOPING A REQUIREMENTS ARCHITECTING METHOD FOR THE REQUIREMENT SCREENING PROCESS IN THE VERY LARGE-SCALE REQUIREMENTS ENGINEERING CONTEXT

Academic dissertation to be presented with the assent of the Doctoral Training Committee of Technology and Natural Sciences of the University of Oulu for public defence in the OP auditorium (L10), Linnanmaa, on 1 December 2017, at 12 noon
Aaramaa, Sanja, Developing a requirements architecting method for the requirement screening process in the Very Large-Scale Requirements Engineering Context.
University of Oulu Graduate School; University of Oulu, Faculty of Information Technology and Electrical Engineering
Acta Univ. Oul. A 700, 2017
University of Oulu, P.O. Box 8000, FI-90014 University of Oulu, Finland

Abstract

Requirements engineering (RE) is an important process in systems development. This research was carried out in the context of Very Large-Scale Requirements Engineering (VLSRE) within the scope of a requirement screening (RS) process. The RS process is defined as a front-end process for screening incoming requests, which are received in a constant flow. The goal of the RS process is to efficiently identify the most promising requests for further analysis, development and implementation while filtering out non-valuable ones as early as possible.

The objective of this study was to understand the challenges related to the RS process and develop solutions to address those challenges. A qualitative research approach was utilised to achieve the research goals. The overall research process follows an action research method, in which each action research cycle includes at least one individually defined and executed case study. Action research and case studies are research methods that are well suited to studying real-life phenomena in their natural settings. This research was carried out in two case companies in the information and communication technology domain. Data from 45 interviews were analysed for preparing publications I–V, which are included in this thesis. In addition, during the longitudinal action research study described in this thesis, data from 26 interviews and 132 workshops were utilised to develop solutions for the RS process, which is an industrial implementation of the VLSRE process. The conducted action research contributes to the field of software engineering, in which such research efforts are currently lacking.

This research has identified a number of significant challenges that different stakeholders face related to requirements processing and decision making in the VLSRE context. Examples of these challenges are the great number of incoming requirements, the lack of information for decision making and the feasibility of utilised tools. To address the identified challenges, a requirements architecting method was developed. The method includes a dynamic requirement template, which gathers structured information content for eliciting requests, documenting and communicating requirements and forming features while considering the needs of different stakeholders. The method was piloted, validated and deployed in industry.

Keywords: action research, case study, empirical research, product management, requirement screening, requirements engineering, requirements management, software engineering, Very Large-Scale Requirements Engineering, VLSRE
Aaramaa, Sanja, Vaatimusten analysointi- ja seunontamenetelmän kehittäminen laajamittaiseen vaatimusmäärittelyprosessiin.
Oulun yliopiston tutkijakoulu; Oulun yliopisto, Tieto- ja sähköteknikan tiedekunta
Acta Univ. Oul. A 700, 2017
Oulun yliopisto, PL 8000, 90014 Oulun yliopisto

Tiivistelmä

Tutkimus toteutettiin laajamittaisen vaatimusmäärittelyprosessin kontekstissa keskittyen vaatimusten seunontaprosessiin. Vaatimusten seunontaprosessi määrittelee tuotekehityksen alkuvaiheen prosessiksi, jossa käsitellään jatkuvana vuonna tulevia kehityspyynnöjä. Vaatimusten seunontaprosessissa pyritään tunnistamaan tehokkaasti lupaavimmat pyynnöt jatkoanalyysin, tuotekehitystä ja toteutusta ajatellen sekä suodattamaan pois niin aikaisessa vaiheessa, kun mahdollista ne pyynnöt, joilla ei ole arvontuotto-odotuksia.


Asiakirjat: empirinen tutkimus, laajamittainen vaatimusmäärittelyprosessi, ohjelmistotutkinto, tapaustutkimus, toimintatutkimus, tuotteenhallinta, vaatimusmäärittely, vaatimusten hallinta, vaatimusten seunonta
brave hearts, free souls, great minds
Acknowledgements

My mom always said life was like a box of chocolates. You never know what you're gonna get.

~Forrest Gump (film)

Do I have what it takes? – at least what I thought it would take. It was almost as if the offered job as a project researcher and opportunity to start my PhD studies was calling me for a duel. My curious mind and eagerness to test my limits tricked me into this journey, where I expected I would learn to collect and analyse data, read and write papers, improve my English skills and get project work experience. I expected to learn all this in a pre-defined manner and in a relatively smooth-running flow of activities. I was a little bit worried whether I would get bored with sitting quietly in an office and taking courses for few more semesters after just completing my master’s studies in Mathematics. How little I knew about what I was diving into! Indeed, I acquired the required research skills; the publications, the thesis and the public defence of my work are the proof of that. However, the environment, the pace of project work, the number of tasks trusted to me and co-operating daily with so many different people was something I did not anticipate. However, as demanding as these things sometimes seemed, they were the ones providing the most important lessons learnt. For example, I learnt how to perform under constant changes, pressure and uncertainty, deal with setbacks, make decisions, stretch time and eventually also prioritising things. These experiences last for a lifetime, as they helped me become a better person. Today I can say that indeed I had what it took, even if the requirements were not clear at the beginning.

I owe a lot to my supervisors, Professor Markku Oivo and Professor Samuli Saukkonen, for getting this work done. Thank you, Samuli for your patience and never ceasing to believe or let me stop believing that one day this would be done. Thank you, Markku, for stepping in for the last crucial steps of this work and the time and effort you invested in for guiding me through those; the last publication included in this thesis and finalising the compilation. You really made this work flow with unswerving proficiency and intensive scheduling, sometimes measuring in minutes. Without your contribution, the last word in this thesis would not have been written. I am grateful for having an incredible follow-up group. Thank you, Professor Jouni Similä, for cheering me up when I was feeling frustrated as well as discussions and academic feedback during this PhD project. Thank you so much, Dr Jouni Markkula, for always being there. You are a person one can come to and
ask anything, knowing that you do not make them feel like complete idiots; one can trust that you will listen to them and give advice that is right to the point. There are two mentors and role models I can never thank enough. The one and only Dr Pasi Kuvaja has been like an academic father, whose advice beyond that related to science has made me a better human being; and Timo Wiander has always seen more in me than I ever will and has been able to convince me to seize new opportunities to develop myself.

I am grateful for the pre-examiners, Professor Marjo Kauppinen, who sacrificed part of the always-too-short Finnish summer, and Professor Paul Grünbacher, who cleared his calendar for reviewing and commenting on my thesis. Your feedback has been valuable, indeed, in improving this thesis. I would like to express my gratitude for Professor Jürgen Börstler accepting the invitation to serve as an opponent in my public defence and challenging me to argue for my work.

I consider myself privileged for having an opportunity to work with so many unique people during my studies. Thank you Jarkko Hyysalo, Nebojša Taušan, Markus Kelanti, Tuomo Kinnunen, Sandun Dasanayake, Pekka Belt and Jari Lehto for writing with me the publications included in this thesis. Thank you Jari for introducing such interesting industrial dilemmas to explore. Thank you Jarkko and Pekka for sharing your experience with me; Jarkko, I must have interrupted your thoughts a thousand times, and you never complained about it. Nebojša and Markus, it was in many ways an enlightening experience to share an office with you. No topic seemed to be taboo for us, and most of the time we managed to keep our conversation at an appropriate sound level, but sometimes we got quite passionate about topics close to our hearts, which boosted the volume a bit. Thank you, Tuomo, for conducting interviews and data analysis, writing project documents and papers, planning projects and providing an engineering outlook for academic life – simply being a friend. Sandun, you and I have seen more airports, buses, metros, and train stations over years than I can remember. Conducting interviews took us to the different corners of Europe, big cities and small towns, where hand gestures were the only way to communicate, and the closest bus stop was a kilometre away from a hotel in the middle of nowhere. Thank you for being part of my PhD journey.

Anna, Elina, Tanja and Pilar, thank you for countless lunch breaks, delicious recipes, practical tips and, yes, being my workmates as well. Thank you for your friendship, which has spanned almost a decade! Vladimir, Burak, Kari and Harri, thank you for the philosophical discussions, support, great courses and study sessions, projects and for reviewing my work over the years. From the bottom of my heart, thank you Marja-Liisa Liedes, Heli Alatalo and Risto Nuutinen for helping me with
the endless administrative rehearsals I have gone through during my studies and as a project manager.

I have carried out my PhD studies entirely in externally funded projects, which has been a great opportunity to access companies, meet marvellous people from all corners of the globe and enjoy the richness of cultural diversity. ITEA and Tekes through their different funding programmes are the enablers of this work.

Finally, I would like to thank my family and friends for being there for me. Every now and then you enabled and even forced me to forget the work and studies for a while. On the other hand, you have inquired the date of the party often enough pushing me to continue writing. You have helped us in moving, renovating, taking care of our children when us parents had conflicting work schedules and enabling a few get-away weekends for us. I thank you, my daughters, for your patience and big hearts while waiting for me to take you shopping, help you with your homework and accepting the fact that your competition or show outfits would be the last ones finished – typically the night before you needed them. I can never thank my husband enough for giving me so much room and time to see this PhD project through and develop my professional skills. I do not know how many men there are in the world who would have survived these years, but I think that you are one in a million at the very least.

2.09.2017                Sanja Aaramaa
## Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR</td>
<td>action research</td>
</tr>
<tr>
<td>cf.</td>
<td>confer</td>
</tr>
<tr>
<td>DfA</td>
<td>Design for assembly</td>
</tr>
<tr>
<td>DfBA</td>
<td>Design for board assembly</td>
</tr>
<tr>
<td>DfFA</td>
<td>Design for final assembly</td>
</tr>
<tr>
<td>DfE</td>
<td>Design for e-commerce</td>
</tr>
<tr>
<td>DfD</td>
<td>Design for delivery competence</td>
</tr>
<tr>
<td>DfE</td>
<td>Design for environment</td>
</tr>
<tr>
<td>DfP</td>
<td>Design for packaging</td>
</tr>
<tr>
<td>DfS</td>
<td>Design for serviceability</td>
</tr>
<tr>
<td>DfSec</td>
<td>Design for security</td>
</tr>
<tr>
<td>DfSM</td>
<td>Design for supply chain M-management</td>
</tr>
<tr>
<td>DfSsSec</td>
<td>Design for software security</td>
</tr>
<tr>
<td>DfT</td>
<td>Design for testability</td>
</tr>
<tr>
<td>DfX</td>
<td>Design for eXcellence</td>
</tr>
<tr>
<td>DP</td>
<td>decision point</td>
</tr>
<tr>
<td>e.g.</td>
<td>exempli gratia</td>
</tr>
<tr>
<td>etc.</td>
<td>et cetera</td>
</tr>
<tr>
<td>Fig.</td>
<td>figure</td>
</tr>
<tr>
<td>ICT</td>
<td>information and communication technology</td>
</tr>
<tr>
<td>ID</td>
<td>identification</td>
</tr>
<tr>
<td>i.e.</td>
<td>id est</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>LSRE</td>
<td>Large-Scale Requirements Engineering</td>
</tr>
<tr>
<td>N/A</td>
<td>not applicable</td>
</tr>
<tr>
<td>PM</td>
<td>product management</td>
</tr>
<tr>
<td>R&amp;D</td>
<td>research and development</td>
</tr>
<tr>
<td>RE</td>
<td>requirements engineering</td>
</tr>
<tr>
<td>RS</td>
<td>requirement screening</td>
</tr>
<tr>
<td>SME</td>
<td>small and medium-sized enterprise</td>
</tr>
<tr>
<td>SW</td>
<td>software</td>
</tr>
<tr>
<td>VLSRE</td>
<td>Very Large-Scale Requirements Engineering</td>
</tr>
</tbody>
</table>
List of original publications

This thesis includes the following five original papers. The papers are referred to throughout the text by their Roman numerals.


The papers listed above have been published in the peer-reviewed proceedings of international conferences. The author of the thesis was the corresponding author of publications II, III and V. In these publications, the author coordinated the writing process as a primary author, was responsible for defining the research problems and questions and forming the theoretical groundwork as well as analysing data for drawing conclusions. The co-authors of publications II, III and V contributed to the data collection and analyses and gave valuable feedback in regard to structure and flow during the internal review process. The author was responsible for coordinating the data collection and had a significant role in the data collection and analysis for publications I, II and III. The author played a considerable role in the data collection and the writing of publications I and IV and actively participated in the data analysis as well.
# Table of contents

Abstract

17

Tiivistelmä

Acknowledgements  9

Abbreviations  13

List of original publications  15

Table of contents  17

1 Introduction  19

1.1 Background and motivation ................................................................. 20

1.2 Research scope .................................................................................. 23

1.3 Structure of the thesis....................................................................... 27

2 Related work and key concepts  29

2.1 Definition of relevant terms and fundamental concepts.................... 29

2.1.1 Problem domain, solution domain and stakeholders .................... 29

2.1.2 Design for eXcellence ................................................................. 29

2.1.3 Product management and product managers ................................ 31

2.1.4 Software architecture and software architects ............................ 31

2.1.5 Requirement screening principles and process .............................. 32

2.1.6 Needs, features and requirements ............................................. 33

3 Research problem and research questions  37

4 Research approach and methods  39

4.1 Action research and case studies in software engineering ............... 40

4.2 Description of the primary case companies ...................................... 42

4.3 The longitudinal action research conducted ..................................... 42

4.3.1 Phase 1 – Diagnosing ................................................................ 46

4.3.2 Phase 2 – Action research cycles ............................................. 46

4.3.3 Phase 3 – Continuous deployment ........................................... 51

4.4 The case study in company B............................................................ 53

4.5 A method to evaluate the rigour and relevance of the research .......... 53

5 Results  55

5.1 An overview of the original publications included in the thesis ........ 55

5.1.1 Publication I: A new way to organise DfX in a large organisation ................................................................. 55

5.1.2 Publication II: Design for eXcellence in the context of Very Large-Scale Requirements Engineering ......................... 56
5.1.3 Publication III: Managing constant flow of requirements: Screening challenges in Very Large-Scale Requirements Engineering ................................................................. 57
5.1.4 Publication IV: Customised choreography and requirement template models as a means for addressing software architects’ challenges ...................................................... 58
5.1.5 Publication V: Requirements volatility in software architecture design: An exploratory case study ............................................................... 58

5.2 The results of the longitudinal action research ........................................ 59
5.2.1 Findings from Phase 1 – Diagnosing ................................................. 59
5.2.2 Findings from Phase 2 – Action research cycles .............................. 61
5.2.3 Cycle 2: Paper piloting – The proof of concept ............................. 62
5.2.4 Cycle 3: Validating the method using the prototype tool ............... 65
5.2.5 Findings from Phase 3 – Continuous deployment ......................... 65

6 Discussion 67

6.1 The challenges of key stakeholders related to the requirement screening process .................................................................................. 67
6.2 Requirement architecture in VLSRE ................................................... 72
6.3 Addressing the challenges identified in the VLSRE context .......... 73
6.4 Improving requirements management practices in VLSRE ............. 75
6.5 Supporting decision making in a VLSRE context ............................. 76
6.6 Validity evaluation .......................................................... 77
6.7 Evaluating the relevance of the conducted research ...................... 79

7 Conclusion 81

7.1 Limitations and future research ...................................................... 82

References 83

Original publications 95
1 Introduction

Tough competition in the information and communications technology (ICT) domain motivates companies to seek ways to improve the efficiency of their software development and remain ahead of their competitors. These goals may be achieved through various means, such as efficient practices and well-organised processes. Research collaboration allows a company to acquire an objective understanding of state-of-the-art technologies and practices. Collaboration between industry and academia provides mutual benefit and enhances technology transfer [1]. In this research issues related to Very Large-Scale Requirements Engineering (VLSRE) have been addressed in two case companies. The majority of this thesis’s research was conducted in case company A, which is a large ICT enterprise. Company A provides software-intensive systems, services and solutions for the global market. Its software development approach is release-based and feature-driven. This is a business-to-business company, and its customers number in the hundreds. A complementary case study has been conducted in case company B, which is a comprehensive solutions provider to both private and business customers. The company operates in a global market offering a number of management tools and services to its business customers as well as various software products to its private customers.

The importance of requirements engineering (RE) in system development has been recognised by practitioners and researchers [2], [3], [4], [5]. Specific RE assessment and improvement models have been created and industrially validated by following generic software process improvement processes [6], [7], [8], [9]. In addition, methods and tools have been developed to help practitioners during RE [10], [11], but they are slowly being transferred from theory to practice [12]. Despite the progress that has been made in different areas of RE, further improvement is needed [9], since generic methods and tools are not applicable to specific contexts [13], [14]. In addition, technologies developed and evaluated by researchers are not easily adopted by practitioners due to the artificiality and lack of realism in the research setting. Ivarsson and Gorschek developed a model that may be utilised by both researchers and practitioners to evaluate the industrial relevance and rigour of empirical research [15].

Research methods, such as experiments, field or case studies and action research, may be utilised to explore real-world phenomena in their original contexts. Action research resembles the traditional software process improvement approach utilised in the ICT industry, and it is therefore considered to suit industry–academia
collaboration well [16]. However, action research has been applied more in information systems studies than in software engineering, where experimentation and case studies are extensively employed [15].

1.1 Background and motivation

In the early 1990s, it was proposed that software engineering and systems engineering should be combined into a new discipline called software systems engineering [17]. The rationale behind the proposal was that both disciplines address the creation of complex software-intensive systems. Nowadays, embedded systems are increasingly software intensive [18], and this also applies to the majority of the products offered by each of the case companies used in this research.

Besides the immense number of incoming requirements in a VLSRE context, distributed development also creates its own challenges when it comes to carrying out RE activities. Globally distributed development denotes locations that are all over the world and in various time zones and represent a diversity of cultures. The distances between the customers, or end users, and the engineers pose problems, but more importantly, due to outsourcing and offshoring, the designers and engineers are often located in different parts of the globe [19].

Due to the increasing complexity of software intensive systems, the number of incoming requests and/or requirements is also increasing [20]. Surely not all requirements can be implemented at once. Therefore, requirements have to be prioritised to determine in what release the selected requirements will finally be implemented [21]. The number of internal and external stakeholders, resources and technical constraints must be considered when decisions are made on the order of implementing requirements [22]. Decision making is not a trivial task, especially not early in the process when the information available is often abstract and uncertain [23]. One fundamental factor in decision making, requirements management and release planning is product management. Thus, product managers play a crucial role when it comes to the success of a company [24].

Requirements engineering (RE) is a branch of systems engineering containing a set of activities for requirements development and management [25]. According to the Institute of Electrical and Electronics Engineers (IEEE) standard requirements engineering is an “interdisciplinary function that mediates between the domains of the acquirer and supplier to establish and maintain the requirements to be met by the system, software or service of interest”. The RE process spans a system’s lifecycle and is concerned with discovering, eliciting, developing,
analysing, determining verification methods, validating, communicating, documenting and managing requirements [26]. RE is said to be the most valuable activity in systems engineering [27]. According to Leffingwell and Widrig, requirements management is a systematic way to elicit, organise and document system requirements as well as a process to establish and maintain agreement between a customer and a systems provider [28]. Many authors emphasise the nature of change in the RE process [25], [29], [30], thus the requirements management process is defined as a process of managing changes to the requirements [31]. On the other hand, requirements management has been defined as including requirements engineering activities from elicitation up to maintenance [28].

The changing nature of requirements is a phenomenon known as requirements volatility, which is defined as “the tendency of requirements to change over time in response to the evolving needs of customers, stakeholders, the organisation and the work environment” [32]. Changing user needs is one of the primary reasons for requirements volatility [33], which in turn causes changes in the content of individual requirements. Changes in the sets of requirements are mainly caused by conflicting stakeholder views and the complexity of organisation [22].

The quality of the developed system has been argued to be dependent on the quality of the development process as a whole [34] and on the quality of the requirements [35]. RE may be considered from two viewpoints. On one hand, RE is an organisational activity, since it should dictate what requirements will be implemented in products and make decisions on what requirements are released. On the other hand, RE is a project-level activity when requirements are actually implemented in products. [4]. According to Ebert [36], the product development process can be divided into two main process categories: 1) upstream processes, which include phases prior to a project launch, and 2) downstream processes, which cover the definition and execution of development projects. The purpose of the upstream processes is to analyse business opportunities and decide upon new development opportunities. Downstream processes focus on the actual development of products. RE is seen as a process that starts before a product is born and lasts throughout the product lifecycle [36].

The literature describes several RE process models [37], [38]. For example, Martin et al. analysed RE activities carried out in two case companies by looking at three projects in each with respect to the three well known RE process models presented in the scientific literature. Several studies indicate that there is a gap between the scientific view on RE and the one actually used in industry [38], [39],
Furthermore, even though RE processes are defined in companies, typically they are not strictly followed [39]. Although RE processes vary a great deal in practice, their fundamental activities are usually carried out in the following common activities:

- **Elicitation.** Identify sources of information and discover their requirements for the system to be developed.
- **Analysis.** Understand the requirements and their overlaps and conflicts.
- **Validation.** Check with the stakeholders that the requirements are what they really need.
- **Negotiation.** Try to reconcile the conflicting views of stakeholders and generate a consistent set of requirements.
- **Documentation.** Write down the requirements so that both stakeholders and developers understand them.
- **Management.** Control the requirements changes that will inevitably arise [41].

The concept of VLSRE emerged in the context of market-driven software development research. Regnell et al. [42] defined the number of requirements as an indicator that determines the scale of the RE where a company operates. An increase in the number of requirements is considered to indicate a higher number of internal and external stakeholders and leads to more interdependencies between requirements. A company is defined as operating in VLSRE mode if the number of items in a database exceeds 10,000 [42]. According to Regnell et al. [42] there are a variety of methods and tools available for practitioners to carry out RE activities, especially elicitation and analysis. However, those methods and tools are often validated in small or medium-scale RE contexts and are not necessarily applicable in a Large-Scale Requirements Engineering (LSRE) context and are even less applicable in a VLSRE context [42]. There are several templates and guidelines to document requirements, such as the requirement specification template by Volere [43] and IEEE guidelines [44]. The Volere template can be uploaded to tools such as IBM® Rational® DOORS®¹, which is widely used by software architects.

Essentially, a requirement (or a request as an expression of need) is information that is communicated between stakeholders. In addition to the content of the requirement itself, a number of other issues have to be taken into account when decisions are made about requirements. Examples of such issues are relevant standards, market analysis and company strategies. Together this leads to the

challenge of considerable information overload in RE. To overcome the information overload challenge and enable the traceability of emerging issues, Regnell et al. [42] proposed a research thread to establish requirements architecture. They defined the concept of “requirements architecture” as the way requirements are organised, including a data model of requirements, their attributes and the relations of those attributes [42]. Another important challenge posed by a great number of requirements is the interdependencies among requirements. These interdependencies affect prioritisation and resource planning and impact analyses, for instance [45].

The empirical cases considered throughout the course of the research for this thesis indicate that the identified gap in the state-of-the-art literature is a reality in large ICT organisations i.e. practitioners have to face the challenges of VLSRE in their daily work. This thesis addresses the VLSRE process in ICT organisations and the challenges faced by their practitioners. The focus is on the elicitation and analysis activities of RE, including management aspects. The thesis contributes to filling an identified gap in the state-of-the-art of software engineering research where empirical evaluations have been reported absent [46], and those that have been published have been accused of being unrealistic [47]. In this research a requirements architecting method, including a supporting tool, has been developed and validated in an industrial context to address the identified challenges in the requirement screening process. The idea of a requirement screening process is to quickly assess the potential of requirements in order to choose the most promising ones on which to work. Thus, the requirement screening process is an industrial implementation of the VLSRE process which is utilised in case company A. The significance of this study is highlighted by the fact that RE and requirements management are considered the most valuable functions of systems engineering [27]. Despite the importance of RE being widely acknowledged by researchers and practitioners, RE research has been criticised for having little impact on industry practices [48].

1.2 Research scope

The research for this thesis has been conducted mainly in the context of VLSRE. The scope of the research is limited to the requirement screening process and its stakeholders. In this research, the requirement screening process is defined as a front-end process for screening incoming requests. The goal of the requirement screening process is to quickly assess requests to identify those that are most
promising for further analysis and implementation. At the same time, the aim is to screen out non-valuable ones while requirements are developed and which then are grouped as features during the analysis. The principal idea of requirement screening is similar to that of a requirements selection process [49] and a requirements triage method [50].

In the first screening iteration of a requirement screening process, the requests with the most potential are chosen for further analysis, while others are rejected (e.g. due to their lack of business potential) or saved to be worked on later (e.g. due to their lack of information). The same logic is applied to requirements and feature candidates as well. Thus, in a requirement screening process, an RE item goes through three stages based on its maturity, and each of the iterations includes three options from which to choose for each RE item. As the core activity in a requirement screening process is to define the priorities for RE items and choose the most promising, this results in an ordered list of feature candidates for forthcoming releases. Consequently, it may be considered a decision-making [4], or filtering process [51].

Similar requirement selection practices, challenges and their solutions as identified in case company A have been typically discussed in the context of market-driven RE [52], [53], [54] and related to a release planning process [55], [22], [35]. The time-to-market pressure and limited resources lead to a situation in which not all the requirements can be implemented at once. This means that requirements have to be prioritised [56], [57], [58], [21]. Time-to-market pressure and the pace of change are emphasised in market-driven RE [59].

The context of the research in case company A is depicted in Fig.1. All major disciplines – Design for eXcellence (DfX), product management (PM), customer relations and software (SW) architects – were engaged during the longitudinal action research. However, the focus was on the requirement screening (RS) process. At the beginning of this research, the RS process included five decision points (DP0–DP4). The developed requirements architecting method led to the process change for omitting DP1, which is greyed out in Fig. 1. Publications I–V (PI–PV) included in this thesis have been placed next to their main data sources. The case company had recognised a need to improve its overall systems development process and proposed a research collaboration to explore issues related to its requirements processing practices. The requirement screening process mainly concerns the software part of system development. However, the hardware part may constrain software development, and certain customer needs may be met by either software or hardware solutions. In case company A, its requirement
screening process is a globally defined process that is utilised per product. On the other hand, different products can also be seen as part of the system, and thus product-specific requirement screening processes have to be synchronised at the systems level as well. However, systems level practices are not as mature as those at the product level.
Fig. 1. The context of the research.

The objective of the research conducted in case company A was to describe the processes related to the requirement screening process, to explore the challenges stakeholders face regarding this process and propose ways to improve the company’s requirement screening process practices. The research problem for this
study was How can requirement screening process practices be improved in a VLSRE context? The context was both complex and, to some extent, vague. Therefore, it was not self-evident where solutions would be found, how many action research iterations would be needed or what types of solutions should be developed. It was agreed that the research collaboration would be carried out systematically until satisfactory solutions were found.

The objective of the research conducted in case company B was to identify decision-making practices in the software architecture design process, understand the challenges software architects face due to requirements volatility and propose solutions to overcome the identified challenges. Case company B operates in LSRE mode and utilises agile practices in its software development. The company is divided into five units, and all were engaged during the case study.

1.3 Structure of the thesis

This PhD thesis describes the entirety of the longitudinal action research that has been carried out in a VLSRE context, including individual case studies, and presents a dynamic requirement template that was developed iteratively during the action research cycles. The intermediate results were published throughout the duration of the research effort, and five of the published papers are included in this thesis. This thesis is divided into seven chapters. Chapter 2 is dedicated to reviewing the related literature and introducing the important terms and concepts. The research problem is described and the research questions are defined in Chapter 3. The research work carried out for this thesis is described in detail in Chapter 4. Chapter 5 gives an overview of the original contributions. In Chapter 6, the empirical findings are discussed in regard to the scientific literature answering the research questions and evaluates the relevance of the research conducted using an assessment model [15] as well as addresses validity concerns. Chapter 7 concludes the thesis and outlines directions for future work.
2 Related work and key concepts

2.1 Definition of relevant terms and fundamental concepts

This research was conducted in the context of VLSRE with the aim of developing new practices for an efficient requirement screening process. During the research effort, various stakeholders were engaged. Next, the relevant stakeholders are defined along with other pertinent terms and concepts that will need to be understood to follow the text.

2.1.1 Problem domain, solution domain and stakeholders

The problem domain is the bounded part of reality where the problem is defined. Usually, the problem should be solved by a system or product. The problem domain addresses only matters of interest to its stakeholders. The proposed solution to the challenges of the problem domain is defined in the solution domain. Thus, the design and implementation of the system to solve the problem take place in the solution domain. Therefore, the solution domain is considered the “sandbox” of developers and designers. In both domains, the language and semantics used originate from the stakeholders, thus the concepts and entities in domains are usually unique. Moreover, the terminology applied may be contradictory within and between domains.

Stakeholders are people who have a stake in a system. The stakeholders are usually divided into external and internal stakeholders, thus two main groups are customers (users and system owners) and developers (analysts, designers, programmers, etc.) [29]. Taking a broader perspective, a stakeholder is defined as a group or an individual that is affected by the achievement of an organisation’s objectives or a group or an individual that can affect them [64].

2.1.2 Design for eXcellence

The Design for eXcellence (DfX) is an approach to designing products to achieve desired quality attributes (“-ilities”) taking into account the product life cycle and means of ensuring the cost-effectiveness of the development, delivery and disposal of the product [65]. Bralla defined DfX as a knowledge-based approach for
maximising the desired characteristics in product design while minimising product lifecycle costs [66]. This definition of DfX is the best known and can be considered a starting point for the modern interpretation of DfX. The philosophy of DfX assumes that excellent products may be created only through an excellent development process. A development process utilises tools for achieving its goals. Therefore, DfX may be considered from three major perspectives: organisation, designer and tools and methods [67].

The principles of DfX have been derived from the idea of serving internal customers with regard to manufacturability and value analysis [68]. The actual “design for” term was used for the first time alongside “assembly” (DfA) by [69]. Design for final assembly (DfFA) and board assembly (DfBA) [70] are examples of recently introduced terms. Various “design for” domains have emerged over the years, like design for packaging (DfP) [71] and design for supply chain management (DfSM) [72]. Some of these were defined decades ago, while others are just few years old. The new disciplines are being established as the need arises. One of the oldest disciplines is design for security (DfSec), which dates back to the 1960s and applies to the physical security of facilities [73], while design for software security DfSec is one of the most recently introduced [74]. Moreover, an emerging discipline is design for e-commerce (DfC), while design for serviceability (DfS) is an example of one of the earliest considerations. Design for delivery competence (DfD) as a discipline has not been discussed as such, but it shares a lot in common with the logistics [75] and supply chain management [72] disciplines.

Eleven disciplines were engaged throughout this research, but the design for environment (DfE) [76] and design for testability (DfT) [77] disciplines were involved in all DfX-related cases. The majority of the DfX disciplines mentioned are tightly connected with the engineering (hardware) domain. However, embedded systems are increasingly software intensive systems, and therefore these disciplines are beginning to have a stake in software development, too, not just in hardware. Therefore, DfX has recently attracted more attention in the software engineering and RE contexts. As suggested in [78], DfX is a way to address the needs of internal customers and manage requirements during a product development process. A study by Möttönen et al. [79] highlighted the importance of a dedicated function to manage the requirements of internal customers. DfX is a means to present stakeholder views using company terms which, in a commensurable form, are meaningful to the systems provider (cf. publication I) [78].
2.1.3 Product management and product managers

Product management plays a key role in software companies. Product managers communicate with stakeholders to ensure that they strive for a common goal defined in the company’s strategy [80]. They are also responsible for managing requirements, release planning and developing a business case. RE and release planning are decision-rich processes that are key to achieving software project goals [4]. In the current competitive ICT market, it is essential for a company to maximise value in each software release implementation. The success of the value creation depends greatly on the ability of product managers to understand the relationships between technical and business aspects. The task is not an easy one and the alignment of product, project and business decisions has been seen as a major challenge in the software industry [81]. Ultimately the success of a product is determined by the skills and competence of its product management [82]. For this reason, it is surprising that despite the acknowledged importance of product managers, they often develop expertise on the changing roles in the company [80].

Earlier studies have indicated that the increasing number of requirements challenges product management in RE and release planning tasks [83]. Studies on the over-scoping of product releases have been identified, causing challenges for RE and leading to project management risks. The consequences of over-scoping may be crucial for a company, since quality issues, project delays and customer dissatisfaction are highlighted as negative effects [84]. It has been suggested that the negative effects of over-scoping could be mitigated by employing agile practices [85]. However, according to Bjarnason et al., in spite of applying an agile RE approach, product management can still have issues with over-scoping [86], [87]. Clearly, there is a need to develop new ways to cope with the constant inflow of requirements and manage requirement databases [54].

2.1.4 Software architecture and software architects

Software architecture forms the foundation for creating a software system. The objective of software architecture design is to offer a cohesive view of the system and its behaviour [88]. Software architecture design is inherently complex, and its complexity continues to increase due to the necessity of addressing various stakeholder concerns [89] that may conflict with the goals of software system development. Software architecture design is primarily the software architect’s responsibility [90]; however, the active involvement of stakeholders, such as
software developers, product managers and customers, is crucial to improve the understanding of the criteria that the architecture has to meet. Software architecture design decisions are mainly based on architecturally significant requirements that are critical for shaping systems architecture [91]. Architecturally significant requirements can be either functional or non-functional requirements, but in practice, most are non-functional requirements due to their ability to affect the whole software system [92].

In a way, software architects are the end users of the information that is gathered during requirements elicitation and further elaborated during other RE phases. Thus, RE and software architecture design are closely connected, and prioritising requirements can affect the architecture design; and, vice versa, the choices made with regard to implementation alternatives can affect the possibility of satisfying the requirements [93], [94]. Including architectural views in requirement specifications has been suggested as a way to improve the understanding of the resistance to change and the consistency, comparability and feasibility of the requirements [95]. According to [96] by including software architecture considerations in RE, analysts could elicit 10% more architecturally significant requirements, 10% more “important” requirements, 7% more crosscutting requirements and more implementation and interoperability requirements in general.

2.1.5 Requirement screening principles and process

The key principle of a requirement screening process is to quickly assess each new requirement to decide whether or not it is worth spending more time on [51]. The goal of establishing an efficient screening process is to ensure that resources are used from the very beginning of RE for the requirements that will eventually be implemented [37]. A corresponding approach to screening is proposed in [50] and referred to as “requirement triage”. The principles of requirement triage have been adopted from the medical field to quickly categorise requirements into three classes based on their criticality.

The underlying theory for screening and triage relies on requirement prioritisation, as typically the number of requirements is greater than what can be implemented in a single release [56]. The decision-making task is not a trivial one since (especially at the beginning) RE requirements are often incomplete, and different stakeholders have divergent priorities and even conflicting expectations [97].
2.1.6 Needs, features and requirements

It has been argued in [98] that the generic term “requirement” should be split into three categories – needs, features and requirements – of which the needs belong to the problem domain and the features and requirements to the solution domain, as illustrated in Fig. 2. Needs are “raw requirements” that describe the needs of customers, users or other stakeholders or the needs derived from standards or other relevant input documentation in terms of the problem domain – that is, in the language of the user or customer [98]. A raw requirement is one that has not yet been processed as a well-formed requirement [44]. Alongside the “raw” term, the literature acknowledges the following requirements as well: “goal-level”, “problem domain-level”, “solution domain-level” and “design-level” requirements [30]. A typical approach is to categorise requirements as functional and non-functional, which are also referred to as quality attributes [3]. Furthermore, requirements may be considered to address either products or processes [31], [99]. These definitions are based on the following principles: 1) the source of the requirement and 2) the target of the requirement or the abstraction level of the requirement.

It is easy for a reader to get lost in the jungle of terminology, as the terms listed above imply a tacit knowledge of domains. In the context of this research, it is not feasible to use “need” and “raw requirement” for those inquiries that customers present to software intensive system developers for two main reasons. First, needs in general exist without engagement with a software intensive system developer, but when a need is expressed, it is recorded in a solution domain documentation system and becomes an entity that has to be handled and answered. Second, (raw) requirements cause confusion, since the term “requirement” alone in the current literature and practice is used as a synonym for a problem domain need, a solution domain recorded need, a requirement specification and so on. Therefore, in this research, the term request is used to indicate those inquiries that stakeholders formally present to software intensive system developers, like using tools to communicate requests to be analysed and solved by system providers.
In addition to introducing the term request, the order of features and requirements in Fig. 2 are reversed in Fig. 3. The rationale for the change is that once requests are communicated to a system provider, the task of analysis begins. The purpose of the analysis process is to understand what is actually required — that is, what the requirements are. While requests from various sources are analysed, the interdependencies between the requirements derived may be identified. This enables the grouping of requirements into logical sets called features. Therefore, by implementing one feature, it is possible to satisfy one or more customer need(s). Based on this reasoning, Fig. 2 is updated as depicted in Fig. 3.
Fig. 3. Needs, request, requirements and features.
3 Research problem and research questions

This thesis is based on research activities carried out in two case companies. Case company A is a large globally operating systems provider, and case company B is a medium-sized enterprise providing software and services for companies and consumers. The RE practices and activities utilised in case company A follow generally recognised RE phases [99]. However, in a large organisation, processes are not as simple as they are typically presented in the literature [31], [30], but there are several parallel, continuous and interconnected processes. In case company A requirements development and management activities actually include a series of parallel and continuous processes that are carried out by several different disciplines. There are five units in case company B with some development dependencies between the units, but typically dependencies are only between two units at a time. Within-unit synchronisation between teams working in different locations is often needed.

The scope of this research is a requirement screening process that covers requirements management and development activities from analysing requests up to feature implementation, thus the requirement screening process is a part of the general RE process. The requirement screening process also includes a feature specification task. During the requirement screening process, the gathered requests are analysed, and those considered valuable and reasonably business savvy from the perspective of stakeholders will be selected and bundled into features. The requirement screening process has been studied by interviewing several stakeholders representing different viewpoints. The information needs for requirements selection have been defined through the data gathered from RE analysts and product managers. The requirements elicitation phase provides information for the requirement screening process. The quality and content of the data have been studied from customer relations and DfX concept viewpoints. The requirement screening process can be seen as a decision-making, filtering or selection process [4], [100], [51].

To build satisfactory requirements architectures, the gathered data from stakeholders must meet the needs of the subsequent phases of systems development. Furthermore, the data must be gathered systematically from all stakeholders to enable commensurable information content for decision making. The research lies in the realm of RE and focuses on requirements analysis and management. Case company A operates in VLSRE and case company B in LSRE modes. While case company A has development teams and other functions distributed globally, case
company B develops its products in three countries. Distributed development led to challenges in coordinating work across sites, spanning national, linguistic and cultural obstacles [101]. The research addresses communication challenges regarding requirements analysis and management by developing a requirements architecting method for VLSRE [42]. The requirements architecting method includes a dynamic requirement template, which provides structured information content for requests, requirements and features analysed and managed in the requirement screening process. Alongside the dynamic requirement template a new way of working, including guidance, was proposed. A prototype tool was created to validate the developed requirements architecting method in case company A.

**Research problem:** The research problem in this study was to determine what information should be gathered in requirements elicitation and how that information should be documented to enable the efficient filtering of valuable requests and to provide a reliable basis for selecting features to be developed in the context of VLSRE and distributed development. In this context, a valuable requirement is defined as one that most likely provides added value for customers and a positive business impact for a solution provider.

**Research questions:**
- RQ1 What are the challenges faced by key stakeholders in VLSRE?
- RQ2 What is requirements architecture in VLSRE?
- RQ3 What are the means of overcoming the identified challenges?
- RQ4 How can requirements management practices be improved in a VLSRE context?
- RQ5 How can decision making be supported in a VLSRE context?

The first research question seeks to identify and describe the challenges stakeholders typically face in a VLSRE context. The second research question aims to define requirements architecture in a VLSRE context. The objective of the third research question is to identify applicable ways to address the identified challenges. The fourth question explores the requirement management practices that can be deployed in industry. Finally, the fifth question aims to identify means to support decision making in a VLSRE context.
4 Research approach and methods

Two methodological approaches were applied in this research: case study and action research. The purpose of the case studies conducted in different companies was to explore phenomena in an industrial context. Most of the cases were carried out in a VLSRE setting; in case company A, more precisely, the requirement screening process and related stakeholders were addressed. The action research method was utilised for developing a requirement architecting method as a solution to overcome the challenges identified in case company A. To assess the validity of the developed requirements architecting method, a prototype tool was created as well. In addition to these, a complementary case study was conducted in an LSRE setting in case company B to understand software architecture design challenges in greater detail.

In the chosen approach, the state-of-the-art literature was combined and compared with the industrial needs identified in the case companies. The action research cycles, case studies and publications I–V included in this thesis are depicted in Fig. 4. This research contributes to the scientific body of knowledge in the form of a developed requirements architecting method.

![Fig. 4. The timeline of the conducted case studies.](image-url)
4.1 Action research and case studies in software engineering

A systematic literature review [102] of nine major software engineering journals and conferences resulted in only 16 studies that could be classified as action research. Many papers did not describe the details of their data collection, the length of the study or the number of action research cycles. However, the authors claim that most of the studies included only one cycle [102]. For example, a study [103] on the knowledge of software process improvements obtained during five software development projects was reported as longitudinal action research. However, in [103] the duration of each project was between six and nine weeks. In two of the publications, [104] and [105], the action research method was used, and the latter [105] claimed that action research was considered a type of case study. Another study, which lasted 19 months, was published as a longitudinal case study [106].

Multiple case studies and the action research method are related. One cycle of action research may be conducted as a case study, so the software engineering case study guidelines are applicable to the realisation of action research cycles, especially the research part of the cycle [107]. Furthermore, in software engineering research, case studies often aim for improvement, as in action research [108]. The diversity of action research methods in information systems [109] allows for my research to be positioned in the field. The characteristics of action research that resemble my research efforts varied, case by case. The research process model used was iterative and linear, and the structure was fluid. A fluid structure means that, for example, activities may be defined quite loosely and many actions may be carried out simultaneously [109].

In most of the cases, my involvement has been facilitative. The primary goals of this research can be looked at from different viewpoints: the case companies seek organisational development, and, as a researcher, I aim for individual learning and scientific knowledge. Based on the characteristics of information systems (IS) action research [109], my research best fulfils the criteria of soft systems methodology and clinical field study. However, it also shares some similarities with participant observation, action learning and process consultant research.

Each case study was designed, conducted and reported by applying guidelines for reporting case studies in software engineering [107]. The design phase included defining the case, its objectives and the research questions to be answered. In addition, during the design phase, the theoretical background, research methods and sources for collecting information were chosen. The preparation phase included the development of a data collection instrument (e.g. questionnaire) for
gathering empirical data, for example, through interviews. In the preparation phase, decisions were made regarding who should be interviewed, for example, and an agreement was made on procedures and schedules. During the collection phase, the data were gathered according to the plan and by utilising the data collection instrument developed. The collected data may be categorised as first-degree data, such as transcripts of interviews; second-degree data, such as meeting records that were not recorded specifically for the case; and third-degree data, such as company documentation and literature. The analysis phase included a comparison of empirical evidence to the existing literature and archival material, the latter of which was provided by the case companies and by interviewees. The results of the cases have been reported for two main audiences: case companies and academics. The transmission of information to the case companies was achieved by organising seminars to present and discuss the results and by writing company (confidential) technical reports. The academic audience was reached through writing and publishing scientific papers in international conferences and journals.

There are two other research approaches, design research [110] and action design research [111], that are also motivated by business needs and resemble the chosen approach. Both the design science research and action design research aim to solve a practical problem identified in practice, as is the case with this research. For example, in design science the purpose is to design and produce an artefact that solves a scoped (pre-defined) practical problem. At the beginning of this longitudinal action research, a well-scoped practical problem could not be defined. Indeed, exploratory case studies were conducted to define challenges in detail to develop solutions for a complex and vague context. The action design research method was developed to respond to the shortcomings identified in design research and action research. Action design research has an objective similar to that of design research, but it also includes a continuous evaluation aspect similar to action research. It also predefines the roles for each participant or group, and the development effort is typically performed by practitioners who represent the same company as the end users. Both design research and action design research aim to generalise the results for building theories in IS. Even though one of the objectives of this research was to develop a solution to overcome practical problems identified in case company A, each individual case was designed as a standalone case study. Furthermore, the development effort of the solution was a joint effort with case company representatives and researchers. Finally, this research did not intend to create a theory, even though the generalisation possibilities have been considered within each case study conducted. Due to these fundamental reasons, this research
approach is neither design research nor action design research but instead a combination of case studies and action research.

4.2 Description of the primary case companies

For this thesis, research was conducted in two primary case companies: A and B. Case company A is a business-to-business operating enterprise in the ICT industry. The company provides an extensive portfolio of software intensive systems to the global market, and its customers number in the hundreds. The customers are an important source of requirements. The system development is done in several parallel and interconnected processes. Some of the processes are more hardware oriented, utilising traditional engineering approaches, while the rest are more related to software. The development is done in releases, which include both new features and enhancements of the existing ones. Requirements engineering in case company A easily exceeds the definition of VLSRE [42].

Case company B has more than 900 employees in 25 offices worldwide. The company carries out product development activities mainly in three countries. The company is structured into business lines that operate as independent entities, with profit and loss responsibilities. Each business line has a flat hierarchy of teams that are mostly organised based on projects. Individual teams are typically self-organising and consist of four to eight people. While teams are free to operate according to their own agendas, they might have to interact and align with other teams, depending on the nature of the project. The teams utilise agile development principles by tailoring them to fit their needs. Some teams have their own architect or Scrum master, but this is not the case for every team. The scale of the requirement engineering in case company B meets the criteria of LSRE [42].

4.3 The longitudinal action research conducted

The phenomena to be investigated in case company A were complex due to the number of stakeholders involved and the interconnected processes and multidimensional organisational structure. Throughout the research, several parallel and consecutive case studies were carried out. An underlying goal of each case was to improve the requirement screening process. The detailed objectives of each case study were agreed upon during planning. Although research on various stakeholders has followed the principles of case studies [107], the overall research approach taken was fundamentally action research [112].
The activities are described in three major phases. **Phase 1 – Diagnosing** includes four cases: Design for eXcellence (DfX), Design for testability (DfT), requirement screening (RS) and order delivery (OD). These stakeholders were engaged in research as they had or were presumed to have been directly involved with the RS process. Then, **Phase 2 – Action research cycles** explains in detail the incremental development of the dynamic requirement template, a new way of working and a prototype tool, which was developed to validate the proposed improvements. **Phase 3 – Continuous deployment** includes three major activities: the integration of new ways of working in case company A and their internal product management tool improvement as well as conducting complementary case studies. The phases and their relationships are illustrated in Fig. 5. In this figure, **Case 4: Order delivery** in Phase 1 is greyed out, as the conducted study showed that actually it did not have a direct connection with the RS process. In addition, product management tool improvement in Phase 3 is greyed out, since the researchers were not involved in the company’s internal improvement project. However, these activities are included in the figure as they help to illustrate the research context in its entirety and are important parts of the whole. The subsections describe the research activities carried out in each of the three major phases. Phases 1 and 2 are part of the research effort carried out under the ITEA ITEI\(^2\) project, while Phase 3 was started during ITEI and then continued under the ITEA AMALTHEA\(^3\) project. Throughout the case studies, relevant literature relating to the empirical evidence was studied. The main goal of Phase 1 was to identify and understand the challenges of requirement screening. Phase 2 focused on incrementally developing solutions to the identified challenges, and in Phase 3 the solutions were deployed in the organisation.

---

\(^2\) ITEI: Information Technologies supporting the Execution of Innovation Projects, https://itea3.org/project/itei.html

\(^3\) AMALTHEA: Model Based Open Source Development Environment for Automotive Multi Core Systems, https://itea3.org/project/amalthea.html
Fig. 5. The phases of the action research conducted.
The composition of the action research team is illustrated in Fig 6. The University of Oulu (University A) and case company A were involved in Phase 1. Members from Tampere University of Technology (University B) joined the team during Phase 2. The number of professors, researchers, students and experts involved varied throughout the study. The utilisation of action research yields findings and new research themes as results [112]. Based on the characteristics of action research described in [109], the process model was iterative, and the structure was fluid. The company strove for organisational improvements in terms of its practices and systems/tools, while the researchers’ primary aim was to create new knowledge in the area of RE. The researchers worked in close collaboration with the company. Some of the tasks (including immediate problem solving) were assigned to the researchers, and others were assigned to the practitioners. The team members were physically located in their “home” institutions, but they regularly met face-to-face and through an e-meeting system. Matters were also discussed over the phone and through e-mail as needed.

![Fig. 6. The composition of action research team.](image)
4.3.1 Phase 1 – Diagnosing

The goal of the diagnosing phase was to determine, understand and describe the problem [109]. Diagnosing in this research was performed by carrying out four different cases using a case study research method. Of these, three provided the most valuable results related to the development of a dynamic requirement template. Data from Case 1: DfX were used in writing publications I and II. Data from Case 2: RS and Case 3: DfT were utilised when developing solutions such as a dynamic requirement template. The fourth, Case 4: Order delivery, addressed a case company internal stakeholder and one of the processes that was presumed to be connected with the RS process. However, this process was eventually found to not be directly connected with the requirement screening process – that is, it did not produce requests or set constraints for the RS process. The cases and number of interviews per case are listed in Table 1. The purpose of the case studies was to understand the viewpoints of different stakeholders regarding requirement screening practices and the challenges they faced with regard to the requirement screening process. Different analysis procedures were applied in the conducted case studies. For example, in Case 1: DfX the data were analysed in two steps. At first, the data were analysed per each DfX discipline to understand, for example, the requirements flow, the stakeholders and the challenges from the viewpoint of each discipline. Then the data were analysed from the viewpoint of different functions, such as processes, services and R&D. The first case study lasted six months and was followed by three concurrent case studies lasting six months. Altogether, phase 1 lasted one year. At this point, it became clear that one action research cycle was not sufficient for developing a solution to the challenges identified. Of the papers included in this thesis, publication I and also partially publication II are based on this phase.

4.3.2 Phase 2 – Action research cycles

In phase 2, the action research cycles were carried out. The goal of the second phase was to iteratively develop a model (the dynamic requirement template) for the challenges identified. The overall duration of this phase was two years.
Table 2. The list of cases and conducted interviews in the diagnosing phase.

<table>
<thead>
<tr>
<th>Cases</th>
<th>Case details</th>
<th>Interviewees</th>
<th>The average experience of participants</th>
<th>Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1: Design for eXcellence)</td>
<td></td>
<td>20</td>
<td>10 years</td>
<td>Transcribed</td>
</tr>
<tr>
<td>Case 2: Requirement screening</td>
<td></td>
<td>9</td>
<td>12 years</td>
<td>Transcribed</td>
</tr>
<tr>
<td>Case 3: Design for testability</td>
<td></td>
<td>14</td>
<td>13 years</td>
<td>Transcribed</td>
</tr>
<tr>
<td>Case 4: Order handling</td>
<td></td>
<td>13</td>
<td>9 years</td>
<td>Transcribed</td>
</tr>
<tr>
<td>Action research team</td>
<td>45 meetings/</td>
<td>N/A</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td>workshops</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Cycle 1**

*Action planning.* At first, it was decided that the lessons learned from the diagnosing phase would be synthesised to form an overview of the requirement documentation practices of the company. The main data sources of this phase were interview transcriptions, meeting notes and example documents provided by case company A. The scientific literature and the results of former studies conducted internally in case company A were also studied.

*Action taking.* The researchers used material gathered during the action planning phase to create the first version of the dynamic requirement template as a Microsoft Excel file. At this point, the data in the template were structured into five major entities: basic data, business data, stakeholders, technical data and constraining factors. Each entity included a number of attributes that described requirement details. Examples of these attributes are “request short description” and “acceptance criteria”. The advancement of the dynamic requirement template was followed in 10 biweekly meetings with case company A representatives. In addition, three workshops were organised.

*Evaluating.* The dynamic requirement template was evaluated through qualitative interviews. Five customer relations experts (cf. Table 2) participated in the interviews. The experts were asked to assess each requirement attribute using specified numerical values. The purpose of the assessment was to learn a) whether it would make sense to record customer requirements with the given attributes, b) whether it was possible to provide the information indicated as requirement attributes and c) if so, with what effort. The interview data were analysed from the viewpoint of the process (defined and actual) as well as the information content
elicited from customers. In addition to collecting feedback about the dynamic requirement template, the interviews provided valuable insight into customer requirements elicitation practices, the information that is collected and how this information is documented.

**Specifying learning.** The data from the evaluation were analysed and presented by the researchers in a workshop for the employees of case company A. It was understood that the dynamic requirement template was a good starting point but that it could not encompass the diversity of the incoming requirements. The managers of the action research team decided that the further development of the dynamic requirement template should be divided into several research threads to gain a more in-depth understanding of the data content to be included in the template and to divide the workload between several researchers. The threads were business reasoning, technical details, stakeholder analysis and work support. The results of each thread were to be communicated to case company A in weekly meetings and workshops. The duration of the first cycle was six months.

<table>
<thead>
<tr>
<th>Participants</th>
<th>Phase 2 Cycle 1 details</th>
<th>Actions</th>
<th>The average experience of informants</th>
<th>Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Customer relations</td>
<td>5 interviews</td>
<td>3 years</td>
<td>Transcribed</td>
<td></td>
</tr>
<tr>
<td>Action research team</td>
<td>13 meetings/ workshops</td>
<td>N/A</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

**Cycle 2**

**Action planning.** The plan for the second cycle was to continue developing the dynamic requirement template through the identified research threads. The synthesis of individual threads would be combined into the second version of the template. An overall solution for improving requirement screening practices in case company A would include the development of context-dependent guidance as well. The second version of the template would be assessed using the requirements provided by a number of DfX disciplines.

**Action taking.** The development of the second version of the template was continued in individual interviews. The researchers familiarised themselves with company A’s documentation regarding business analyses, technical analyses, process definitions and requirement descriptions. In addition, literature reviews were conducted on stakeholder analysis and workflow, for example. Weekly meetings with case company A were continued.
Evaluating. Representatives of two DfX disciplines (DfE: Design for Environment and DfT: Design for Testability) participated in the evaluation of the second version of the dynamic requirement template. One representative provided high-level information about requirements data, while the other provided a well-defined example of a requirement description. The task was performed in the series of workshops facilitated by the author. Audio recordings from the workshops were used to list comments the participants made about data fields, provided written guidance and the feasibility of the template. Observations during the workshops were written as notes, and these were combined with the data from the audio recordings to improve the dynamic requirement template. This cycle included six interviews that aimed to study DfX requirements definition practices. The data from these interviews were cross-analysed with company process documentation as well as with data and findings from Case 1. At the end of the second cycle, four product management representatives participated in qualitative interviews. The purpose of the interviews was to discover the challenges faced by product managers regarding the requirement screening process. The focus was on information and decision-making aspects of requirements screening. A summary of the actions in the second cycle is provided in Table 3. The NVivo software tool was utilised for analysing data from product management interviews. The transcribed interview files were imported in NVivo using the auto-coding option, which was set to follow the interview questionnaire structure. These pre-structured data were then coded by themes, such as process phases and challenges.

Table 4. The summary of actions in Cycle 2.

<table>
<thead>
<tr>
<th>Participants</th>
<th>Phase 2 Cycle 2 details</th>
<th>Records</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Interviewees</td>
<td>The average experience of participants</td>
</tr>
<tr>
<td>Evaluation Design for</td>
<td>3 workshops</td>
<td>10 years</td>
</tr>
<tr>
<td>Environment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evaluation Design for</td>
<td>3 workshops</td>
<td>2 years</td>
</tr>
<tr>
<td>Testability</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DfX process</td>
<td>6 interviews</td>
<td>11 years</td>
</tr>
<tr>
<td>Product managers</td>
<td>4 interviews</td>
<td>11 years</td>
</tr>
<tr>
<td>Action research team</td>
<td>13 meetings/</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>workshops</td>
<td></td>
</tr>
</tbody>
</table>

4 http://www.qrinternational.com/products_nvivo.aspx
Specifying learning. Paper pilots of the second version of the dynamic requirement template provided the proof of concept of the applicability of the template and introduced new ways of working. Interviews with product managers showed that the information in the template met the needs of the requirement screening process. However, it was understood that appropriate guidance must be provided as well. In addition, to enhance technology transfer, the validation of the dynamic requirement template and new way of working needed to be performed in a more realistic manner. Therefore, a conceptual model for the new requirement screening practices was created and used to develop a prototype tool, which would be used for industrial validation of the proposed improvements. Scientific articles were written based on the findings of the third cycle, including publication III and part of publication IV. The duration of the second cycle was one and a half years.

Cycle 3

Action planning. The plan was to divide the validation of the dynamic requirement template and the development of the prototype tool into three steps based on defined RE items – request, requirement and feature – which are in line with requirement screening process decision points DP0, DP2 and DP3. In other words, the third cycle was a small action research effort in itself. While the first part was tested with the practitioners, the development of the second part was done in parallel. Besides developing the prototype and fixing bugs, the guidance for using the template with context-aware rationale for new ways of working was improved.

Action taking. The researchers involved in this part came from the University of Oulu and Tampere University of Technology. Weekly meetings took place between the researchers and representatives of case company A. The author was responsible for observing and guiding the practitioners during prototype testing. Throughout the research, the researchers from the University of Oulu developed new ways of working through the research threads, taking into account the notes and feedback from the observations. The researchers from Tampere University of Technology were responsible for modelling and coding tasks of the prototype tool.

Evaluating. Representatives from five DfX disciplines (DfE [2 groups], DfT, DfP: Design for Packaging, DfA: Design for Assembly) participated in the validation of the requirements architectoring method using the prototype tool, which was developed in three increments. All three steps began with a kick-off meeting with the practitioners. In these meetings, the data to be recorded in each step were introduced, and the prototype tool increment was presented. The testing itself was
done during two or three meetings with representatives from each DfX discipline. The practitioners worked independently, and the author mainly observed the flow of work and took notes about necessary improvements and bugs in the prototype, for example. The meetings were mostly face-to-face, but one of the representatives participated via an e-meeting system. The researchers from the University of Oulu and Tampere University of Technology met on an as-needed basis but at least once a week to share gathered experiences and to decide upon actions for the next increments. At the end of each step, the author presented feedback regarding the lessons learned to all the DfX disciplines that were established in company A during the organisation-level monthly meetings. Feedback was also received in these meetings and the comments were taken into account to revise solution proposals. The summary of the actions in Cycle 3 is provided in Table 4.

**Specifying learning.** At the end of the third cycle, the managers of case company A decided that the developed practice was sufficient and that a prototype should be developed for an industrial-scale tool that would be part of a designed work support environment. It was agreed that further development of the prototype would be done internally within case company A. Scientific articles were written based on the lessons learned in the pilots regarding the developed practice and tool [113], [114], [115]. At the end of this six-month cycle, case company A started progressively deploying the practice and tool. The extensive empirical data gathered during the research collaboration made it possible to perform several analyses from different viewpoints.

### 4.3.3 Phase 3 – Continuous deployment

In Phase 3, the researchers’ involvement in case company A gradually decreased. However, co-operation with case company A continued throughout the duration of the case studies conducted in other companies under international R&D projects. After the proof of concept had been accepted and the dynamic requirement template had been validated in prototype tool supported pilots, continuous deployment began. In other words, case company A started an internal improvement project to implement applicable enhancements to the globally used product management tool to support decision making. Together with product management tool development, the training for the new practice was implemented unit by unit and team by team. In addition, the group of researchers continued working on the prototype tool, which was eventually developed for use as a commercial tool.
During the continuous deployment phase in case company A, a case study addressing software architecture challenges was conducted. The results of the case study have been published in publication IV. Besides case company A, two other companies and a software consultant participated in the interviews. The other companies are small and medium-sized enterprises (SMEs). The first focuses on the full development cycle and maintenance of complex software solutions, and the other is oriented toward adapting various tool environments to the specific needs of the client organisations. The software consultant engaged has an in-depth practical knowledge of software architectures. The interviews done in the SME companies and with the software consultant expanded the view on software architecture practices in industry. The thematic analysis of the interview data was done in a three-step process. At first explicitly mentioned challenges and known issues in software architecture design were highlighted. These highlighted parts were re-coded and organised into categories based on commonalities in the underlying causes of the challenges. Lastly, the findings from the data were interpreted based on the researchers’ experience in field.

Table 5. The summary of actions in Cycle 3.

<table>
<thead>
<tr>
<th>Evaluation</th>
<th>Steps and details</th>
<th>Step 1</th>
<th>Step 2</th>
<th>Step 3</th>
<th>Average experience of informants</th>
<th>Records</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design for Environment 1</td>
<td>4 workshops</td>
<td>3 workshops</td>
<td>2 workshops</td>
<td>11 years</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Design for Environment 2</td>
<td>2 workshops</td>
<td>2 workshops</td>
<td>2 workshops</td>
<td>10 years</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Design for Testability</td>
<td>1 workshop</td>
<td>3 workshops</td>
<td>2 workshops</td>
<td>3 years</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Design for Testability</td>
<td>3 workshops</td>
<td>2 workshops</td>
<td>2 workshops</td>
<td>20 years</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Design for Assembly</td>
<td>3 workshops</td>
<td>2 workshops</td>
<td>2 workshops</td>
<td>20 years</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>Design for eXcellence</td>
<td>2 meetings</td>
<td>1 meeting</td>
<td>1 meeting</td>
<td>N/A</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Action research team</td>
<td>8 meetings/ workshops</td>
<td>7 meetings</td>
<td>6 meetings</td>
<td>N/A</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>
4.4 The case study in company B

The motivation for conducting a case study in company B emerged from the aim of obtaining a more comprehensive view of the aspects of software architecture and going beyond VLSRE. The objective of the case study was to understand what kinds of challenges requirements volatility pose for software architecture design in an LSRE setting. The details of the case study are described in publication V. The data collection was done using a semi-structured thematic interview guide [116]. Five units and 15 interviewees from three different company B sites took part in the case study. The case is one of multiple case studies conducted in five different companies in three countries during ITEA MERgE.\(^5\) To carry out a systematic data analysis, NVivo software was utilised. The transcribed audio files were imported in the tool structuring data based on the thematic interview guide used in the interviews. Then the data were re-coded based on the research questions as well as the themes emerging during the analysis process.

4.5 A method to evaluate the rigour and relevance of the research

A method to evaluate the rigor and relevance of technology evaluations was published in [15]. The method was validated by a full systematic literature review of technology evaluations in RE. In the model, aspects related to rigour, such as context, study design and validity, are evaluated based on how the research efforts are described in the reviewed articles. All the aspects are graded as 0 (no description), 0.5 (partial description) or 1 (good description). Relevance, on the other hand, is evaluated by the following aspects: subjects, context, scale and research method. These are graded as 0 (does not contribute to relevance) or 1 (contributes to relevance). In total, 349 articles were evaluated in the systematic literature review. Of those, 116 were graded as 0 with respect to both rigour and relevance. Only six articles received full grades. In nine articles, the applied research approach was action research. None of the articles using action research received grades in the top quarter [15]. This model was utilised in the research to evaluate the relevance of the action research conducted. The results of this assessment are described in Chapter 6.

\(^5\) MERgE: Multi-Concerns Interactions System Engineering, https://itea3.org/project/merge.html
5 Results

The purpose of this chapter is to give an overview of the original papers included in the thesis and present the results of the longitudinal action research conducted. The descriptions of the original papers focus on the motivation for the cases and their main contributions. The relationships between the research questions posed in this thesis and the publications that contribute to answering them are presented in Table 5. Each paper is linked to the research phases and case studies that were described in Chapter 4.

Table 6. The relationship between the publications and research questions.

<table>
<thead>
<tr>
<th>Research question</th>
<th>Paper</th>
<th>Phase (Px): Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>RQ1: What are the challenges faced by key stakeholders in a VLSRE?</td>
<td>PI</td>
<td>P1: Case 1</td>
</tr>
<tr>
<td></td>
<td>PII</td>
<td>P1: Case 1 &amp; P2: Cycle 2 DfX process</td>
</tr>
<tr>
<td></td>
<td>PIII</td>
<td>P2: Cycle 2 PM &amp; P3: Cont. depl. SW architects</td>
</tr>
<tr>
<td></td>
<td>PIV</td>
<td>P3: Case company B study</td>
</tr>
<tr>
<td></td>
<td>PV</td>
<td></td>
</tr>
<tr>
<td>RQ2: What is requirements architecture in VLSRE?</td>
<td>PIII</td>
<td>P2: Cycle 2 Product management</td>
</tr>
<tr>
<td></td>
<td>PIV</td>
<td>P2: Cycle 2 PM &amp; P3: Cont. depl. SW architects</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Thesis</td>
</tr>
<tr>
<td>RQ3: What are the means of overcoming the identified challenges?</td>
<td>PII</td>
<td>P1: Case 1 &amp; P2: Cycle 2 DfX process</td>
</tr>
<tr>
<td></td>
<td>PIV</td>
<td>P2: Cycle 2 PM &amp; P3: Cont. depl. SW architects</td>
</tr>
<tr>
<td></td>
<td>PV</td>
<td>P3: Case company B study</td>
</tr>
<tr>
<td>RQ4: How can requirements management practices be improved in a VLSRE context?</td>
<td>PI</td>
<td>P1: Case 1 &amp; P2: Cycle 2 DfX process</td>
</tr>
<tr>
<td></td>
<td>PII</td>
<td>P1: Case 1 &amp; P2: Cycle 2 DfX process</td>
</tr>
<tr>
<td></td>
<td>PIV</td>
<td>P3: Case company B study</td>
</tr>
<tr>
<td>RQ5: How can decision making be supported in a VLSRE context?</td>
<td></td>
<td>Thesis</td>
</tr>
</tbody>
</table>

5.1 An overview of the original publications included in the thesis

5.1.1 Publication I: A new way to organise DfX in a large organisation

This case was the starting point for the co-operative research on studying the requirement screening process and its stakeholders in case company A. The DfX
disciplines had been identified as important stakeholders of the requirement screening process, and different DfX disciplines were found to be facing challenges related to requirements communication. However, the relationship between DfX processes and the requirement screening process was not initially clear. Furthermore, it was noticed that there were some issues related to the requirements flow in case company A. Thus, it was important to study how DfX was organised in case company A to understand whether DfX approaches provide the means for coping with constant time pressures and changing requirements while also producing high-quality products.

The purpose of Case 1 was to study the concept of DfX from a software engineering viewpoint, since DfX requirements are traditionally related to the hardware side of product development. The purpose was to identify bottlenecks in requirement communication regarding the requirement screening process. Case 1 revealed that DfX is organised in the case company in a unique way (cf. publication I), thereby contributing scientific knowledge from an organisational viewpoint. The findings of Case 1 motivated discussions of managerial views [79], historical aspects [67] and sustainability [70]. An important finding was that aspects related to product development hardware dominated the DfX literature. However, in systems engineering hardware design decisions affect software design and vice versa. Since hardware and software are interconnected, they cannot be built in isolation. Thus, DfX requirements were introduced to software development and the requirement screening process. This case study motivated a research thread related to stakeholder aspects [113] and challenged traditional requirements documentation practices [117]. The main contribution of this paper is its identification of a novel way of organising DfX in the development context, where hardware and software can no longer be treated in isolation. Publication I is one of the results of Phase 1 (diagnosing) of the longitudinal action research. The case contributed to the development of the dynamic requirement template and new ways of working by pointing out directions for further studies and by improving our understanding of the variety of requirements that had to be treated in the requirement screening process.

5.1.2 Publication II: Design for eXcellence in the context of Very Large-Scale Requirements Engineering

The first DfX case focused on understanding the concept of DfX. This was continued with the DfT study, where the objective was to understand issues related
to testing and especially those caused by shortcomings in the elicitation and analysis phases of RE. Based on the results of those cases, it was understood that DfX disciplines have a structured approach to defining their requirements. The results motivated a revisiting of the data to explore whether the RE process in a VLSRE setting actually means a series of RE processes within a large organisation. Besides the original 20 interviews, an additional six interviews with three experts were conducted. The additional interviews were conducted to form a better understanding of the process of formulating DfX requirements in a VLSRE context. The interviews provided a rich picture of DfX practices related to requirements definitions. The data were revisited together with the earlier DfX cases. The findings of these cases are explained in publication II. The additional interviews particularly contributed to depicting the DfX process and formed the basis for the VLSRE framework. Publication II is partially the result of the first phase (diagnosing) and the second phase (action research Cycle 2) of the action research conducted. The contribution of this paper is threefold. It describes a DfX RE process in a VLSRE context, identifies challenges related to the process and proposes a conceptual VLSRE framework to address the identified challenges.

5.1.3 Publication III: Managing constant flow of requirements: Screening challenges in Very Large-Scale Requirements Engineering

The preceding case studies, including the ones published in publications I and II, addressed the stakeholders who set the requirements to be processed in the requirement screening process. In addition, information issues and decision-making aspects were studied in the requirement screening case during the diagnosing phase (cf. Fig. 5). In the second phase of the action research conducted (Cycle 2), an exploratory case study was conducted to understand how product managers cope with the constant high inflow rate of requirements. The topic of scaling the requirements engineering process had been discussed earlier, mainly in the context of market-driven development. This case study resulted in a description of an industrial requirement screening process, which was established to manage requirements in a VLSRE context. In addition, it was possible to identify practical management challenges during the requirement screening process. Based on the product management interviews, the decision making practices became better understood, which in turn helped to structure the business-related information content of a requirement based on the process steps.
5.1.4 Publication IV: Customised choreography and requirement template models as a means for addressing software architects’ challenges

In the third phase, continuous deployment, the research was continued by further studying issues related to the definition of software architecture. The software architects use requirements to design software architecture for a system. Thus, they are one of the key stakeholders of the requirement screening process. The objective of this case study was to understand the challenges software architects encounter during the software development lifecycle of a software architecture design. The goal was also to develop domain-specific models as solution proposals to overcome the challenges identified. The study identified four main challenge categories, all consisting of a number of practical challenges. A dynamic requirement template was developed to overcome challenges related to information mismatches between software architects and stakeholders, such as customer relations people, who often lack technical knowledge. The other domain-specific model was a choreography model developed to address challenges on a systems level. This case contributed to the development of the dynamic requirement template by providing an understanding of how to structure the technical parts of requirements content and how to collect the necessary information from the time of the request initiation.

5.1.5 Publication V: Requirements volatility in software architecture design: An exploratory case study

One of the challenges identified in publication IV related to the fact that software architecture design is created in uncertain conditions, where the available information is not necessarily reliable. During the third phase, a complementary case study was initiated in case company B. The purpose of the exploratory case study was to study the phenomenon of requirements volatility in software architecture design. The objective was to gain a more comprehensive understanding of the interplay of requirements engineering and software architecture design. As a result of this study the factors causing requirements volatility in an LSRE context were revealed. The study also revealed challenges that requirements volatility poses to software architecture design. The paper discusses the means of overcoming the identified challenges based on the existing literature in regard to the empirical data gathered from case company B. This case
study improved our understanding of the consequences of an inadequate requirement engineering process later on in the development process.

5.2 The results of the longitudinal action research

This section discusses the findings from the action research conducted (phases 1–3) and the case studies. Emphasis is placed on the action research cycles and the development of the dynamic requirement template as well as on the VLSRE practices. The cases, explained earlier in this chapter when describing the original publications, are mentioned only briefly.

5.2.1 Findings from Phase 1 – Diagnosing

The first case in Phase 1 addressed the DfX concept. Publication I and largely also publication II are based on Case 1: DfX. Since the details of the case are included in sections 5.1.1 and 5.1.2, which provide overviews for publications I and II, the details are not repeated here.

Case 2: Requirement screening

The purpose of the second case was to understand requirement screening activities. The interview questions addressed four major topics: requirement types; processing and prioritisation; tools and documents; and stakeholders and communication. A few questions specifically addressed aspects of DfX. One aim was to discover how DfX requirements are seen from the perspective of requirement screening experts. It was found that a basic principle of the requirement screening process corresponds to the ideas presented in [51] and [50]. The types of requirements varied from small changes to those leading to a new set of features. The descriptions of the requirements ranged from a few sentences to full technical specifications. Consequently, the requirements were not systematically documented or stored, and appropriate systems-level practices were not established. A global tool intended to support the requirement screening process was found to be inadequate in terms of its usability, structure and functionality. These findings initiated a review of the available requirements management tools and a study on documenting the attributes of requests, requirements and features. These attributes were utilised in the development of a requirements architecting method, namely in defining the content and structure for a dynamic requirement template.
Case 3: Design for Testability

The third case was run in parallel to Case 2. In this case, DfX viewpoints were studied further, focusing on one of the X’s testing. It was found that due to changing requirements (cf. e.g. [3]), information about the criteria against which a system should be tested is lost. Another important finding was that DfX as a concept was not well known by requirement screening experts and that its requirements were typically given low priority. This is because DfX requirements aim for reductions in lifecycle costs, while customer requirements provide revenues in a shorter period of time. Thus, the business logic or basis for defining monetary value is completely different. This discovery led to developing business reasoning data content for a dynamic requirement template.

Fig. 7. The data blocks in the dynamic requirement template and context-dependent work support.

The case studies conducted in Phase 1 demonstrated that the systematic documentation of requirements and communication practices must occur. An overall concept of the solution (i.e. the first draft of the dynamic requirement template structure as it was defined at the end of the diagnosing phase) is illustrated in Fig. 7. It presents the information types that were considered necessary by the interviewed practitioners as well as the researchers’ analysis of the collected data. The most of attributes included in a basic data block concerned request, covering typical information such as short description, initiator and ID. Most of the attributes in business rationale and stakeholder data blocks related to requirement. Examples of these attributes were development costs and description of known stakeholders. The data block of technical details was about feature, and the attributes covered, for example, system description. The constraining factors covered dependencies and conflicts, while context depended guidance gave instructions and examples what kind of information was expected to be provided.
5.2.2 Findings from Phase 2 – Action research cycles

Cycle 1: The first version of the dynamic requirement template

The first version of the dynamic requirement template was a Microsoft Excel file. It was based on company material and inspired by the literature (e.g. [3], [118], [99]). The data covered five major thematic sections, which included specific attributes as described in Section 4.3.2 Phase 2 – Action research cycles. The attributes defined the data fields to be filled in by a request initiator. The attributes were divided into mandatory and optional attributes to ensure that the necessary information was described, which in turn ensured efficient requirement screening. There were 18 mandatory attributes and 20 optional attributes. The attributes were named, and a cell with the data expected to be filled in was provided. In addition, brief explanatory guidance on the type or form of information that was expected was included. An example row, Request ID, which was used in the Cycle 1 evaluation step, is provided in Table 6. Mandatory attributes included, for example, rationale i.e. the reasoning for initiating request, risks and anticipated business value expressed in a three-level scale. The optional attributes covered, for example, complexity, test cases and exceptions. In addition, the template included a column in which to collect a numerical assessment of the mandatory attributes so it could be used during the interviews with customer relations experts.

Table 7. An illustrative table of the dynamic requirement template used at the beginning of Cycle 1.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Insert/select data here</th>
<th>Short guideline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request ID</td>
<td>Name of the request. Unique identification of the request will be automatically created by tool.</td>
<td>The request needs to be uniquely identified. The request should include, for example, a name and an identification number through which the request can be referred. Forward traceability to subsequent phases depends upon each request having a unique identifier. There needs to be a commonly agreed practice for naming requests.</td>
</tr>
</tbody>
</table>

Customer relations interviews

The purpose of the interviews was to learn which type of information is collected when a customer is contacted and what is communicated to the requirement screening process as well as to collect feedback on the template. The interviews
revealed that a global elicitation process defined by the company is generally followed, but it is often altered locally. The major difference was the existence of a person who serves as the main contact for customers. The estimated complexity of the request influences how many details are gathered from customers. The respondents explained that the elicitation process should be able to block requests that do not have any business value. Analysing business factors is a major task requiring careful consideration and should take into account the viewpoints of both the company and its customers. It also defines procedures for the critical requirements of the business.

Table 7 summarises the numerical feedback for the mandatory attributes included in the dynamic requirement template and groups it by topic. Each of the attributes was evaluated using the following legend: 0 (already collected), 1 (not possible to collect), 2 (possible to collect but requires additional effort) and 3 (possible to collect but requires considerably more effort). Basically, the assessment results suggest that the main topics are discussed with customers in the elicitation phase. Only technical data were considered possible to collect but required some additional effort. However, based on the interview data, the discussion does not necessarily possess the level of detail that is needed for the requirement screening process. Overall, the results of these interviews gave positive feedback for the idea of the requirements template and its intended content. Thus, the research was continued through the following threads: business reasoning, technical details, stakeholder analysis and work support.

<table>
<thead>
<tr>
<th>Data type</th>
<th>Modes, observations and notes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic data</td>
<td>0  Basic textual description is provided. However, the level of details depends on the individual.</td>
</tr>
<tr>
<td>Business data</td>
<td>0  Each attribute is addressed when customers are contacted, but the documented details vary greatly.</td>
</tr>
<tr>
<td>Stakeholders</td>
<td>0  Customers are recorded, but other relevant stakeholders are not analysed.</td>
</tr>
<tr>
<td>Technical data</td>
<td>2  Some of the attributes are not currently addressed in customer communication. Information could be collected, but it would require additional effort.</td>
</tr>
</tbody>
</table>

### 5.2.3 Cycle 2: Paper piloting – The proof of concept

The dynamic requirement template was elaborated upon based on the experiences of Cycle 1. At the beginning of Cycle 2, the requirements data were structured in
four columns in one Microsoft Excel spreadsheet. The first column was the name of the data field, the second was to be filled in during the evaluation, the third gave short context-dependent guidance and the fourth was used by the researchers to record their observations during the evaluation and by practitioners to record their remarks. The first part of the template included administrative data like those provided in Table 8. This part also included requirement descriptions and stakeholder and related systems information. A systematic literature review was conducted on stakeholder understanding [119], and its results were used to create a stakeholder analysis section for the dynamic requirement template. The stakeholder analysis section also described the objectives and expected value of a stakeholder. Examples of other data include system modifications, use cases, risks and business-related information.

Table 9. The caption of the template at the beginning of Cycle 2.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Request field value</th>
<th>Short guideline</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request ID</td>
<td>R1</td>
<td>Give a unique and explanatory name for the request.</td>
</tr>
</tbody>
</table>

The rest of the data to be recorded were divided into classes according to how the information should be considered during software development. The first class covered information related to existing agreements between a customer and systems provider. The data in this part addressed customer business reasoning and technical aspects, among other things. The second class made detailed references to the relevant standards or parts of the standards that had to be followed. The third class described the dependencies of certain requirements on other requirements, including what a requirement will affect if implemented. The remaining two classes included directives and obligations. These classes described companywide guidelines, laws and regulations.

The paper piloting was a great opportunity to test the developed template and context aware guidance with real requirements provided by case company A experts. It was discovered that it is feasible to provide the intended data, but the attributes in the Excel file were not divided in an optimal way between three RE items: request, requirement and feature. Thus, based on the experience gained in the paper piloting step, the template was restructured. Major changes were made to the template at this point. Some of the results were published in [115]. The action research team learned that business understanding increases in parallel to the clarification of technical details. Therefore, the business reasoning related to a requirement was divided into sections based on the defined decision points of the
process. To make the first decision (request), product managers had to identify an opportunity for sales. For the second decision (requirement), added value for the case company as well as the customer needed to be clarified. For the third decision (feature), the business reasoning had to be verified. As an example, a part of the business reasoning information section of the template is illustrated in Table 9. Other additions to the data were communicated by a new requirement related to a change to an ongoing product release. In addition to the template, an explanatory guideline document was created for case company A to rationalise the new method of working and to describe the necessary concepts and underlying theories.

The results of the DfX interviews were explained in Section 5.1.2. in the overview of publication II. From the template and practice development point of view, the DfX continuum interviews yielded valuable insights. Due to the findings of this case, the constraining factors and request classes were added to the dynamic requirement template. This case also contributed to the creation of the context-dependent guidance and defined the new way of working.

Cycle 2 also included product management interviews that were conducted to understand the challenges faced by product managers during the requirement screening process. The focus was on the information content of requests, requirements and features as well as decision-making aspects. Publication III is based on this case study. Publication IV is also partially based on the results of these interviews. The overviews of the papers are provided in sections 5.1.3 and 5.1.4. The results of this case helped to create a conceptual model for new requirement practices. The model was used to support the development of the prototype tool.

Table 10. Example of the basic business reasoning data fields.

<table>
<thead>
<tr>
<th>Value identified</th>
<th>Request field value</th>
<th>Short guideline</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Business reasoning</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feature handler defined</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Checklist for early (go-)kill indicators</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Request uniqueness</td>
<td>Yes / No</td>
<td>Is similar request been analysed earlier?</td>
</tr>
<tr>
<td>On product roadmap</td>
<td>Yes / No</td>
<td>Check if the corresponding feature is already on the product roadmap.</td>
</tr>
</tbody>
</table>
5.2.4 Cycle 3: Validating the method using the prototype tool

After the conceptual model had been created, a prototype tool was developed to support the testing of the dynamic requirement template and the proposed new ways of working. The prototype was not the main focus of this research but rather the means to industrially validate the developed requirements architecting method. The development and testing of the prototype was done iteratively in three steps following the logic of the requirements architecting method. The purpose of the first step was to test the request data fields and basic functionality of the prototype. The request information content was designed so that a request initiator would be able to provide the needed data and give enough information to the product managers to enable an efficient screening process. Based on the request data product managers should be able to either reject a request or start analysing it in detail to define the requirements. In the second step, more detailed business data and identified stakeholders were recorded. In this process, each request led to one or more requirements, and their dependencies were managed through the simple list of relations. The requirement information content was intended to provide enough data to gauge the potential of a requirement from a requirement business viewpoint. Based on this knowledge, product managers should be able to make a decision in terms of whether to initiate a detailed technical analysis, which is performed by the software architects. The third step addressed feature information content, which included detailed technical descriptions and explained overlaps and dependencies between requirements. Typically, several requirements were bundled into one feature proposal, which was then included in the planned release content list. The prototype testing was successful, and the new way of working was well received. A decision was made to start a continuous deployment phase where company A started their internal tool development and the group of researchers continued working on the prototype tool. Scientific articles were written based on the lessons learned from the pilots regarding the practices and tool (cf. e.g. [113], [114], [120]).

5.2.5 Findings from Phase 3 – Continuous deployment

During Phase 3, case company A continued to internally improve the globally used product management tool by implementing parts of the requirement architecting method into the tool as they saw fit. Throughout the tool improvement project, the new ways of working were integrated into the requirement screening process. The
enhanced product management tool was designed to support model-driven development. Another round of product management interviews was also conducted to define the functionality of the industrial-scale tool, for example. The impact of novel practices was measured. The researchers reviewed the existing requirement management tools, and the results were published in [121]. The technical aspects of the requirement data were studied further in case company A, and complementary interviews were conducted in other companies, too. The case study on software architectures led to the development of software choreography models c.f. publication IV and motivated the creation of a domain-specific language [122].
6 Discussion

It has been claimed that RE is the most valuable function in systems engineering [27]. Therefore, its key activities, such as decision making [4], requirements selection [51, 50, 100] and tool development [13], have been researched widely over the years. In spite of the available methods and tools, practitioners face challenges when performing their daily tasks (cf. publication I, publication III, [81]). Longitudinal action research was conducted to explore and develop solutions for requirement screening challenges in a VLSRE context. As a result, new research directions, such as stakeholder practices [113], business case analysis [115] and cognitive work support [114], were identified, studied and reported in published articles. The most tangible results were the development of the dynamic requirement template and a novel way of working. The dynamic requirement template and context-dependent guidance were implemented as a prototype tool, which was used to validate the dynamic requirement template and context-dependent guidance in an industrial setting by the intended users.

6.1 The challenges of key stakeholders related to the requirement screening process

The requirement screening process is an important link in the chain of processes in the case company, as its purpose is to process requests from different stakeholders, define requirements and propose a prioritised list of features to be implemented. The key stakeholders are, for example, internal and external customers who express their needs, requirements analysts and other experts who process requests and software engineers who define systems architecture and implement its features. The case company had internally realised that it faced some issues related to its requirement screening process and acknowledged the need to address those issues through research co-operation. To develop solutions, the challenges had to be explored. The case studies conducted provided invaluable insight into the practitioners’ challenges when it came to requirement screening process activities. These can be categorised based on generic RE process phases, such as elicitation, analysis or validation. The challenges may also be looked at from the perspective of necessary RE activities, such as documentation, negotiation or communication. In addition, a decision-making (i.e. prioritisation) or organisation viewpoint may be chosen. In this research, the challenges of the key stakeholders in a requirement
screening process have been analysed from different perspectives to gain a comprehensive understanding about the complex phenomenon.

According to publication II, which studied DfX, the challenges in the elicitation phase were related to unclear and incomplete requirements. This is a widely acknowledged issue in the RE literature, which is also denoted as an ambiguity in the literature [123]. One of the challenges DfX faced was related to creating a compelling argument for internal improvements to be made compared to customer features. It is much easier to show the business value of customer features than that of improvements aiming at long-term savings at a company level. This issue was described in more detail in publication I. According to product managers, one of their major challenges is dealing with an immense number of incoming requests and their incomplete information content, especially with regard to technical details. Due to the lack of information, further communication is required to clarify the actual need. Discussions often take place via informal channels, which tend to lead to shortcomings in requirements’ reliability, quality or traceability. The increasing number and continuous inflow of requirements have been reported to stress RE and release planning tasks [83], especially in the context of market-driven software development. Product management challenges in a VLSRE context were discussed in publication III. The lack of information in the early stages of a requirement’s lifecycle also hinders software architecture definition work, as discussed in publications IV and V.

The analysis task may be divided into subcategories such as technical and business analysis. In the case company, first an initial screening took place. This was a quick assessment of the business potential of a request. The biggest challenge in the initial screening was again a lack of necessary information about not only the request itself but also complementary data, for example, information about the market situation to make decisions as fast as intended. A wicked challenge product management faces related to making a crucial decision under heavy time pressure and based on incomplete information has been acknowledged by several researchers [124], [81]. Discussions for the purposes of clarification and negotiation with customers often led to changes in requirements, which is a phenomenon referred to in the literature as requirement uncertainty [125], [126]. The challenges pertain to a more detailed analysis of requests to define requirements, and finally features relate to the defining value, which will be realised years after the initiation of the requests. This applies to managing requirement dependencies across different products or system parts and time constraints – that is, how much time a requirement analyst may invest per
requirement and how many R&D resources may be allocated for technical analysis instead of implementation.

The larger and/or more complex a system becomes, the more stakeholders, requirements and viewpoints have to be considered over the system’s lifecycle. This leads to tasks such as prioritisation, since not all requirements can be fulfilled, and the order of those that are implemented has to be decided. The size and/or complexity also affects the required negotiation task, which has to be carried out to reconcile the conflicting stakeholders’ views or contradictory needs, for example. Defining priorities for different types of requirements, such as business and product requirements [30], is often rather challenging in practice [57]. DfX requirements are typically quite high-level or non-functional, and according to the interviewed practitioners, it was challenging to create compelling business argumentation to support such requirements. Therefore, the priority of these requirements typically was not high.

Similar issues have also been discussed in [127]. The product management study also revealed challenges in prioritisation and negotiation tasks. Regarding prioritisation, it was considered difficult to define the priority for each requirement – that is, what would be the value of different types of requirements received from various sources. Requirements tend to change over time due to internal and external factors, such as product strategies and the market situation. Due to the changes in requirements and their priorities as well as in criteria for setting priorities. Re-prioritising requirements and managing the constant changes is quite time-consuming and laborious work. Regarding negotiation, the biggest issue product management faces is related to communicating to customers decisions regarding turning down their requirements or trying to reach a consensus about alternative solutions. Time is also a crucial factor here, since the longer a customer waits for a decision, the more frustrated he/she will be if the decision is negative. From the DfX viewpoint, the negotiation challenge was about trying to convince product managers to take their requirements into account early enough in the product development process – in other words competing with the requirements of external customers.

In all cases, the practitioners explained that the available tools do not provide adequate support in the daily tasks of documenting, communicating and managing requirements. The challenges are related to performance, the functionalities of the tools provided, the synchronisation needs between processes and organisations and the information content of RE items as well as managing different types of dependencies. Alongside the tools, much essential information was documented in
standard Microsoft Office formats, such as PowerPoint presentations, Word documents and Excel spreadsheets. These “offline” documents were considered outdated, and globally defined templates were typically tailored by each product team and sometimes even by individuals within them. Shortcomings in tools and documentation affect processes from the first contact with a customer to software architecture definition and testing. Issues related to tools and documentation practices have been extensively discussed in the literature [128], [129], [130], [131] and [132].

**Validation** is about confirming that gathered requests and defined requirements are what stakeholders need. During development, new stakeholders become involved, which means that validation is not a one-time effort but an ongoing process to ensure that the system being developed is fit for the purpose. In a VLSRE context, there are several parallel processes that depend on the results of the others. In systems development, hardware teams often prefer traditional systematic development approaches, while software teams tend to work in a more flexible way. Nowadays, companies respond quickly to changing market situations, for example, by updating their product portfolio and business acquisitions. Due to such decisions, different practices have to be adjusted and fitted together. Frequent changes may lead to a situation where roles and responsibilities are not clearly defined. Aspects of organisational issues have been discussed in, for example, [132], [133] and [134]. Moreover, individual teams often tailor globally defined processes to improve their efficiency. Such tailoring may serve its purpose on a team level; however, it often introduces challenges at the unit or company level, as described in [123] and [135]. Synchronisation challenges were discussed in more detail in publications II, III and V.

Two of the case studies conducted addressed software architecture challenges in VLSRE and LSRE contexts. The objective of the first study was to understand the software architecture design challenges in practice and propose domain-specific models to address the identified challenges (publication IV). The second case explored the interplay of RE and software architecture design, defining factors causing requirements volatility, revealing challenges and proposing means to overcome the identified challenges (publication V). Through these two studies, it was possible to form a comprehensive view of a software architecture design process in an industrial setting. The context of the first study was VLSRE, while in the second case it was LSRE. Despite this difference in the scale of the RE, the software architects faced similar challenges to some extent. In both contexts, software architecture design is done in uncertain conditions, without all the
necessary information and sometimes receiving requirements considerably late in the process. Missing information early in the process leads to refinements and changes later in the development process that cause requirements volatility [136].

In both contexts, the challenges related to distributed development teams were identified, even if the number and the size of the teams were smaller in the LSRE context than in the VLSRE context. In distributed development, challenges are related to cultural and communication issues [137], [138], [139], for example. In both cases, software architects saw the lack of face-to-face communication and language issues as representing challenges, while divergent understandings of the same concepts was only identified as a challenge in the VLSRE context. The dependencies between developer teams were identified in the LSRE context, but these dependencies were not considered to critically jeopardise business goals. However, a complex network of dependencies between internal and external stakeholders significantly hampered the development process in the VLSRE context. Frequent changes, such as in requirements, was an important issue in both contexts; however, due to the complexity of VLSRE, the consequences were more widespread in the organisation. Insufficient tool support was also identified as a problem in both studies.

Table 11. Summary of identified challenges.

<table>
<thead>
<tr>
<th>Viewpoint</th>
<th>Challenge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elicitation</td>
<td>• unclear and incomplete requirements, especially regarding technical</td>
</tr>
<tr>
<td></td>
<td>details</td>
</tr>
<tr>
<td></td>
<td>• creating a compelling argument for internal improvements</td>
</tr>
<tr>
<td></td>
<td>• informal communication channels</td>
</tr>
<tr>
<td>Analysis</td>
<td>• lack of information about the market situation</td>
</tr>
<tr>
<td></td>
<td>• requirement uncertainty/changing requirements</td>
</tr>
<tr>
<td></td>
<td>• time constraints</td>
</tr>
<tr>
<td>Prioritisation/</td>
<td>• the number or requirements exceeds what can be implemented</td>
</tr>
<tr>
<td>negotiation</td>
<td>• conflicting stakeholder views</td>
</tr>
<tr>
<td></td>
<td>• incomparable requirements</td>
</tr>
<tr>
<td></td>
<td>• changing requirements or their priorities</td>
</tr>
<tr>
<td>Validation</td>
<td>• synchronisation when utilising several development approaches at a</td>
</tr>
<tr>
<td></td>
<td>time</td>
</tr>
<tr>
<td></td>
<td>• frequent changes, e.g. in organisational structures</td>
</tr>
<tr>
<td></td>
<td>• tailoring processes may benefit a team but challenge a company</td>
</tr>
<tr>
<td>Software architecture</td>
<td>• uncertain conditions, without all the necessary information</td>
</tr>
<tr>
<td>design</td>
<td>• receiving requirements late</td>
</tr>
</tbody>
</table>
### 6.2 Requirement architecture in VLSRE

Product managers and other experts involved in the requirement screening process perform various analysis tasks, but more importantly, they make critical decisions about a product and its features. To make such decisions, a lot of information from internal and external sources is needed. Some types of information that are needed have to do with market situations and trends, technical, human or financial constraints and company strategies. What is a key, is that the correct information must be available on time for the relevant stakeholders in the format that best supports the processing of RE items and facilitates decision making. Appropriate information content, structure and format are here referred to as requirements architecture.

This thesis has explained the development and piloting of the dynamic requirement template, which is a tangible part of requirements architecture in a VLSRE context. The parts related to a technical feasibility analysis in the dynamic requirement template and from the software architects’ viewpoint have been elaborated in more detail in publication IV. Considering software architecture in RE is expected to benefit the consistency, comparability and feasibility of requirements as well as the contemplating alternative design choices [95]. By considering software architecture in the early phases of the development process, it is possible to elicit 10% more architecturally significant requirements, 7% more crosscutting requirements and more implementation and interoperability requirements than if addressed later in the process [96]. In addition, frameworks like software architecture concepts can impose constraints related to testability, for instance, or lead to new systems requirements [140]. Changes made late in the development process cost significantly more than those made in the early phases, like in RE. Therefore, understanding the constraints early enough helps to avoid unnecessary losses later on.

An important issue in the VLSRE context is the constant inflow of requirements, which stresses the requirements engineering and release planning.
tasks of product management [83]. Several studies have reported the overscoping challenges in the context of market-driven software development [84], [37]. However, in publication III, we showed that similar challenges are present in a VLSRE context even if the case company does not operate in a market-driven mode. The case company in this example had globally well-defined processes, the product managers had a clear understanding of the objectives of the process and R&D experts were also involved in the early phases of product development. These aspects had already been suggested to be the means for addressing the issue of overscoping [86]. This finding encouraged the development of a dynamic requirement template that could be used right from the start, when requirements are formulated, thereby providing the proper support for technical and business reasoning to relevant stakeholders. The idea was to address the challenge of there being too much “wrong” information rather than the lack of “right” information.

Requirements architecture in VLSRE includes an updated RS process, a dynamic requirement template that is based on the evolution of RE items from requests through requirements to features and context dependent guidance to support integration of new way of working.

### 6.3 Addressing the challenges identified in the VLSRE context

In this research, the requirement screening process was studied from different viewpoints to identify its related challenges. Based on the findings of these cases, some solutions have been developed and validated in an industrial context. The driving vision has been to tackle information overflow by proposing an information structure for requests, requirements and features in a way that supports the stakeholders involved with requirement screening process tasks. The dynamic requirement template is a tangible result, but other means have been identified as well.

In publication I, we explained what kinds of benefits a company may gain by organising DfX disciplines in a novel way. Contrary to a traditional approach where DfX has been utilised within R&D, case company A manages it in Operations. This solution, where the organisation (which actually pays for the costs of the assembly, maintenance or disposal of the products at the end of the day) takes global responsibility for DfX, is in line with studies supporting total cost model optimisation [141] and those reporting challenges, placing the responsibility for product development squarely on the shoulders of individual product lines [142], [143], [144]. The DfX concept, with its globally defined processes, practices,
guidelines and so on, is a systematic way to address the challenge of internal customers not being recognised on par with external customers [72]. The DfX RE process was explained in greater detail in publication II, which defined a VLSRE framework as well. In the VLSRE framework, internal and external stakeholders establish stakeholder-specific RE processes to define their requirements. These requirements from various sources are fed into a unified information channel for requests. The framework includes the set of receiver processes, taking into account their need for information to process requests further. The key is the unified information channel that structures request data into two major parts: a) the common part for all types of requests and for all stakeholders (senders and receivers) and b) categorised data depending on the receivers’ information needs.

The common challenges software architects face in the VLSRE and LSRE contexts are related to communication, including a lack of appropriate tool support, changing requirements, product or team dependencies and the distributed development setting in general. Accurate and timely communication is essential within and between development teams as well as with customers. Choosing advanced collaborative tools and establishing adequate processes help in communication [139], [137]; however, it is more important to understand what information is necessary and why. One way to address the ambiguity and misinterpretation of concepts in use is to define a customised choreography model with domain-specific concepts. Domain-specific concepts may support the conception and readability of specifications [145], [146]. As the work is done by humans, the performance of the distributed team relies on trust among team members as well as networking within the team [34]. Moreover, it is crucial to maintain a strong and mutually beneficial relationship with customers to manage their changing needs [147]. One way to mitigate the consequences of changing requirements is to elicit gaps in those changes [148] and reuse existing requirements to identify the difference between elicited requirements and actual user needs [149]. Managing requirements’ dependencies can be supported through impact analysis [150]. The means and solution proposals for addressing software architecture challenges were elaborated in more detail in publications IV and V.

A simplified picture of a VLSRE framework is depicted in Fig. 1, where the requirement screening process (industrial implementation of VLSRE) is connected to important stakeholders and immediate processes preceding it or taking place after it. In reality, the number of processes and stakeholders is far greater that can be fitted into one figure or researched in the scope of one thesis.
6.4 Improving requirements management practices in VLSRE

In recent decades, a variety of methods and tools have been developed in RE to support practitioners in software development, which is by far not an easy task [57], [124], [81]. Despite the progress achieved, the importance of RE both in research and practice is recognised, for example, due to the increasing popularity of agile development methods [151]. Moreover, the available methods and tools are typically validated in small- or medium-scale settings [152], or in student projects [96]. The methods and tools validated in such contexts cannot be applied intrinsically in LSRE or VLSRE contexts [42]. Fierce price erosion, severe competition [153] and dynamic markets are the attributes that are globally used to characterise the ICT sector. Such environments force companies to continuously improve their internal efficiency. This is essential in large organisations, which do not emerge overnight but rather evolve from small basic functions through organising, improving and revising themselves repeatedly in order to survive in the changing market. When discussing improvements in companies, they must be considered from organisation, process, method and tool points of view, at the very least.

Throughout this research, the VLSRE context was studied from the viewpoint of different stakeholders. This has allowed improvement proposals to be formulated for organising functions and revising the requirement screening process as well as the development of a requirements architecting method that led to implementing updates into a globally utilised product management tool in case company A. Successful validation of the developed requirements architecting method in industry motivated the group of researchers to work on the prototype, developing a commercial tool which supports a model-based development approach. Publication I demonstrated how a large systems provider successfully organises DfX in Operations instead of R&D, which is the more typical approach. The novel way of organising DfX improves the recognition of internal customers together with external ones, which is a challenge that was pointed out by [72]. More importantly, if products are developed in individual product lines the decisions are driven by direct cost optimisation, leading to inefficiency on a company level and poor customer service. Even if the individual product lines have a common goal, it may turn out that they have contradictory plans in terms of how to achieve those goals [142], [143], [144]. Kaski proposes a total cost model, as business process activities also relate to product architecture and sales volumes [141]. In publication II, a VLSRE framework was presented as a suggestion for overcoming the
identified challenges. Publication V listed factors causing requirements volatility, presented the challenges posed to software architecture design and elaborated proposals to address those challenges. These proposals included choosing appropriate elicitation techniques [154], recording design rationale [155] and identifying architecturally significant requirements [156], [157]. In this thesis, the development of a requirements architecting method has been described, including an updated requirement screening process, which was described in detail in publication III. The thorough analysis of requirement screening process practices allowed the identification of issues related to documenting inquiries from different stakeholders, and the suboptimal structuring of the information content of RE items led to defining the term request and omitting one decision point from the RS process itself.

6.5 Supporting decision making in a VLSRE context

RE is a decision-rich process that plays a critical role in the success of software projects. RE has a dual role; from the organisation point of view, it dictates which requirements will be implemented in products, and from an implementation point of view it is a project activity [4]. The number of requirements proposed per release is typically greater than what can be fitted within the scope of the release [56], and the prioritisation of requirements is done informally in many phases, based on various factors and limited information [57]. Making decisions about which requirements will be included in the forthcoming releases is a major challenge in the software industry [81], since there are a number of stakeholders, resources and technical constraints to be considered in the process [22]. The task becomes even harder if decisions have to be made based on abstract or uncertain information [23], or the number of requirements from which to choose is vast, as in VLSRE. Short development cycles have been proposed to ease prioritisation problems in dynamic markets [158]; however, contradictory results suggest that agile practices actually lead to overscoping challenges [84]. In addition to product managers, software architects must make significant decisions based on requirements [88]. In the “twin peaks” model, RE and software architecture design are developed iteratively and considered equally important [93]. A close connection between RE and software architecture design means that decisions made on requirements can affect software architecture and vice versa [94], [159]. According to [160] software architecture design challenges include, for example, crosscutting and intertwined design decisions, the high cost of change and violations of design rules and constraints.
Putting on a record a software architecture design rationale may be critical during software evolution and maintenance [155].

This thesis has described in detail the longitudinal action research process and case studies conducted to develop a requirements architecting method to support decision making in a VLSRE context. The developed method includes a dynamic requirement template that is used for gathering requests, analysing requirements and forming feature candidates to be developed in forthcoming releases. The feasibility of the developed method was assessed at first through paper piloting. Promising results indicating the possibility of screening out non-valuable requests earlier in the process encouraged the continued development of requirements architecting method further. A prototype tool including context-specific guidelines was developed and used to validate the requirements architecting method in an industrial context. After successful validation of the developed requirements architecting method in case company A, the company continued internal improvement by adding a request element, revising some of the data fields and integrating context-dependent guidelines into the globally used product management tool. The requirements architecting method supports decision making in two main ways. First, it includes a dynamic requirement template, which allows decision makers and information receivers to define what information is necessary for them to make decisions. Examples of these information receivers are product managers and software architects. Secondly, it offers means by which to consider the viewpoints of internal and external customers in a uniform way. The advantage the developed template offers compared with the ones such as Volere [43] and IEEE [44] is due to the well-defined evolvement of RE items, where information content matches the receiver’s need and provides context-aware guidance for the information provider.

6.6 Validity evaluation

The validity of the research had to be considered from the very beginning of the study. Aspects of validity relate to construct, internal and external validity and reliability [107]. Some of the intermediate results were reported earlier, including a discussion on validity [114], [115], [79], [113]. In addition to these, each of publications I–V included in this thesis has been published as a case study and includes a discussion of validity. Thus, the main focus here is on the validity of the action research cycles described in the thesis. The purpose of Phase 2 was to develop solutions for the issues identified in Phase 1.
Threats to construct validity relate to the extent to which subjects and researchers understand the topics under discussion in the same way. The years of close collaboration between the case company and the researchers mitigated any threats to the construct validity. In addition, the interview questions used in each set of interviews were always prepared by several researchers and reviewed by representatives of the case company. One pilot interview was done prior to each round of interviews to ensure that the questions yielded the information that the researchers intended to gather.

According to [107], internal validity only has to be addressed in explanatory and causal studies, not in the case of descriptive and exploratory ones. The aim of the case studies conducted here was descriptive and exploratory, therefore internal validity is not a concern.

Regarding external validity, the results of this type of industrial study always have limited generalisability. However, the lessons learned can be utilised by companies experiencing similar challenges to adjust their practices to meet their needs. For example, the logic of presented a requirement template, which may be tailored by any company to correspond to their own processes and their stakeholders’ needs.

This study provides results that are useful for researchers since it presents a detailed description of how to conduct industrial co-operation through longitudinal action research, which is not widely discussed in the software engineering domain.

Reliability relates to the extent to which the results depend on a specific researcher. Researcher bias was mitigated through review processes and feedback loops throughout the cycles. For example, all data collection instruments were reviewed by professors and practitioners, and a pilot interview was carried out to test each instrument prior to conducting the actual interviews. The large number of academics and practitioners engaged during this research mitigated the bias caused by an individual researcher because it was always possible to reflect on the findings with other researchers. In addition, the results were reported to the case companies in the form of technical reports, and these were also presented in workshops. The practitioners involved gave feedback both on the reports and the presentations. The feedback received was used to verify the accuracy of the results [161]. The generalisability of results is typically weak in case studies. In theory, it is possible that another researcher could produce the same results and draw the same conclusions if he or she used the same methods and tools. However, one should take into account differences in time and context. Thus, even if the subjects were exactly the same in a replication of these case studies, their knowledge would have
changed over time. In addition, by its very definition, the primary goal of action research is to improve a situation in a case company, so the context of the subjects would include factors that had changed.

6.7 Evaluating the relevance of the conducted research

Ivarsson and Gorschek [15] developed and tested a model to analyse the rigour and relevance of published empirical research. They found that the majority of reported research is neither industrially relevant nor rigorous. This section assesses the relevance of the Phase 2 action research cycles against the evaluation method presented in [15]. In the evaluation, the six articles that received the highest scores in [15] are used as reference papers [162], [163], [164], [165], [166], [167].

According to the evaluation method, the research either contributes (1) or does not contribute (0) to relevance regarding four aspects: subjects, context, scale and research method. Thus, the maximum points a study may accrue is 4.

The case studies published in the reference papers were conducted within companies or by using company data, and the subjects were industry professionals. In this research, action research cycles were conducted in case company A. The archival data utilised included, for example, process descriptions, decision criteria and process artefacts. The subjects in Phase 2 were the intended users, who used authentic data during the development and validation of the prototype. Therefore, the conducted action research is evaluated as 1 with regard to both context and subjects.

In this research, the requirements architecting method was validated using a prototype in an industrial setting. However, the application was not industrial-scale. A down-scaled industrial case was also conducted in [166]. The goal of the prototype testing was to provide a solid basis for the new way of working, not to validate the applicability of an industrial-scale tool. In Phase 3, a product management tool used in the requirement screening process in company A was improved by integrating proof of concept ideas into it to support the new way of working in an industrial environment. Furthermore, the development of the prototype was continued, resulting in a commercial tool. Thus, the scale of this research is evaluated as 0.5. This highlights the feasibility of the developed requirements architecture method from a practical point of view.

The research methods utilised in Phase 2 were action research, a case study and interviews, all of which are considered to contribute to the relevance of the research. Five of the six reference papers applied case study as a research method – [162],
Thus, the action research conducted is rated 1. The overall grading for Phase 2 is 3.5 out of 4. Table 10 summarises the reasoning and grading of Phase 2.

Table 12. Summary of relevance evaluation of the action research cycles carried out in Phase 2.

<table>
<thead>
<tr>
<th>Aspect</th>
<th>Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Contribution (1)</td>
</tr>
<tr>
<td>Subjects</td>
<td>Representatives of intended users, i.e. industry professionals.</td>
</tr>
<tr>
<td>Context</td>
<td>Evaluation is done in an industrial setting</td>
</tr>
<tr>
<td>Scale</td>
<td>Applications used are industrial scale.</td>
</tr>
<tr>
<td>Research method</td>
<td>Facilitates investigating real settings and is relevant for practitioners, action research, case study, interviews, etc.</td>
</tr>
</tbody>
</table>
7 Conclusion

The research problem defined in this thesis was to determine what information should be gathered in requirements elicitation and how that information should be documented to enable the efficient filtering of valuable requests to provide a reliable basis for selecting features to be developed in the context of VLSRE and distributed development. In this research, a valuable requirement has been defined as one that most likely provides added value for customers and a positive business impact for a solution provider.

The question about what information should be gathered in requirements elicitation was looked at both from two viewpoints: that of information providers, such as DfX disciplines, and that of information consumers like requirement screening experts. Based on these needs, the first list of mandatory and optional attributes was created, and the feasibility of these proposed attributes was studied in the conducted customer relations interviews. To answer the second part of the problem, how the information should be documented to enable the efficient filtering of valuable requests, the list of attributes was updated and re-structured as a requirement template based on product management interviews. The requirement template was paper piloted with DfX disciplines to assess the suitability of the proposed structure. To provide reliable bases for selecting features to be developed in a distributed development setting and in a VLSRE context, RE items; request, requirement and feature information content with context dependent guidance, were implemented in a prototype tool for validating the developed requirements architecting method in an industrial context. The relevance of the conducted research was evaluated using the method introduced by Ivarsson and Gorschek [15].

In addition to the research results, this thesis contributes to software engineering research methodology by explaining in detail an action research effort that spans several years. The description includes four cases conducted in the diagnosing phase, three cycles that were carried out to identify stakeholders’ challenges related to the requirement screening process and to develop solutions to address those challenges as well as a continuous deployment segment as a follow up phase. Action research has a long tradition in IS research [109], but it is not extensively applied in software engineering. Action research efforts are often reported as case studies [168], or the reporting is incomplete [102], when, for example, only one cycle is described. Longitudinal studies are in the minority in software engineering. Studies lasting around a year are defined as longitudinal [169].
This type of industry–academia collaboration benefits all participants. For practitioners, research collaboration provides an opportunity to gain first-hand knowledge about existing scientifically evaluated methods and tools that they can exploit. For researchers, industry collaboration provides a means through which to test the developed technologies in an authentic environment with experts. Continuing collaboration for years requires that the results are mutually interesting and provide value. We have learned that individuals have a great impact on this type of collaboration and propel it forward. For instance, trust between organisations is easier to build if key personnel have a good history of working collaboratively.

7.1 Limitations and future research

One limitation of this study is that most of the results are based on the research conducted in case company A. Therefore, the generalisability of the results is limited. The limitation of the empirical data was mitigated by two means: conducting a complementary study in case company B (described in publication V) and interviewing experts from three other companies (described in publication IV). It is possible to consider the timeframe of the study as a limitation as well, as the research was conducted in the rapidly changing ICT domain. However, longitudinal action research studies may overcome the barriers between practitioners and academics and improve technology transfer from academia to industry [170].

Future work should evaluate the applicability of the proposed solutions to other products beyond the ICT domain as well as within companies that take different development approaches.
References

21. C. Wohlin and A. Aurum, “What is important when deciding to include a software requirements in a project or release?,” Empirical Software Engineering, pp. 246-255, 2005.


Original publications


Reprinted with permission from (I) and (III) Springer, (II) SCITEPRESS, (IV) IARIA XPS Press and (V) ACM.

Original publications are not included in the electronic version of the dissertation.
684. Karppinen, Pasi (2016) Studying user experience of health behavior change support systems: a qualitative approach to individuals’ perceptions of web-based interventions


686. Taušan, Nebojša (2016) Choreography modeling in embedded systems domain

687. Ylänne, Henni (2017) Herbivory control over tundra carbon storage under climate change


689. Singh, Sujeeit Kumar (2017) Conservation genetics of the Bengal tiger (Panthera tigris tigris) in India


691. Hens, Hilde (2017) Population genetics and population ecology in management of endangered species


694. Schneider, Laura (2017) Mechanocatalytic pretreatment of lignocellulosic barley straw to reducing sugars


697. Havia, Johanna (2017) Trace element analysis of humus-rich natural water samples: Method development for UV-LED assisted photocatalytic sample preparation and hydride generation ICP-MS analysis

698. Dong, Yue (2017) Bifunctionalised pretreatment of lignocellulosic biomass into reducing sugars: use of ionic liquids and acid-catalysed mechanical approach

699. Leinonen, Marko (2017) On various irrationality measures

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/
Sanja Aaramaa

DEVELOPING A REQUIREMENTS ARCHITECTING METHOD FOR THE REQUIREMENT SCREENING PROCESS IN THE VERY LARGE-SCALE REQUIREMENTS ENGINEERING CONTEXT