Requirements Prioritisation and Retrospective Analysis for Release Planning Process Improvement

Karlsson, Lena

2006

LUND UNIVERSITY

PO Box 117
221 00 Lund
+46 46-222 00 00

Requirements Prioritisation and Retrospective Analysis for Release Planning Process Improvement

Lena Karlsson

LUND UNIVERSITY

Department of Communication Systems
Lund Institute of Technology

ISSN 1101-3931
ISRN LUTEDX/TETS--1081--SE+192P

Printed in Lund, Sweden by E-kop

September 2006

To mum and dad

This thesis is submitted to the Research Education Board of the Faculty of Engineering at Lund University, in partial fulfilment of the requirements for the degree of Doctor of Philosophy in Engineering.

Contact Information:

Lena Karlsson
Department of Communication Systems
Lund University
P.O. Box 118
SE-221 00 LUND
Sweden

Tel: +46 46 222 03 65
Fax: +46 46 14 58 23
E-mail: lena.karlsson@telecom.lth.se

Abstract

The quality of a product can be defined by its ability to satisfy the needs and expectations of its customers. Achieving quality is especially difficult in market-driven situations since the product is released on an open market with numerous potential customers and users with various wishes. The quality of the software product is to a large extent determined by the quality of the requirements engineering (RE) and release planning decisions regarding which requirements that are selected for a product. The goal of this thesis is to enhance software product quality and increase the competitive edge of software organisations by improving release planning decision-making.

The thesis is based on empirical research, including both qualitative and quantitative research approaches. The research contains a qualitative survey of RE challenges in market-driven organisations based on interviews with practitioners. The survey provided increased understanding of RE challenges in the software industry and gave input to the continued research. Among the challenging issues, one was selected for further investigation due to its high relevance to the practitioners: requirements prioritisation and release planning decision-making. Requirements prioritisation techniques were evaluated through experiments, suggesting that ordinal scale techniques based on grouping and ranking may be valuable to practitioners. Finally, a retrospective method called PARSEQ (Post-release Analysis of Requirements SElection Quality) is introduced and tested in three case studies. The method aims at evaluating prior releases and finding improvement proposals for release planning decision-making in future release projects. The method was found valuable by all participants and relevant improvement proposals were discovered in all cases.

Acknowledgements

This work was funded by Lund University Faculty of Engineering as well as the Swedish Agency for Innovation Systems (VINNOVA) under the grant for the Center for Applied Software Research at Lund University (LUCAS) and the ITEA project for Embedded Systems Engineering in Collaboration (MERLIN).

First and foremost, I would like to thank my supervisor Björn Regnell for sharing his great creativity and knowledge. Thanks also to Martin Höst, Per Runeson, and Thomas Thelin for support and assistance. Many thanks to all past and present colleagues, co-authors, and friends at the Department of Communication Systems, at Blekinge Institute of Technology, at University of Skövde, and in the MERLIN project. Thanks to all anonymous individuals who have contributed to the thesis: interviewees and focus group participants, experiment subjects, industrial case study participants, and reviewers. A special thanks to Per Klingnäs and Mikael Jönsson for excellent development of the PARSEQ tool. Finally, thanks to my family for constant encouragement, and to friends who have stood by me through rain and sunshine.

Lena Karlsson
September, 2006

Contents

Introduction

Software continually becomes a more and more important part of an increasing number of products. Several different domains need to deal with software development, e.g. the automotive industry, developers of medical IT, and developers of commercial products such as mobile phones and digital cameras. The intangible and flexible nature of software causes software projects to be over-represented in project failure statistics. Typical problems include lack of functionality, poor quality, budget overruns and missed deadlines (The Standish Group, 2001).

Quality can be defined as *the degree to which a system, component, or process meets customer or user needs or expectations* (IEEE, 1990). Even if a product is delivered on time and within budget, it may be a failure due to poor quality if it does not meet customer and user expectations. In particular, *market-driven* organisations, which release their products on an open market with numerous potential customers and users, experience challenges with quality. Satisfying customer expectations is very difficult when the customers are diverse and have different opinions.

In software product development the customer expectations are elicited, analysed, specified, and validated in an activity called requirements engineering (RE) (Sommerville, 2001). The activity lays the foundation for successful planning and development of the product before release to the market. In a competitive environment, such as the one experienced by market-driven organisations, it is essential to plan product releases with time-to-market in mind. Release planning is where requirements engineering for market-driven software product development meets the market perspective. Selecting a subset of requirements for realisation in a certain release is as complex as it is important for the success of the product (Carlshamre, 2002).

The main goal of the research presented in this thesis is to enhance software product quality and to increase the competitive edge of software organisations. By developing and applying methods and techniques for assessing and improving RE and release planning, the software product quality is expected to improve. The main contributions are: increased understanding of RE challenges in the software industry based on a qualitative survey, evaluation of requirements prioritisation techniques based on experiments, and a method for retrospective analysis of release planning decision-making evaluated in case studies.

The thesis starts with an introduction to the research area and the research focus. Following the introduction there are three parts, each including between one and three papers. In order to avoid repetition and redundancy, each part has one introduction and one concluding section. That is, when two or three papers are combined, the introduction and conclusion sections have been integrated. In total, six papers are included partly or completely in the thesis.

- **Introduction.** This section describes the background of the research. Section 1 presents the research focus and research questions examined in the thesis. Section 2 continues with a description of related work to put the research into context. The research approach and validation issues are described in Section 3, before the research results and contributions are presented and future research is discussed in Section 4.

- **Part I: Requirements Engineering Challenges in Industry.** The first part presents a qualitative survey of RE challenges in market-driven organisations (Paper 1). The paper describes challenging areas within RE experienced by 14 interviewed practitioners. Among the many challenges we find issues related to release planning and requirements prioritisation.

- **Part II: Evaluation of Requirements Prioritisation Techniques.** The work in Part II is focused on evaluating different techniques for requirements prioritisation since it is an important activity in release planning. The second part includes two studies: the first one describes two experiments comparing different requirements prioritisation techniques (Paper 2) and the second one presents an archive analysis examining results from prioritisation sessions (Paper 3).

- **Part III: Retrospective Analysis for Release Planning Decisions.** Release planning process improvement is investigated in the third part. A method for retrospective analysis of release planning decision-making is presented, as well as results from three industrial case studies (Paper 4 and 5). In addition, we present tool support which was used and evaluated in one of the case studies (Paper 6).

List of Papers

The thesis is based on the following six papers:

1. REQUIREMENTS ENGINEERING CHALLENGES IN MARKET-DRIVEN SOFTWARE DEVELOPMENT – AN INTERVIEW STUDY WITH PRACTITIONERS
Lena Karlsson, Åsa G. Dahlstedt, Björn Regnell, Johan Natt och Dag, Anne Persson
Accepted for publication in Information and Software Technology: Special Issue on Understanding the Social Side of Software Engineering, Qualitative Software Engineering Research, 2007.

2. PAIR-WISE COMPARISONS VERSUS PLANNING GAME PARTITIONING - EXPERIMENTS ON REQUIREMENTS PRIORITISATION TECHNIQUES
Lena Karlsson, Thomas Thelin, Björn Regnell, Patrik Berander, Claes Wohlin
Accepted for publication in Empirical Software Engineering Journal, 2006.

3. EVALUATING THE PRACTICAL USE OF DIFFERENT MEASUREMENT SCALES IN REQUIREMENTS PRIORITISATION
Lena Karlsson, Martin Höst, Björn Regnell
Proceedings of the 5th ACM-IEEE International Symposium on Empirical Software Engineering (ISESE'06), Rio de Janeiro, Brazil, September 2006.

4. CASE STUDIES IN PROCESS IMPROVEMENT THROUGH RETROSPECTIVE ANALYSIS OF RELEASE PLANNING DECISIONS
Lena Karlsson, Björn Regnell, Thomas Thelin
Accepted for publication in International Journal of Software Engineering and Knowledge Engineering: Special Issue on Requirements Engineering Decision Support, December 2006.

5. RETROSPECTIVE ANALYSIS OF RELEASE PLANNING DECISIONS IN A PRODUCT LINE ENVIRONMENT - A CASE STUDY
Lena Karlsson, Björn Regnell

Submitted, 2006.

6. INTRODUCING TOOL SUPPORT FOR RETROSPECTIVE ANALYSIS OF RELEASE PLANNING DECISIONS
Lena Karlsson, Björn Regnell
Proceedings of the 7th International Conference on Product Focused Software Process Improvement (PROFES'06), Amsterdam, the Netherlands, June 2006, pp. 19-33.

Related Publications

The following publications are related but not included in the thesis:

7. UNDERSTANDING SOFTWARE PROCESSES THROUGH SYSTEM DYNAMICS SIMULATION: A CASE STUDY
Carina Andersson, Lena Karlsson, Josef Nedstam, Martin Höst, Bertil I Nilsson
Proceedings of the 9th IEEE Conference and Workshop on the Engineering of Computer-Based Systems (ECBS'02), Lund, Sweden, April 2002, pp. 41-48.
(This paper presents a simulation model which worked as foundation for the model presented in Paper 10.)

8. CHALLENGES IN MARKET-DRIVEN REQUIREMENTS ENGINEERING - AN INDUSTRIAL INTERVIEW STUDY
Lena Karlsson, Åsa G. Dahlstedt, Johan Natt och Dag, Björn Regnell, Anne Persson
Proceedings of the 8th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'02), Essen, Germany, September 2002, pp. 37-49.
(This paper presents intermediate results from Paper 1 and is based on the first seven interviews.)

9. POST-RELEASE ANALYSIS OF REQUIREMENTS SELECTION QUALITY - AN INDUSTRIAL CASE STUDY
Lena Karlsson, Björn Regnell, Joachim Karlsson, Stefan Olsson
Proceedings of the 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03), Velden, Austria, June 2003, pp. 47-56.
(This paper presents the first of the case studies described in Paper 4.)

10. AN ANALYTICAL MODEL FOR REQUIREMENTS SELECTION QUALITY EVALUATION IN PRODUCT SOFTWARE DEVELOPMENT
Björn Regnell, Lena Karlsson, Martin Höst
Proceedings of the 11th IEEE International Conference on Requirements Engineering (RE'03), Monterey Bay, California, the USA, September 2003, pp. 254-263.
(This paper is summarised in the Introduction, Section 2.3. It presents results from a survey, which motivates the evaluation and improvement of requirements selection quality.)

11. MARKET-DRIVEN REQUIREMENTS ENGINEERING PROCESSES FOR SOFTWARE PRODUCTS - A REPORT ON CURRENT PRACTICES
Åsa G. Dahlstedt, Lena Karlsson, Johan Natt och Dag, Björn Regnell, Anne Persson
1st International Workshop on COTS and Product Software (RECOTS'03), Monterey Bay, California, USA, September 2003.
(This paper presents intermediate results from Paper 1 and is based on the first seven interviews. The paper compares the discovered challenges to the characteristics of market-driven software development reported in literature.)

12. IMPROVING REQUIREMENTS SELECTION QUALITY IN MARKET-DRIVEN SOFTWARE DEVELOPMENT
Lena Karlsson
Licentiate thesis, ISRN LUTEDX/TETS-1063-SE+132P, Dept. of Communication Systems, Lund University, Sweden.
(The licentiate thesis includes Papers 7, 8, 9, 10, and an early version of Paper 13.)

13. REQUIREMENTS PRIORITISATION: AN EXPERIMENT ON EXHAUSTIVE PAIR-WISE COMPARISONS VERSUS PLANNING GAME PARTITIONING
Lena Karlsson, Patrik Berander, Björn Regnell, Claes Wohlin
Proceedings of the 8th International Conference on Empirical Assessment in Software Engineering (EASE'04), Edinburgh, UK, May 2004, pp. 145-154.
(This paper presents the first of the two experiments in Paper 2.)

14. INVESTIGATION OF REQUIREMENTS SELECTION QUALITY IN MARKET-DRIVEN SOFTWARE PROCESSES USING AN OPEN SOURCE DISCRETE EVENT SIMULATION FRAMEWORK
Björn Regnell, Bengt Ljungquist, Thomas Thelin, Lena Karlsson
Proceedings of the 5th International Workshop on Software Process Simulation and Modeling (ProSim'04), Edinburgh, UK, May 2004, pp. 84-93.
(This paper introduces a simulation framework for the analytical model of requirements selection quality presented in Paper 10.)

15. ALIGNING THE REQUIREMENTS ENGINEERING PROCESS WITH THE MATURITY OF MARKETS AND PRODUCTS
Lena Karlsson, Björn Regnell
Proceedings of the 10th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'04), Riga, Latvia, June 2004, pp. 69-74.
(This paper describes market-driven RE from a lifecycle perspective, since it was discussed by interviewees in Paper 8.)

16. HOW EVALUATION TECHNIQUES INFLUENCE THE RE-TOOL EVALUATION: AN EXPERIMENT.
Raimundas Matulevicius, Lena Karlsson, Guttorm Sindre
Proceedings of the European Software Process Improvement Conference (EuroSPI'04), Trondheim, Norway, November 2004, pp. I3B11-I3B16.
(This paper investigates RE tools, which is related to the topic in Paper 6.)

17. COMPARING ORDINAL AND RATIO SCALE DATA IN REQUIREMENTS PRIORITISATION
Lena Karlsson, Björn Regnell
Proceedings of the 3rd International Workshop on Comparative Evaluation in Requirements Engineering (CERE'05), Paris, France, August 2005, pp 21-29.
(This paper presents the measures for comparing prioritisation results from different measurement scales, which are further evaluated with a larger data set in Paper 3.)

18. A CASE STUDY IN RETROSPECTIVE ANALYSIS OF RELEASE PLANNING IN AN AGILE PROJECT
Lena Karlsson, Björn Regnell, Thomas Thelin
1st Workshop on the Interplay of Requirements Engineering and Project Management in Software Projects (REProMan'05), Paris, France, August 2005.
(This paper presents the second of the case studies presented in Paper 4.)

Contribution Statement

The author of the thesis is the main author of the six included papers. This means responsibility for running the research process, dividing the work between coauthors and conducting most of the writing. Paper 1 and 2 were produced in cooperation with other universities and have five authors each. In both cases, a lot of the design and analysis was performed together with coauthors, while most of the writing and division of work was performed by the main author. The research in Paper 3, 4, and 5 was performed primarily by the main author, who designed and conducted most of the work, as well as reported on the studies. Paper 6 describes a tool which was designed by the author, but developed by two external developers. The paper is written primarily by the main author.

1. Research Focus

The goal of this research is to find means for improving the RE and release planning processes. The purpose is to enhance software product quality and increase the competitive edge of software organisations. The main research questions that have been investigated are:

RQ1. Which challenges related to RE are experienced by practitioners in the market-driven software development industry?

RQ2. How can requirements prioritisation techniques be characterised and compared?

RQ3. How can retrospective analysis be used to evaluate and improve the release planning process?

The research questions RQ1-RQ3 correspond to Part I-III in the thesis. The relation between the research questions is illustrated in Figure 1.

RQ1 was posed in order to discover and understand RE challenges experienced by practitioners and was used to select focus for the research. Among the many challenges that appeared in the qualitative survey, issues regarding requirements prioritisation and release planning emerged. Therefore, these areas were targeted in the continued research.

RQ2 was examined to understand and compare requirements prioritisation techniques since requirements prioritisation is a vital activity for release planning. The characteristics of different techniques were identified in experiments and their potential support for release planning decision-making was examined.

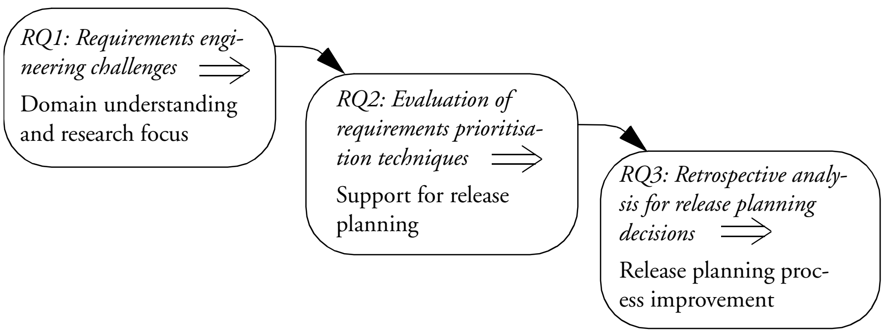

Figure 1. *The three parts of the thesis*

RQ3 aims at improving the release planning process through retrospective analysis. The results from RQ2 were used to create a method for evaluating the release planning process and discovering possible improvements. The retrospective analysis method was applied in three industrial case studies with different characteristics in order to investigate its possibilities and limitations.

2. Related Work

This section provides some theoretical background to the requirements engineering area and describes the context of the research in the thesis. The successive subsections describe related work in general, while in Section 4 related work is discussed in relation to the thesis findings.

Software engineering is an engineering discipline whose goal is to cost-effectively develop software systems. This includes all aspects of software development; from the early stages of system specification through to maintaining the system after it is put into use (Sommerville, 2001). *Software requirements* are by the Software Engineering Body of Knowledge (SWEBOK, 2004) defined as properties that must be exhibited in order to solve some problem of the real world. In other words, requirements define what the system should do, i.e. what functionality and qualities the system shall include. Thus, *requirements engineering* regards the process of identifying, analysing, documenting, validating and managing these software properties (Lauesen, 2002).

Systems engineering is concerned with all aspects of computer-based systems development, including hardware, software, electrical and mechanical engineering, thus software engineering is part of this area. The software in these systems is *embedded* in a hardware system and must respond, in real-time, to events from the system's environment (Sommerville, 2001). In the thesis, the term *product* is also used to refer to a system that partly or completely consists of software.

2.1 Requirements Engineering

Traditionally, RE takes place in the beginning of every project, and results in a specification that defines the product to be developed. This view is based on the *Waterfall model* (Royce, 1970), where requirements engineering is followed by design, implementation, testing and maintenance activities. However, this cascade process may not be the most appropriate in practice, since the flexible nature of software requires the development process to be more iterative and evolutionary. New and changed requirements appearing during development calls for continuous RE efforts.

The four main activities in the RE process, as defined by Sommerville (2001), are illustrated in Figure 2. The feasibility study is performed before starting with elicitation, and the activities are, in practice, performed iteratively in order to handle changing requirements. In addition, requirements management is performed continually throughout the product life-cycle to understand and control requirements changes.

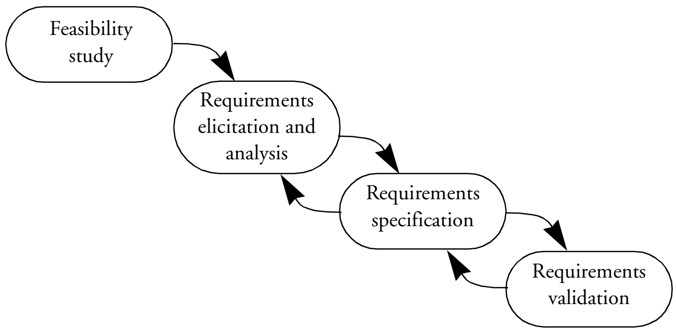

Figure 2. *The requirements engineering process*

- *Feasibility study* is performed to decide whether or not it is worth carrying on with development. The system should contribute to the overall objectives of the organisation and be possible to implement with the current technology and within the given cost and schedule constrains.

- *Requirements elicitation and analysis* starts with gaining application domain understanding and moves on to collecting requirements from stakeholders for the system. Next, the requirements are classified and conflicting views among stakeholders are resolved. In any set of requirements, some are more important than others. Prioritisation is performed to discover the most important requirements. Finally, the requirements are checked for completeness, consistency, and accordance with the stakeholders' wishes.

- *Requirements specification* involves documenting the elicited functional and non-functional requirements in detail. Non-functional requirements are also called quality requirements and affect how well a system must perform its functions (Lauesen, 2002). In addition, the specification may include purpose and scope of the product, user characteristics, and development constraints.

- *Requirements validation* involves showing that the requirements actually define the system which the customer wants and is concerned with finding problems with the requirements. Validation can be performed with different techniques such as reviews or prototyping.

2.2 Market-Driven Requirements Engineering

Products can be divided into different categories depending on the type of market where the product is vended. Among the early work that characterises the differences between *customer-specific* and *market-driven* development is the field study by Lubars et al. (1993). The authors investigated differences between the two types of development in the areas of requirements definition, specification and validation.

In the customer-specific case (also called *bespoke* or *contract-driven*) the product is ordered by a specific customer and the supplier develops and maintains the product for that customer. The customer often represents a large organisation such as a military, governmental or financial institution. A product contract is negotiated with the customer, describing what the product shall include, when it shall be delivered, and how much it will cost.

Market-driven software systems or products (also called *packaged* or *commercial-of-the-shelf*) are developed for an open market. The customer may be another organisation or a consumer and the products may be, for example, computer packages, development tools, or mobile phones. In both cases there is a large range of *potential* customers on a mass market and suppliers need to take diverse needs into account.

The characteristics of market-driven development have been described by Lubars et al. (1993), Potts (1995), Sawyer (2000), and Carlshamre (2001). Some of their findings are summarised below.

The characteristics of market-driven RE include, for example, that the mass market product often has a life cycle with several consecutive releases and it lasts as long as there is a market for it. Therefore, release planning is an important activity. A highly important issue is to have shorter time-to-market than the competitors, in order to yield high market shares and be successful on the market.

Requirements are often invented by developers or elicited from a set of potential customers since there is no single customer to ask. This may yield too many requirements with respect to the available resources for one release. It is necessary to make estimations of implementation effort and market value in order to prioritise and select a set that will fit the market and corporate strategy. Requirements are prioritised within the market-driven developing organisation before release planning, while in the customer-specific situation the requirements are negotiated and contracted with the customer.

Many organisations do in fact deal with both market-driven and customer-specific projects. In this thesis, we focus mainly on the market-driven aspects of these organisations.

2.3 Release Planning Decision-Making

Release planning is one of the specific RE activities conducted in market-driven organisations, along with prioritisation and cost estimation. Release planning can be described as selecting an optimal subset of requirements for realisation in a certain release. Thus, it is a major determinant of the success of the software product (Carlshamre, 2002).

Wohlin and Aurum (2005) identified relevant criteria for release planning based on a survey with practitioners. One of the criteria regarded as relevant to all respondents in the survey was the actual cost-benefit trade-off for implementing a requirement. This is similar to the criteria used by Karlsson and Ryan (1997) in the cost-value approach. The approach is based on optimising the relation between the requirements' value and cost in order to achieve stakeholder satisfaction.

The requirements selection and release planning process is supported by requirements prioritisation, which can be defined as the activity during which the most important requirements for a system are identified (Sommerville, 2001). Issues that determine the priority of a requirement include importance to users and customers, implementation costs, logical implementation order, and financial benefit (Lehtola et al., 2004). There

Table 1. Summary of prioritisation techniques

Technique	Description	References
Planning game	Grouping and ranking requirements on an ordinal scale. Usually based on the criteria value, cost and risk.	Beck, 2005
Pair-wise comparisons	Comparisons between all pairs of requirements. Based on the Analytical Hierarchy Process (AHP) and the result is on a ratio scale. Usually based on cost and value criteria.	Saaty, 1980; Karlsson, 1997
Numeral Assignment	Grouping requirements in e.g. 3 or 5 groups, usually based on customer value. The result is presented on an ordinal scale.	IEEE, 1998; Karlsson, 1996
$100 test	Also called cumulative voting and is suitable in distributed environments. Based on assigning fictional money to requirements and the result is on a ratio scale.	Leffingwell and Widrig, 2000
Wiegers' method	Combines the customer value, penalty if the requirement is not implemented, implementation cost, and risks. The result is on a ratio scale.	Wiegers, 1999

are several prioritisation techniques described in literature, but prioritisation practice is informal in many companies (Lehtola and Kauppinen, 2006). Some of the prioritisation techniques are summarised in Table 1. For more details regarding requirements prioritisation techniques, see e.g. Moisiadis (2002) and Berander and Andrews (2005), as well as Part II of this thesis.

The selection of requirements for a release is often made in several steps of the RE process, since release plans are revised and changed throughout development as more knowledge is gained about market expectations and development progress. Starting out with a large set of potential requirements, the selection brings the number down for each activity in the requirements process. This is illustrated in Figure 3, which is adapted from Regnell et al. (2003) (Paper 10). The discarded requirements are typically stored in a database for investigation in future releases. During *elicitation*, decisions concern which stakeholder representatives to consult to elicit ideas for new features. Then there is often a *screening* activity performed to make a first quick assessment to decide whether a requirement is worth spending more time on. Requirements that are clearly out of scope for the next release are rejected

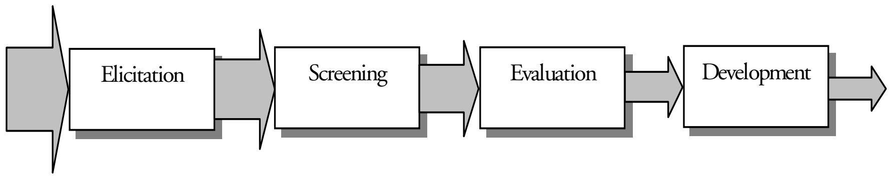

Figure 3. *The RE process from a requirements selection viewpoint*

in order to avoid overload in the process (Regnell et al., 1998). The *evaluation* activity includes prioritising requirements and identifying requirements that are interdependent. Finally, the requirements can be re-assessed during *development* and decisions regarding postponing or removing requirements can be made based on the information gained during implementation.

In Regnell et al. (2003), the requirements selection process is described as a queuing network model, with parameters for arrival rates, service rates, number of servers, and probability for a requirement to remain in the process. The parameters were estimated based on a survey with practitioners from companies developing software-intensive products. The survey indicates that, on average, only 21% of all incoming requirements are good enough to be implemented with regard to market opportunities, product strategy and development resources. Evidently it is difficult to determine which of the incoming requirements to select for implementation. Furthermore, a majority of the respondents estimate that only 25-50% of the requirements selection decisions made in their organisation are correct. It appears that there is a large opportunity for improving the release planning process and thereby the requirements selection quality.

2.4 Retrospective Analysis for Process Improvement

Several different approaches to Software Process Improvement (SPI) have been suggested. Among the common ones are maturity models such as the Capability Maturity Model (CMM) (Paulk et al., 1993) and standards such as ISO9000 (www.iso.org). These approaches aim at assessing and improving the process through a set of principles or practices. However, many of those who have applied CMM have been disappointed about the time and costs required for the assessment and improvement (Herbsleb and Goldenson, 1996).

An alternative to using pre-defined principles and practices is to base the process improvement effort on the experiences of your own organisation or project. One such approach is the Experience Factory (Basili et al., 1994). It is aimed at capitalising and reusing lifecycle experiences and products through processing project information and data, and giving feedback to project activities. Basili et al. (1994) state that reuse of knowledge is the basis of improvement. Another approach based on internal experiences is the *retrospective analysis*, acknowledged as one of the most important steps toward improving the software process (Kerth, 2001). Retrospective analysis has been used under different names such as *postmortem analysis* (Birk et al., 2002), *postmortem project evaluation* (Ulrich and Eppinger, 2000), and *postmortem review* (Dingsoyr, 2005). Retrospective analysis is usually performed after the project is finished and may consist of an open-ended discussion of the strengths and weaknesses of the project plan and execution. Sometimes it is facilitated by an outside consultant or someone with an objective view of the project. At the end of the session, a postmortem report is prepared as a formal closing of the project, which is then used in the project planning stage of future projects (Ulrich and Eppinger, 2000).

Retrospective analysis is also recognised as a valuable method for *knowledge management,* which promotes an organisation's intellectual capital (Rus and Lindvall, 2002). It focuses on the individual as an expert and bearer of important knowledge that he or she can systematically share within the organisation. The retrospective analysis has also been used in the agile community, which uses the concept *process refactoring* (Collins and Miller, 2001) in order to emphasise the continuous approach to process improvement. Instead of waiting until a project is finished, process refactoring takes place *during* the project in order to improve before it is too late. Retrospective analysis is discussed further in Part III of the thesis.

Some of the more recent work focus on the outcome of the retrospective analysis, i.e. how the findings are reported and used. Dingsoyr (2005) describes a case study at a company where every project wrote an experience report, but these were seldom read by other projects. It seems important to consider how the results from the retrospective analysis are reported and transferred to future projects. One way to make postmortem analysis results more accessible is presented by Schalken et al. (2006). The authors present a method to derive findings from a set of postmortem review reports and transform the qualitative information into quantitative information. The results may provide guidance in a process improvement initiative. Desouza et al. (2005) describe an evaluation of two different outcomes from the retrospective process: traditional reports and stories. The differences in structure, cost of preparation, richness of knowledge, and ease of comprehension require each project to decide which one is more appropriate.

3. Research Approach

This section gives an overview of the different research approaches that are used in the thesis. It also describes certain validity issues that need to be considered for each approach. The section starts with an introduction of some research methodology concepts and continues with three sections describing the research approach and validity for each research question. The validity is discussed from a research design perspective, i.e. which measures that were taken during design to increase validity of the results. In Section 4, the validity is discussed from the perspective of the results, e.g. limitations of the results and the results in relation to literature.

One main research approach is used for each research question. The first research question is answered by a qualitative survey, the second by experiments, and the third by a series of case studies.

3.1 Research Designs

There are two main types of research designs: *fixed* and *flexible* designs (Robson, 2002). The fixed design, also called *quantitative*, deals with designs that are highly pre-specified and prepared. A conceptual framework or theory is required in order to know in advance what to look for. It is often concerned with quantifying a relationship or comparing two or more groups and the results are often *prescriptive*, i.e. it suggests a solution, method or tool that is more appropriate than another.

The flexible design, also called *qualitative*, relies on qualitative data and requires less pre-specification. The design is intended to evolve and develop during the research process as the researchers gain more knowledge. The flexible design is concerned with studying objects in their natural setting and is often *descriptive*, i.e. it describes some issue of the real world. Qualitative data may include numbers, but are to a large extent focused on words. Many times, however, a design may include both fixed and flexible methods and yield both quantitative and qualitative data.

The research in this thesis uses three major types of methods: *survey*, *experiment* and *case study*. Surveys and case studies can be both fixed and flexible, while experiments are typically fixed (Wohlin et al., 2000). Pure qualitative research designs include strategies such as *grounded theory* and *ethnography*, which are discussed briefly in Part I of this thesis and more in depth by Robson (2002).

All research designs and approaches have certain validity issues that need to be considered. In fixed designs there are mainly four types of validity: conclusion, internal, construct and external validity. In flexible designs there is a different set of validity issues, which regards description, interpretation and theory (Robson, 2002). In addition to these, there are matters of respondent and researcher bias, i.e. when people involved in the research, deliberately or not, affect the results.

3.2 Survey on RE Challenges

This section describes the qualitative survey methodology, which was used to answer the first research question. It also discusses validity issues for flexible designs and for the performed survey.

SURVEYS. Surveys can be both flexible and fixed, i.e. include different degrees of pre-specification. Survey is a wide term that includes everything from open-ended interviews (typically flexible) to questionnaires with closed questions (typically fixed). While questionnaires can reach a large set of people and provide data that is easy to analyse, there is a risk of low response rates and questionnaires can be prone to misunderstandings. Interviews have higher response rates and the interviewer may explain and clarify questions during the session to avoid misunderstandings. However, there are disadvantages such as high time consumption and that the interviewer may impose a bias (Robson, 2002). The purpose of surveys is to understand, describe, explain or explore the population (Wohlin et al., 2000).

VALIDITY IN FLEXIBLE DESIGNS. The three main validity issues in flexible designs are presented here. A more detailed presentation is available in Robson (2002).

1. *Description* is regarding the accuracy and completeness of the data. Thus, if interviews occur they should be audio-taped and possibly transcribed in order to keep all data for reference and analysis.

2. *Interpretation* implies that frameworks and theories shall emerge from the knowledge gained during the research, instead of being predetermined and biased. This is achieved by analysing and demonstrating how the interpretation was reached.

3. *Theory* is closely related to interpretation, but regards the threat of not considering alternative explanations of the phenomena under study. This is confronted by continuously revising and refining the theory until it accounts for all known cases.

RESEARCH QUESTION 1.
Which challenges related to RE are experienced by practitioners in the market-driven software development industry?

The first research question was investigated by conducting a flexible survey at eight different Swedish companies involved in market-driven software development. Fourteen practitioners participated in interviews, which were recorded and later transcribed and analysed with support from a commercial data analysis tool. The interviews were open-ended and had a flexible structure, and took different shape depending on the responses. To validate intermediate results, a focus group meeting was held with five RE practitioners.

The data was stored both on audio tape and as printed transcriptions in order to keep data complete and accurate. Thus, the *description validity* is taken into consideration. In most interviews, two or three researchers participated and extensive notes were taken. During analysis, three researchers with different research interests examined the data and drew different conclusions that were later discussed. Such discussions from different angles help to ensure that conclusions were not emerged through prejudices, but through the knowledge gained during the study. Thereby, we believe that the *interpretation validity* is regarded.

The eight companies were of different size and age, and from different business areas, and at six of the companies, interviews were held with two people in different organisational positions. In that manner, we believe that the gained knowledge is based on many different aspects and the results reflect a broad image of the reality. The selection of companies from different categories of age, size and business were made with the intent to regard *theory validity*.

Since the study is based on a flexible design we do not intend to generalise the results to a larger population, but only to the setting under study. The intention with the survey was to gain understanding of RE and describe RE challenges. Based on the gained understanding we could find areas in need of further research.

3.3 Experiments on Prioritisation Techniques

This section describes the methodology of experiments as it was used to answer the second research question. Although Paper 3 is not a controlled experiment but an archive analysis, the methodology and validity issues of experiments can be applied. It also discusses validity in fixed designs and presents how the validity threats were handled in the studies.

EXPERIMENTS. Experiments are used when we want control over the situation and wish to manipulate the behaviour. Results are often reported as averages and proportions, thus it is a quantitative design. Controlled experiments involve more than one treatment to compare the outcomes and enable statistical analysis. As other fixed designs, the experiment is theory-driven and requires a substantial amount of conceptual understanding from the start (Robson, 2002).

The design of an experiment should be made so that the objects involved represent all the methods or tools we are interested in. The strength of an experiment is that we can investigate in which situation the claims are true and they provide a context in which certain methods or tools are recommended for use (Wohlin et al., 2000).

VALIDITY IN FIXED DESIGNS. Some of the validity problems encountered in fixed designs are briefly described here, while more thorough presentations are available in Robson (2002) and Wohlin et al. (2000).

1. *Conclusion validity* is concerned with the relationship between the treatment and the outcome. We want to make sure that there is a statistical relationship, i.e. with a given significance. The threats are concerned with choice of statistical tests, sample sizes and care taken in the implementation and measurement of the experiment.

2. *Internal validity* is needed to make sure that the relationship between the treatment and outcome is a causal relationship, i.e. that the treatment actually caused the result. Threats to internal validity concern issues such as how the subjects are selected and divided into classes, and how subjects are treated during the experiment, i.e. factors that can make the experiment show behaviour that is not due to the treatment.

3. *Construct validity* is concerned with the relation between the theory and the observation and refers to the extent to which the experiment setting actually reflects the construct under study. Using multiple strategies to measure the same thing may improve the construct validity and ensure that the result is an effect of the treatment.

4. *External validity* regards generalisability of the setting and the subjects. Although internal validity is regarded, i.e. there is a causal relationship between the cause and effect, the results might not be valid outside the context of the specific study. Therefore, scalability from small, individual tasks to large tasks performed by teams need to be regarded. Similarly, the transfer from e.g. students to practitioners must be elaborated for different cases.

RESEARCH QUESTION 2.
How can requirements prioritisation techniques be characterised and compared?

The second research question was examined in two separate studies. The first study aimed at comparing requirements prioritisation techniques in two controlled experiments. In total, 46 academics participated in the experiments. The design was fixed, i.e. prepared and well-defined.

Controlled experiments are fixed in nature and apply to all four validity issues described earlier. *Internal validity* is considered by isolating the treatment from other influencing factors to ensure that the outcome is actually caused by the treatment. A typical example of threats to internal validity is that the groups given different treatments already differ from each other in one way or another. This was regarded by sampling the subjects based on pre-tests so that the groups' characteristics were as similar as possible and additionally the subjects were given treatments in different orders.

Conclusion validity was regarded by plotting the data and conducting appropriate hypothesis tests, which present significance of the results. The experiments were performed with a rather small and specific set of subjects. It increases the homogeneity of the subjects and thus the conclusion validity, although it reduces the *external validity* of the experiment since the subjects are not selected from a general population. The simplified setting and task might not be scalable to industrial usage. However, it is likely that the practitioners who are intended to use the techniques would perform similarly to the subjects (in this case mainly PhD students and master's students in their final year). Therefore, it would be appropriate to evaluate the techniques in industry.

Construct validity concerns the extent to which the experiment setting reflects the construct under study. In other words, we need to consider if we measure what we want to measure. The measures need to be defined and the treatments need to be applied carefully. Time-consumption is an objective and well-defined measure, which was tested using a watch. The subjects were aware of the measure, thus there may have been an interaction between testing and treatment. Ease-of-use is a subjective and well-defined measure, which was tested after the experiment using a questionnaire. No interaction between testing and treatment was present since the subjects were not aware of the test during the experiment. Similarly, no interaction between testing and treatment was present for the subjective measure of accuracy. However, the accuracy for a prioritisation technique is less well-defined, since there is no correct prioritisation key to compare with to determine accuracy. Construct validity is regarded by testing the same measures in two separate experiments. Performing the experiment with another set of requirements or another set of subjects would further increase the construct validity.

The second study was designed as an *archive analysis* (Robson, 2002) in which prioritisation data from requirements prioritisation assignments were used to evaluate the use of different measurement scales. The design was fixed, but differs from a controlled experiment since the subjects, as well as the researchers, were unaware of the usage of the data at the time of data collection. *Conclusion validity* was regarded since statistical tests were used when appropriate, and measures and treatments are considered reliable. However, the statistical power would be higher if more subjects were involved. *Internal validity* is less applicable in this case since the subjects were unaware of the analysis. Thus, threats such as learning effects and repeated testing are reduced. Threats to *construct validity* concerns whether the used measures actually reflects what we want to measure. We believe the presented measure of skewness to be well-defined and valid for comparing prioritisation results from different distributions, since it is based on the standard deviation.

Finally, the *external validity* can be discussed. Since the data are taken from a small-scale prioritisation task performed by students and PhD students it is difficult to generalise to an industrial setting. However, we believe that the study indicates that the presented measures are ready to be evaluated in industry.

3.4 Case Studies on Retrospective Analysis for Release Planning Decision-Making

The third research question was answered primarily by flexible case study methodology and the validity issues are mainly the same as presented in Section 3.2. This section describes the validity issues considered in the three case studies investigating the developed retrospective analysis method.

CASE STUDIES. A case study is conducted to investigate a single case within a specific time space and can be either fixed or flexible. The researcher typically collects detailed information on one single project, and different data collection procedures may be applied. Case studies differ from experiments in that the variables are not being manipulated, i.e. the case study samples from variables representing the typical situation. A case study is an observational study and may be easier to plan than a controlled experiment because it requires less pre-specification. However, it may be more difficult to interpret the result and generalise to other situations. Also, the effects of a change can only be assessed at a high level of abstraction and might not be possible to measure immediately (Wohlin et al., 2000). Case study methodology typically involves multiple data collection methods such as observation, interview and documentary analysis (Robson, 2002).

VALIDITY IN CASE STUDIES. The validity for case studies depends on the specific design. In this thesis, the case studies are mainly flexible. The collected data are primarily subjective and statistical methods are not applicable. The validity for flexible designs is discussed in Section 3.2. In addition, we can discuss *analytic generalisation* for multiple case studies

(Robson, 2002). The purpose of multiple case studies is not to gather a sample of cases so that generalisation to a population can be made, but to seek complements to the first study by focusing on an area not originally covered. In that manner it is possible to develop a theory which helps understanding other cases or situations. The strategy has similarities with performing multiple experiments in an attempt to replicate or complement earlier studies. Thus, the results may either confirm the theory or lead to revision and further development of the theory.

RESEARCH QUESTION 3.
How can retrospective analysis be used to evaluate and improve the release planning process?

The third research question was investigated in three different case studies in which the retrospective analysis method PARSEQ (Post-release Analysis of Requirements SElection Quality) was developed and applied. The method aims at finding improvements for the release planning process through retrospective analysis of already developed releases. A sample of requirements that were candidates for the investigated releases is re-estimated to find release planning decisions that would be made differently in retrospect. Those incorrect release planning decisions are investigated in a root-cause analysis where reasons for incorrect decisions are discussed, and improvements are suggested.

We used a flexible design and the three participating organisations have different characteristics, which required us to adapt the method to the different situations. The organisations were selected with respect to their differences, in order to discover limitations and possibilities of the method.

Description validity was regarded by taking extensive notes. In addition, charts and diagrams that were created during workshops were saved for further analysis. Since the method is based on the participants' knowledge and experiences, it is important to select the right people. In all cases several practitioners participated, such as product managers, project managers, chief developers, system architects, and users. They were carefully selected with respect to their experience of release planning for the investigated product. *Interpretation validity* was considered in two of the three cases since two researchers participated and could discuss the results afterwards to prevent misinterpretations. The results were also fed back to the participants for validation. *Theory validity* was handled by conducting multiple case studies with different characteristics. In addition, employees from different departments and in different roles participated in all studies. *Analytic generalisation* was regarded by seeking complementary cases so that different situations could be analysed. The case studies complement each other and give a comprehensive picture of the usage of the PARSEQ method.

4. Research Results

In this section, the main research results and contributions are summarised and plans for further work are described. Section 4.1 describes contribution C1-C3 which corresponds to the research questions RQ1-RQ3. Similarly, further research FR1-FR3, in Section 4.2, corresponds to future plans for RQ1-RQ3.

4.1 Main Contributions

This section describes the main contributions in the thesis C1-C3, corresponding to the three research questions. The results are discussed in relation to related research, some of which were presented in Section 2. Validity of the results is discussed in more comprehensive terms as a complement to the detailed validity discussion from a research design perspective in Section 3.

C1: Increased understanding of RE challenges in market-driven software development

The first research question aims at discovering challenges experienced by RE practitioners in software development. A flexible survey was performed with practitioners in industry. The paper reports on findings from interviews with 14 practitioners involved in RE at eight different software-developing companies. A large number of challenging issues were found, which were organised into twelve areas. Some of the challenges are also acknowledged by other sources. For example, the difficulty of writing understandable requirements is discussed by Al-Rawas and Easterbrook (1996). Further, the communication gap between marketing staff and developers is also described by Hall et al. (2002). In addition, the problem of implementing and improving RE in an organisation is also identified by Kauppinen et al. (2002).

Hence, several of the challenges discovered in the survey have also been identified in related research. However, some challenges have not been discussed in other surveys, probably because the challenges are special to market-driven development. For example, release planning is noticed as problematic for several of our participating companies, because it is often based on uncertain estimates of cost and value. All participants discuss requirements prioritisation, although most of the companies use an *ad hoc* approach based on grouping requirements. Similarly, managing the constant flow of incoming requirements suggestions, and handling the balance between eliciting requirements from potential users and inventing new ones in-house were referred to as challenging.

Challenges confirmed by other sources increase the credibility of the results, and new ones help us increase the understanding of the RE challenges experienced by practitioners. The sampling was made with respect to the differences between participants since we wanted to discover an as broad spectrum of challenging issues as possible. It is possible that other challenges would appear if other companies participated. Thus, the picture of industrial RE challenges we provide is not a universal one. However, it has suited its purpose of increasing the understanding of the area, and helping to find research areas in need of further investigation. The area of requirements prioritisation and release planning decision-making was selected for the continued research because it was one of the four areas which were discussed by all participants and it received the highest number of quotations in the interviews.

C2a: Report on characteristics of different requirements prioritisation techniques

The contribution to the second research question is divided into two parts: C2a and C2b. The first one is investigated in experiments, evaluating the differences in time, ease of use, and accuracy between three requirements prioritisation techniques. The results suggest that Pair-wise comparisons with tool-support (Telelogic, 2006) is the fastest of the three investigated techniques, and the Planning game is the second fastest. The two mentioned techniques do not differ regarding ease of use. The manual Pair-wise comparisons technique is the most time-consuming and least easy to use among the investigated techniques. The accuracy of the prioritisation results does not differ among the techniques.

These results contradict some earlier work. In Karlsson et al. (1998), a technique similar to the Pair-wise comparisons technique is found to be superior to a technique similar to the Planning game regarding ease of use and reliability of results. However, Pair-wise comparisons is the most time-consuming technique in their evaluation, which is also true in our case. Our results are supported by Lehtola and Kauppinen (2006) who discovered in their case study that pair-wise comparisons were difficult and time-consuming to perform, especially with more than 20 requirements. Some users also argued that pair-wise comparisons are pointless and it would have been easier for them to just select the most important requirements without comparisons. On the other hand, dividing requirements into three groups, as is done in the Planning game, is often used in practice (Lehtola and Kauppinen, 2006; IEEE, 1998; Karlsson, 1996). Techniques based on grouping and ranking, such as the Planning game, may be more efficient to introduce than the Pair-wise comparisons, since grouping is often already used in practice.

In summary, two studies confirm that pair-wise comparisons is a time-consuming technique (Karlsson et al., 1998; Lehtola and Kauppinen, 2006). Therefore, we regard the results on time-consumption as trustworthy. However, the results regarding ease of use is confirmed in one study (Lehtola and Kauppinen, 2006) and contradicted in another (Karlsson et al., 1998). Therefore, further studies are needed to investigate this difference in results before validity can be assured.

C2b: Evaluation of measurement scales in requirements prioritisation

The second research question is also investigated in an archive analysis, examining the decision-support provided by different requirements prioritisation techniques using different measurement scales. The measurement scales relevant to requirements prioritisation are the ordinal scale and the ratio scale, which are further described in Part II and in (Fenton and Pfleeger, 1997).

Results from prior prioritisation exercises were re-examined in an archive analysis. Four different data sets, with 36 data points in total, resulting from prioritisation with a ratio scale technique were investigated. The purpose was twofold: to investigate the skewness of the different ratio scale prioritisation results, and to compare the cost-value approach for ordinal and ratio scale data. The paper presents a measure for the skewness based on the standard deviation from a baseline distribution. The measure indicates that some of the subjects tend to get a more skewed distribution, i.e. they use the extreme values on the ratio scale more than others. The subjects expressed that one reason for using modest values is lack of domain knowledge. The measure can be used to evaluate in which situations it is worth the added effort of using the ratio scale compared to the ordinal scale.

The evaluation of the cost-value approach compares cost-value diagrams drawn from ordinal scale data to diagrams drawn from ratio scale data. It indicates that the ordinal cost-value diagram agree substantially to the one based on ratio scale data. Thus, decisions based on ordinal scale data would be substantially similar to decisions based on ratio scale data.

The results speak in favour of using the ordinal scale, at least in situations when domain knowledge is weak and it may be sufficient with ordinal scale data. The cost-value approach can then be used as decision-support. Lehtola and Kauppinen (2006) support this view by describing that some practitioners found it difficult to estimate which number to give to factors when using a ratio scale technique. Further, they conclude that prioritisation techniques are valuable for putting a set of requirements in order, and using the results as a basis for discussion.

The presented approaches to compare ordinal and ratio scale data are novel and we need more data to confirm the conclusions. Industrial usage is needed to validate whether the ordinal scale can bring the same decision-support for practitioners as the ratio scale.

C3a: Method for retrospective analysis of release planning decision-making

The contribution to the third research question is divided into two parts: C3a and C3b. The third research question investigates how retrospective analysis can be used to improve the release planning process. A method called PARSEQ was developed for the purpose of analysing the release planning process and improve the requirements selection quality in a structured manner.

The method is evaluated in three separate industrial case studies with different characteristics. The first case examined a small software developer, the second investigated an in-house software project, and the third examined product line development for an embedded software product. In the first case a handful of improvement suggestions for the release planning process were found. The second case was found to have made successful release plans for the product and the study focused on the positive experiences from the project. In the third case, a large number of root-causes and improvement suggestions were found.

The application of the PARSEQ method differed between the cases. The first one was supported by a requirements management (RM) tool and a ratio scale prioritisation technique was used. Since we wanted a faster approach, not depending on a commercial RM tool, the ordinal scale techniques were considered. The studies for RQ2 indicated that the ordinal scale seemed sufficient for our purposes, and therefore the other two cases used a more agile approach based on the Planning game and ordinal scale cost-value diagrams.

Retrospective analysis has shown fruitful in other areas such as project management (Kerth, 2001), knowledge management (Rus and Lindvall, 2002), and agile development (Collins and Miller, 2001). Our results indicate that it is also successful in finding improvements for the release planning process. The three case studies are performed at different organisations with different characteristics. The method seems feasible for the investigated cases and therefore it is likely that it will work in other situations as well, although further cases need to be investigated to find the possibilities and limitations of the method.

C3b: Tool-support for the retrospective analysis method

Based on the experiences from the first two case studies, a tool was developed with the purpose of making the process more efficient and increase possibility of visualisation. The tool handles all steps from importing a sample of requirements to exporting process improvement suggestions. The tool uses a number of windows to guide the user through the process. The re-estimation can be performed with one of three different requirements prioritisation techniques: the Planning game, the $100 method, and the Pair-wise comparisons. Two criteria of ones own choice can be entered: one to maximise and one to minimise, e.g. value and cost. After re-estimation, the tool illustrates the results in a cost-value diagram, which is then used for analysis. The discussion regarding root-causes and improvement suggestions can be documented in a root-cause matrix, which in turn can be exported along with the cost-value diagram from the tool.

The tool was used and evaluated in the third case study. The evaluation shows that it did speed up the process and decrease manual labour. It was valuable to be able to select prioritisation technique and criteria during the workshop. It also provided good visualisation support through the automatically generated cost-value diagram. However, some drawbacks were also found, such as lack of support for distributed workshops. Further development is needed before it can be used as support in all steps of the method.

4.2 Further Research

This section describes how the research can be continued and evolved in the future. All included papers have possibilities of deeper investigation, which is further detailed in the different parts. This section is arranged in three sub-sections, describing further work for each research question.

FR1: Increasing survey sample with focus on diversity and good experiences

In order to provide a more comprehensive picture of practitioners' RE challenges, it would be valuable to add further interviews with people in other organisations. Although the sample in the conducted qualitative survey is broad, additional medium-sized organisations would further extend the sample. Organisations that develop embedded systems, as well as products sold on consumer markets would further increase the range of the sample. The agile development approaches are gaining land and further experiences from agile and incremental development would be valuable.

The performed study focused on challenges in RE. However, it could be even more valuable for practitioners to report on good experiences from successful projects. Therefore, future studies could focus on projects and organisations which can share their knowledge and demonstrate encouraging examples of solutions to the stated challenges. A large scale study of that kind could end up in a guidebook, where practitioners could recognise challenges and find examples of possible solutions based on industrial experiences.

FR2: Industrial validation of requirements prioritisation techniques

The results from the investigation on requirements prioritisation techniques would benefit from industrial validation. It is not certain that the results achieved in the papers are valid for industrial use since the scale and domain are different. The number of requirements investigated is small compared to most situations in a real project and the domain of high-level requirements for mobile phones is a simplified representation of real requirements. An industrial case study could involve a combination of the investigated techniques, i.e. to use a simple technique (such as the Planning game) for assigning requirements in groups and then use a more rigorous technique (such as Pair-wise comparisons) for the requirements that need more detailed evaluation.

There are other techniques that could be compared in further experiments, such as Wiegers' method and the $100 method. Both are based on a ratio scale but Wiegers' method takes several criteria into consideration, while the $100 method focuses on one criterion at the time. The criteria in Wiegers' method are customer value, penalty if the requirement is not implemented, cost of implementation, and risks. An interesting procedure would be to compare Wiegers' method to the cost-value approach, where cost and value are estimated with e.g. $100 method, to evaluate which one gives more valuable support for release planning. In addition, time-consumption and ease of use can be measured.

To further validate the usage of different measurement scales for requirements prioritisation, data sets from industrial requirements prioritisation sessions may be used. It would then be possible to investigate whether some people use the more extreme values on the ratio scale, while others are more modest, also in industrial requirements prioritisation. Industrial requirements prioritisation data could also be used to further evaluate the usage of the ordinal cost-value diagrams. It may be possible to set up a case study in industry to evaluate if the practical decision-support achieved by the ordinal scale prioritisation techniques is sufficient. Interviews could then reveal the practitioners' opinions after using different techniques.

FR3: Possibilities and limitations for the PARSEQ method

The PARSEQ method could be applied in additional organisations to examine its possibilities and limitations. The method need to be adaptable to different situations such as different development approaches, different project types, and different product types. Therefore, organisations and projects with different characteristics need to be involved. If a large number of case studies is performed, the method could be used as a ground for finding general improvement areas to the release planning process. It may be possible to see patterns between certain organisational characteristics and certain process improvements. In that case, general recommendations could be developed regarding release planning process improvements.

The tool support for the method also needs more evaluation and improvement. Improvements are needed e.g. to be able to perform PARSEQ in a distributed manner. Modifications can be developed as part of a Master's student project. Thereafter, the tool needs industrial validation in further case studies.

Available resources for future research

As discussed above, this research can be carried on in a number of ways. To assist others who aim at investigating this area we recommend the following available resources. This thesis is available online along with the included publications at http://serg.telecom.lth.se/research/publications/. In addition, the design and other material used in the experiments in Part II can be found at http://serg.telecom.lth.se/research/packages/ReqPrio/. The PARSEQ tool and source code, along with guidelines and the development report, can be downloaded from http://serg.telecom.lth.se/research/packages/ParseqTool.