TKK Dissertations 15
Espoo 2005

INTRODUCING REQUIREMENTS ENGINEERING INTO PRODUCT DEVELOPMENT: TOWARDS SYSTEMATIC USER REQUIREMENTS DEFINITION

Doctoral Dissertation

Marjo Kauppinen

Helsinki University of Technology
Department of Computer Science and Engineering
Software Business and Engineering Institute

TKK Dissertations 15
Espoo 2005

INTRODUCING REQUIREMENTS ENGINEERING INTO PRODUCT DEVELOPMENT: TOWARDS SYSTEMATIC USER REQUIREMENTS DEFINITION

Doctoral Dissertation

Marjo Kauppinen

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Computer Science and Engineering for public examination and debate in Auditorium AS2 at Helsinki University of Technology (Espoo, Finland) on the 18th of November, 2005, at 12 noon.

Helsinki University of Technology
Department of Computer Science and Engineering
Software Business and Engineering Institute

Teknillinen korkeakoulu
Tietotekniikan osasto
Ohjelmistoliiketoiminnan ja -tuotannon laboratorio

Distribution:
Helsinki University of Technology
Department of Computer Science and Engineering
Software Business and Engineering Institute
P.O. Box 9210
FI - 02015 TKK
FINLAND
Tel. +358-9-4511
Fax +358-9-451 4958
URL: http://www.soberit.tkk.fi/
E-mail: reports@soberit.tkk.fi

ISBN 951-22-7922-3
ISBN 951-22-7923-1 (PDF)
ISSN 1795-2239
ISSN 1795-4584 (PDF)
URL: http://lib.tkk.fi/Diss/2005/isbn9512279231/

TKK-DISS-2060

Otamedia Oy
Espoo 2005

HELSINKI UNIVERSITY OF TECHNOLOGY P.O. BOX 1000, FI-02015 TKK http://www.tkk.fi	ABSTRACT OF DOCTORAL DISSERTATION

Author Marjo Kauppinen

Name of the dissertation

Introducing Requirements Engineering into Product Development:
Towards Systematic User Requirements Definition

Date of manuscript	Date of the dissertation 18.11.2005
☐ Monograph	☑ Article dissertation (summary + original articles)

Department	Computer Science and Engineering
Laboratory	Software Business and Engineering Institute
Field of research	Software Engineering
Opponent(s)	Professor Ilkka Haikala and Professor Jukka Paakki
Supervisor	Professor Reijo Sulonen
(Instructor)	

Abstract

Without knowing the requirements of customers and users, it is difficult to build the right product. Although requirements engineering (RE) is considered a critical activity in product development, the state of RE practices seems to be immature in many organizations. For several years, researchers have tried to understand why so many companies have informal RE processes and why it is so difficult to introduce RE technology into mainstream practice.

This thesis investigates how RE can be introduced into organizations that develop market-driven products. The results are based on the experiences gathered from four Finnish organizations that considered it essential to improve their product development processes by investing in RE. To gain a deep understanding of RE process improvement in real product development contexts, we conducted four longitudinal case studies using an action research approach.

One of our main findings is that introducing RE into product development appears to involve a cultural change. By this we mean that development personnel need to adopt a new way of thinking and working when defining requirements systematically from the customers' and users' points of view. Furthermore, this cultural change involves such human factors as beliefs, attitudes, motivation, and commitment of development engineers and managers.

One way of supporting the cultural change is to define a simple RE process model that links business goals to technical requirements via user needs and user requirements. The purpose of the process model is to give an overview of RE, support communication by providing common terminology, and emphasize the importance of systematic user requirements definition. On the basis of the lessons learned from the four case studies, we also recommend a set of RE practices that support the systematic definition of user requirements. Furthermore, the thesis provides a model of factors that affect organization-wide implementation of RE practices and describes challenges organizations may face when introducing RE into product development.

The main conclusion drawn from this work is that changing the perspective from technical requirements to user requirements can be difficult for product development personnel. Furthermore, it can take several years for the cultural change towards systematic user requirements definition to spread throughout the whole product development organization. However, the experiences from the case studies show that the organization-wide adoption of RE practices can be enhanced by offering Just-in-Time training and an RE expert's assistance for development teams when they are defining user requirements for the first time.

Keywords requirements engineering, software process improvement, user requirement, user need, action research

ISBN (printed) 951-22-7922-3	ISSN (printed) 1795-2239
ISBN (pdf) 951-22-7923-1 (PDF)	ISSN (pdf) 1795-4584 (PDF)
ISBN (others)	Number of pages 110 + publications

Publisher Software Business and Engineering Institute

Print distribution Software Business and Engineering Institute

☑ The dissertation can be read at http://lib.tkk.fi/Diss/ 2005/isbn9512279231/

Preface

"Doing research is hard work. It is also fun and exciting. In fact, nothing can compare to the joy that comes from discovery."

- Anselm Strauss and Juliet Corbin (1997)

Moments of discovery are not based only on a huge number of lonesome hours, but also arise as a result of discussions with colleagues and fellow researchers. This work contains significant influences from the people who have shared their views and knowledge with me, and I am deeply indebted to all of them.

First and foremost, I wish to express my deepest gratitude to Professor Shosta Sulonen, who, as the supervisor of this dissertation, supported and encouraged me throughout the long process. The discussions with Shosta opened my eyes many times and inspired me to see things in a new way.

The one person who made the most impact on the ideas behind this work is my dear colleague Sari Kujala. We have worked together for the past seven years, being at the same time each other's most supportive partner and fiercest critic. Sari, you are given my warmest thanks – you know how much I appreciate your exceptionally close cooperation.

I would also like to express my thanks to rest of the QURE team: Tapani Aaltio, Jarmo Hurri, Laura Lehtola, Virve Leino and Teemu Seppälä. I have really enjoyed working with you. You have contributed to this work by participating in data collection, reading my research reports and providing many invaluable comments and insights. Special thanks go to Laura for her stimulating questions and the fruitful discussions we had together during the hectic writing process.

Beyond the QURE team, I am grateful to Tero Kojo, Professor Tomi Männistö and Docent Timo Soininen, with whom I have had many enjoyable and interesting discussions. Working with them has also raised my interest in product families – a new and exciting research area. I also wish to thank Helena Hölsö, Anneli Kaarresalo and Jonna Lehtola for their everyday support in practical things at work. The coffee breaks we enjoyed were especially important to me. Without them, my writing process would have been much less interesting.

Professor Jyrki Kontio helped me to discover the fascinating world of research methods. I am really grateful that he asked the many challenging questions that forced me to better understand, for example, the principles of the action research approach. Jyrki also recruited me to the QURE project. It is possible that without him, I would still be working in industry, struggling to work out how to combine research work with practical product-development work.

I would also like to thank the pre-examiners of my thesis, Professors Jukka Paakki and Björn Regnell, for their valuable comments. Their questions helped me look at my research work from a new perspective.

I express my kind thanks to Ruth Vilmi, who has done an excellent job checking the English of my individual publications and this thesis. With her support, I have learnt that the key characteristics of scientific writing are clarity and simplicity.

As an action researcher, I have had a great opportunity to work with experienced practitioners in many organizations. I would like to thank the industrial partners of the QURE project for their fruitful cooperation. This thesis would have been much less interesting without their cooperation.

This work was funded mainly by the National Technology Agency of Finland (Tekes) and the Helsinki Graduate School in Computer Science and Engineering (HeCSE). The support of both is gratefully acknowledged.

I would like to express my warmest gratitude to my family and all my friends for making my life rich and joyful. My dear parents, Sirkka and Kyösti, have always supported me. My wonderful sisters, Mervi and Päivi, encouraged me to move forward with my thesis. I am, however, especially happy with the moments during the writing process when I spent time with them and their family members, Kalle, Iida, Jarno, Konsta and Timo. Such moments helped me to forget my thesis and research work for a while.

Finally, special thanks go to my dearest friend Immo. His great sense of humour helped me through the difficult moments along the way. I hope we can now spend more time on all those things that might be even more exciting than doing research. Moments of discovery make the hunting of them worthwhile, but football, especially English football, is a matter of life and death.

Helsinki, October 2005

Marjo Kauppinen

Supervisor

Professor Reijo Sulonen
Department of Computer Science and Engineering
Helsinki University of Technology
Espoo, Finland

Reviewers

Professor Jukka Paakki
Department of Computer Science
University of Helsinki
Helsinki, Finland

Associate Professor Björn Regnell
Department of Communication Systems
Lund University
Lund, Sweden

Opponents

Professor Ilkka Haikala
Department of Information Technology
Tampere University of Technology
Tampere, Finland

Professor Jukka Paakki
Department of Computer Science
University of Helsinki
Helsinki, Finland

TABLE OF CONTENTS:

List of Publications

This thesis is based on the following publications, which are referred to in the text by Roman numerals. The papers are reprinted with permission from the publishers: INCOSE, Springer Science and Business Media, IEEE Computer Society, and Elsevier.

I Kauppinen, M. and Kujala, S. (2001) Assessing Requirements Engineering Processes with the REAIMS Model: Lessons Learned. Proceedings of the Eleventh Annual International Symposium of the International Council on Systems Engineering (INCOSE2001), 7 pages in the electronic proceedings.

II Kauppinen, M., Aaltio, T., and Kujala, S. (2002) Lessons Learned from Applying the Requirements Engineering Good Practice Guide for Process Improvement. Proceedings of the Seventh European Conference on Software Quality (QC2002), Springer-Verlag, pp. 73-81.

III Kauppinen, M. and Kujala, S. (2001) Starting Improvement of Requirements Engineering Processes: An Experience Report. Proceedings of the 3rd International Conference on Product Focused Software Process Improvement (Profes2001), Springer-Verlag, pp. 196-209.

IV Kauppinen, M., Kujala, S., Aaltio, T., and Lehtola, L. (2002) Introducing Requirements Engineering: How to Make a Cultural Change Happen in Practice. Proceedings of the IEEE Joint International Requirements Engineering Conference (RE' 02), IEEE Computer Society, pp. 43-51.

V Kauppinen, M., Vartiainen, M., Kontio, J., Kujala, S., and Sulonen, R. (2004) Implementing Requirements Engineering Processes throughout Organizations: Success Factors and Challenges. Information and Software Technology, 46(14), 937-953.

The author of this thesis is the principal author of all the included publications and is responsible for writing all the text of the papers except the subsection related to active user needs elicitation in Paper IV. Sari Kujala wrote this subsection and also participated in all the phase of the research related to this work. Furthermore, she made valuable suggestions to improve all the included publications. Tapani Aaltio participated in the research related to Paper II and IV. He also provided suggestions to improve both of these publications. Laura Lehtola participated in the research related to Paper IV, and made improvement proposals for it. Matti Vartiainen helped in finding an appropriate structure and contributed several important insights to Paper V. Jyrki Kontio and Reijo Sulonen made valuable suggestions to improve Paper V.

1 Introduction

For successful growth and an increase in company profits, it is not enough only to develop products, but to develop products that satisfy the needs of the products' intended users. Discovering real user needs and transforming them into requirements can, however, be challenging. For example, the Standish Group's research (1994) indicates that lack of user involvement, incomplete requirements and changing requirements are the major reasons why information technology projects do not deliver all of their planned functionality on schedule and within budget.

Requirements engineering (RE), a branch of software engineering, offers models, methods, and practices for organizations that want to define and manage requirements systematically. This thesis investigates how organizations can introduce requirements engineering into product development and why it might be difficult to apply RE technology in practice.

In this section, the background of the thesis is first covered by describing why requirements engineering is important and what the state of RE practice is. Section 1.2 then presents a set of key terms that are necessary to communicate the scope and contributions of this work. The following three sections define the research focus, state the research problem and describe how this problem was studied. The main contributions of this thesis are briefly described in Section 1.6. Finally, Section 1.7 gives an overview of the structure of the thesis.

1.1 Background and motivation

Systematic requirements definition and management have long been recognized to be a crucial part of software systems development. Scharer pointed out in 1981 that good requirements definition can be an important factor in the success of system development projects. Similarly, Brooks (1987) states in his classic essay that the most important function that the software builder performs for the client is the iterative extraction and refinement of product requirements.

The value of good requirements and the criticality of defining them well have increased dramatically with the size and complexity of the systems being developed (Dorfman 1997). In the same way, Nuseibeh and Easterbrook (2000) point out that effective requirements engineering will continue to play a key role in determining the success or failure of projects and in determining the quality of systems that are delivered.

The high-level purpose of requirements engineering is to ensure that customer and user needs steer the product development team towards building the right system. Broadly speaking, software systems requirements engineering is the process of discovering the purpose for which the software system is intended by identifying stakeholders and their needs, and documenting these in a form that is amenable to analysis, communication, and subsequent implementation (Nuseibeh and Easterbrook 2000). The RE process covers all activities related to the elicitation, analysis, documentation, validation, and management of requirements throughout the product development life cycle. Requirements engineering is also concerned with how requirements relate to business processes, project management, architecture design, and testing.

A systematic RE process can be the key to successful product development, but poorly defined requirements can lead to poor product quality, schedule and budget overruns. According to the Standish Group's widely cited research (1994), the three major contributors to project success are user involvement, executive management support and a clear statement of requirements. In addition, Verner et al. (1999) report on the basis of structured discussions with twenty senior software development professionals that well-defined requirements are one of the major factors leading to success in software development projects. Nearly 50% of the respondents had mentioned good requirements gathering when discussing successful projects.

Systematic requirements definition can also improve cycle times and productivity. The results of Blackburn et al. (1996) show that faster firms and firms with higher productivity devote more time and effort in the early stages of the project to learning what customers want in a software product and shaping the specifications to meet those needs. Furthermore, the research of Blackburn et al. (2000) strongly indicates that additional time and effort in the early stages of a project result in reduced time-to-market and greater coding productivity.

Organizations that implement effective RE processes can enjoy multiple benefits (Wiegers 1999a). Firstly, a requirements specification, which is the primary output of requirements engineering (Hsia et al. 1993), defines what the system is to do and provides the basis for software specification and design. The second benefit of RE processes is that well-documented requirements support verification and validation. In other words, a requirements specification can be an instrument to check whether the implemented system is what has been ordered (Bubenko 1995).

The third benefit of systematic RE processes and perhaps the greatest reward comes from reducing rework during the later stages of development (Wiegers 1999a). RE is a particularly important stage of the software process as errors at this stage lead to later problems in the system design and implementation. For example, Davis (1993) compiled the results of three empirical studies, indicating that it may be up to two-hundred times more expensive to detect and repair errors in the maintenance phase, compared to detecting and repairing them during the requirements engineering phase. Similarly, Lubars et al. (1993) point out that organizations can save a lot of time and money if they can detect and correct a fraction of the errors already in the requirements phase.

Although RE is considered a critical activity of product development work, organizations do not seem to have explicitly defined RE processes, and so their RE practices can be considered immature (e.g. Hsia et al. 1993; Sommerville and

Sawyer 1997; Stevens et al. 1998; Juristo et al. 2002). For example, Sommerville and Sawyer (1997) state on the basis of their experience that very few organizations have an explicitly-defined RE process. Recently, Sommerville and Ransom (2005) have repeated the same statement by saying that a very large number of organizations have poorly defined and informal RE processes. Their empirical study confirmed their statement by showing that the RE maturity level of all the nine companies that participated in the study was low. Similarly, Juristo et al. (2002) report on the basis of their survey that immaturity still defines current RE practices.

Researchers began thinking about how to discover, analyze and formally specify system and software requirements in the 1970s. For example, Bell and Thayer (1976) found in their empirical study that problems with requirements are frequent and important. They pointed out that new software engineering techniques are clearly needed to improve both the development and statement of requirements. During the last thirty years, researchers have developed a wide range of methods and techniques to solve RE problems. According to van Lamsweerde (2000), most of the research has been devoted to techniques of modelling and specification. Similarly, Nuseibeh and Easterbrook (2000) report that formal description techniques have received considerable attention in RE research, but have not yet been widely adopted into RE practice.

In 1993, Hsia et al. pointed out that there is a wide gap between RE research and RE practice. Over the last ten years, the RE research community has tried to understand why it is so difficult to introduce RE research results into practice and how to support technology transfer from research to practice (e.g. Davis and Hsia 1994; Siddiqi and Shekaran 1996; Miller 1997; Morris et al. 1998; Kaindl 2000; Kaindl et al. 2002; Pinheiro et al. 2003).

The technology transfer problem does not concern only RE research. The issues related to it have been discussed in software engineering as well (e.g. Potts 1993; Pfleeger 1999; Pfleeger and Menezes 2000). According to Potts (1993), software engineering research is often more solution-driven than problem-focused. He also states that real-world problems are seldom found where researchers expect them to be. Similarly, Davis and Hickey (2002) state that many RE researchers fail to understand current practices. Both Potts (1993) and Davis and Hickey (2002) recommend researchers close involvement with industrial projects, which lets researchers emphasize what people actually do or can do in practice, rather than what is possible in principle.

The aim of this thesis is to investigate how RE can be introduced in real product development contexts. The focus is on product development organizations that do not have systematic RE processes and practices in place. The particular interest is in the improvement of market-driven RE processes. In other words, the thesis concentrates on organizations that develop software-intensive products for a large market of customers and users.

1.2 Key concepts

This section describes a set of key terms that are necessary for understanding the research questions of this thesis and communicating the scope and contributions of the work. In addition to the key concept descriptions, the thesis includes a glossary of the terms in Appendix A.

We have applied three principles when defining the key terms. First, we have attempted to use the existing definitions found in the software engineering and requirements engineering literature. Secondly, we have attempted to include definitions that cover all important aspects of a particular concept and are unambiguous enough from the perspective of research. Thirdly, we have also included definitions that are simple enough from the perspective of practitioners. The second and third principles are, to some extent, conflicting. Therefore, we have included two definitions for some of the terms in the glossary (Appendix A).

1.2.1 Requirements

The literature offers many definitions of the term "requirement" (e.g. Abbott 1986; IEEE Std 610.12 1990; Davis 1993; Thayer and Thayer 1997). These definitions differ in their emphasis, which indicates that no common view of the term exists. Here we represent three definitions (Abbott 1986; IEEE Std 610.12 1990; Davis 1993) to highlight different aspects of requirements.

The IEEE Standard Glossary of Software Engineering Terminology (IEEE Std 610.12 1990) defines the term as:

1. A condition or capability needed by a user to solve a problem or achieve an objective.
2. A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.
3. A documented representation of a condition or capability as in (1) and (2).

This definition emphasizes that requirements must be documented. In addition, it points out that there are two different sources from which requirements can be derived. In the first part of the definition, the origin of requirements lies in user needs, while the second part of it recognizes that requirements can also be derived from a formal document such as a contract or standard.

Types of requirements

Davis (1993) defines a requirement as:

> "A user need or a necessary feature, function, or attribute of a system that can be sensed from a position external to that system."

The Davis' definition emphasizes that requirements describe external behaviour of a system. In addition, it lists four different kinds of requirements: user need, feature, function and attribute. Abbott (1986) provides a definition that resembles the Davis' definition. He defines a requirement as

> "Any function, constraint, or other property that must be provided, met, or satisfied to fill the needs of the system's intended user(s)."

This definition emphasizes that a system must satisfy user needs. It also lists three different types of requirements: functions, properties, and constraints.

Robertson and Robertson (1999) also classify requirements into three categories: functional requirements, non-functional requirements, and constraints. Furthermore, according to Brackett (1990), requirements cover not only the desired functionality of a system or software product, but also address non-functional issues, constraints on the design, and constraints on the implementation. Figure 1 represents the classification of the different types of requirements that is used in this thesis. The purpose of the classification is to help understand the overloaded term "requirement".

A functional requirement specifies a function that a system or system component (i.e., software) must be capable of performing (Brackett 1990). According to Robertson and Robertson (1999), functional requirements describe what the product must do – an action that the product must take if it is to provide useful functionality for its users. Functional requirements are also called behavioural or operational requirements (Davis 1993).

Non-functional requirements are those relating to performance, reliability, security, maintainability, availability, accuracy, error-handling, capacity, ability to be used by a specific class of users, acceptable level of training or support, or the like (Brackett 1990). According to Robertson and Robertson (1999), non-functional requirements are properties, or qualities, that the product must have such as appearance, speed or accuracy.

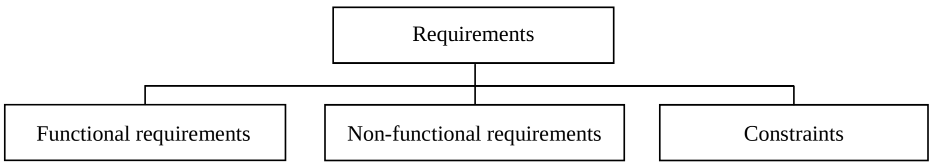

Figure 1. Types of requirements

Design and implementation constraints are boundary conditions on how the required software is to be constructed and implemented (Brackett 1990). Examples of design constraints include the fact that the software must run using a certain database system. Constraints are global requirements that restrict the way that products are produced. For example, the budget for development is a constraint as it restricts the number and sophistication of the requirements.

As a summary, the diverse definitions of the term requirement suggest that there is no universally accepted definition of what a requirement is. On the other hand, researchers of the RE community seem to agree relatively widely on the division of requirements into functional and non-functional. Many researchers also classify constraints as one of the requirements types, although they might have different views as to what constraints are.

Levels of requirements

In addition to the different requirement types shown in Figure 1, an increasing number of sources point out that there are different levels of requirements as well (e.g. Rombach 1990; Stevens et al. 1998; Wiegers 1999a, 2003; Leffingwell and Widrig 2000; Sommerville 2001). Figure 2 illustrates the requirements levels found in the literature. Furthermore, Appendix B provides definitions for the different terms summarized in Figure 2.

	Rombach (1990)	**Stevens et al. (1998)**	**Wiegers (1999a, 2003)**	**Leffingwell and Widrig (2000)**	**Sommerville (2001)**
Business perspective		Business requirements	Business requirements		
	Software needs			Stakeholder needs	
User perspective	Customer/user -oriented requirements	User requirements	User requirements	Features	User requirements
Development perspective	Developer-oriented requirements	System requirements	Functional requirements	Software requirements	System requirements

Figure 2. Levels of requirements

Sommerville (2001) points out that some of the problems that arise during the RE process result from a failure to make a clear separation between different levels of requirements. Figure 2 shows that there are different views as to how many levels are needed to organize requirements information. On the other hand, Figure 2 reveals that many sources recommend requirements to be defined at least from two different perspectives. Firstly, user requirements describe user goals or tasks that the users must be able to perform with the product (Wiegers 2003). They should be written using natural language and simple intuitive diagrams, because they are meant for people who do not have a detailed technical knowledge of the system (Sommerville 2001). Secondly, system requirements, which are more detailed descriptions of the user requirements, are defined from the developers' point of view. They form a model of the system that is usually too large and technical for users to understand (Stevens et al. 1998). A data-flow model and an object model are examples of such a system model.

Requirements can also be defined from a business perspective. Business requirements represent high-level objectives of the organization or customer requesting the system or product (Wiegers 1999a). A typical example of a business requirement is "The product should take 30% of the South American market by 1999" (Stevens at al. 1998).

In this thesis, we focus on user requirements. Stevens et al. (1998) state that user requirements are the first step towards defining a system, and, to be successful, every system needs to satisfy its end users. Moreover, a historical examination into the book on Software Engineering written by Sommerville (1996, 2001) reveals an increasing awareness of user requirements. In the fifth edition, Sommerville (1996) does not mention user requirements explicitly, whereas in the sixth edition (2001) he introduces the concept of user requirement and provides guidelines for writing them.

1.2.2 Requirements engineering (RE)

The literature offers a number of definitions for the term "requirements engineering" (e.g. Hsia et al. 1993; Sommerville and Sawyer 1997; Thayer and Thayer 1997; Zave 1997). The coverage and focus of these definitions vary widely. Here we present four definitions and compare them.

One of the most comprehensive definitions for RE is provided by Zave (1997):

> "Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behaviour, and to their evolution over time and across software families."

Zave's definition covers several important aspects of requirements engineering. First, it highlights the importance of real-world goals. Real-world goals describe the reasons why a system is needed, and motivate, therefore, the development of the system. Zave’s definition also emphasizes that specifications of software behaviour must be precise. In addition, it covers the evolution of specifications over time and across software families. This points to the need for requirements change management and requirements reuse.

The focus of Zave’s definition is on software engineering. Thayer and Thayer (1997) provide a definition for RE in the context of system engineering:

> “In system engineering, requirements engineering is the science and discipline concerned with analyzing and documenting requirements. It comprises needs analysis, requirements analysis, and requirements specifications.”

Thayer and Thayer’s definition focuses on a set of RE activities, whereas Zave does not express these explicitly in her definition for RE. In the same way as Thayer and Thayer, Sommerville and Sawyer (1997) provide a definition for RE that emphasizes RE activities:

> “Requirements engineering covers all of the activities involved in discovering, documenting, and maintaining a set of requirements for a system. The term engineering implies that systematic and repeatable techniques should be used to ensure that system requirements are complete, consistent, relevant etc.”

In addition to RE activities, Sommerville and Sawyer’s definition lists three characteristics of system requirements: completeness, consistency, and relevance. Furthermore, it highlights the importance of systematic and repeatable techniques. Similarly, Hsia et al. (1993) point out the systematic usage of principles and methods:

> “Requirements engineering is the disciplined application of proven principles, methods, tools, and notations to describe a proposed system’s intended behaviour and its associated constraints.”

This definition also emphasizes that principles, methods, tools, and notations should be proven. In other words, RE technology should have been evaluated to be useful in practice.

Table 1 summarizes how different aspects are covered in the four definitions of the term requirements engineering that were found in the literature. When these four definitions are compared from the perspective of research, we find the Zave’s definition the most comprehensive. From the perspective of practitioners, the Zave’s definition is found relatively long and difficult to understand. In addition, it does not specify RE activities explicitly. From the perspective of practice, we select the Sommerville and Sawyer’s definition for RE to be used in this thesis (see Appendix A) because it specifies a set of important RE activities and emphasizes the importance of systematic and repeatable usage of techniques.

Table 1. Summary of the definitions for the term “requirements engineering”. The symbol “-” means that the definition does not specify the aspect explicitly.

Aspect of RE	Hsia et al. (1993)	Sommerville and Sawyer. (1997)	Thayer and Thayer (1997)	Zave (1997)
Context	-	-	System engineering	Software engineering including requirements evolution over time and across software families
Deliverables (outcome)	Description of system’s behaviour and its associated constraints	A set of system requirements	Needs and requirements	Specifications of software behaviour including ▪ real-world goals ▪ functions ▪ constraints ▪ relationships of goals, functions, and constraints
Characteristics of deliverables	-	Complete, consistent, relevant	-	Precise
Activities	Describing system’s behaviour and its associated constraints	Discovering requirements Documenting requirements Maintaining requirements	Analyzing needs Analyzing requirements Specifying requirements	-
Methods	Disciplined application of proven principles, methods, tools and notations	Systematic and repeatable usage of techniques	-	-

1.2.3 RE processes and practices

In order to obtain a structured view of requirements engineering, we find it useful to divide it into requirements definition and requirements management (Figure 3). The outcome of requirements definition is the formulation of functional requirements, non-functional requirements, and constraints, i.e. a requirements document or a set of requirements documents. Requirements management is intended to keep track of requirement changes and ensure that changes are made to the requirements document in a controlled way (Kotonya and Sommerville 1998). In addition, it includes activities such as tracking the requirement status and keeping project plans current with the requirements (Wiegers 2003).

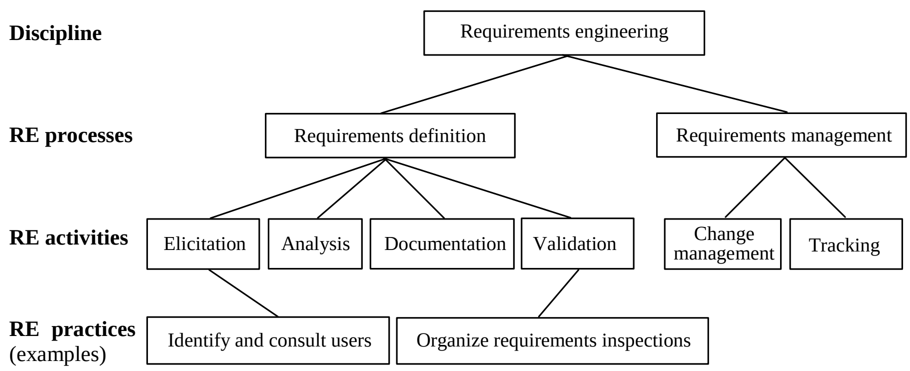

Figure 3. A hierarchical decomposition of requirements engineering

The idea for the hierarchical decomposition presented in Figure 3 comes from Wiegers (1999a). He divides requirements engineering into requirements development and requirements management. We followed Brackett (1990) and Sommerville (2001) in using the term "requirements definition" to rename requirements development. Furthermore, we included the level of RE practices in the hierarchical decomposition of RE.

Requirements definition can be further divided into elicitation, analysis, documentation, and validation activities:

- Elicitation – The activity of discovering requirements by communication with customers, users and other stakeholders who will be affected by the system and who have a direct or indirect influence on the requirements (Sommerville and Sawyer 1997). Furthermore, requirements can be discovered from system documents, domain knowledge, and market studies (Kotonya and Sommerville 1998).
- Analysis – The activity of analyzing an initial set of requirements for conflicts, overlaps, omissions, and inconsistencies (Sommerville and Sawyer 1997). In addition, different stakeholders negotiate to decide on which requirements are to be accepted (Kotonya and Sommerville 1998).
- Documentation – The development of a document that clearly and precisely records each of the requirements of the system (Thayer and Dorfman 1997). Another name for this activity is specification (e.g. Thayer and Dorfman 1997; Wiegers 2003).
- Validation – The activity of checking the requirements document for consistency, completeness, and accuracy (Kotonya and Sommerville 1998). According to Thayer and Dorfman (1997), this activity is called verification. They define verification as the process of ensuring that the software requirements specification is in compliance with the system requirements,

conforms to document standards of the requirements phase, and is an adequate basis for the architectural design phase.

The RE processes and activities provide a high-level view of requirements engineering. For practical application additional specifics are needed. We call these more-detailed actions “practices”. Practices can range from simple actions, such as setting a unique identifier to each requirement to using a method to discover and analyze requirements.

1.2.4 Software process improvement (SPI)

Over the past twenty years, there has been an increasing interest in SPI. According to Sommerville (2001), process improvement means understanding existing processes and changing these processes to improve product quality and/or reduce costs and development time. The underlying assumption of SPI is that the quality of the process directly affects the quality of the developed software. In other words, the better the software development process, the better the software system.

Through the years, different kinds of frameworks, such as the Capability Maturity Model (CMM) and the ISO/IEC 15504 standard (also known as Spice) have been developed for incremental and systematic SPI. The purpose of these frameworks is to guide organizations in assessing the current state of their process and identifying the issues critical to improving their software process and software quality.

There is also an RE-specific framework called the Requirements Engineering Good Practice Guide (REGPG), which provides a gradual approach to RE process improvement (Sommerville and Sawyer 1997). The development of the REGPG was motivated by the conviction that the most effective way to help practitioners is to help them identify how they could make best use of existing good RE practices (Sawyer et al. 1997).

The REGPG extends the principles of software process improvement into the requirements process. It draws on existing SPI models and is consistent with the CMM, SPICE and ISO 9000 (Sommerville and Sawyer 1997; Sawyer et al. 1997; Sawyer et al. 1999a). According to Sawyer et al. (1997), existing life-cycle standards and SPI frameworks are a valuable source of good practices, but the REGPG aims to provide more focused coverage of how the good practices can be integrated in an RE process, what their benefits and costs will be and what problems may be encountered.

1.3 Research problem and questions

Although RE is considered a critical activity in product development, most organizations seem not to have explicitly defined RE processes, so their RE practices can be considered immature (e.g. Hsia et al. 1993; Sommerville and Sawyers 1997; Stevens et al. 1998; Juristo et al. 2002). Over the years, the RE research community has tried to find reasons why the state of RE practice is poor in many organizations and why RE technology does not find its way to these organizations. (e.g. Hsia et al. 1993; Morris et al. 1998; Kaindl et al. 2002; Pinheiro et al. 2003).

The goal of this study is to investigate how RE can be introduced into product development organizations that do not have systematic RE practices in place, but are willing to start RE process improvement. The particular interest of the study is in user requirements. To our knowledge, the RE literature provides little empirical data on how user requirements can be defined systematically in real product development projects.

This research is based on the assumption that organizations need well-defined processes in order to manage product development projects and develop high-quality systems. Therefore, a process improvement approach is applied to introduce RE into product development. We considered three frameworks for supporting systematic RE process improvement: 1) the CMM (Paulk et al. 1997) combined with the IDEAL model (McFeeley 1996), 2) the ISO/IEC 15504 standard (ISO/IEC TR 15504-1,2,3,4,7:1998), and 3) the REGPG (Sommerville and Sawyer 1997).

We selected the REGPG because, in comparison with the CMM and ISO/IEC 15504, it focuses specifically on RE processes and contains, therefore, a larger set of RE practices. Furthermore, it provides more-detailed guidance on the RE practices than the CMM and ISO/IEC 15504. Although the REGPG offers practical information for RE process improvement, we could not find any empirical studies on it at the time we started our research on RE process improvement in 1998. From the research point of view, this motivated us to evaluate the REGPG in real product development organizations.

To make the research problem of how organizations that do not have systematic RE practices can introduce RE into product development more precise, it has been divided into the following four research questions:

1. How does the REGPG (Requirements Engineering Good Practice Guide) help organizations in RE process improvement?
2. Which practices support systematic user requirements definition?
3. What are the potential success factors that support the introduction of RE into product development?
4. Which challenges do organizations face when they introduce RE into product development?

Table 2 shows how the research questions are covered in the different publications of the thesis. The first two publications focus on the REGPG. The third paper describes improvement experiences gathered during the early phases of the improvement projects, and the fourth one evaluates factors that support and prevent the cultural change towards systematic definition of user requirements. Finally, the fifth paper covers issues that relate to the organization-wide implementation of RE processes. Table 3 summarizes the objectives of the publications.

Table 2. Relations of the research questions and the publications

Id	Research question	Publication				
		I	II	III	IV	V
1	How does the REGPG help organizations in RE process improvement?	x	x			
2	Which practices support systematic user requirements definition?		x	x	x	
3	What are the potential success factors that support the introduction of RE into product development?			x	x	x
4	Which challenges do organizations face when they introduce RE into product development?				x	x

Table 3. Research objectives of the publications

Id	Name of publication	Objective
I	Assessing Requirements Engineering Processes with the REAIMS Model: Lessons Learned	Evaluate the usefulness of the REAIMS maturity model, which is one of the main components of the REGPG.
II	Lessons Learned from Applying the Requirements Engineering Good Practice Guide for Process Improvement	Identify the strengths and weaknesses of the REGPG.
III	Starting Improvement of Requirements Engineering Processes: An Experience Report	Investigate the factors that influence the success of the RE process improvement at the beginning of the improvement projects.
IV	Introducing Requirements Engineering: How to Make a Cultural Change Happen in Practice	Evaluate the factors that support and prevent the cultural change towards systematic definition of user requirements.
V	Implementing Requirements Engineering Processes throughout Organizations: Success Factors and Challenges	Examine the factors that affect the organization-wide implementation of RE processes.

1.4 Research scope and focus

The research presented in this thesis was carried out within the discipline of software engineering (SE), which is concerned with all aspects of software production from the early stages of system specification through to maintaining the system after it has gone into use (Sommerville 2001). The focus of the research is requirements engineering, which is a sub-discipline of software engineering.

The particular interest of the thesis lies in how to introduce RE into product development organizations that do not have systematic RE processes and practices. This research is based on the assumption that organizations need well-defined processes in order to manage product development projects and develop high-quality systems. Therefore, a process improvement approach is applied to introduce RE into product development.

Requirements engineering focuses on the early phases of software development, where decisions are made on what to implement by the software system, and where the foundation is laid for the later phases that determine how to implement it (Regnell 1999). In this thesis, we concentrate on the requirements definition process and do not cover the requirements management process. The assumption behind this restriction is that the quality of the requirements definition process affects the quality of the requirements management. In other words, the better requirements are defined in the first place, the fewer requirement changes will occur during the later phases of product development.

This study does not focus only on pure software systems but also covers products that include both software and hardware. Therefore, product development and systems engineering form the wider context of the research in addition to software engineering. However, these two disciplines are excluded because including them would have extended the scope of the thesis considerably.

Similarly, management science is intentionally left beyond the scope of this thesis, although it has a long tradition of investigating issues related to organizational development and change. This restriction was made to keep the focus sufficiently narrow.

Although requirements engineering is a sub-discipline of software engineering, we assume that RE literature offers methods and practices that are suitable not just for pure software systems but also for software-intensive systems that include hardware as well. This assumption is based on the following RE principles: The system is seen as a black box during user requirements definition; requirements can be implemented by either software or hardware or both.

All in all, this thesis focuses on requirements engineering. The particular interest is in systematic user requirements definition. In addition, the thesis utilizes the process improvement approach from the SE discipline. The research covers the RE process improvement from the assessment of existing RE practices to the organization-wide implementation of new RE practices.

1.5 Research methodology

The aim of this research was to investigate how organizations can introduce RE into product development. First of all, the purpose of the research was to understand real-life problems in organizations that are interested in starting to improve their RE processes. Moreover, we wanted to conduct detailed field studies in order to gain experience and construct a picture of the situation in which practitioners apply RE practices. Therefore, this work was conducted in real product development organizations.

This section first gives an overview of the research project and the case organizations in which the research work was conducted. Furthermore, it describes the research approach and methods used. Finally, issues concerning internal and external validity are addressed.

1.5.1 Research context and case organizations

This research was conducted in the QURE (Quality through Requirements) project. The QURE project started in January 1999 and ended in September 2002. It was performed at the Software Business and Engineering Laboratory, which is a unit of the Department of Computer Science at the Helsinki University of Technology. QURE was funded by the National Technology Agency of Finland (Tekes) and industrial partners. The aim of the project was both to help the industrial partners to improve their RE practices and to conduct research work into requirements engineering. The high-level research goal of the project was to investigate how organizations can develop products that better satisfy customer and user needs.

The experience drawn on in this thesis comes from work with four industrial partners of the QURE project. The industrial partners were medium-size or large Finnish companies (Table 4). These companies are internationally known and have a significant market share globally in their own field. All of them focus mainly on products that are developed to a large market of customers. Occasionally, they develop customer-specific systems. Companies B, C and D develop interactive systems, and the products of Company A have both real-time embedded and interactive components. The companies represent four different kinds of application domain (Table 4).

Table 4. Description of the case companies

Company	Number of employees	Application domain
A	23 600	Transportation systems for buildings (elevators and escalators)
B	1 200	Measurement systems for meteorology, environmental science and traffic safety
C	450	Information management systems for building, public infra and energy distribution designers
D	3 200	Patient monitoring systems for anaesthesia and critical care

At the beginning of the study, all the case companies had an explicitly-defined product development process, but none of them had a documented RE process at the time when the research co-operation with the QURE project began. Even though the RE processes of the case companies were not explicitly defined and the RE practices were ad hoc, they have developed successful products for years. The product development organizations had experts who understood the application domain well, and who effectively shared their knowledge with others.

However, because the product development environment had been changing, the case organizations started to improve their RE processes, rather than continue to rely on the tacit knowledge of the domain experts. The main changes in the product development environment that drove the case companies to improve their RE processes were the following:

- Products are bigger and more complex than before and therefore domain experts are unable to handle all the requirements in their heads.
- Projects are bigger and there are more people that need the tacit knowledge of the domain experts.
- Personnel changes are faster than before and it is possible that domain experts are not available throughout the product development project.
- Product development projects are faster and there is less time to correct requirements mistakes in the later phases of the projects.
- Members of product development projects work in different countries and therefore they are unable to share the tacit knowledge as easily as in the past.

All the case companies started a process improvement project in order to introduce requirements engineering into research and product development (R&D) units. The number of the employees of these product development units varied from 25 to 160. The representatives of the different units and different occupational groups, such as project managers, product development engineers, and product managers formed an improvement team in each case organization. The researchers of the QURE project became members of these teams from the very beginning of the improvement projects. The research co-operation with the case organizations lasted from 1.2 to 3.7 years (Table 5). Thus the research results described in this thesis are based on four longitudinal case studies.

Table 5. Scope of the research co-operation with the case organizations

R&D organization	Number of R&D units	Research co-operation	
		Period	Duration
A	One	Feb 1999 – Sep 2002	44 months (3.7 years)
B	Two	Feb 1999 – Sep 2002	44 months (3.7 years)
C	Three	Jun 2000 – Nov 2002	30 months (2.5 years)
D	One	Aug 2001 – Sep 2002	14 months (1.2 years)

1.5.2 Research approach, method and procedure

The need for a deep understanding of how organizations can introduce requirements engineering into product development calls for a qualitative research approach, and more precisely an action research method (Avison et al. 1999; Stringer 1999). According to Avison et al. (1999), a particular strength of qualitative methods is their value in explaining what goes on in organizations, and action research can address complex real-life problems and immediate concerns. In addition, qualitative methods permit the evaluator to study selected issues in depth and detail (Patton 1990). Also, Potts (1993) suggests an "industry-as-laboratory" research approach, where researchers identify problems through close involvement with industrial projects, and create and evaluate solutions in an almost indivisible research activity. This lets researchers emphasize what people actually do or can do in practice, rather than what is possible in principle.

According to Avison et al. (1999), action research is an iterative process involving researchers and practitioners acting together on a particular cycle of activities, including problem diagnosis, action intervention, and reflective learning. The researchers defined a simple process improvement procedure to guide the systematic RE process improvement of the case organizations. Figure 4 shows how the process improvement procedure relates to the action research activities. The process improvement procedure in Figure 4 does not show all the iterations. In practice, there were no distinct boundaries between the activities. Development and piloting in particular were interleaved, and there was a great deal of iteration between these two activities.

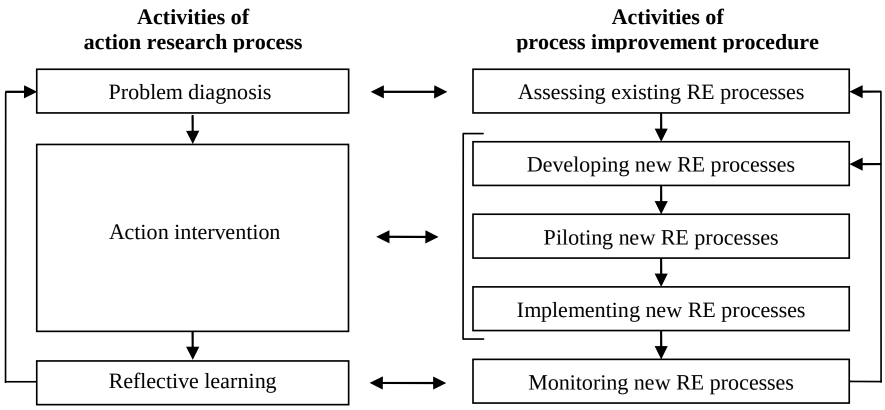

Figure 4. Action research activities and process improvement procedure

The process improvement procedure combines tasks from the IDEAL model (McFeeley 1996) and the ISO/IEC 15504 standard (ISO/IEC TR 15504-7:1998). The first activity of the procedure was to analyze the current state of RE practices and to identify their strengths and weaknesses. Based on the assessment results, the organization developed an RE process that included a set of basic RE practices. The new practices were piloted to ensure the practicality and usefulness of the developed RE process. After piloting, the new process was implemented throughout the organization. The purpose of the implementation was to make all process users aware of the new practices and to ensure they could apply them in product development projects. The last step of the improvement cycle was to collect feedback about the RE process and to evaluate the impact of the improvement efforts.

The researchers participated in all the activities of the RE process improvement procedure. The role of the researchers was to act as facilitators assisting the case organizations to improve their RE processes. During the assessment and monitoring activities, the researchers were responsible for interviewing people, for analyzing requirements documentation and for reporting the assessment results and lessons learned. They also introduced the REGPG in each case organization at the beginning of the improvement projects. During the development, piloting and implementation activities, the researchers were members of the process improvement teams. Their role was to provide information about RE practices and methods, and to support improvement actions. In addition, the researchers were also responsible for observing process improvement activities and capturing potential lessons learned.

1.5.3 Data collection and analysis

The findings of this study are based on the data collected through observations, informal conversations, formal interviews, and analysis of both RE process documentation and requirements documents (Figure 5). The purpose of the interviews and document analysis was to gain information on how practitioners defined and managed requirements in practice and what the strengths and weaknesses of the existing RE practices were. Participant observation and informal conversations provided detailed information on how the process improvement activities were performed in practice and how real product development projects applied new RE practices.

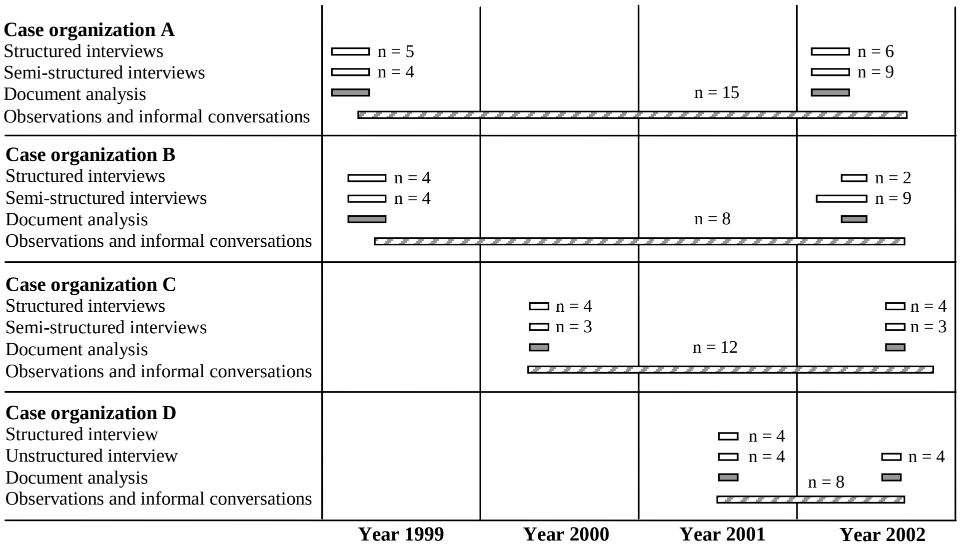

Figure 5. Data collection in the case organizations, n = number of the informants

The researchers interviewed project managers, product managers, domain experts, usability experts, product development managers, product development engineers, and persons who were responsible for coordinating RE process improvement. The total number of interviews was 43, eleven of which were group interviews. The total number of different interviewees was 53.

In addition to the interviews, the researchers collected data by observing the process improvement work of the case organizations. Figure 5 shows the number of informants that were observed during the process improvement projects. The total number of different persons that were either interviewed or observed was 28 in Organization A, 21 in Organization B, 20 in Organization C, and 10 in Organization D. All in all, the total number of the informants of this study was 79.

A preliminary understanding of the issues related to RE process improvement was gained through a literature review. The literature review led to a list of possible success factors. In addition to the four research questions, the success factors found in the RE and software process literature were used to guide the data collection. However, the data gathering was not confined to these factors; the researchers also aimed at identifying other important issues that related to RE process improvement.

The collected data were analyzed at two levels: within cases and across cases. During the within-case analysis, we wrote summaries of the findings for each case organization. During the cross case analysis, we identified the similarities and differences between the cases.

We applied the grounded theory methodology to data analysis. The grounded theory methodology is intended for researchers in various disciplines who are interested in building theory through qualitative data analysis (Strauss and Corbin, 1998). It also offers techniques and procedures to those researchers who want to do qualitative analysis but who do not wish to build theory. The grounded theory methodology includes three different types of coding procedures: open, axial and selective coding. In this thesis, we applied these coding procedures to identifying and analyzing factors that affect the successful introduction of RE processes and practices.

During the open coding phase, we first identified the concepts related to the successful introduction of RE processes and practices. After that, we grouped these concepts into categories. We identified, for example, that the usefulness of the RE process is critical to its successful introduction into product development. In addition, practitioners wanted to have a simple RE process that is easy to learn and use. We grouped these two concepts, usefulness and simplicity, into the category that is called characteristics of the RE process. According to Strauss and Corbin (1998), grouping concepts into categories is important because it enables the analyst to reduce the number of units with which he or she is working.

During the axial coding phase, we identified relationships between the identified categories. We discovered, for example, that a set of improvement activities such as piloting improve the usefulness of the RE process. According to Strauss and Corbin

(1998), it is important to discover the ways that categories relate to each other. This helps analysts to contextualize the phenomenon under study.

During the selective coding phase, we discovered a central category and refined the relationships between the identified categories. According to our research, human factors such as motivation, commitment and enthusiasm of practitioners form the main category that explains why practitioners are either willing or reluctant to change their way of defining requirements. Strauss and Corbin (1998) emphasize the identification of the central category because this has the analytical power to pull the other categories together to form an explanatory whole.

1.5.4 Internal validity

Here, internal validity means the validity of the findings within the individual case organization, i.e. it denotes the property of an empirical study where the result is consistent within its local context (e.g. Votta and Porter 1995; Berander and Wohlin 2003). We address the internal validity of the findings from four perspectives. Firstly, we used the triangulation of data sources and data collection techniques to improve the internal validity of the findings within each case organization. Our study intermixed interviewing, document analysis, informal conversations and observation. By combining different data collection techniques, we were able to cross-check findings in the individual case organization.

Secondly, to increase the internal validity of the findings, a rather large number of informants were selected using a typical case sampling strategy (Patton 1990). We interviewed 18 persons from Organization A, 17 persons from Organization B, 12 persons from Organization C, and 6 persons from Organization D. In each case organization, the interviewees were selected with the co-operation of the key informant who knew the employees of the organization. These key informants helped identify persons that had recently defined requirements for typical product development projects. Most of the interviewees were product managers, project managers and domain experts. In the case organizations, these persons were typically responsible for defining requirements.

Thirdly, the study was carried on in the case organizations over a long period, between 1.2 and 3.7 years (Table 5), which improved further the internal validity of the findings. The long-term view allowed us to verify the observations made at the beginning of the study and discover new issues that relate to the research questions.

The fourth validity issue concerns investigator triangulation. Because of the limited budget and the longitudinal nature of the study, we were able to use investigator triangulation in only a very restricted way. The author of the thesis designed interview questions. To avoid bias and misinterpretation of the questions, another researcher reviewed them. The author conducted most of the 43 interviews and analyzed the collected data. To improve internal validity, the findings were discussed with another researcher who had participated in the RE process improvement work of all the case organizations.

1.5.5 External validity

External validity is concerned with the degree of generalization, i.e. it denotes the property of an empirical study where the result is generalizable to other contexts (e.g. Yin 1994; Votta and Porter 1995; Wohlin et al. 2000). To improve the external validity of the research results, this study involved four separate organizations and the research results described in this thesis have been derived from the cross-case analysis of the findings of the individual cases. To make the findings comparable, we used the same research procedure, data collection techniques and interview questions in Organization A, B, and C. In Organization D, the duration of the research co-operation was shorter than in the other case organizations, and therefore, the problem diagnosis and reflective learning activities were lighter (Figure 5).

The case organizations were selected using a convenience sampling strategy. The organizations were the industrial partners of the QURE research project, which meant that these organizations considered RE essential. Convenience sampling is the least desirable sampling strategy (Patton 1990), and is a threat to external validity. However, we believe that the case companies represented typical companies that develop market-driven products.

To increase the external validity of the research results, we combined iterative literature reviews with empirical case studies. A preliminary literature review was conducted at the beginning of the research, and a detailed literature review after the empirical studies. This allowed us to compare the empirical observations and lessons described in the existing literature with our results, and vice versa.

Patton points out that there are advantages and disadvantages to reviewing the literature before, during, or after fieldwork – or on a continual basis throughout the study (Patton 1990). The results of the literature review may bias the researcher’s thinking and lead to “fishing”. On the other hand, the results of the fieldwork may affect what data the researcher discovers from the literature; in other words, it may similarly lead to fishing.

To decrease the bias between the literature reviews and empirical part of the study, the researchers adopted a stance of neutrality with regard to the phenomenon under study. Neutrality means that the investigator does not set out to prove a particular perspective, or manipulate the data to arrive at predisposed truths (Patton 1990). Rather, the investigator’s commitment is to understand the world as it is, to be true to complexities and multiple perspectives as they emerge, and to be balanced in reporting both confirming and disconfirming evidence (Patton 1990).

1.6 Contributions

This chapter presents a summary of the scientific and practical contributions of this research. The primary contribution is novel information about the improvement of market-driven RE processes. To our knowledge, this thesis is one of the most comprehensive empirical studies in RE process improvement. The research is based on the improvement experiences gathered from four different case organizations. Furthermore, this study is longitudinal, covering RE process improvement from the assessment of the existing RE practices to the organization-wide implementation of new RE practices.

The thesis presents an information model that clarifies the basic RE terminology. First, the model makes a clear distinction between customers and users. Furthermore, it separates user requirements from user needs and links them to business goals and technical requirements. In addition to the information model that emphasizes the importance of user needs and user requirements, the thesis provides a set of RE practices that support systematic user requirements definition. The identification of these practices can be considered one of the main practical contributions.

This research offers further insights into RE technology transfer as well. We have built a model of the factors that affect the organization-wide implementation of RE processes and practices. We believe that this model can give some explanations as to why it is difficult to transfer RE technology from research to practice. From the perspective of practice, the thesis provides a set of guidelines for RE process improvement and implementation. The guidelines can serve as a checklist for organizations that want to introduce requirements engineering into their product development. Both the model and the guidelines are based on the broad literature review and our research results.

The empirical evaluation of the REGPG in real product development contexts is one of the contributions of the thesis. Although the REGPG was published in 1997 and over 10000 copies of it have been sold (Sawyer 2004), there still seems to be little empirical evidence on how it supports organizations in RE process improvement. The thesis describes how the REGPG supports the different process improvement activities. Our results indicate that performing an assessment with the REGPG raises practitioners' awareness of RE, and motivates them to improve their RE processes.

Finally, the thesis also has an important methodological contribution. It demonstrates how relevant research in the SE and RE field can be based on action research. For the time being, the action research approach seems to be very rarely employed in RE and SE research. The thesis provides an example how to combine this scientific approach with practical process improvement. We hope that this will encourage other researchers to apply action research in their case studies.

1.7 Outline of the thesis

The remainder of the thesis is organized as follows. Section 2 provides an overview of prior literature that is relevant to this work. The research results are summarized in Section 3 and compared with previously published work in Section 4. Thereafter, the contributions of the work are presented from the perspective of research and practice, and further research areas are presented in Section 5. The five publications annexed to the thesis follow as appendices.

2 Review of the literature

This section provides an overview of the literature that is relevant to this work. The purpose of the section is to put the presented work into context and to help the reader understand the contents of this thesis. In addition, it identifies a number of sources that have made a major contribution to the study of the research questions. Figure 6 shows how this section relates to the research questions and to Section 3, which presents the main results of the study.

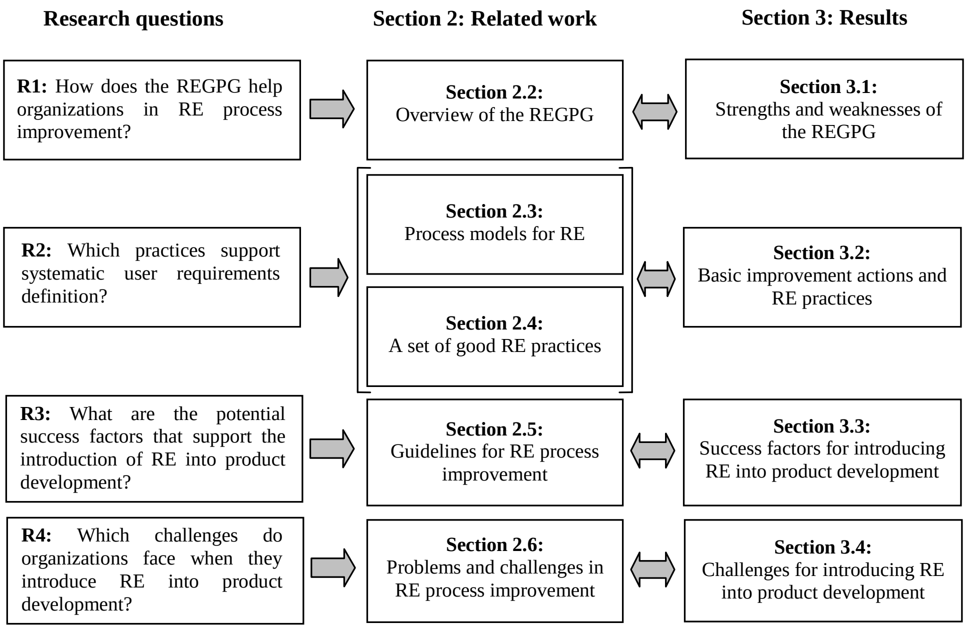

Figure 6. Structure of Section 2 and its relation to the research questions and Section 3

2.1 Relationship between literature reviews and case studies

The literature review for this thesis has been done iteratively and simultaneously with the case studies (Figure 7). According to Patton (1990), there are advantages and disadvantages to reviewing the literature before, during, and after fieldwork – or on a continual basis throughout the study. Sometimes a literature review may not take place until after data collection, because reviewing the literature may bias the researcher's thinking and reduce openness to whatever emerges in the field. Alternatively, the literature review may go on simultaneously with fieldwork, permitting a creative interplay among the process of data collection and literature review.

The iterative literature reviews supported both collection and analysis of empirical data. At the beginning of the study, the literature, especially case study reports related to software process improvement, provided lessons learned that indicated what to observe in the field. In the later stages of the study, we also compared our observations and findings systematically with the literature. In other words, the literature allowed us to extend, validate and refine our findings.

Figure 7 does not show all the review iterations. Instead, it highlights the most important results of the reviews. At the beginning of the study, the literature review focused on process improvement frameworks and models such as the IDEAL model (McFeeley, 1996), the CMM (Paulk et al. 1997), the REGPG (Sommerville and Sawyer 1997) and ISO ISO/IEC 15504 standard (ISO/IEC TR 15504-1,2,3,4,7:1998). Based on the first review, we selected the REGPG to support RE process improvement in the case organizations. In addition, we defined a process improvement procedure to guide the systematic RE process improvement of the case organizations, because the REGPG offers only very general suggestions for facilitating process change. The process improvement procedure that is presented in Figure 4 combines tasks from the IDEAL model (McFeeley 1996) and the ISO/IEC 15504 standard (ISO/IEC TR 15504-7:1998).

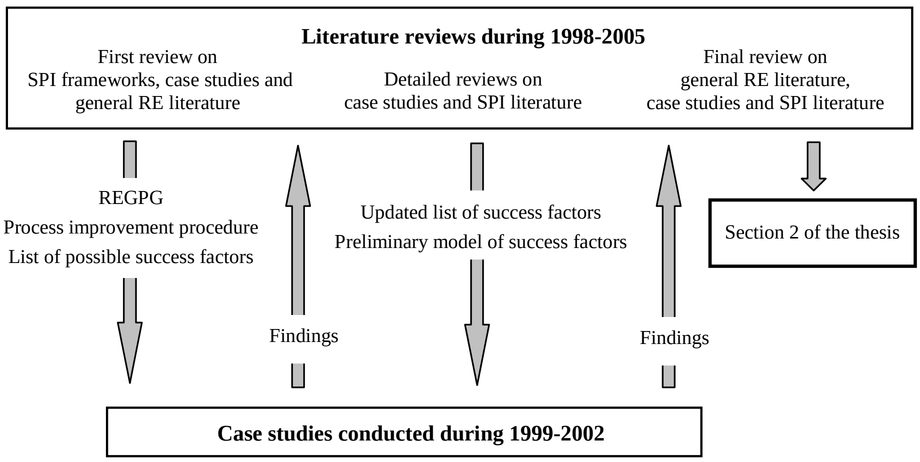

Figure 7. Iterative literature reviews performed simultaneously with the case studies

The first literature review also led to a list of possible success factors in RE process improvement. These success factors were used to guide the data collection. However, the data gathering was not confined to these factors; the researchers also aimed at identifying other important issues that related to RE process improvement. During the case studies, we updated the list of the possible success factors by analysing new case studies published in the RE and SPI literature (see Appendix C and D). Furthermore, we built a preliminary model that illustrated the relationships between the success factors. This first version was based on the analysis of the case study reports published by other researchers. We modified and extended it based on our own findings. The final version of the model is described in Section 3.3.8.

The final literature review was performed after fieldwork. The rest of this section shows the results of the final review. Section 2.2 describes briefly the REGPG, which forms one of the cornerstones of this research. A set of RE process models found in the RE literature is analysed in Section 2.3. Thereafter, a set of good RE practices is described. Section 2.5 provides guidelines for RE process improvement. These guidelines are based on factors that have been found important for successful process improvement in the RE and SE literature. Section 2.6 summarizes the most common problems encountered in the case studies of RE process improvement.

2.2 Overview of the REGPG

The REGPG is a framework for gradual RE process improvement (Sommerville and Sawyer 1997). Sawyer et al. (1999b) state that requirements processes are less well understood, less well supported by standards, and less mature than other software processes. This motivated the REAIMS project to develop the REGPG to extend the principles of software process improvement into the requirements process.

The REGPG is designed to help organizations to assess their RE processes and to plan and implement improvements (Sawyer et al. 1997). The main components are a set of good RE practices and the REAIMS maturity model (Figure 8). The REGPG also includes a simple process and three checklists for assessment. In addition, it provides a small set of general guidelines for process improvement. These improvement guidelines support the implementation of the good practices and the usage of the REAIMS maturity model, assessment process and checklists (Figure 8).

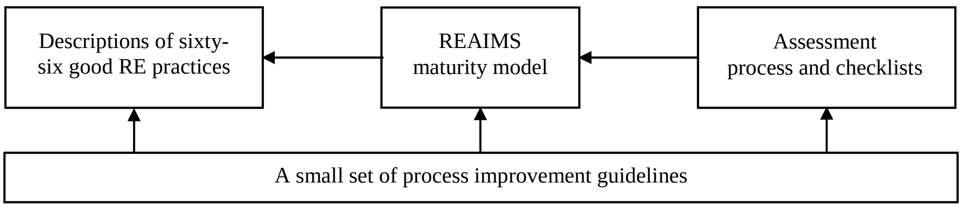

Figure 8. The main components of the REGPG

The following subsection describes why the REGPG was selected as the framework to be used in RE process improvement. After that, the components of the REGPG are described briefly. At the end of this section, we compare the REGPG with the Rational Unified Process (RUP) and Extreme Programming (XP). Both the RUP and

XP have gained a lot of attention among practitioners. Thus, it is interesting to look at what kind of RE practices they recommend.

2.2.1 Why the REGPG?

This research is based on the assumption that the quality of the process directly affects the quality of the developed system. Therefore, we selected a process improvement approach as a starting point for the study. To support systematic RE process improvement from assessment to implementation of improvements, we examined three improvement frameworks: 1) the CMM (Paulk et al. 1997) combined with IDEAL model (McFeeley 1996), 2) the ISO/IEC 15504 standard (ISO/IEC TR 15504-1,2,3,4,7:1998), and 3) the REGPG (Sommerville and Sawyer 1997).

We selected the REGPG for several reasons. Firstly, it covers RE processes and activities more widely and contains a larger set of RE practices than the CMM and ISO/IEC 15504 standard. Secondly, it provides more-detailed guidance on how the RE practices can be implemented, what their benefits and costs will be and what problems may be encountered. In addition, the REGPG draws on existing SPI models and is consistent with the CMM, ISO/IEC 15504, and ISO 9000 (Sommerville and Sawyer 1997; Sawyer et al. 1997; Sawyer et al. 1999a).

One of the reasons for selecting the REGPG as the improvement framework was that it does not require any particular RE process model to be used in an organization. In order to provide a structured view to RE, the REGPG contains a short description of RE activities and it classifies its sixty-six good practices according to these activities. Sommerville and Sawyer (1997), however, point out that there are many ways to organise RE processes.

When considering the REGPG as the improvement framework, we also found important that the REGPG supports incremental process improvement. Sommerville and Sawyer (1997) state that revolutionary approaches to process improvement cost too much and are far too risky for most organizations. Our practical experience supported this statement.

2.2.2 Good RE practices

The key idea of the REGPG is that organizations can solve problems in RE and improve the quality of their requirements documents by adopting good RE practices incrementally. The developers of the REGPG have drawn together sixty-six RE practices from existing standards, reports of requirements practices, studies of RE processes and the practical experience of their industrial partners (Sawyer et al. 1997; Sawyer et al. 1999a).

To support the gradual adoption of the RE practices, the developers of the REGPG have classified them into basic, intermediate and advanced practices. Basic practices are relatively simple activities that provide the foundation for a repeatable RE process (Sawyer et al. 1997). They are usually relatively cheap to introduce and use, and they should almost always be the first to be adopted (Sommerville and Sawyer 1997). Intermediate practices are typically more complex, but help make the RE process more systematic (Sawyer et al. 1998). Advanced practices are intended to support the continuous improvement of RE processes, or they require substantial specialist expertise. The REGPG includes 36 basic, 21 intermediate and 9 advanced practices.

In addition to the classification of basic, intermediate, and advanced practices, they are also organized according to the process deliverables or activities to which they mainly contribute (Sawyer et al. 1999a). The REGPG covers four RE activities, which are requirements elicitation, analysis, representation and validation. In addition, it provides practices for structuring and organizing requirements documents and managing requirements information throughout the development life-cycle. The REGPG also includes guidelines for system modelling and for handling requirements for critical systems whose failure can threaten human life or can significantly disrupt the running of an organization (Sommerville and Sawyer 1997). All in all, the REGPG covers eight RE areas. These RE areas and the number of the RE practices within each area are shown in Table 6.

Table 6. RE areas of the REGPG and the number of RE practices within each RE area

RE areas of the REGPG: RE process deliverable or RE activity	**Number of basic practices**	**Number of intermediate practices**	**Number of advanced practices**	**Total number of RE practices**
Requirements document	8	0	0	8
Requirements elicitation	6	6	1	13
Requirements analysis & negotiation	5	2	1	8
Requirements representation	4	1	0	5
System modeling	3	3	0	6
Requirements validation	4	3	1	8
Requirements management	4	3	2	9
RE for critical systems	2	3	4	9

For organizations starting RE process improvement, the REGPG offers the top ten practices (Table 7). Sommerville and Sawyer (1997) think that these ten practices are so important that they should be implemented in all organizations. In addition, they recommend that organizations should start their process improvement program by implementing them. According to Sawyer et al. (1997), the list of the top ten practices represents judgments they have made on the basis of existing practice as represented in standards and their partner-organizations’ experience.

To support systematic and incremental RE process improvement, the REGPG describes the good practices in the form of improvement guidelines. The guideline descriptions vary from two to six pages, and provide the following information:

- The benefits of the practice. This outlines the improvements that can be expected by adopting the practice.
- The cost of introducing the practice. This indicates the level of investment needed to integrate the practice in an existing process.
- The cost of applying the practice. This indicates the effort required to use the practice effectively once it has been introduced.
- Advice for implementation. This provides information about what the practice means and how it can be implemented.

Table 7. Top ten practices of the REGPG. Low-cost practices should involve less than 10 days of effort to introduce or apply, moderate-cost involves 10-100 days of effort and high-cost involves more than 100 effort-days (Sommerville and Sawyer 1997).

Practice	**Key benefit**	**Cost of introduction**	**Cost of application**
Use a standard structure in requirements documents.	Higher quality, lower cost of requirements documents	Moderate to high	Low
Make the document easy to change.	Reduced costs of changing requirements	Low	Very low
Uniquely identify each requirement.	Provides unambiguous references to specific requirements	Very low	Very low
Use policies for requirements management.	Provides guidance for all involved in requirements management	Moderate	Low
Use standard templates for representing individual requirements.	Requirements are presented in a consistent way so they are more understandable	Moderate	Low
Use language simply, consistently and concisely.	Requirements are easier to read and understand	Fairly low	Low to moderate
Organize formal requirements inspections.	Finds a high percentage of requirements problems	Moderate	Moderate
Use validation checklists.	Helps to focus the validation process	Low to moderate	Low
Use checklists for requirements analysis.	Faster, more complete analysis of requirements	Low to moderate	Low
Plan for conflicts and conflict resolution.	Faster resolution of requirements problems	Low	Low

Sawyer et al. (1999a) state that the guidelines are designed to help organizations make a rational assessment of which practices offer the best cost and benefit tradeoffs for their practical needs. In other words, the organization has to compare practices and make a decision about which of them are likely to be the most cost-effective for it. According to Sommerville and Sawyer (1997), this must be based on knowledge of the RE process and the process maturity level, budget and timescale for improvements, and of the people involved in implementing the RE process improvements.

2.2.3 The REAIMS maturity model

The REGPG provides the REAIMS maturity model for RE process assessment. The REAIMS maturity model has three levels that help characterize an RE process and set out a strategy for its improvement (Sawyer et al. 1999a). The levels of the REAIMS maturity model are called initial, repeatable and defined. Sommerville and Sawyer (1997) state that initial-level organizations have an ad hoc RE process and requirements problems are common. According to them, repeatable-level organizations have defined standards for requirements documents and the quality of their requirements documents should be good. Finally, organizations that are at the defined level have documented RE process models based on good practices and the quality of the requirements documents should be constantly high.

In general, initial-level organizations should focus on introducing the basic practices, and repeatable-level organizations should have implemented most of the basic practices and be in a position to implement the intermediate practices (Sawyer et al. 1997). Repeatable-level organizations should have implemented almost all basic practices and all appropriate intermediate practices, and they may improve their process by introducing the advanced practices. Sawyer et al. also point out that some organizations will find it cost-effective to try to increase their level of process maturity, while others will find it best to stay at a particular level and to improve their processes within that level.

2.2.4 Assessment process, checklists and improvement guidelines

An important activity in systematic process improvement is to assess the state of current practice (ISO/IEC TR 15504-1 1998; Zahran 1998). The assessment provides information on the basis of which organizations can set improvement goals and plan improvement actions. To help organizations in the assessment activity, the REGPG offers three checklists: one for the basic practices, one for the intermediate and one for the advanced practices (Sommerville and Sawyer 1997). The checklists help practitioners analyze how widely the RE practices of the REGPG are used in the organization. To help perform an assessment in a large organization, the REGPG also recommends the process consisting of the following activities:

- *Prune guideline checklists*. This activity is intended to identify RE practices that are obviously never used.

- *Select people to interview*. This activity is intended to identify people who know the extent to which RE practices are used in the organization.
- *Score practices against checklists*. This initial scoring should be "quick and dirty" to identify the practices that are uncontroversial and those where there is uncertainty about how widely the practices are used in the organization.
- *Resolve areas of uncertainty*. This activity is designed to clarify how widely the RE practices are used in the organization.
- *Compute process maturity*. The process maturity is calculated by summing the numerical scores for each practice. The numerical scores are presented in Table 8.

Table 8. Usage scope of the RE practices and their related scores (Sawyer et al. 1999a)

Usage scope	Description	Score
Standardized	The practice has a documented standard in the organization and is checked as a part of a quality management process.	3
Normal use	The practice is widely followed in the organization, but it is not mandatory.	2
Discretionary	Some project managers may have introduced the practice, but it is not universally used.	1
Never	The practice is never or very rarely applied.	0

The REGPG also provides some general advice for facilitating process change. It defines, for example, the following four process improvement guidelines explicitly (Sommerville and Sawyer 1997):

- Find an evangelist who can convince other practitioners to accept the changes.
- Try out changes on pilot projects and find out the advantages and disadvantages of the change.
- Allow enough time to make the change and do not use projects with very tight deadlines as pilot projects.
- Respect professional skills and emphasize that the point of changes is to help people improve the quality of their work.

2.2.5 The Rational Unified Process versus the REGPG

The Rational Unified Process (RUP) is a software engineering process that provides a disciplined approach to assigning tasks and responsibilities within a development organization (Kruchten 2004). The RUP is a specific and detailed instance of a more generic process called The Unified Software Development Process, which is described in the textbook written by Jacobson et al. (1998).

The RUP is a comprehensive framework that provides guidelines for the entire software engineering process. According to Kruchten (2004), it captures many of the best practices in modern software development and presents them in a tailorable form that is suitable for a wide range of projects and organizations. Requirements management is one of the six important best practices that are recommended by the RUP.

Requirements management is described in the RUP as a systematic approach to eliciting, organizing, communicating, and managing the changing requirements of a software-intensive system or application (Kruchten 2004). In other words, requirements engineering is called requirements management in the RUP.

Table 9 provides an overview of the requirements management activities of the RUP. These activities are mostly included in the REGPG. Furthermore, the REGPG appears to cover requirements validation practices better, for example. In addition to the six requirements management activities summarised in Table 9, the RUP provides a process model that specifies how these activities relate to each other.

The RUP covers three different levels of requirements: 1) requests and needs, 2) features, and 3) software requirements, whereas the REGPG handles requirements as one set. The RUP also lists ten artefacts to be produced during requirements management. The needs and features are captured in a vision document. The software requirements are specified in a use-case model and other supplementary specifications, which capture those requirements that do not fit well in the use cases. Complementary to the foregoing artefacts, a glossary, storyboards and a requirements management plan are also developed. However, the most significant activity in the requirements management process is the development of a use-case model, because use cases drive many activities throughout the development cycle (Kruchten 2004).

As a summary, the RUP classifies requirements management as a core discipline in the software engineering process. The RUP provides a more detailed and complex process model for RE than the REGPG. It also specifies a large set of artefacts to be produced during RE. Alternatively, the REGPG seems to include a more extensive set of RE practices than the RUP. However, this statement is based on the information gathered from the book written by Kruchten (2004). Kruchten points out that his book is not the complete RUP, but rather a small subset to introduce the RUP.

Table 9. Comparison of the requirements management activities of the RUP with the practices of the REGPG.

Requirements management activities of the RUP (Kruchten 2004)	**Related practices of the REGPG (Sommerville and Sawyer 1997)**
Analyze the problem: Gain agreement on a statement of the problem. Identify stakeholders. Identify the boundaries and constraints of the system.	Use business concerns to drive requirements elicitation. Identify and consult stakeholders. Define system boundaries.
Understand stakeholder needs: Gather stakeholder requests and obtain a clear understanding of the real needs of the users and the stakeholders.	Identify and consult stakeholders.
Define the system. Establish the set of system features to be considered for delivery. Determine the criteria that will be used to prioritize system features. Identify actors and use cases needed for each feature.	Prioritize requirements. Use scenarios to elicit requirements.
Manage the scope of the system: Collect important information from stakeholders and maintain those as requirements attributes to be used in prioritizing and scoping the agreed-upon set of requirements.	Record requirements source. Record requirements rationale. Assess requirements risks.
Refine the system definition: Detail the software requirements using a use-case model. Capture non-functional requirements and design constraints.	Specify non-functional requirements quantitatively. Look for domain constraints.
Manage changing requirements: Use requirements attributes and traceability to assess the impact of changing requirements. Use a central control authority to control changes to requirements. Maintain agreement with the customer and set realistic expectations on what will be delivered.	Document the links between requirements and system models. Maintain a traceability manual. Use change management policies.

2.2.6 Extreme Programming versus REGPG

Extreme Programming (XP) has attracted a lot of attention among practitioners and researchers since it was introduced in 1999 (Beck 1999) and described in more detail in 2000 (Beck 2000). XP is a light-weight methodology for small-to-medium-sized teams developing software in the face of vague or rapidly changing requirements (Beck 2000).

XP is implemented with 12 practices. Beck (1999) points out that the individual practices in XP are not by any means new. The innovation of XP is that 1) it puts all these practices under one umbrella, 2) makes sure they are practiced as thoroughly as possible, and 3) makes sure the practices support each other to the greatest possible degree (Beck 2000).

Our analysis of the XP practices indicates that five of them relate to RE (Table 10). Especially, the Planning Game contains several RE-specific tasks. During the Planning Game, the customer first writes stories that describe what the system needs to do (the descriptions of features, i.e. requirements). After that, the programmers estimate how long the stories will take to implement. Finally, the customer prioritizes the stories and chooses a set of them to be implemented in the next release.

Table 10. Comparison of the RE-related practices of XP with the practices of the REGPG.

RE-related practices of XP (Beck 2000)	Related practices of the REGPG (Sommerville and Sawyer 1997)
The Planning game: Quickly determine the scope of the next release by combining business priorities and technical estimates.	Use scenarios to elicit requirements. Plan for conflicts and conflict resolution. Prioritize requirements. Uniquely identify each requirement. Assess requirements risk.
Small releases: Put a simple system into production quickly, then release new versions on a short cycle.	Prototype poorly understood requirements. Use prototyping to animate requirements.
Metaphor: Guide all the development with a simple story of how the whole system works. The metaphor captures part of the system architecture.	Model the system architecture.
Testing: Programmers continually write unit tests, which must run flawlessly for development to continue. Customers write functional tests demonstrating that features are finished.	Propose requirements test cases.
On-site customer: Include a real, live user on the team, available full-time to answer questions.	Identify and consult system stakeholders. The end-users are one of the stakeholder groups.

Similarly, the REGPG recommends the use of scenarios. According to Sommerville and Sawyer (1997), scenarios can be thought of as stories which explain how the system is used. In addition, the REGPG suggests the prioritization of requirements into three categories in the same way XP does. Effort estimation, which is one of the key tasks in XP, is covered only lightly in the REGPG. The REGPG suggests that schedule risks should be considered as a part of requirements risk analysis.

One of the key principles of XP is to get feedback from users as quickly as possible, and it therefore recommends small releases. The REGPG suggests that prototypes should be built during requirements elicitation and validation. The goal of prototyping is to get feedback from users and better understand their real needs (Sommerville and Sawyer 1997). However, prototypes are meant to be used for a short time, whereas the key idea of XP is to put the system into real use as quickly as possible.

In XP, the customer writes functional tests demonstrating that features are finished. Similarly, proposing requirements test cases is one of the good practices in the REGPG. However, the objective of proposing test cases during requirements definition is to validate the requirement rather than the system (Sommerville and Sawyer 1997).

XP emphasizes the role of customers. A real customer who will really use the system must sit with the development team and be available full-time to answer questions (Beck 2000). The REGPG treats all stakeholders equally and does not highlight the importance of users and customers as key stakeholders.

As a summary, XP does not state explicitly that requirements definition is important. Instead, it nominates coding, testing, listening, and designing as the four basic activities of development (Beck 2000). However, XP includes a small set of RE-specific tasks. For example, stories, which are one way to document requirements, form the key element of XP. Furthermore, the role of customers and users is emphasized in XP. The customer, who is supposed to sit with the development team full-time, is responsible for writing stories and functional tests. Furthermore, the customer sets the implementation priority for the requirement and decides when each requirement is satisfied.

As one can assume, XP includes a clearly smaller set of RE-related tasks than the REGPG. Furthermore, these tasks are not explicitly specified to relate to RE. Therefore, practitioners may not find them when they are looking for good practices that support requirements definition and management.

2.3 Process models for RE

Kotonya and Sommerville (1998) define a process model to be a simplified description of a process. The model can be developed from different perspectives and can, therefore, represent different kinds of information. Among the information that people want to extract from a process model are: What is going to be done? Who is going to do it? How and why will it be done? (Curtis et al. 1992). The type of model which may be produced depends on the purpose for which the model will be used. Five basic uses for process models are (Curtis et al. 1992; Krasner et al. 1992):

- To facilitate human understanding and communication of the process.
- To support continuous process improvement.
- To support process management.
- To facilitate automated guidance.
- To support automated execution.

Curtis et al. (1992) point out that in practice, most process descriptions have employed narrative text and simple diagrams to express process.

The RE literature provides several process models for RE (e.g. Christel and Kang 1992; Pohl 1996; Kotonya and Sommerville 1998; Stevens et al. 1998; Robertson and Robertson 1999; Wiegers 2003). These models represent different aspects of the requirements definition process. Moreover, they are generic, i.e. they have not been developed for any specific company or organization. In addition to these generic process models, the RE literature describes a small set of company-specific RE process models (e.g. Yeh 1992; Hutchings and Knox 1995; Regnell et al. 1998; Carlshamre and Regnell 2000). This section shows two examples of the generic models to illustrate possible inputs, outputs, activities and stakeholders of the requirements definition process. In addition, it represents two examples of RE process models developed in industry.

Kotonya and Sommerville (1998) have defined a coarse-grain model of the RE process, which shows the principal activities and their approximate sequencing (Figure 9). Kotonya and Sommerville emphasize that their model does not tell how to enact the RE process, but provides an overall picture of the process. It shows that there is different kind of input information to the requirements definition process. Some of this information, such as regulations and standards, are well documented, whereas user needs and domain information may often be tacit knowledge in people's heads.

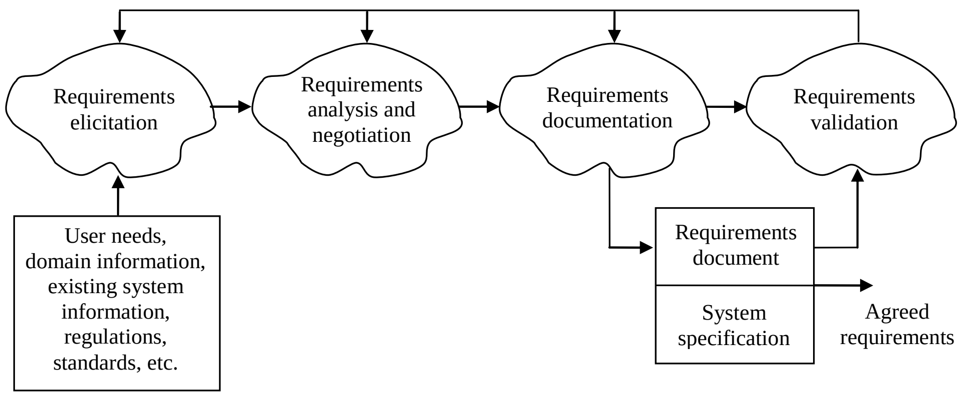

Figure 9. Coarse-grain activity model of the RE process (Kotonya and Sommerville 1998)

The process model in Figure 9 also represents two outputs. Firstly, Kotonya and Sommerville point out that the requirements document must describe the agreed requirements in a way that all system stakeholders can understand. According to them, this usually means that the requirements must be documented using natural language and diagrams. Secondly, the system specification contains more detailed system documentation such as system models.

Kotonya and Sommerville show RE activities in their process model using cloud icons because they want to indicate that there are no distinct boundaries between these activities. Furthermore, they emphasize that, in practice, the activities are interleaved and there is a great deal of iteration and feedback from one activity to another.

Figure 10 presents an iterative process model developed by Pohl (1996). According to him, RE is a process in which the RE team learns about the current and/or future reality. Pohl's model depicts the four activities and their relations. Furthermore, it represents some potential stakeholders that can participate in the RE process.

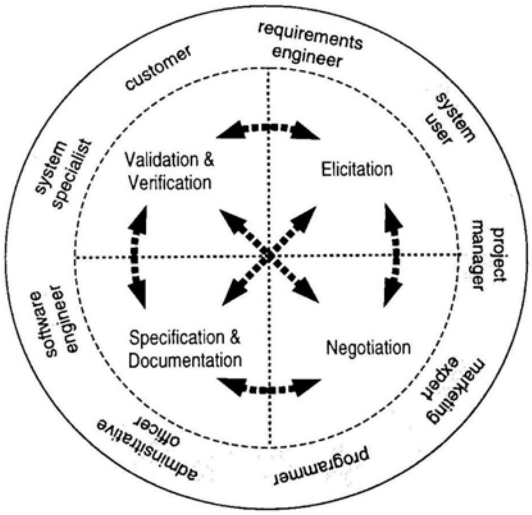

Figure 10. An iterative RE process (Pohl 1996)

The Pohl's iterative process model resembles the product requirements definition process developed at the Digital Equipment Corporation (Figure 11). This re-engineered process was designed to ensure customer satisfaction and business readiness before a line of code is written, an electronic circuit designed, or a service defined (Hutchings and Knox 1995).

Prior to the re-engineered process presented in Figure 11, Digital developed a detail nine-step process model (Hutchings and Knox 1995). The nine steps were piloted, and two problems were encountered. Firstly, many product development personnel had expectations that knowledge-driven, creative processes such as requirements engineering are similar to deterministic, manufacturing-style processes. Secondly, the work of the cross-functional team focused on engineering deliverables. Consequently, marketing personnel did not see their work showing up in the RE process, which caused difficulties in holding the cross-functional teams together. Because of these problems, the original nine steps were hidden in the background of the process descriptions, while the process was simplified into three broad phases and its scope was widened. The main idea of the re-engineered model is the concept of "whole product", which broadens the requirements definition scope to include requirements for marketing messages, pricing, and packaging, as well as service scenarios.

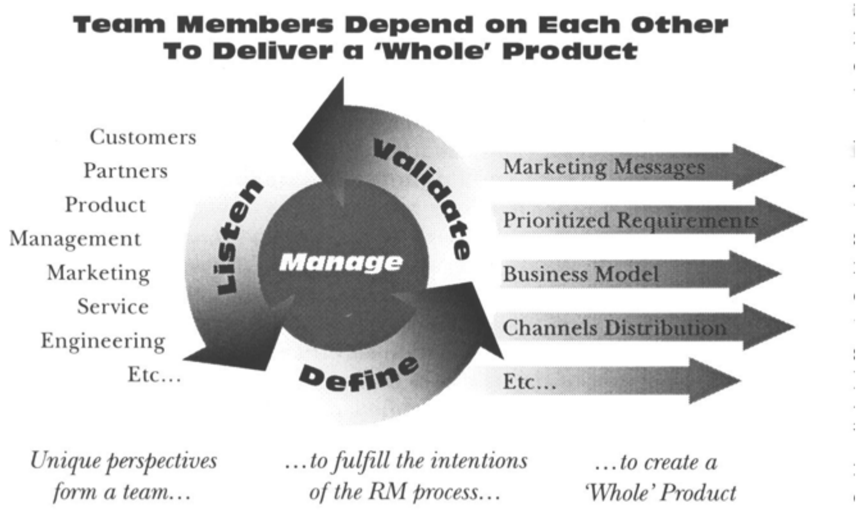

Figure 11. The "Whole Product" perspective of requirements management at the Digital Equipment Corporation (Hutchings and Knox 1995)

Figure 12 shows the REPEAT requirements lifecycle model of Telelogic AB. REPEAT covers typical RE activities such as elicitation, documentation, and validation (Regnell et al. 1998). However, instead of using a traditional activity-based model, Telelogic has developed this state-oriented life cycle model for continuous requirements definition and management. The basic idea is that a requirement can be issued at any time. Furthermore, each requirement is stored in the in-house-built database system, and has a life cycle processing through the specific states set out below.

- New: The initial state of a requirement after it has been issued and given an initial priority.
- Assigned: The requirement has been assigned to an expert for classification.
- Classified: A rough estimate of cost and impact is attached to the requirement.
- Rejected: An end-state indicating that the requirement has been rejected.
- Selected: The requirement has been selected for implementation with a certain priority and results from detailed cost and impact estimations. A selected requirement may be deselected due to requirements changes and then re-entered into the classification state or get rejected.
- Applied: An end-state indicating that the requirement has been implemented and verified.

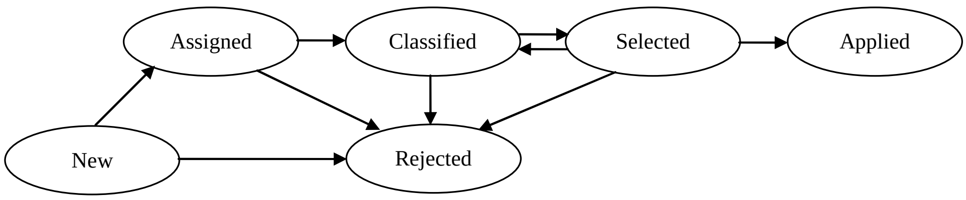

Figure 12. The REPEAT requirements lifecycle model of Telelogic AB (Regnell et al. 1998)

In addition to Telelogic, Ericsson Radio Systems AB has developed a similar kind of state-oriented model for fostering requirements from invention to release (Carlshamre and Regnell 2000). According to Carlshamre and Regnell, one of the potential benefits of this lifecycle view is that it represents a methodology-independent approach to RE. In the Ericsson case, a major reason for introducing a lifecycle approach was the fact that software developers were tired of having new methods thrown at them every second year or so.

Carlshamre and Regnell also point out that there are a number of challenges for organizations who would like to introduce a lifecycle approach similar to the one presented in Figure 12. One of the main challenges is that the lifecycle model handles individual requirements, and therefore, requirements can become very fragmented. Regnell et al. (1998) suggest that requirements need to be packaged into coherent bundles in order to give them a structure that reflects functionality as seen by the user.

As a summary, there seems to be a common view that the requirements definition process is iterative and includes elicitation, analysis/negotiation, documentation/specification, and validation/verification activities. The outcome of the process can vary from a detailed technical system specification to business models and marketing messages. In addition, a characteristic of the RE process is that it can involve many stakeholders, such as requirements engineers, customer and user representatives, domain experts, project managers, developers, testers, sales and marketing personnel. In other words, the RE process involves people that have a different background and expertise. This is in contrast to other software processes, such as system testing, where the majority of the people involved in the processes have a common technical background and a shared goal of demonstrating that the system meets its specification (Kotonya and Sommerville 1998).

2.4 Good RE practices

The previous section presented four high-level process models for requirements definition. These models provide an overview of RE but do not describe how to elicit, analyze, document or validate requirements in practice. This section describes briefly a small set of RE practices that provide more-detailed guidance on how to define requirements in product development projects. The set of RE practices is based on the analysis of three publications (Sommerville and Sawyer 1997; Hofmann and Lehner 2001; Wiegers 2003). Each of these publications provides a varying number of RE practices.

Firstly, as discussed earlier in Section 2.2 Sommerville and Sawyer (1997) describe sixty-six good practices that are based on the experiences of the REAIMS project where the focus was on safety-critical systems. They have also adopted practices from the standards and other field studies. Secondly, Hofmann and Lehner (2001) have identified ten RE practices that contribute to software project success. These practices are based on the data collected in fifteen RE teams in nine software companies and development organizations in the telecommunication and banking industries. Finally, Wiegers (2003) describes forty-six practices for requirements definition and management based on his RE process improvement experience.

Because the focus of our study is on the requirements definition process, the analysis of the three publications concentrated on practices that support systematic requirements definition. We identified seven practices that are common to all of the three publications:

- Adopt a template for documenting requirements.
- Prioritize requirements.
- Create prototypes during requirements definition.
- Apply use cases to elicit and validate requirements.
- Review and inspect requirements documents.
- Develop complementary models.
- Create and maintain a requirements traceability matrix.

The remainder of this section briefly summarizes the practices.

Adopt a template for documenting requirements. Organizations should define a standard document structure to support practitioners in writing good-quality requirements documents. A standard document structure should encapsulate what the organization thinks is the best way to organize a requirements document (Sommerville and Sawyer 1997). According to Wiegers (2003), many organizations begin with the template described in IEEE Standard 830-1998 (IEEE Std 830-1998). Because the size of projects and the type of systems can vary, there might be a need to define several templates (Sommerville and Sawyer 1997; Wiegers 2003).

Prioritize requirements. According to Hofmann and Lehner (2001), requirements prioritized by stakeholders drive successful RE teams. This allows the RE team to decide which requirements to investigate when and to what degree of detail. Requirements can be classified into three categories, for example, such as essential, useful and desirable (Sommerville and Sawyer 1997). “Essential” means that the requirement must be included in the system, “useful” that the system will be less effective without the requirement, and "desirable" that the requirement is not a core system facility, but makes the system more attractive to users. Hofmann and Lehner report that prioritization of requirements, however, caused the most difficulty for RE teams. For example, several teams that participated in the study mentioned the inability to consistently execute RE according to stakeholders' priorities rather than their own interpretation of what is important.

Create prototypes during requirements definition. When developers are not certain about the requirements, they can construct a prototype to make the concepts and possibilities more tangible (Wiegers 2003). Users who evaluate the prototype help the developers achieve a better understanding of the real needs of system users. The prototypes can range from simple mock-ups to operational prototypes (Hofmann and Lehner 2001). According to Sommerville and Sawyer (1997), there are various problems that can arise with prototyping. For example, it can increase the time-to-market, and some requirements such as real-time requirements can be difficult or impossible to prototype.

Apply use cases to elicit and validate requirements. Wiegers (2003) proposes the exploration together with user representatives of the tasks they need to accomplish with the software i.e. their use cases. These stories that explain how the system is used are also called scenarios (Sommerville and Sawyers 1997) Furthermore, Hofmann and Lehner (2001) report that successful RE teams use scenarios to validate requirements. According to Sommerville and Sawyer, the scenario-based approach requires users to spend a lot of time interacting with requirements engineers, and a problem can be that they are not able to take enough time away from their normal job to work through scenarios.

Review and inspect requirements documents. According to Wiegers (2003), formal inspection of requirements documents is one of the highest-value software quality practices available and informal preliminary reviews are also useful. Similarly, Hofmann and Lehner (2001) report that successful RE teams validate and verify requirements with multiple stakeholders by organizing peer reviews. Wiegers (2003) also points out that inspections are not one of the easiest new practices to implement. The major problem is ensuring a broad spectrum of involvement as some people may be unable to take time from their other work to participate in the requirements reviews and inspections (Sommerville and Sawyer 1997).

Develop complementary models. As part of the system modelling process, several models should be created to illustrate different aspects of the system (Sommerville and Sawyer 1997). Such models include, for example, data-flow diagrams, entity-relationship diagrams, state-transition diagrams or sequence diagrams (Wiegers 2003). The process of developing different types of models forces the making of different kinds of analysis of requirements (Sommerville and Sawyer 1997). These different types of analysis are likely to reveal incorrect, inconsistent, missing, and superfluous requirements. The main problem with models is that people without a technical background often have problems understanding them (Sommerville and Sawyer 1997).

Create and maintain a requirements traceability matrix. According to Hofmann and Lehner (2001), successful RE teams maintain a requirements traceability matrix to track a requirement from its origin through its specification to its implementation. In addition, the traceability matrix makes it easier to find which requirements and models are affected by change (Sommerville and Sawyer 1997). The major problem with the traceability matrix is that the requirements engineers who must develop the links between the models and requirements do not, in the short term, benefit from this information (Sommerville and Sawyer 1997). Therefore, they are likely to give this practice a low priority under time pressure.

Most of the practices described above handle requirements in general and do not include the users' point of view. There are two practices that explicitly mention the users' role. First, users can evaluate prototypes in order to help the developers achieve a better understanding of the real needs of system users. Secondly, the developers can explore with user representatives the tasks users need to accomplish with the product and represent these tasks as use cases.

2.5 Guidelines for RE process improvement

This section summarizes the factors found in both the RE and software engineering (SE) literature to be important for process improvement. The literature review focuses on the case studies of the RE process and the lessons learned from successful software process improvement (SPI). The review was conducted according to the recommendations of Webster and Watson (2002). First, we analyzed each reference and recorded the lessons learned and the explicitly reported success factors derived from each of them. In addition, we searched and recorded important issues related to process improvement. Finally, we analyzed all the recorded items, and clustered them around the identified key concepts. The following two sections summarize the key concepts in the form of guidelines.

2.5.1 Key factors found in the RE literature

Even though research into requirements engineering has been active throughout the 1990s, there are not many case studies concerning RE process improvement. We found eight case studies (Hutchings and Knox 1995; Salo and Käkölä 1998; Claus et al. 1999; Jacobs 1999; Calvo-Manzano et al. 2002; Damian et al. 2002, 2004; Berander and Wohlin 2003; Daneva 2003) that deal with issues relating to the success of RE process improvement. In addition to these papers, we also used as reference sources two RE books (Sommerville and Sawyer 1997; Wiegers 1999a) that offer guidelines for RE process improvement.

The following eight guidelines summarize the most frequently identified factors that affect the success of RE process improvement. In other words, we first detected critical factors for process improvement from each reference. After that, we listed the factors in a table and classified them into the order that shows how many times the factor has been identified in the studied literature (see Appendix C). The guidelines are presented in the order of the number of references to them found in the studied RE literature.

Involve users of RE processes in improvement work. One of the main factors contributing to the institutionalization of a process is the involvement of future process users and management in the development of the process from the very beginning (Claus et al. 1999). User involvement is critical for two reasons. First, it helps to develop a process that is useful to the people that have to execute this process (Claus et al. 1999; Damian et al. 2002). Second, involvement increases the acceptance of the developed process (Claus et al. 1999). Several authors also point out that RE process improvement should be a team effort (e.g. Hutchings and Knox 1995; Sommerville and Sawyer 1997; Damian et al. 2002).

Provide benefits for all users of RE processes. The studied RE literature stresses the significance of benefits for people involved in RE processes. For example, Sommerville and Sawyer (1997) state that one should always try to introduce techniques where everyone involved (not just managers) sees some benefits. Further, Hutchings and Knox (1995) report that the root cause of the difficulty of maintaining the committed participation of the marketing function in the requirements management process of the Digital Equipment Corporation was that the marketing people could not see the value of their participation in terms of the deliverables of their function.

Improve RE processes continuously based on feedback. Calvo-Manzano et al. (2002) encourage companies to manage process evolution by the application of metrics and corrective actions. Similarly, Sommerville and Sawyer (1997) point out that organizations need to establish procedures to collect feedback on improvements and ensure that action is taken in response to this feedback in order to correct any identified problem.

Test RE processes and practices in pilot projects. According to Claus et al. (1999), one of the main success factors of process definitions is that at least one software development project is involved from the start of the process improvement initiative and applies the new processes. This ensures that the defined processes are feasible and actually benefit development projects, rather than slow them down. Sommerville and Sawyer (1997) also point out that it is important to introduce process changes in pilot projects in order to discover the advantages and disadvantages of the change.

Train all users of RE processes. Damian et al. (2002) report that once the RE process was revised, training and leadership were essential for change management. According to them, this aspect is often overlooked and becomes a cause of failure of organizational change efforts. In addition, Jacobs (1999) reports that training only a few persons and putting all hope in the multiplier-effect is likely to fail. He points out that all parties to be involved in requirements engineering therefore have to participate in adequate training.

Ensure management commitment and support: According to Calvo-Manzano et al. (2002), the commitment process is a fundamental requirement for a successful improvement process. The objective of this commitment process is to obtain the support of senior management in carrying out the improvement project. In addition, Damian et al. (2002, 2004) report that the commitment and support of middle management i.e. the project manager was one of the success factors for RE process improvement at the Australian Center for Unisys Software.

Cope with resistance to change. This problem is discussed in four case studies (Hutchings and Knox 1995; Claus et al. 1999; Damian et al. 2002; Daneva 2003). At first, our literature review indicated that resistance to change is not a critical problem in RE process improvement. However, the detailed analysis of the case studies indicates that the acceptance of new RE practices can be one of the key challenges in RE process improvement. For example, Hutchings and Knox (1995) reports that overcoming people's natural resistance to change is critical for RE process improvement success The issues related to the acceptance of new RE processes and practices are covered in more detail in Section 2.6.

Define simple RE processes. According to Salo and Käkölä (1998), the presence of multiple stakeholders from several functional organizations, some of which participate in requirements processes in a minor role, implies that these processes, methods, and tools should be as simple as possible. In addition, Hutchings and Knox (1995) report that the Digital Equipment Corporation had a detailed nine-step requirements management process that focused primarily on engineering deliverables such as a requirements document and a functional specification. In order to support the participation of the marketing personnel in requirements management, marketing deliverables were included as outputs of the process, the original nine steps were hidden in the background of the process description and the requirements management process was simplified into three broad phases.

Prepare to make a cultural change. The results of three case studies (Hutchings and Knox 1995; Claus et al. 1999; Jacobs 1999) show that the introduction of requirements engineering involves not just a change of process or technology, but also a change of culture. Such cultural change means two things: Firstly, product development personnel need to understand the importance of customer and user requirements and, secondly, they must commit to defining and managing requirements systematically. The results of the case studies (Hutchings and Knox 1995; Jacobs 1999) also indicate that the cultural change towards systematic customer requirements management is challenging.

Use an evolutionary improvement strategy. Sommerville and Sawyer (1997) remark that it is not realistic to expect organizations to invest a lot of time and money in improvements whose value is difficult to assess. Therefore, they recommend organizations to introduce small-scale improvements with a high benefit/cost ratio before expensive new techniques. Wiegers (2003) aligns with these statements and points out that, instead of aiming at perfection, it is important to develop a few improved procedures and to get started with implementation.

2.5.2 Key factors found in the SE literature

Software process research grew up during the 80s to address the increasing complexity and criticality of software development activities (Fuggetta 2000) and has a longer tradition than RE process research. Therefore, software process literature is more extensive than RE process literature and offers knowledge valuable to the RE research community. This part of the review is based on fourteen case studies (Humphrey et al. 1991; Basili and Green 1994; Johnson 1994; Basili et al. 1995; Tanaka et al. 1995; Haley 1996; Diaz and Sligo 1997; Jakobsen 1998; Sakamoto et al. 1998; Wiegers 1999b; Jakobsen 2000; Kautz et al. 2000; O'Hara 2000; Basili et al. 2002).

In addition, we include three SPI books (Humphrey 1989; Paulk et al. 1997; Zahran 1998), one technical report (McFeeley 1996) and two papers (Curtis 1997; Conradi and Fuggetta 2002) that summarize the process improvement experience gained over several years by three researchers.

The analysis of the SPI literature was conducted in the same way as the analysis of the RE process literature. In other words, we first detected critical factors for software process improvement from each reference. After that, we listed the factors in a table and classified them into the order that shows how many times the factor has been identified in the studied literature (see Appendix D).

According to the studied literature, there are six factors that are common to both software and RE process improvement: 1) user involvement, 2) management commitment and support, 3) resistance to change, 4) training and education, 5) continuous process improvement, and 6) evolutionary process improvement. In addition to these six factors, we identified four factors that were classified important to process improvement in the SE literature but not emphasized in the RE literature: 1) measurement, 2) process improvement goals, 3) motivation, enthusiasm and pride of personnel, and 4) infrastructure for process improvement. These four factors are covered below in more detail.

Measure the impact of the improvement efforts. Measurement of improvement efforts is widely covered in the software process literature. According to Humphrey (1989), sustained progress is not possible until the process is under statistical control. In addition, several other authors state the importance of measurement to successful process improvement (e.g. Haley 1996; Zahran 1998; O'Hara 2000; Basili et al. 2002). On the other hand, some authors report that it is difficult to quantify the impact of process improvement efforts (e.g. Wiegers 1999b; Kautz et al. 2000; Conradi and Fuggetta 2002).

The studied RE literature discusses the difficulties of quantifying the effect of the RE process changes (e.g. Sommerville and Sawyer 1997; Jacobs 1999; Calvo-Manzano et al. 2002), but it does not point out the importance of measurement to the success of RE process improvement. A possible explanation for this difference is that the software process case studies of the review describe improvement experiences that are mainly gained from large organizations (e.g. Humphrey et al. 1991; Haley 1996; Basili et al. 2002). These large organizations have been able to implement and invest in long-term measurement programs, despite measurement being expensive (Humphrey 1989) and requiring a rigorous process and professional staff (Basili et al. 2002).

Align process improvement goals with business goals. A number of authors state explicitly that it is essential to align process improvement goals with the business goals of the organization (Paulk et al. 1997; Zahran 1998; O'Hara 2000; Conradi and Fuggetta 2002). In other words, the studied SE literature emphasizes the importance of aligning process improvement goals with the business goals of the organization, whereas the RE literature points out tangible and short-term benefits for RE process users. Furthermore, several RE case studies recommend testing RE processes in pilot projects to ensure that the defined processes are feasible and actually benefit development projects.

Support motivation, enthusiasm and pride of personnel. According to McFeeley (1996), one of the biggest challenges in process improvement is to maintain motivation and enthusiasm of personnel across and between all levels of the organization. On the other hand, achievements and successes in process improvement motivate people. Jakobsen (1998) reports that many people become addicted to their improvement successes and thus can be powerfully motivated by a chance to achieve more of them. Furthermore, Humphrey et al. (1991) point out that pride is the most important result. According to them, improvements are one-time achievements, but pride feeds on itself and leads to continuous measurable improvement.

Create an infrastructure to support process improvement. Software process improvement is a significant undertaking for any organization, and it is almost impossible to accomplish anything without a supporting infrastructure (McFeeley 1996). According to Paulk et al. (1997), infrastructure is the underlying framework of an organisation including organisational structures, policies, standards, training, facilities, and tools that supports its ongoing performance.

2.6 Problems and challenges in RE process improvement

A lot of research has been conducted on methods or techniques that support mainly a single RE activity such as requirements modelling or requirements elicitation. Relatively few case studies that concern the improvement of the whole RE process have been described in the literature. We found nine studies that report experiences on RE process improvement that concern market-driven product development (Table 11).

Table 11. Case studies that describe experiences in RE process improvement

Publication	High-level problem	Solution
Yeh (1992) describes a requirements process for the planning and product development organizations of IBM Lines and Business.	There were many problems in the process of determining requirements: Who and where are the sources of requirements? What are they really asking for? Given resources, market, and schedule constraints, how much can be done?	Developing a well-structured requirements planning process with clearly defined tasks, controls, and work products.
Hutchings and Knox (1995) report a re-engineering case study of the product requirements definition process at Digital Equipment Corporation.	On average, 40% of the requirements specified in the feasibility and requirements phase of the lifecycle were redefined in the subsequent four lifecycle phases.	Re-engineering the requirements management process. Broadening it to include requirements for marketing messages, pricing, packaging, and service scenarios.
Regnell et al. (1998) describe a market-driven RE process developed for Telelogic AB.	Telelogic had an ad hoc process for managing requirements and faced a number of challenges related to release precision and product quality.	Developing an RE process that manages requirements with a state-oriented life-cycle model which defines a ladder of states for each requirement climb.
Claus et al. (1999) describe experiences in introducing systematic requirements management at the Deutsche Bahn.	A procedure to change requirements had been defined, but was rather cumbersome and therefore only used for major changes. Small changes were handled ad-hoc.	Defining a requirements management process with the goal of both supporting the projects and satisfying the CMM requirements.
Jacobs (1999) reports a case study on improving requirements engineering at Ericsson Eurolap.	Missing understanding of customer needs was identified as a main obstacle for decreasing fault density and lead time.	Introducing the “Gilb Style” method. Putting all projects on ice for a week. All software developers participated in a one-week training course in RE.
Carlshamre and Regnell (2000) describe and compare the market-driven RE processes of Ericsson Radio Systems AB and Telelogic AB.	Telelogic had problems with delayed product releases. Ericsson Radio Systems AB launched an improvement programme with the aim of reducing lead time and increasing productivity and quality.	Developing a state-oriented life cycle model for RE (Figure 12). The states represent the refinement level of each individual requirement in its progress towards release.
Damian et al (2002, 2003, 2004) describe an industrial experience in RE process improvement at the Australian Center for Unisys Software.	Requirements crept throughout the development life cycle. The quality of delivered software was less than optimal. Initial over commitment and a poor change impact analysis led to schedule overruns and dropped functionality.	Undertaking significant changes in the requirements management process, which include the introduction of group session approaches to requirements analysis and a structured method for writing requirements.
Daneva (2002, 2003, 2004) summarizes experiences in making a generic RE model alive process at Telus Mobility.	The company did not have a disciplined organization-wide RE process.	Adopting a generic ERP (Enterprise Resource Planning) RE process.
Weber and Weisbrod (2003) summarize problems and solutions for RE at DaimlerChrysler.	The complexity of electronic components had increased. Consequently, the complexity of specification activities surpassed what conventional text-processing systems can support in terms of management and tracing functionality.	Developing a tool-supported RE process. Establishing an RE team to support the company’s business units.

In this section, we summarize problems encountered in RE process improvement. Here we focus on challenges that have been identified in several case studies and are not specific to a particular RE approach or process. The analysis of problems related to RE process improvement revealed four challenges:

- How to measure the business benefits of RE processes and practices.
- How to support practitioners in accepting new RE processes and practices.
- How to involve different stakeholders in RE processes.
- How to plan the content of releases and prioritize requirements.

These challenges are described in more detail in the following four subsections.

2.6.1 Business benefits of RE

The results of the case studies indicate that it seems to be difficult to measure the business benefits of RE process improvement efforts. In five case studies (Hutchings and Knox 1995; Regnell et al. 1998; Jacobs 1999; Carlshamre and Regnell 2000; Damian et al. 2004), the aim of the RE process improvement was to enhance product quality and productivity, or reduce development time. Regnell et al. (1998) report that Telelogic AB gained a measurable improvement in delivery precision and product quality after they had launched a new RE process and that the authors were convinced that the introduction of the new RE process was the major explanation of these achievements.

Jacobs (1999) reports that the measurements showed a clear improvement in product quality, project duration and effort since Ericsson Eurolap introduced the new way of RE. However, Jacobs also points out that several other factors had changed at the same time and that it was not possible to quantify the changes caused by the new way of RE. Similarly, Davis and Hsia (1994) point out that the time between the requirements phase and product delivery is usually too long to pinpoint how specific RE techniques contribute to a product’s success or failure.

2.6.2 Acceptance of RE processes and practices

Daneva (2003) reports that the key point in RE process adoption is not the process itself but its acceptance by those concerned. To overcome a sceptical attitude towards the new RE process and to cope with organizational inertia, Daneva (2003, 2004) recommends three actions. First, have RE teams blend new RE practices into the existing ones. Second, try to use known and proven practices. Thirdly, raise the RE team’s awareness of the RE practices that are critical for project success.

Hutchings and Knox (1995) observed that overcoming people’s natural resistance to change was the key element of the human side of introducing a process such as requirements management into an organization. They also report that the new requirements management process was very different from the Digital process that teams had been used to. Therefore, a change agent had to continually show RE teams the benefits of the process and reassure them that the uncertainty they were experiencing and the time they were expending would indeed yield a valuable and rewarding experience.

Similarly, Claus et al. (1999) found that the real problems were those of change management (managing the internal, psychological aspect of change). They report that they at first underestimated the difficulty of making the change happen and then stick, not just for a few developers but on a large scale. According to them, the technical aspects of RE were comparatively trivial because the requirements management process and the templates defined were fairly simple. Claus et al. point out that they could think of more complex, more formal solutions that would be better from a purely technical point of view. However, they state that the comparatively small advantage of a very formal approach to RE would not be sufficient to convince a large number of developers that it is worth the effort of learning and applying a new method in addition to all the other tools and techniques they have to learn and apply.

2.6.3 Stakeholder involvement in RE processes

It seems to be difficult to involve stakeholders outside product development in RE processes. Hutchings and Knox (1995) report that cross-functional teams reduce misunderstandings related to requirements and the time it takes to correct them. On the other hand, they point out that it is difficult to hold the cross-functional teams together over the entire development period. According to their study, the function that most often departed early was the marketing function, because marketing personnel did not see their work showing up in the requirements definition process. Therefore, Digital re-engineered their RE process by broadening the requirements definition scope to include requirements for marketing messages, pricing, and packaging, as well as deployment and service scenarios.

Similarly, Jacobs (1999) reports that the biggest problem in introducing a new way of RE was communication problems with internal customers and with other related projects. According to Jacobs, this miscommunication made two issues obvious. First, even a good RE approach can be problematic if not all parties are involved and trained. Second, requirements specifications are not only the basis for the design but may be used in other ways, for example, as a foundation for marketing information.

In addition, Damian et al. (2002, 2004) report that active involvement of staff across development platforms and from several functional areas, i.e. engineering, testing, and product information was found as one of the strengths of the improved RE practice since it enabled an understanding of features within the entire system. Furthermore, they point out that better communication, collaboration and participation of other stakeholder groups such as the Business Initiative Group, marketing groups, and customers is necessary for further improvement.

2.6.4 Release planning and requirements prioritization

Carlshamre and Regnell (2000) point out that release planning is one of the most important and complex tasks in market-driven RE because it deals with the essential questions of which requirements should be included in the next release. According to Carlshamre and Regnell, release planning is complex because requirements usually have many dependencies. Therefore, it is not possible, for example, to just select a subset of the requirements having the highest priority. Carlshamre and Regnell also report that another challenge is how to synthesize needs and opportunities from different market segments before planning the requirements strategy for the next release.

Similarly, Yeh (1992) states that requirements prioritization is one of the most crucial and, at the same time, difficult tasks that face the decision makers because of the numerous sources of requirements, the rapidly changing and competitive environment, and limited resources. Similarly, Daneva (2002) reports that requirements prioritization activities were identified as the most difficult ones because of stakeholders' concerns. According to her observations, business representatives were reluctant to prioritize requirements because of fears that implementers would automatically restrict the project to the must-have items and the nice-to-have features would never be implemented. In addition, consultants were reluctant to ask for priorities because they did not want to admit they could not implement it all in the time available.

2.7 Summary

The purpose of the literature review was to provide an overview of previous research related to the research questions of the study. First, we described briefly the REGPG, which was a starting point for this research. We also analyzed a set of the existing RE process models and good RE practices. Furthermore, we presented a set of guidelines based on factors found important for successful process improvement in the RE and SE literature. Finally, we summarized the most common problems encountered in the case studies of RE process improvement.

There are several publications that provide information about the REGPG and its development (Sommerville and Sawyer 1997; Sawyer et al. 1997, 1998, 1999a, 1999b). The literature review, however, revealed that there is still little empirical evidence on the usefulness of the REGPG. Daneva (2002, 2003) reports how the REGPG was used for assessing and understanding an ERP (Enterprise Resource Planning) requirements engineering process. Furthermore, Sommerville and Ransom (2005) describe an empirical study where the REGPG was used to support RE process assessment and improvement in nine companies. Based on their empirical data, Sommerville and Ransom show that the REGPG is useful in supporting maturity assessment and in identifying process improvements.

The RE literature proposes a number of RE models that contain valuable information about different aspects of RE. However, there are only two models that explicitly specify user requirements as an output of the requirements definition process (e.g. Stevens et al. 1998; Wiegers 1999a, 2003). In addition, the literature review revealed that user requirements definition and issues related to it have received surprisingly little attention in empirical RE research so far. We were able to find only two case studies that focus on user requirements (e.g. Forsgren and Rahkonen 1995; Coble et al. 1997).

A lot of RE research seems to focus on methods and techniques developed for large customer-specific systems. In practice, RE is also performed in projects that develop market-driven products. Therefore, several RE researchers have raised the need to widen the research scope in RE from custom-designed system development to market-driven product development (e.g. Potts 1995; Nuseibeh and Easterbrook 2000; Kaindl et al. 2002). For the time being, relatively few case studies related to market-driven RE processes have been presented in the RE literature. The organization-wide implementation of RE processes and practices in particular has received little attention in RE research. We found five case studies that focus on making the RE model a life process and describe factors that affect the organization-wide implementation of the RE process (e.g. Hutchings and Knox 1995; Claus et al. 1999; Jacobs 1999; Damian et al. 2002, 2003, 2004; Daneva 2003, 2004).

As a summary, our literature review supports the statement of Hofmann and Lehner (2001), who point out that most RE research is conceptual and concentrates on methods or techniques, primarily supporting a single activity. Instead of developing new RE methods and techniques, our aim is to investigate how existing RE methods and practices can be applied in real product development contexts. Furthermore, we want to gain further understanding of the improvement of the whole RE process by following up the improvement work from the assessment phase to the organization-wide implementation phase.

3 Results

This section gives an overview of the contents of the included papers, and summarizes the main research results. The results are grouped in relation to the research questions stated in Section 1.3. Section 3.1 describes how the REGPG supports organizations in RE process improvement. In Section 3.2, we recommend a set of basic improvement actions and RE practices that support systematic user requirements definition. Success factors and challenges related to the organization-wide adoption of RE processes and practices are described in Section 3.3 and 3.4.

3.1 Strengths and weaknesses of the REGPG

The first research objective of this thesis was to evaluate how the REGPG supports organizations in RE process improvement. This research question has been answered in two publications, Paper I and Paper II. Paper I concentrates on the assessment phase of the improvement cycle. It evaluates the usefulness of the REAIMS maturity model, which is one of the key components of the REGPG. Paper II describes how the REGPG supports the whole improvement cycle from assessment to implementation and monitoring of improvements. This section summarizes the strengths and weaknesses of the REGPG and shows how the REGPG supports systematic RE process improvement.

3.1.1 Main strengths of the REGPG

We identified four strengths of the REGPG when it was applied for RE process improvement in the four case organizations:

- A REAIMS assessment raises personnel awareness of requirements engineering.
- REAIMS assessment results provide information for process improvement planning.
- The REGPG supports organizations in defining a first requirements definition process.
- The REGPG includes relevant RE practices for different kinds of application domain.

One of the benefits of the REGPG is that conducting an assessment with the REAIMS maturity model raises personnel awareness of requirements engineering. The REGPG contains 66 RE practices covering eight RE areas. Both REAIMS interviews and assessment results familiarize practitioners with these eight areas and the practices related to them. Therefore, a REAIMS assessment can be particularly useful for organizations that are just starting to improve their RE processes. In addition, REAIMS assessments are a way to motivate practitioners for process improvement. Firstly, assessment results give practitioners an overall view of the state of requirements engineering in their organization. Secondly, they reveal RE areas and practices that can be developed.

The second benefit of the REGPG is that assessment results that are based on the REAIMS maturity model provide information for process improvement planning. REAIMS assessment results show which practices of the REGPG are used in an organization and to what extent. This information helps practitioners to identify new RE practices that can be included in the RE process. Furthermore, assessments results also improve communication and knowledge transfer within organizations by revealing good RE practices that have already been used in some product development projects.

The third benefit of the REGPG is that it supports organizations in developing a first process model for requirements definition. The REGPG recommends that a good RE process includes the following activities: elicitation, analysis combined with negotiation, and validation. These activities seem to suit organizations that want to define a process model in order to facilitate their personnel's understanding of the basics of requirements engineering. The purpose of the first RE process model is to give practitioners an overview of the main activities and clarify terminology related to requirements engineering. Common terminology and a common view of the RE process enables communication between different stakeholders that are supposed to participate in requirements definition and management.

One of the strengths of the REGPG is that it appears to include relevant RE practices for a range of application domains. Our case organizations represent four different kinds of application domain. Table 12 shows how the practitioners of these organizations assessed the state of the practices of the REGPG. The results presented in Table 12 are based on quick self-assessments where a small group of practitioners in each case organization evaluated how widely the practices of the REGPG are used in their product development unit. The results of these quick self-assessments show that the practitioners found only a small number of the practices irrelevant. In Organization C, people classified all nine RE practices that are specific for critical systems irrelevant to their product development because their products were not safety critical.

Table 12. Usage scope of the practices of the REGPG according to the quick self-assessments. The self-assessment covered all the 66 practices of the REGPG in Organizations A, B and C, and the basic 36 practices in Organization D.

	Organization A **2002**	**Organization B** **2002**	**Organization C** **2002**	**Organization D** **2001**
Number of systematically used practices	5	10	2	2
Number of normally/usually used practices	12	18	12	9
Number of sometimes used practices	38	23	20	10
Number of never used practices	10	14	22	13
Number of irrelevant practices	1	1	10	2

Further evidence of the relevance is that we identified thirteen basic practices that were used to some extent in all the case organizations (Table 13). Furthermore, more than half of the basic practices have been used at least in three organizations according to our analysis. In addition, we identified only six basic practices that had not been applied in any of the case organizations. These figures provide evidence that the REGPG includes relevant RE practices for product development.

Table 13. Usage scope of the basic practices of the REGPG according to the researcher's assessments.

	Number of the basic practices	**Percentage from the 36 basic practices**
Practices that have been used in all the case organizations	13	36 %
Practices that have been used in only three case organizations	8	22 %
Practices that have been used in only two case organizations	2	5 %
Practices that have been used in only one case organization	7	20 %
Practices that have not been used in any case organizations	6	17 %

3.1.2 Main weaknesses of the REGPG

In addition to the four strengths presented in the previous section, we identified three weaknesses of the REGPG when it was applied for RE process improvement in the case organizations:

- REAIMS assessment results are dependent on assessors.
- Selecting a realistic set of RE practices based on the REGPG is a demanding task.
- The REGPG offers a very limited set of process improvement guidelines.

One of the weaknesses of the REGPG is that REAIMS assessment results are dependent on assessors. Organizations are also interested in using the REAIMS maturity model for long-term monitoring and benchmarking. However, we do not recommend practitioners to use the model for these purposes as such because the results of REAIMS assessments are subjective and dependent on how assessors interpret the guideline descriptions of the REGPG. To enhance the reliability and repeatability of assessment results, the REAIMS maturity model needs to be developed further by augmenting it with a set of indicators. The indicators would help assessors to interpret the descriptions of the RE practices more unambiguously and to judge the state of the RE practices more objectively. A systematic assessment process also improves the reliability of assessment results.

The second weakness of the REGPG is that selecting a realistic set of improvement actions based on it is a demanding task. To convert the assessment findings into recommendations and further into improvement actions is difficult and requires time, effort, and expertise of requirements engineering. The REGPG introduces sixty-six practices and organizations can select too many of them to be implemented at the same time. This can cause an improvement cycle to take too long, in which case people can become frustrated or even lose interest in RE process improvement. Therefore, organizations need more advice on how to select RE practices that are the most critical and beneficial ones for them.

The REGPG offers the top ten practices for organizations starting RE process improvement. Sommerville and Sawyer (1997) state that these ten practices are so important that they should be implemented in all organizations and that organizations should start their process improvement program by implementing them. However, they do not provide empirical evidence for this statement.

Table 14 shows how widely the top ten practices of the REGPG were used in the case organizations at the end of the process improvement project. Our results indicate that there are organizations that do not find all the top ten practices critical. The top-ten list includes two practices that have never been used in the case organizations. In addition, two practices were only used in some product development projects. Our research results suggest that the list of the top ten practices may need modifications and that more empirical evidence may convince practitioners to apply them.

Table 14. Usage scope of the top ten practices of the REGPG in the case organizations

Top ten practice	How widely is the practice used in the organization?			
	Organization A	**Organization B**	**Organization C**	**Organization D**
Use a standard structure in requirements documents	Normally	Normally	Normally	Normally
Make the document easy to change	Never	Never	Never	Never
Uniquely identify each requirement	Normally	Systematically	Normally	Normally
Use policies for requirements management	Sometimes	Sometimes	Sometimes	Sometimes
Use standard templates for representing individual requirements	Normally	Sometimes	Sometimes	Normally
Use language simply, consistently and concisely	Sometimes	Sometimes	Sometimes	Sometimes
Organize formal requirements inspections	Sometimes	Never	Never	Sometimes
Use validation checklists	Sometimes	Never	Never	Never
Use checklists for requirements analysis	Never	Never	Never	Never
Plan for conflicts and conflict resolution	Never	Never	Sometimes	Never

The third weakness of the REGPG is that it offers only four explicitly defined guidelines for process improvement. The REGPG emphasizes that it is important to find an evangelist and try out changes in pilot projects. In addition, it points out that organizations should allow enough time to make the process change and to respect the skills of professionals involved in requirements engineering. Our research results indicate that these four guidelines supports RE process improvement. However, they are not enough to ensure systematic and successful process improvement. Therefore, organizations that want to develop their RE process systematically need to support the use of the REGPG with other improvement frameworks.

3.1.3 Summary

Figure 13 summarizes how the REGPG supports the different process improvement activities. The research results indicate that the REGPG supports the assessment phase in particular. REAIMS assessments raises practitioners' awareness of requirements engineering, motivates them for RE process improvement, and improves communication of the existing good practices within an organization. In addition, the REGPG supports the development of the RE processes by introducing a large set of good RE practices that can be included in RE processes of organizations.

On the other hand, a large set of practices also causes problems. Practitioners have difficulties in identifying the most critical and beneficial practices, and they tend to implement too many of them at the same time.

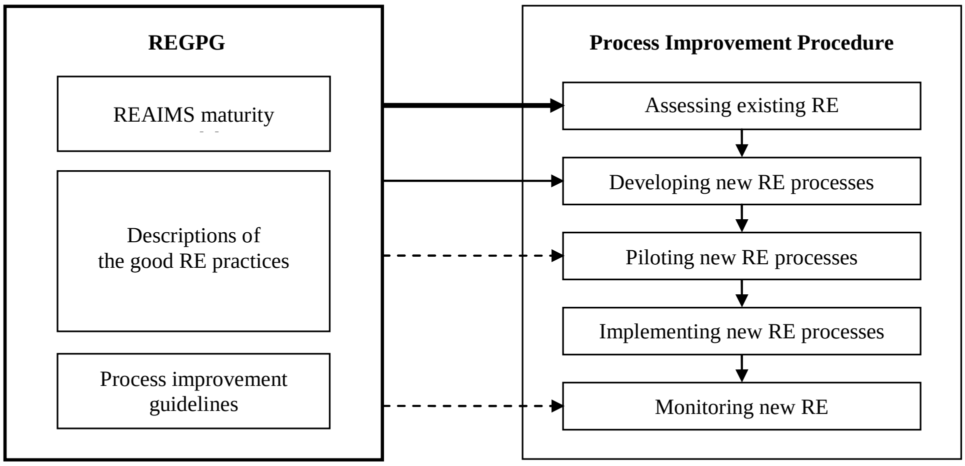

Figure 13. Process improvement activities supported by the REGPG

The REGPG supports the piloting activity of the process improvement procedure in a limited way (Figure 13). The REGPG emphasizes that it is important to introduce process changes in pilot projects and identify the advantages and disadvantages of the change. In addition, it points out that projects with very tight deadlines should not be used as pilot projects.

Because results of REAIMS assessments are subjective and dependent on how assessors interpret the guideline descriptions of the REGPG, the REGPG supports only in a limited way the monitoring activity of the process improvement procedure (Figure 13). The results of two assessments can be made more comparable if these assessments are made by the same person, and a systematic assessment process is used. In particular, it is important that the assessor documents the evidence of the existing RE practices that justifies the assessment score for each practice.

3.2 Basic improvement actions and RE practices

The second research objective of this thesis was to identify basic RE practices that support systematic user requirements definition. This objective has been addressed in three publications of the thesis. Paper II provides a list of fourteen RE practices that the case organizations selected as their improvement actions. Paper III proposes four improvement activities for companies that are beginning to invest in their RE processes. And, finally, Paper IV describes a small set of RE practices that support the systematic definition of user requirements. This section summarizes basic improvement actions and recommends a set of RE practices for organizations that want to introduce requirements engineering into their product development and support systematic user requirements definition.

3.2.1 An information model for RE

One of our main findings relates to the differing understanding of requirements. Practitioners do not mean the same thing when they discuss requirements. In most cases, to product development personnel, requirements meant technical requirements that describe the internal functions and properties of the system and were in effect design solutions. On the other hand, some practitioners, for example, product managers who have direct contacts with customers, used the term to mean external properties of the system to be developed.

Another important finding was that requirements were mainly documented from a technical point of view in the case organizations. In most cases, customer- and user-related information was tacit knowledge in experienced people's heads. Figure 14 shows an information model the aim of which is to support organizations in making the customer- and user-related tacit information more explicit. The model is based on the good experiences gathered from the case organizations.

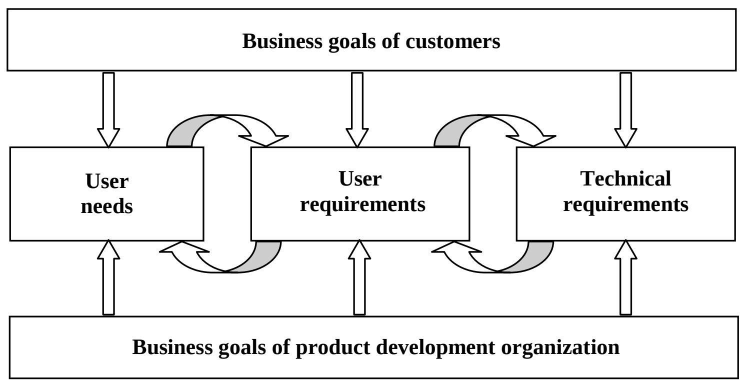

Figure 14. A simple information model for RE

First of all, it can be useful to make a distinction between customers and users because they may not be the same. The IEEE Standard 830-1998 offers suitable definitions. Customers are persons who pay for the product. Users are persons who operate or interact directly with the product.

It is also beneficial to document high-level goals from the points of view of both customers and the product development organization. Documenting these high-level goals is the first step towards making tacit business information explicit. Customers have objectives that define the reasons for buying a new system. Customers may, for example, want to use the new system to reduce costs or to improve the quality of business services. The product development project also has its business goals that define reasons for producing the new system from the company's perspective. The company may, for example, want to increase market share by developing a new version of the system for a new user group.

The customer's primary goal is usually to buy a system, which supports users in their tasks (Coble et al. 1997). However, customers do not typically know what users really need and what is essential to their tasks. To connect customers' business goals to technical requirements, two types of user information were found to be useful (Figure 14). Firstly, user needs describe problems and opportunities related to the current situation and context of use. "Context of use" refers to user characteristics, users' goals, tasks, equipment, and the environment in which a new system will be used. Secondly, the user needs are analyzed and refined to user requirements that define the external behaviour of the new system from the users' point of view.

In other words, user needs refer to problems that hinder users in achieving their goals, or opportunities to improve the likelihood of users achieving their goals (Kujala 2002). User needs can be considered to be raw requirements, which can be wishes and demands explicitly articulated by users. They can also be hidden needs that users are not able to express. User requirements are externally visible functions, properties, or constraints that the system must provide to fill the needs of its intended users. Technical requirements describe the internal functions, properties, and constraints of the system. Some of them can be derived from user requirements and some of them can originate, for example, from domain knowledge, standards, or regulations.

User needs are often more informally documented than user requirements. Product development personnel also find it important to create traceability links between user needs and user requirements, and also to technical requirements (Figure 14). A prerequisite for forward and backward traceability is a unique identifier for each user need, user requirement and technical requirement.

3.2.2 A process model for user requirements definition

All the case organizations defined a process model to give an overview of requirements engineering. The main idea was to keep the RE process model simple. Because the personnel of the case organizations were not aware of requirements engineering, the simple model helped practitioners to understand the basics of the systematic requirements definition. Three organizations used the RE activities of the REGPG as a basis for their first RE process model. The REGPG recommends that a good RE process includes the following activities: elicitation, analysis combined with negotiation, and validation (Sommerville and Sawyer 1997). Organization D decided to tailor the requirements subprocess of the Rational Unified Process (RUP) for its purposes, because they were interested in implementing the RE tools developed and marketed by Rational Software.

Figure 15 presents a simple process model that is based on the experiences of RE process modelling and the observations of RE activities in the case organizations. The purpose of this model is to facilitate practitioners' understanding of the basics of RE and to give a structured overview of RE activities. In practice, RE activities are interleaved.

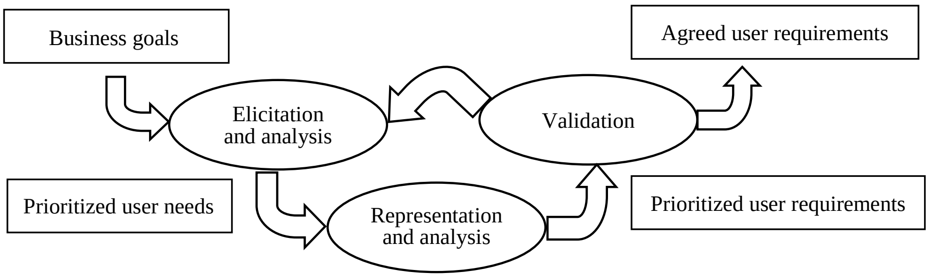

Figure 15. A simple process model for user requirements definition

3.2.3 A basic set of good RE practices

In this section, we first show the practices of the REGPG that have been used in all the case organizations (Table 15). Then we represent the main observations related to the most widely used RE practices. However, we do not have observations on how to model system architecture and how to develop complementary system models (row 13 and 14 in Table 15), because the focus of the thesis is on user requirements.

Uniquely identify each requirement. At the beginning of this study, many requirements documents were based on narrative text and it was difficult to recognize individual requirements from the text. One of the first steps towards unambiguous requirements was to set an identifier to each of them. This helped readers of the requirements document to recognize individual requirements. In addition, identifiers made review meetings more fluent. Actually, it was difficult to organize inspections if requirements did not have a reference number. Without reference numbers people had great difficulties in pointing out which requirement they were talking about.

Prioritize requirements. All the case organizations defined an attribute called priority to be linked to each requirement. Requirements were prioritized in three categories such as high, medium, and low. The high means that the requirement is critical and must be included in the system, the medium that the requirement is important, and the low that the requirement is desirable but can be left out from the release if there are schedule problems. In some cases, the categories were named in a way that the practitioners had difficulties in understanding what they meant and what the difference between them was. In addition, the practitioners seemed to classify most of the requirements as critical, while only very few requirements were assigned a low priority.

Table 15. Practices of the REGPG that have been used in all the case organizations

Id	Practice	How widely is the practice used in the organization?			
		Organization A	Organization B	Organization C	Organization D
1	Uniquely identify each requirement	Normally	Systematically	Normally	Normally
2	Prioritize requirements	Sometimes	Systematically	Systematically	Normally
3	Use a standard structure in requirements documents	Normally	Normally	Normally	Normally
4	Check that requirements documents meet organization’s standard	Sometimes	Normally	Normally	Normally
5	Use standard templates for representing individual requirements	Normally	Sometimes	Sometimes	Normally
6	Identify and consult system stakeholders	Normally	Sometimes	Sometimes	Normally
7	Use software to support negotiations	Sometimes	Normally	Normally	Sometimes
8	Use scenarios for representing requirements	Normally	Sometimes	Sometimes	Sometimes
9	Use multi-disciplinary teams to review requirements	Normally	Sometimes	Sometimes	Sometimes
10	Make a business case for the system	Sometimes	Normally	Sometimes	Sometimes
11	Use policies for requirements management	Sometimes	Sometimes	Sometimes	Sometimes
12	Use language simply, consistently and concisely	Sometimes	Sometimes	Sometimes	Sometimes
13	Model system architecture	Sometimes	Sometimes	Sometimes	Normally
14	Develop complementary system models	Normally	Sometimes	Sometimes	Sometimes

Use a standard structure in requirements documents. All the case organizations defined a template for requirements documents, and practitioners were willing to use these templates. However, using them was not easy. A template with short guidelines on how to use it did not seem to be enough. The practitioners also wanted to have real document examples from the organization’s own application domain. Another way to support the adoption of the template was to involve an RE expert in the team when it was using the template for the first time.

Check that requirements documents meet organization’s standard. None of the case organizations had quality management procedures where these kinds of checks would have been formally carried out. However, all the case organizations had an unwritten principle that product development projects were supposed to use the requirements document template, and people seemed to be aware of this principle. In addition, one of the case organizations had created a checklist for requirements validation, and one of the items in this checklist was to verify that the standard template had been used.

Use standard templates for representing individual requirements. The case organizations had not specified a standard way to describe individual requirements. However, the writers of the requirements documents had tended to use the same structure for similar kinds of requirements. For example, a structure “the user does…” had been used in some of the use case documents. In some of the system requirements documents, the structure “the system shall be able to…” had been applied.

Identify and consult system stakeholders. None of the case organizations provided guidelines for product development projects on how to identify essential stakeholders and on how to systematically discover their needs. In many product development projects, internal domain experts had, however, been consulted during requirements definition. In addition, the focus had started to shift from technical requirements towards user requirements. This meant that some product development projects had had direct contacts with customers and users in order to discover their real needs. However, users were not good at articulating their needs. Therefore, in addition to being interviewed, users were observed in their own environment. The focus was not on what users said they wanted or needed, but on understanding their goals and current tasks, while recognizing users as experts in their tasks.

Use software to support negotiations. In all the case organizations, the practitioners used electronic mail to discuss requirements. Usage of email was dependent on individuals, and there were no guidelines on when it was good to use email and when it was good to have direct contacts. Some product development projects tried to speed up the RE process by sending the requirements document to a small group of the stakeholders and asking for feedback through email. In these cases, the stakeholders did not react and did not give any feedback. On the other hand, in one case organization, there was a requirements management team that used email effectively to exchange information about, and to comment on, requirements documents.

Use scenarios (use cases) for representing requirements. Representing requirements from the users’ point of view was not easy because software engineers were used to specify technical details. All the case organizations piloted use cases in representing user requirements. Even though the use case approach is simple, it did not automatically support engineers in describing the external behaviour of the system. In some projects, software designers went on to describe the internal behaviour of the system, and sometimes they defined details of the user interface using use cases. To overcome this granularity problem, the basic principles of use cases had to be emphasized. The basic principles included in the training material of the case organizations were as follows:

- Use cases are a way of specifying functionality from a user's point of view (Rumbaugh 1994).
- The system is considered as a black box: We are interested in externally visible behaviour (Rumbaugh 1994).

Furthermore, documented user needs helped software engineers write use cases at the appropriate level and from the users' point of view. Descriptions of users' present tasks provided information that was especially useful for use cases.

Use multi-disciplinary teams to review requirements. All the case organizations had organized informal reviews where requirements were discussed. People from different backgrounds like designers, project managers, test managers, domain experts, product managers, and internal customers had participated in these reviews. In addition to informal reviews, one of the case organizations tried to organize formal requirements inspections. The main problem was that practitioners did not have time to read requirements documents before the inspection meeting.

Make a business case for the system. According to the REGPG, the requirements document should always summarize the business objectives. In two case organizations, the requirements document template contained a section where reasons for the system development were supposed to be described. Some of the product development projects had used this section for describing business goals, but some projects had not written anything in this section, even though they had used the template. Furthermore, one of the case organizations had a separate document for business information called a business plan. Some of the business plans had been written very thoroughly and they included valuable information for requirements definition.

Use policies for requirements management. All the case organizations defined a high-level process model for requirements definition and two of them had detailed instructions on how to gather, analyze and validate requirements. The practitioners were aware of the high-level process model and found it suitable for most of the product development projects. The detailed instructions were not used. The practitioners found them either impractical or too heavy.

Use language simply, consistently and concisely. The quality of the language used in the requirements documents was mainly dependent on the writing skills of individuals. Some projects had paid special attention to writing requirements clearly and understandably. However, none of the organization had guidelines for ensuring the quality of the language used in requirements documents. In addition, the practitioners seemed to be satisfied with the language used in the requirements documents.

3.2.4 Summary

Based on the experiences of the case organizations, we recommend the following four improvement actions for organizations to begin with RE process improvement:

- Define a template for user requirements documents.
- Define standard structures for representing individual user requirements.
- Define a simple process for user requirements definition including a small set of RE practices.
- Pilot the requirements definition process and collect examples of good requirements and requirements documents.

Figure 16 summarizes the inputs, outputs, and main activities of the requirements definition process. In addition, it shows a set of RE practices that have been found useful on the basis of the experience gathered from the case organizations. The focus of the process model is on user requirements. We found two RE practices that especially support the systematic definition of user requirements: 1) discovering user needs actively, and 2) representing user requirements in the form of use cases. The main principle of the active user need elicitation is that needs are gathered directly from real users in their own environment.

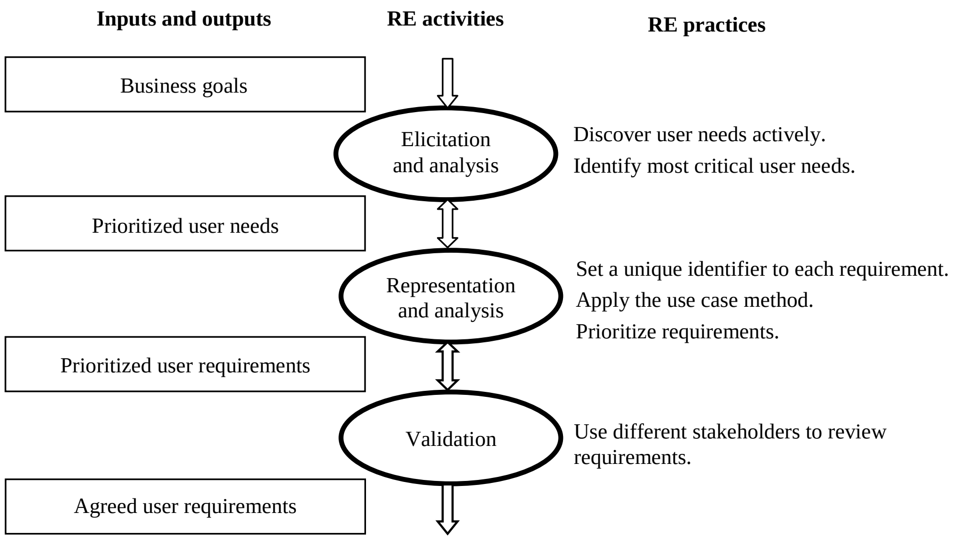

Figure 16. A requirements definition process and a set of basic RE practices

3.3 Success factors for introducing RE into product development

The third research objective of this thesis was to identify factors that support the introduction of requirements engineering. This research objective has been addressed in three publications. Paper III describes a small set of possible success factors for RE process improvement and investigates how these factors influence the success of the RE process improvement at the beginning of the improvement projects. In Paper IV, we describe what kind of cultural change the introduction of requirements engineering can require and which factors support the success of the cultural change. Paper V focuses on factors affecting the organization-wide implementation of RE practices. The following seven sections describe briefly factors critical to the successful introduction of requirements engineering.

3.3.1 Human factors

One of our main findings was that introducing requirements engineering can involve a cultural change. Such a cultural change requires that requirements are defined systematically, not only from a technical point of view, but also from the customers' and users' points of views. It is challenging to make a change of behaviour happen in practice, because both managers and product development engineers can hold beliefs that prevent systematic user requirements definition. For example, people can assume that users do not have any needs for new products. Therefore, instead of discovering real user needs systematically, product development engineers invent user requirements themselves. These kinds of beliefs are the most challenging obstacles to cultural change, because they also relate to values and attitudes.

Another important finding was that the change in behaviour has to happen first at the individual level, then at the project level, and finally at the organizational level. Therefore, a key issue in the RE process improvement is to emphasize that the purpose of the new process is to help practitioners do their job. When the new RE process is introduced, it is essential to respect the skills of the practitioners, and not to point out what they have done poorly in the past.

In order to make new RE practices permanent, the RE awareness needs to spread throughout product development management and personnel. Firstly, product development personnel need to understand what RE means and how they can benefit from the new RE practices. In addition, raising management awareness of requirements engineering is vital. If management does not understand why they should invest in requirements engineering, the implementation of the new RE practices throughout the organization might stop.

Practitioners are afraid of wasting time doing unpractical things that do not solve any existing problem or provide any benefits. When the practitioners can see results from using the new RE practices, they become motivated to apply the practice again in the future. Enthusiasm and pride support people's commitment to the new RE practices.

The lesson learned is that the change required by the RE process starts from individuals, and therefore it is vital to treat people as experts and respect their skills. If the individuals are motivated and enthusiastic in performing the new RE practices, it is more probably that these practices will become permanent.

3.3.2 Usefulness of the RE process

We identified three factors that affect how useful people judge the new RE process. The first factor concerns how soon people can see the benefits of the new RE practices. People are more motivated to change their way of working if they can see the results of the change in the near future. If the benefits of the new RE practices can be achieved only in the long term, people give them up more easily.

The second factor related to usefulness concerns those for whom the new practices are beneficial. People are more motivated to change their behaviour if they can see that the results of the change are likely to be useful to themselves or to other project members. If the benefits of the new RE practices can be seen only at product development or company levels, people seem to give them up more easily.

The third factor related to usefulness is how valuable and measurable the benefits of the new RE practices are. The managers in particular want to have measurable data from the new RE process. Product developers are ready to change their way of defining requirements if they have piloted the new practices and have gained even such fairly abstract benefits as better understanding of the goals of the project or improved communication about the requirements. Reporting positive experiences of other projects or other companies is a good way to sell the new RE practices to project teams.

The lesson learned is that the more immediate, personal, and concrete the benefits of new RE practices are, the better the chances that organizations will succeed in implementing them permanently throughout the organization.

3.3.3 Practicality of the RE process

Our research results highlight that practitioners want to have RE practices that are easy to learn and use. People do not have time for long training courses because of the tight schedules of the product development projects. The practitioners are satisfied if they can understand the new RE practice, method, or technique in a day or even in half a day. It is also important that the new RE practices do not require weeks of effort to be performed. If the practice takes weeks or months of effort, it is important that it can be performed step by step.

People also want to have a flexible RE process, which means that the process can be tailored to the needs of their project. Practitioners pointed out that the same RE practices do not automatically suit all kinds of situations. For example, a project developing a new version of a product might need more lightweight RE practices than a project developing an entirely new product.

The lesson learned is that the simpler and easier to learn and apply the new RE process is, the more willing practitioners are to use it. In addition, practitioners want to have a flexible RE process that they can adjust according to the needs of their product development project.

3.3.4 Training

The research results indicate that introducing the basics of requirements engineering seem to require a day's training course. The purpose of the basic training is to describe why RE is important, and to give an overview of the RE process. However, the traditional classroom style of teaching is ineffective when the objective is to integrate the RE process with daily routines. "Just-in-Time" training combined with learning-by-doing is required to change the way in which product development teams define requirements in practice. Hutchings et al. (1993) introduced the principle of "Just-in-Time" training, according to which teaching must fully support the team's work and occur at the time the team needs it.

It is also important to train all the persons that will participate in requirements definition and management. This facilitates people who mean the same thing when they are talking about requirements, and have a common view on how to define and manage requirements in practice.

The lesson learned is that the traditional classroom style teaching suits training that aims at raising personnel awareness of requirements engineering. "Just-in-Time" training is needed if the organization wants to implement the new RE practices permanently.

3.3.5 Support

The experiences in all the case organizations show that management support for requirements engineering is vital. In the case organizations, where the support of the senior managers was concrete and visible, the implementation occurred more smoothly than in the organization where senior managers did not have time to be involved in the RE process improvement. It is also important that project managers support people in performing the RE practices.

The experiences also indicate that the more demanding RE practices require an RE expert's support when people perform these practices for the first time. In addition to management and RE expert support, people also want to have document templates, examples, and practical guidelines for defining requirements.

The main lesson learned is that concrete and visible support from all management levels is vital for successful RE process implementation. In addition, an RE expert's assistance, a requirements document template combined with real examples, and practical guidelines support the implementation of the RE process.

3.3.6 Implementation strategy

The experiences in the case organizations indicate that the successful implementation of the new RE practices requires a systematic approach. If the implementation approach is haphazard, only some people become aware of the new RE practices, and furthermore, there would be a considerable risk that product development projects would give up the new practices under the pressure of tight schedules. The systematic approach covers such issues as how the training and support should be organized. It should also include a plan of how to collect feedback from the new RE practices and how to improve the RE processes continuously.

Furthermore, the success of the RE process implementation is dependent on the magnitude of the change. In the case organizations, where the changes to the current state were small and incremental, people were more willing to apply the new practices than in the organizations that tried to make a large change in one go.

In one case organization, people had strong attitudes unfavourable to new processes. On the basis of their earlier experiences, the practitioners thought that the new process was merely bureaucracy and defined from a management perspective. The experiences showed that people changed their minds if they could see how the new process helped them to do their job better.

The lesson learned is that successful implementation of RE practices is based on a systematic, incremental and people-oriented strategy.

3.3.7 Improvement activities

The experiences in all the case organizations show that it is important to involve representatives of all process user groups in defining the new RE process. The role of the practitioners is to bring knowledge of the existing RE practices and to ensure that the new RE process is practical and satisfies the needs of product development projects. In addition, it is important to pilot RE practices before they are implemented organization wide. Piloting demonstrates the benefits and the shortcomings of the new RE practices. It is also important to modify the new RE process on the basis of the feedback gathered during the piloting.

The lesson learned is that it is essential to involve representatives of all process user groups in defining the new RE process. It is also important to pilot new RE practices before they are implemented organization wide.

3.3.8 Summary

The research results indicate that implementing the RE process throughout the organization is a complex phenomenon. Its success depends on many factors, and, furthermore, these factors are interrelated. The key factors can be grouped into categories (Figure 17). The aim of Figure 17 is to give a structured view of the key factors and propose their main relationships. According to Strauss and Corbin (1998), grouping concepts into categories is important because categories have the potential to explain and predict about the phenomenon under study.

One of the main findings is that RE process change is bottom-up. The change of behaviour needs to happen first at the individual level, from which it can spread to project teams and finally to the entire organization. Therefore, we classify human factors as a central category (Figure 17). We also suggest that these human factors can explain why people may resist a process change. For example, if people do not understand the reasons for a process change, i.e. why they have to change their way of working and what they gain from the new process, they probably will not be willing to adopt it. The new process can also require new skills. If people are not offered adequate training, they can be reluctant to change their working practices.

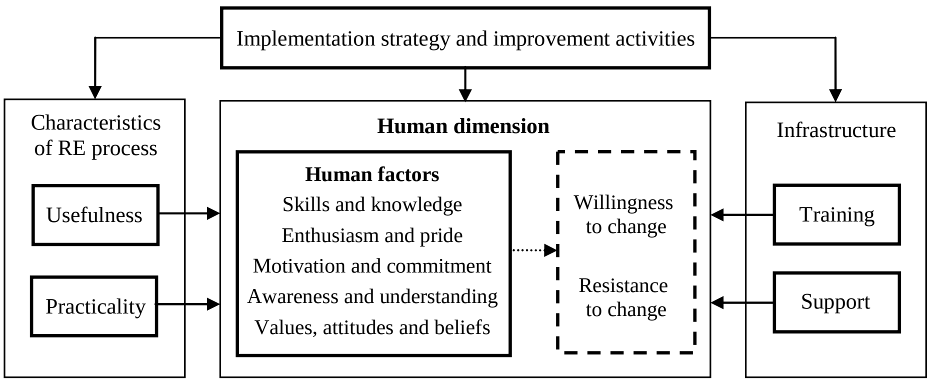

Figure 17. Model of the factors affecting organization-wide implementation of RE processes

Figure 17 shows only a limited set of the relationships between the success factors and categories. Our research suggests that there are several other relationships between the factors. For example, if the process is simple, it is also easy to learn and apply, and then it does not require massive training and support. Human factors are also dependent on each other. First, people must be aware of the new RE process, understand its importance and how to use it. They can then become motivated and committed to performing the new RE practices.

In addition to Figure 17, Appendix E also summarizes the success factors. It provides the guidelines for RE process implementation from the perspective of practice. The guidelines are based on the factors found in the software process and RE process literature, and supplemented by the lessons learned from the four case organizations of this study.

3.4 Challenges for introducing RE into product development

The fourth research objective of this thesis was to identify challenges organization can face when introducing requirements engineering into product development. This research objective has been addressed in two publications of the thesis. In Paper IV, we describe what kind of cultural change the introduction of requirements engineering might require and which factors prevent the success of cultural change. Paper V focuses on factors affecting the organization-wide implementation of RE practices. The following six sections summarize critical challenges that appeared in the case organizations when they started to introduce requirements engineering into product development.

3.4.1 Duration of the change

Our research results imply that the introduction of RE takes time. In particular, cultural change takes place slowly. Product development personnel are so used to focusing on technical details that they have difficulties defining user requirements. In addition, people can have beliefs and attitudes related to the systematic definition of user requirements, which makes the change deeper and more time-consuming.

The risk related to the long duration of the change is that managers may start to lose their interest because they cannot perceive the high-level benefits deriving from the RE process improvement. If the management stops emphasizing the importance of requirements engineering, product development personnel start losing their confidence in the value of defining requirements systematically.

The lesson learned is that organizations need to allow enough time to make the change, especially if they are just starting to introduce requirements engineering. In addition, introducing RE appears to involve a cultural change, and therefore the organization-wide implementation of the RE process might take several years.

3.4.2 Personnel changes

Personnel changes took place in all the case organizations during RE process improvement. For example, in one case organization, a representative of the senior management was actively involved in the project of the RE process improvement. Because of the organizational change, this manager could not continue to support the RE process improvement. In this same case organization, the manager of the process improvement project changed. The position of the new person was not strong, because she was not named formally as a project manager. The implementation of the RE process slowed down after these personnel changes.

The risk related to personnel changes is that key persons leave and the RE process disappears with them. The lesson learned is that organizations need to have several people responsible for improving the RE process and should not rely on just one change agent or evangelist.

3.4.3 Training and support resources

The research results suggest that "Just-in-Time" training and RE support require persons that have good knowledge about the RE process and practices of the organization. In addition, experience of product development and knowledge about application domain increase these persons' capability to help project teams to tailor the RE process and to apply RE practices to the needs of the projects.

The risk related to training and support resources is that organizations find "Just-in-Time" training and RE support too expensive or too difficult to organize, because all the skilled persons are tied up with product development work. Another risk is that product development teams expect the RE expert to carry out the requirements definition work. Therefore, the expert can become a bottleneck or, in the worst case, people do not learn to apply the RE practices themselves.

The lesson learned is that "Just-in-Time" training and RE support require skilled persons and investment. If "Just-in-Time" training combined with RE support is well planned and organized, it can pay off and be more cost-effective than the traditional classroom style of teaching and external RE courses.

3.4.4 Scope of the RE process

The experiences from all the case organizations showed that requirements engineering also relates closely to business processes such as strategic planning and roadmapping. In addition, sales, sales support, and marketing personnel have valuable knowledge and information for customer and user requirements definition. However, it can be difficult to involve these stakeholders in requirements definition because they are not able to see how to combine the RE practices with their existing tasks, and what they can gain from participating in requirements definition.

The risk related to the scope of the RE process is that widening the scope by linking the RE process with business processes makes the change bigger, more difficult, and more time-consuming.

The lesson learned is that RE processes sometimes concern not only product development but also such organizational units as sales, sales support, and marketing. Linking requirements engineering to a company's business processes can increase the usefulness of the RE process and support its organization-wide implementation as well.

3.4.5 Implementation of an RE tool

All the case organizations were interested in acquiring an RE tool. One case organization defined the RE process first, and subsequently acquired an RE tool. The use of the tool made a set of RE practices systematic and supported the organization-wide implementation of the RE process. Some practitioners were very satisfied with the tool, while others would have improved its implementation. The practitioners emphasized that their own company had to invest in tailoring the RE tool to the needs of product development projects. They wanted to have, for example, company-specific document templates and reports ready for product development projects.

One of the case organizations bought an RE tool to support the handling of the so-called raw requirements received from different sources. This organization defined its RE process after it had bought the tool. Some practitioners were satisfied with the tool. Some practitioners, however, said that it supported only partly the RE process of the organization, and that it was difficult get an overview of the requirements stored in the tool.

One case organization piloted an RE tool. People that participated in the piloting had different views on how useful the tool was. Some practitioners could not see real benefits to be received from the use of the RE tool, while others found it fundamental for managing a large proportion of requirements. The piloting results were so promising that the manager of the RE process improvement project recommended that top management acquire the RE tool. However, top management decided not to buy the tool because it could not free any resources for the tool support.

The risk related to the implementation of an RE tool is that practitioners expect the tool to solve, for example, traceability and requirements management problems automatically. Another risk is that organizations underestimate resources needed for tool implementation and support.

The lesson learned is that an RE tool can support the organization-wide adoption of RE practices if it is well integrated with the RE process and thoroughly implemented.

3.4.6 Measurement

Managers are interested in evaluating the benefits and costs of the new RE process. However, it is difficult to set measurable goals for RE process improvement and to evaluate the impact of process improvement efforts. A reason for these difficulties is that organizations do not have quantitative data from their existing RE practices. On the other hand, it is unrealistic to expect organizations that are just starting to improve their RE process to have measured their RE practices.

The risk related to measurement is that people might fear that the purpose of measurement is to evaluate their performance instead of evaluating the quality and the benefits of the RE process. Personal incentives that are given on the basis of the measurement data can increase this fear. Another risk related to measurement is that organizations collect data without improving the RE process based on this data.

The lesson learned is that measuring the benefits and costs of the RE process could support the organization-wide implementation of the process. However, it is a difficult task to perform in practice.

4 Discussion

This chapter presents the main findings of the study and compares them with the results of previous research. The findings are discussed here according to the research questions which were formulated as follows:

- How does the REGPG help organizations in RE process improvement?
- Which practices support systematic user requirements definition?
- What are the potential success factors that support the introduction of RE into product development?
- Which challenges do organizations face when they introduce RE into product development?

At the end of the chapter, the strengths and weaknesses of the action research method and the limitations of the study are discussed. Furthermore, we discuss how the underlying assumptions of the study may have affected the research results.

The REGPG

Our results indicate that conducting an RE assessment with the REGPG is useful for organizations that are just starting to improve their RE processes because it raises personnel awareness of RE and existing RE practices. This result does not support one of the assessment problems described by Fayad and Laitinen (1997), according to whom assessments can be a waste of money and time for organizations just starting because, in an immature organization, the results are meaningless. They also state that all assessment models are artificially derived and include idealized lists of practices. According to our study, the REGPG seems to include a large set of relevant RE practices for organizations starting to introduce RE into product development.

On the other hand, the large set of RE practices also causes problems. Practitioners tend to implement too many of them at the same time. However, the problem moving from assessment results to realistic improvement actions does not seem to be specific just to the REGPG. For example, Zahran (1998) points out that moving from assessment to improvement planning is often difficult for many organizations. According to him, one of the reasons is that organizations try to address too many problems at once. Similarly, Turner (2003) also highlights in his article concerning pitfalls in the hunt for best practices that it is not easy to evaluate how appropriate a practice is for a particular organization.

Sawyer (2004), one of the developers of the REGPG, reports that the assessment method of the REGPG has been recognized to be too passive, and it was modified, therefore, to include an explicit step for selecting good practices. In addition, this was backed up with the development of a decision support tool that processed the analysis data and listed those best practices most likely to provide solutions (Sawyer, 2004).

Recently, Sommerville and Ransom (2005) have reported the results of the first systematic evaluation of the REGPG. According to their study, the change from a passive assessment model to an active improvement model and the development of tool support increase the practical usefulness of the REGPG. Furthermore, their results confirmed the usefulness of this RE process maturity model and improvement framework across a range of different types of companies. Similarly, our results indicate that the REGPG includes relevant RE practices for different kinds of application domains.

Cultural change towards systematic user requirements definition

One of our main findings was that introducing requirements engineering appears to involve a cultural change in product development. It was observed that product development engineers are used to documenting requirements from a technical point of view, but customer- and user-related information is often tacit knowledge in experienced people's heads. The cultural change means that product development engineers have to change their way of thinking as well as of working. Instead of describing requirements only from a technical point of view, they should define requirements systematically from the customers' and users' points of view also. Two other case studies (Hutchings and Knox 1995; Jacobs 1999) point out that the cultural change towards systematic customer requirements management is a key challenge in RE process improvement.

In order to make user requirements definition more systematic, we recommend organizations to develop a separate document template for user requirements. Our findings support the research result reported by Kamsties et al. (1998). According to them, a requirements document is the basic requisite for all other RE activities. In addition to implementing a separate document template for user requirements, we identified two RE activities that especially support a cultural change in practice: 1) eliciting user needs actively, and 2) representing user requirements systematically in the form of use cases. The main principle of the active user need elicitation is that needs are gathered directly from real users in their own environment. Also, Lubars et al. (1993) report that most informants in their field study felt that they understood the requirements best when they interacted directly with users. Furthermore, Keil and Carmel (1995) found that the more successful projects employed more links to customers and users than did the less successful ones.

Use cases are a good way of representing user requirements because they support practitioners in specifying externally visible functionality of a system from the users' point of view and in a systematic way. In addition, user requirements, represented in the form of use cases, seem to serve as a means of communicating the functions of a new system or a new version of the system with designers, testers, user-manual writers, and sales personnel. The experiences of the case organizations support the findings of Weidenhaupt et al. (1998) who report that practitioners apply use cases for several different kinds of purposes. Furthermore, the survey performed by Neill and Laplante (2003) revealed that over 50 percent out of 194 respondents used scenarios or use cases in the requirements phase. This might indicate that use cases provide concrete benefits for practitioners, and is therefore perhaps one of the most widely used RE techniques in practice.

Success factors

Another main finding of the study was that RE process change is bottom-up. The change of behaviour needs to happen first at the individual level, after which it can spread to project teams and finally to the entire organization. Therefore, such human factors as motivation, commitment and enthusiasm are fundamental for the success of RE process implementation. Our findings support those of Hutchings et al. (1993), Basili et al. (1995), and O'Hara (2000). For example, Basili et al. (1995) report that software process change is bottom-up and direct input from developers is a key factor in change. According to O'Hara (2000), winning the hearts and minds of people is crucial to successful software process improvement. Furthermore, Hutchings et al. (1993) point out that process improvement changes the way people work and is fundamentally as much a human concern as it is a technical one.

We identified two characteristics of the RE process that are vital for its successful implementation. First, the process must be useful to its users. If it offers concrete benefits, people can become motivated and committed to applying it in practice. Similarly, Zahran (1998) emphasizes that any change to the current processes must carry with it benefits for project managers and software engineers. Furthermore, Sakamoto et al. (1998) report that the most effective way to convince stakeholders, including software developers and top management, is to present evidence of problems in the current development and evidence of improvement.

Another important characteristic of the RE process is practicality. Practicality means that a process must be both simple and flexible. A simple RE process facilitates an understanding of the basics of RE by personnel and gives them an overview of RE. According to Armour (2001), projects that are lacking awareness cannot use a detailed process because they do not know what process might work. Furthermore, Ward et al. (2001) point out that processes must be simple because complex processes are difficult to follow and update and quickly become unsuitable for the operations for which they were originally specified.

Our results indicate that successful implementation of RE processes requires training. "Just-in-Time" training combined with learning-by-doing seems to be an effective way to help people applying the RE practices in real product development projects. Hutchings et al. (1993) report that teams will invest themselves in the sometimes painful change of adopting new processes or techniques when 1) the teaching occurs when they are ready, 2) the teaching material is focused on their actual problem or situation, and 3) the teaching is accompanied by expert facilitation and consulting. The experiences in our case organizations support these findings of Hutchings et al.

The results of the study also show that concrete and visible management support is vital for successful RE process improvement, especially when the RE practices are implemented throughout the organization. Our findings also indicate that management support is particularly important when RE process improvement concerns not only product development, but also such organizational units as sales, sales support, and marketing. In practice, management can show concrete support by, for example, offering resources for "Just-in-Time" training.

Main challenges

One of the main challenges related to RE process improvement is the duration of the change. In particular, cultural change takes place slowly. Product development personnel are so used to focusing on technical details that it takes time to learn to document user requirements systematically. In addition, people can have beliefs and attitudes related to the systematic definition of user requirements, which makes the change deeper and more time-consuming. Similarly, Sommerville and Sawyer (1997) point out that organizations must budget for several years of improvement effort in order to gain the benefits after new RE processes have come into general use.

One of the main risks related to RE process improvement is that key persons leave the organization, and the RE process disappears with them. There are many publications that recommend a change agent or evangelist (e.g. Humphrey et al. 1991; Sommerville and Sawyer 1997; Zahran 1998). However, our research results indicate that organizations need to have several people responsible for improving the RE process and should not rely on just one change agent or evangelist.

Our findings indicate that an RE tool can support the organization-wide adoption of RE practices if it is well integrated into the RE process and implemented thoroughly. According to Weber and Weisbrod (2003), requirements management tools are the number one instrument for leveraging RE practices. On the other hand, Hofmann and Lehner (2001) report that commercially available RE tools interfered with rather than supported RE activities. They believe that either a lack of well defined RE processes or the RE team members' lack of training in using the selected tools caused this undesired effect. Our findings support the experience of Weber and Weisbrod (2003). According to them, tool support represents both an opportunity and a risk in RE process improvement.

At the beginning of this study, we had an ambitious goal to measure the impact of process improvement efforts. During the first year, we discovered that it was really demanding to set measurable goals for RE process improvement because the organizations did not have quantitative data from their existing RE practices. Furthermore, the analysis of the existing literature revealed that it can be difficult to show how RE process improvements contribute to a project's and product's success (e.g. Davis and Hsia 1994; Jacobs 1999). Measurement can also be both expensive and disruptive (Humphrey 1989). For these reasons, we gave up measuring the business benefits of RE process improvements. Instead, we monitored the impact of improvement efforts by assessing the maturity level of the RE process of the case organizations at the beginning and at the end of the study. Furthermore, we collected feedback about the usefulness of the new RE practices by interviewing people who had applied these practices. The results of the assessments and interviews are described in Paper V.

Action research

The research results presented in this thesis are based on four case studies that were carried out using an action research method, which seems to be very rarely employed in RE and SE research. For example, Glass et al. (2002) examined 369 papers in six leading research journals in the SE field, and they could not find any publication that employed action research methods. On the other hand, Potts (1993) proposes for the SE research community an "industry-as-laboratory" research approach where researchers identify problems through close involvement with industrial projects and create and evaluate solutions in an almost indivisible research activity. In addition, Potts points out that action research is a respected mainstream branch of research in other fields such as organizational behaviour, ergonomics, urban planning, and some of the more influential branches of economics.

In the discipline of information systems (IS), action research is gaining a growing interest (e.g. Baskerville and Wood-Harper 1996; Avison et al. 1999; Baskerville and Myers 2004). For example, Baskerville and Myers (2004) believe that action research methods provide one potential avenue to improve the practical relevance of IS research. On the other hand, Baskerville and Wood-Harper (1996) point out that a number of problems confront the action researcher, such as a lack of neutrality, a lack of discipline, confusion with consulting, and its context-bound nature. They also state that these problems confront researchers using alternative methods as well.

Kock (2003) highlights the challenges associated with action research's dual goals of serving practitioners and the research community. Here, we discuss three issues that Kock deals with in his article. First, he points out that the researcher's actions may strongly bias the results. Similarly, our presence may have affected the internal and external validity of the results. We acted as facilitators and participant-observers in the case organizations. From the perspective of the case organizations, our role was to support improvement activities and provide knowledge about requirements engineering. From the perspective of research, we were observers that adopted a stance of neutrality. In addition, the research periods varied from 1.2 to 3.7 years, which allowed us to validate the findings made in each case organization, in other words to enhance the internal validity of the research results and to reduce the bias of the results.

Secondly, Kock (2003) states that action research is not an “efficient” approach and may require significantly more time and effort from the researcher than other, more traditional research approaches. Our research experiences are somewhat different from Kock’s experiences. At the beginning of this research, we aimed at conducting empirical experiments to investigate the strengths and weaknesses of the REGPG. We spent a reasonable amount of time on defining empirical design for the experiments using the Goal-Question-Metric paradigm (e.g. Basili 1992; Basili 1996). After collecting data for some time, we noticed that it was difficult and time-consuming to gather some of the data in practice. In addition, we realized that some of the gathered data was irrelevant from the perspective of practice. Therefore, we changed our research approach towards action research. Based on our experience, we offer the opinion that action research can be an efficient approach. It allowed us to ensure that the gathered data and research results were relevant also for the practitioners. Furthermore, it allowed us to be open and sensitive to new important issues that arose during the study.

Thirdly, Kock (2003) suggests that action research is particularly well-suited for researchers with previous industry experience and who want to do research related to the solution of complex problems in settings they are familiar with. Our experience similarly indicates that action research is suitable for the investigation of a complex phenomenon such as the introduction of RE into product development organizations. In addition, the author of this thesis has over ten years’ experience in working in industry and is, therefore, interested in research problems that are essential from the perspective of practice. The action research method, the four cases, and the long-term view of this study allowed us to examine RE process improvement and organization-wide implementation of RE practices in depth and from the perspective of practice.

In order to help other RE and SE researchers employ action research, we highlight two issues. Our experiences indicate that action research requires both sensitivity and flexibility from the researcher. At the beginning of the study, the researcher sets up his or her research problem and may have a solution to be offered for case organizations. It is, however, possible that the real problem is not where it was expected to be or that the solution is not appropriate for the case organizations. Therefore, the researcher needs to be sensitive enough to listen to all the practitioners whom the problem concerns. Furthermore, the researcher needs to be ready to modify the solution or even alter the research focus and seek a new solution.

Flexibility also includes a pitfall. The researcher can end up doing pure consultation and providing services that have nothing to do with the original research problem. Based on our experiences, we recommend that the researcher maintains a simple research plan and evaluates the actions performed in the case organizations against the plan. We also emphasize that the statements of the research problem and the more detailed research questions form the core content of the plan and be used as the tool to guide action research. Emphasizing the importance of the research problems and questions may seem to be obvious. However, the researcher can easily lose the focus in action research because the close co-operation with case organizations provides a huge amount of data.

Limitations

In this research, RE process improvement was investigated from the perspective of primary users of RE processes. The sample size can be considered to be rather large for qualitative research. The total number of different informants that were either interviewed or observed in each case organization varied from 10 to 28. However, most of the informants represented the users of the RE process, i.e. persons who were responsible for defining requirements. The research covered only superficially the perspective of other stakeholders such as product development managers and RE process developers.

The external validity of the results is difficult to determine. In other words, one cannot generalize from the four cases, especially in the quantitative sense of word. On the other hand, one can learn from the study of four cases. According to Strauss and Corbin (1998), if concepts are abstract enough, then they are likely to occur in similar or variant forms in other organizations. Moreover, Patton (2002) points out that studying a relatively small number of special cases that are successful at something and can, therefore, be a good resource of lessons learned.

The results presented in this study are based on the experiences gained from Finnish organizations, and therefore there might be issues that may not be appropriate to other cultures. For example, technology-oriented product development is claimed to be characteristic to Finnish companies. Our research results indicate that product development personnel tend to define technical requirements and defining requirements from the customers' and users' points of view can be a challenge for them. However, this phenomenon does not seem to be typical only to the four Finnish companies. Hutchings and Knox (1995), for example, report issues concerning the attainment of cultural change, such as when the Digital Equipment Corporation sought to move from a technology-centric view of product development to a customer-centric view. Furthermore, Stevens et al. (1998) state that somehow it seems to be difficult for engineers to write good user requirements, even when they understand the theory.

Even though the results of the thesis are based on a longitudinal study, we were able to investigate the success of the RE process improvement in the case organizations for less than 3.7 years. A longer research period would provide more knowledge about process improvement aimed at organizational change, in other words, a change that would affect the entire organization permanently. For example, two case studies (Carlshamre and Rantzer 2001; Basili et al. 2002) show that process improvement can be a success for a long time and the most challenging problems can appear over a period of years.

We assume that the results of the thesis can be dependent on many context-specific factors such as the maturity level of the existing RE processes, process improvement tradition, company size, product maturity, product type, and application domain. Here we describe factors that are common to all the four case organizations and discuss their possible effects on the results. We are not, however, able to state exactly how these factors have affected the results, because we did not observe them systematically during the case studies.

All the case organizations had defined a product development process and they had had earlier experiences on process improvement. Furthermore, the case organizations were the product development organizations of medium-sized or large companies, and they were able to invest in systematic RE process improvement. These factors can explain the willingness of the case organizations to apply the process improvement approach and the REGPG. In other words, the results related to the REGPG might be valid only for medium-sized and large organizations that are familiar with process improvement. Furthermore, it is possible that our process improvement guidelines shown in Appendix E are too extensive for small organizations that are able to invest a very limited amount of resources in RE process improvement.

At the beginning of the study, the maturity of the RE processes was relatively low in all the case organizations. This factor might explain why the case organizations were seeking simple RE processes and practices.

All the case organizations had developed products for a large market of customers for many years. In addition, all of them had several experts that understood the application domain well. These factors can explain why cultural change towards systematic user requirements definition took place slowly. In other words, organizations that are developing a totally new product or customer-specific systems may not face the challenge of cultural change.

Underlying assumptions of the study

This study was based on three assumptions. These assumptions and their possible effects on the research results are discussed here.

The first assumption was that the quality of the system depends on the quality of the development process. Therefore, a process improvement approach was applied to introduce RE into product development. Based on this underlying assumption, we also selected the REGPG and defined the process improvement procedure to guide the systematic RE process improvement of the case organizations.

Having analyzed the use of the systematic process improvement approach, and the process improvement procedure afterwards, we can draw attention to two points. First, the process improvement procedure supported systematic research work and research co-operation with the case organizations. Secondly, the systematic process improvement approach might have affected the duration of change. In other words, it might have made the process change take longer than if a small set of RE practices would have been introduced into product development without assessing existing RE processes and piloting these RE practices.

Having assessed the selection of the REGPG afterwards, we can emphasize two further points. First, the REGPG introduced a large set of RE practices that the practitioners were interested in. On the other hand, introducing these practices might have made the practitioners seek for optimal solutions and consequently the process change might have taken longer than if just a small set of RE practices had been recommended to the case organizations.

The second assumption was that the quality of the requirements definition process affects the quality of the requirements management. We were not able to observe RE practices related to requirements change management in real product development projects, and none of the case organizations were such that we could compare the number of requirements changes before and after RE process improvements. For these two reasons, the question of whether better requirements definition decreases the number of requirements changes still remains open.

The third assumption was that RE methods and practices, although they originate from software engineering, can be used for systems that also include hardware. One of the case organizations developed pure software systems and three of them developed software-intensive systems. During the study, we could not identify any RE practice that did not suit systems that included hardware. In other words, our research results indicate that many RE methods and practices developed in the context of software engineering can be applied for software-intensive systems as well.

Furthermore, we could not identify any challenges that are specific either to the user requirements definition of pure software systems or to software-intensive systems that contain hardware also. However, we cannot draw any conclusion as to whether there are special RE challenges related, for example, to pure software systems, because the number of case organizations was small and our study covered only a small set of RE practices. For example, we were not able to observe RE practices related to project management. Therefore, we cannot either confirm or disprove the statement of Faulk (1997), according to which the arbitrary and invisible nature of software makes it difficult to anticipate which requirements will be met easily and which will decimate the project’s budget and schedule if, indeed, they can be met at all.

5 Conclusions

This chapter first summarizes the main contributions of this thesis, and then, further research topics are discussed.

5.1 Main contributions and implications

One of the main contributions of this work is that it provides new knowledge about RE process improvement. This study supplements the relatively few case studies related to the enhancement of the market-driven RE processes (e.g. Yeh 1992; Hutchings and Knox 1995; Regnell et al. 1998; Jacobs 1999; Carlshamre and Regnell 2000; Damian et al. 2002, 2003, 2004; Daneva 2002, 2003, 2004; Weber and Weisbrod 2003). To our knowledge, this thesis is one of the most comprehensive empirical studies in RE process improvement in terms of duration and the number of case organizations.

First of all, our results highlight the importance of cultural change. The cultural change means that requirements are defined also from the customers' and users' points of view and not just from the technical point of view. User requirements definition and issues related to it have received surprisingly little attention in RE research so far. This thesis provides novel knowledge about how to define user requirements systematically in practice and what kind of challenges can relate to the change from a technical perspective to the users' perspective.

Based on the improvement experiences gathered from the case organizations, we identified a set of RE practices that support systematic user requirements definition. We would especially recommend three practices: 1) discover user needs actively, 2) document user requirements separately from technical requirements, and 3) represent user requirements using use cases. These practices might suit organizations that are able to invest only a limited amount of resources and money for RE process improvement.

Our four cases show that cultural change towards systematic user requirements definition can be time-consuming and requires more than the introduction of a new RE process and new RE technology. The study revealed that the most demanding obstacles to the cultural change are beliefs that both product development managers and engineers tend to hold. To overcome these beliefs is especially crucial because RE process change is bottom-up. In other words, the change of behaviour needs to take place first at the individual level, and it can then spread to project teams, and finally to the entire organization.

Another main contribution of the thesis is that it provides further insights into technology transfer of RE. It proposes a conceptual model of the factors affecting organization-wide implementation of RE processes and practices. The model classifies the critical factors into categories and thus provides a structured overview of them. In addition, it shows a set of relationships between the categories, which sheds light on the complex phenomenon of the organization-wide implementation of RE practices. And, finally, the model highlights the significance of human factors, and provides explanations why people may resist a process change and why the introduction of RE technology can be difficult.

From the perspective of practice, we have transformed the model of the success factors into a set of practical guidelines for RE process improvement and implementation. These guidelines can serve as a checklist when planning to introduce requirements engineering into product development. We do not suggest that these guidelines are complete, or that they will solve all the problems associated with the organization-wide adoption of RE practices. Careful consideration of them can, however, enhance the chances of success in implementing a new RE process throughout an organization.

One of the contributions of this thesis is that it offers empirical evidence on the strengths and weaknesses of the REGPG. Although the REGPG was published in 1997, there is still little empirical data on how it supports organizations in RE process improvement. Sommerville and Ransom (2005) report the results of the first systematic evaluation of the REGPG across a range of companies. Our study supplements the Sommerville and Ransom’s empirical study. Our results show that performing an assessment with the REGPG raises practitioners’ awareness of RE, and motivates them to improve their RE processes. Furthermore, the study indicates that the REGPG offers relevant RE practices for organizations that want to introduce requirements engineering into their product development.

However, the application of the REGPG is not straightforward. Our research revealed that it is difficult to select a realistic set of practices from over 60 practices and that practitioners tend to select too many of them to be implemented at the same time. From this, we conclude that the REGPG should be developed further to support practitioners in identifying RE practices essential to their organizations. Furthermore, our research results indicate that practitioners do not find some of the practices included in the REGPG useful. Therefore, it would be beneficial to collect empirical data on the usage of the sixty-six practices and remove rarely used ones. This would make the REGPG simpler to apply.

This study contributes mainly to the discipline of requirements engineering. The results can also be looked at from a wider software engineering point of view. We applied the software process improvement approach developed in the field of software engineering. From the perspective of software process research, the study makes two contributions. First, it provides evidence that most of the success factors of software process improvement are equally essential and applicable to RE process improvement.

Secondly, we suggest that the model of the factors affecting organization-wide implementation of RE processes is not only specific to RE process improvement but also provides new knowledge about the relationships of the success factors reported in the SPI literature. We built the first version of the model based on the success factors found in the SPI literature. Later we extended the model based on our own findings and finally validated it with the lessons learned from RE and software process improvement. From this we conclude that our model offers some new explanations of why people may resist a process change.

In software process improvement, resistance to change is a widely-recognized problem and is considered to be one of the biggest challenges. Our results suggest that a deeper understanding of such human factors as values, beliefs, attitudes and motivation is essential when one needs to cope with resistance to process changes. Therefore, we believe that the research results from other disciplines such as social psychology and sociology offer valuable knowledge for both software process and RE process improvement.

The final conclusion to be drawn from this work is that the introduction of the RE process into product development is a demanding and slow task. Shifting the focus from technical requirements to user requirements is likely to be especially challenging for product development personnel. Furthermore, there are human, organizational, technological, and economic factors that affect the success of the organization-wide implementation of RE. Organizations can, however, gain benefits from requirements engineering by defining a simple RE process, by focusing on a small set of useful RE practices, and by supporting the systematic usage of these practices. To support the organization-wide adoption of the RE practices, organizations can invest in Just-in-Time training and offer an RE expert to help product development teams when they are defining user requirements for the first times.

From the research perspective, this study suggests that practitioners are willing to apply RE methods and practices if they can see what benefits they can gain from these methods and practices. Furthermore, practitioners seem to search for simple and practical solutions for their RE problems. In other words, they are searching for RE methods and practices that are easy to learn, easy to use and that support them in doing their work better. Thus, we RE researchers should evaluate RE technology in real product development contexts for a sufficiently long period of time, provide more empirical evidence on its concrete benefits and report possible problems related to its application.

5.2 Future challenges

The results of the study point to several challenges for future research. First, one of our future research goals is to acquire further confidence in our results by validating them with new case organizations. For example, we would find it interesting to extend the model of the factors affecting organization-wide implementation of RE processes by gaining a deeper understanding of the interrelationships of the factors involved. Furthermore, it would be interesting to investigate, especially from the practice perspective, what can be done if a success factor is difficult to achieve in a case organization.

An important direction for future research is to gain further insights into human factors. Social scientists have studied changes in organizations from a people perspective, so valuable lessons can be learnt from other disciplines such as social psychology. Widening the scope of the RE process improvement research towards organizational change is also important. Management science has a long tradition of investigating issues related to organizational development and business process development. We believe that the research results from that discipline offer valuable knowledge for software process and RE process improvement.

Furthermore, it would be interesting to analyze the role of the RE processes. Processes are often seen as a way to control chaotic projects and make them to more predictable and effective (e.g. Paulk et al. 1997; Zahran 1998). On the other hand, Conradi and Fuggetta (2002) state that appropriate perspectives should be applied to different parts of the software process – for example, discipline and rigor to inspections and configuration management, and creativity and collaboration to requirements engineering and high-level software design. We find this statement very attractive and so one of our future research challenges is to investigate what kind of RE process supports creativity and how it might enhance communication and co-operation between different stakeholders during RE.

In this study, we identified a small set of practices that support the systematic definition of user requirements. We are interested in continuing research into RE practices as well. One of the research topics is to investigate how user needs and user requirements might be connected more closely to business information. Furthermore, we will continue to analyze how practitioners can create traceability between user requirements and technical requirements.

The REGPG was one of the cornerstones of this study. Our results indicate that it provides valuable support for organizations that want to apply a process improvement approach when introducing RE into product development. One of the future challenges is to gather more empirical evidence as to the usefulness of the REGPG and especially its practices. From the perspective of practitioners, it would be attractive to investigate which of the RE practices are specific to certain contexts and whether it is possible to identify a small set of beneficial RE practices that can be recommended to most organizations. Furthermore, it would be interesting to study how easily the REGPG can be applied in organizations that are not interested in applying the systematic process improvement approach.

In this study, we were not able to measure the business benefits of RE process improvements. Other research reports also imply that it can be difficult to show how RE process improvements contribute to a project's or product's success (e.g. Davis and Hsia 1994; Jacobs 1999; Sommerville and Ransom 2005). Thus, one of the challenging research topics for the RE research community is to find effective and reliable ways to assess the business benefits of RE technology.

This study focused on requirements definition for a single product and product development project. However, all the case organizations involved in the study develop product families as well, but do not have systematic practices for handling requirements for them. Therefore, one of our long-term research challenges is to widen the research scope to cover requirements definition for product families.

A long-term research area that concerns the whole RE research community is the continuation of the investigation of why it is difficult to transfer RE technology into practice. On the one hand, our results indicate that organizations are seeking for simple solutions that are easy to learn and use, while on the other, the analysis of the existing RE literature revealed that academic research often concentrates on advanced technology. This difference suggests that organizations have different kinds of needs for RE technology. In order to support the successful transfer of RE technology, we, as RE researchers, should better understand how requirements are currently defined and managed in different kinds of organizations and what kind of RE technology practitioners need and are ready to apply. Based on the analysis of needs, we could classify organizations into different kinds of target groups. After that, it might be easier to market both simple RE practices and more advanced RE methods and techniques to their right target groups.

6 References

ABBOTT, R. J. (1986) An Integrated Approach to Software Development. New York: John Wiley.

ARMOUR, P. G. (2001) The Laws of Software Process - Getting from Here to There: Putting a Boundary around Process. Communications of the ACM 44 (1), 15-17.

AVISON, D., LAU, F., MYERS, M., and NIELSEN, P. A. (1999) Action Research. Communications of the ACM 42(1), 94-97.

BASILI, V.R. (1992) Software Modeling and Measurement: The Goal/Question/Metric Paradigm. Technical Report CS-TR-2956, Department of Computer Science, University of Maryland, College Park, MD 20742.

BASILI, V.R. (1996) The Role of Experimentation in Software Engineering: Past, Current, and Future. Proceedings of the 18th International Conference on Software Engineering (ICSE'96), IEEE Computer Society, pp. 442-449.

BASILI, V. and GREEN, S. (1994) Software Process Evolution at the SEL, IEEE Software 11(4), 58-66.

BASILI, V., ZELKOWITZ, M., MCGARRY, F., PAGE, J., WALIGORA, S., and PAJERSKI, R. (1995) SEL's Software Process – Improvement Program. IEEE Software 12(6), 83-87.

BASILI, V. R., McGARRY, F.E., PAJERSKI, R., and ZELKOWITZ, M. V. (2002) Lessons Learned from 25 Years of Process Improvement: The Rise and Fall of the NASA Software Engineering Laboratory. Proceedings of the 24th International Conference on Software Engineering (ICSE'02), ACM Press, pp. 69 –79.

BASKERVILLE, R. and MYERS, M. D. (2004) Special Issue on Action Research in Information Systems: Making IS Research Relevant to Practice – Foreword. MIS Quarterly 28(3), 329-335.

BASKERVILLE, R. L. and WOOD-HARPER, A. T. (1996) A Critical Perspective on Action Research as a Method for Information Systems Research. Journal of Information Technology 11(3), 235-246.

BECK, K. (1999) Embracing Change with Extreme Programming. IEEE Computer 32(10), 70-77.

BECK, K (2000) Extreme Programming Explained: Embrace Change. Upper Saddle River, NJ: Addison-Wesley.

BELL, T. E. and THAYER, T. A. (1976) Software Requirements: Are They Really a Problem? Proceedings of the 2nd International Conference on Software Engineering (ICSE-2), IEEE Computer Society Press, pp. 61-68.

BERANDER, P. and WOHLIN, C. (2003) Identification of Key Factors in Software Process Management – A Case Study. Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03), IEEE Computer Society, pp. 316-325.

BLACKBURN, J. D., SCUDDER, G. D., and VAN WASSENHOVE, L. (1996) Improving Speed and Productivity of Software Development: A Global Survey of Software Developers. IEEE Transactions on Software Engineering 22(12), 875-885.

BLACKBURN, J. D., SCUDDER, G. D., and VAN WASSENHOVE, L. (2000) Concurrent Software Development. Communications of the ACM 43(11), 200-214.

BRACKETT, J. W. (1990) Software Requirements. SEI Curriculum Module SEI-CM-19-1.2, Carnegie Mellon University, Software Engineering Institute.

BROOKS, F. (1987) No Silver Bullet: Essence and Accidents of Software Engineering. IEEE Computer 20(4), 10-19.

BUBENKO, J. A. (1995) Challenges in Requirements Engineering. Proceedings of the Second IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, pp. 160-162.

CALVO-MANZANO, J. A., CUEVAS, G., SAN FELIU, T., DE AMESCUA, A., GARCÍA, L., and PÉREZ, M. (2002) Experiences in the Application of Software Process Improvement in SMES. Software Quality Journal 10(3), 261-273.

CARLSHAMRE, P. and RANTZER, M. (2001) Dissemination of Usability: Failure of a Success Story. Interactions 8(1), 31-41.

CARLSHAMRE, P. and REGNELL, B. (2000) Requirements Lifecycle Management and Release Planning in Market-Driven Requirements Engineering Processes. Proceedings of the 11th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, pp. 961-965.

CHRISTEL, M. G. and KANG, K. C. (1992) Issues in Requirements Elicitation. Technical Report, CMU/SEI-92-TR-12, ESC-TR-92-012, Carnegie Mellon University, Software Engineering Institute.

CLAUS, C., FREUND, M., KAISER, M., and KNEUPER, R. (1999) Implementing Systematic Requirements Management in a Large Software Development Programme. Proceedings of the Fifth International Workshop on Requirements Engineering: Foundations of Software Quality (REFSQ'99), Presses Universitaires de Namur, pp. 33-42.

COBLE, J. M., KARAT, J., and KAHN, M. G. (1997) Maintaining a Focus on User Requirements throughout the Development of Clinical Workstation Software. Proceedings of the CHI'97 Conference on Human Factors in Computing Systems, ACM Press, pp. 170-177.

CONRADI, R. and FUGGETTA, A. (2002) Improving Software Process Improvement. IEEE Software 19(4), 92-99.

CURTIS, B. (1997) Software Process Improvement: Methods and Lessons Learned. Proceedings of the 19th International Conference on Software Engineering, (ICSE 1997), pp. 624-625.

CURTIS, B., KELLNER, M., and OVER, J. (1992) Process Modeling. Communications of the ACM 35(9), 75-90.

DAMIAN, D., ZOWGHI, D., VAIDYANATHASAMY, L., and PAL, Y. (2002) An Industrial Experience in Process Improvement: An Early Assessment at the Australian Center for Unisys Software. Proceedings of the 2002 International Symposium on Empirical Software Engineering (ISESE'02), IEEE Computer Society, pp. 111-123.

DAMIAN, D., CHISAN, J., VAIDYANATHASAMY, L., and PAL, Y. (2003) An Industrial Case Study of the Impact of Requirements Engineering on Downstream Development. Proceedings of the 2003 International Symposium on Empirical Software Engineering (ISESE'03), IEEE Computer Society, pp. 40-49.

DAMIAN, D., ZOWGHI, D., VAIDYANATHASAMY, L., and PAL, Y. (2004) An Industrial Case Study of Immediate Benefits of Requirements Engineering Process Improvement at the Australian Center for Unisys Software. Empirical Software Engineering, 9(1), 45-75.

DANEVA, M. (2002) Using Maturity Assessments to Understand the ERP Requirements Engineering Process. Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE'02), IEEE Computer Society, pp 255-262.

DANEVA, M. (2003) Lessons Learnt from Five Years of Experience in ERP Requirements Engineering. Proceedings of the 11th IEEE International Conference on Requirements Engineering, IEEE Computer Society, pp. 45-54.

DANEVA, M. (2004) ERP Requirements Engineering Practice: Lessons Learned. IEEE Software 21(2), 26-33.

DAVIS, A. M. (1993) Software Requirements: Objects, Functions, and States. Upper Saddle River, NJ: Prentice Hall.

DAVIS, A. M. and HICKEY, A. M. (2002) Requirements Researchers: Do We Practice What We Preach? Requirements Engineering 7(2), 107-111.

DAVIS, A. M. and HSIA, P. (1994) Giving Voice to Requirements Engineering. IEEE Software 11(2), 12-16.

DIAZ, M. and SLIGO, J. (1997) How Software Process Improvement Helped Motorola, IEEE Software 14(5), 75-81.

DORFMAN, M. (1997) Requirements Engineering, IN THAYER, R. H. and DORFMAN, M. (Eds.) Software Requirements Engineering, second edition. Los Alamitos, CA: IEEE Computer Society Press, pp.7-22.

FAULK, S. R. (1997) Software Requirements: A Tutorial. IN THAYER, R. H. and DORFMAN, M. (Eds.) Software Requirements Engineering, second edition. Los Alamitos, CA: IEEE Computer Society Press, pp. 158-179.

FAYAD, M. and LAITINEN, M. (1997) Process Assessment Considered Wasteful. Communications of the ACM 40 (11), 125-128.

FORSGREN, P. and RAHKONEN, T. (1995) Specification of Customer and User Requirements in Industrial Control System Procurement Projects. Proceedings of the 2nd IEEE International Symposium on Requirements Engineering (RE'95), IEEE Computer Society Press, pp.81-88.

FUGGETTA, A. (2000) Software Process: A Roadmap. Proceedings of the Conference on the Future of Software Engineering, ACM Press, pp.25-34.

GLASS, R. L., VESSEY I., and RAMESH, V. (2002) Research in Software Engineering: An Analysis of the Literature. Information and Software Technology 44(8), 491-506.

HALEY, T. J. (1996) Software Process Improvement at Raytheon. IEEE Software 13(6), 33-41.

HOFMANN, H. F. and LEHNER, F. (2001) Requirements Engineering as a Success Factor in Software Projects. IEEE Software 18(4), 58-66.

HSIA, P., DAVIS, A., and KUNG, D. (1993) Status Report: Requirements Engineering. IEEE Software 10(6), 75-79.

HUMPHREY, W. S. (1989) Managing the Software Process. Reading, MA: Addison-Wesley.

HUMPHREY, W. S., SNYDER, T. R., and WILLIS, R. R. (1991) Software Process Improvement at Hughes Aircraft. IEEE Software 8(4), 11-23.

HUTCHINGS, T., M., HYDE, G., MARCA, D., and COHEN, L. (1993) Process Improvement That Lasts: An Integrated Training and Consulting Method. Communications of the ACM 36(10), 105-113.

HUTCHINGS, A. F. and KNOX, S. T. (1995) Creating Products Customers Demand. Communications of the ACM 38(5), 72-80.

IEEE Std 610.12.-1990 IEEE Standard Glossary of Software Engineering Terminology.

IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

ISO/IEC TR 15504-1:1998(E) Information Technology – Software Process Assessment – Part 1: Concepts and Introductory Guide, Technical Report.

ISO/IEC TR 15504-2: 1998(E) Information Technology – Software Process Assessment – Part 2: A Reference Model for Processes and Process Capability, Technical Report.

ISO/IEC TR 15504-3: 1998(E) Information Technology – Software Process Assessment – Part 3: Performing an Assessment, Technical Report.

ISO/IEC TR 15504-4: 1998(E) Information Technology – Software Process Assessment – Part 4: Guide to Performing Assessments, Technical Report.

ISO/IEC TR 15504-7:1998(E) Information Technology – Software Process Assessment – Part 7: Guide for Use in Process Improvement, Technical Report.

JACOBS, S. (1999) Introducing Measurable Quality Requirements: A Case Study. Proceedings of the 4th IEEE International Symposium on Requirements Engineering, IEEE Computer Society, pp. 172-179.

JACOBSON, I., BOOCH, G., and RUMBAUGH, J. (1998) The Unified Software Development Process. Reading, MA: Addison-Wesley.

JAKOBSEN, A. B. (1998) Bottom-Up Process Improvement Tricks. IEEE Software 15(1), 64-68.

JAKOBSEN, A. B. (2000). Software Processes: Live and Let Die. IEEE Software 17(3), 71-75.

JOHNSON, A. (1994) Software Process Improvement Experience in the DP/MIS Function: Experience Report. Proceedings of the 16th International Conference on Software Engineering (ICSE 1994), IEEE Computer Society Press, pp. 323-329.

JURISTO, N., MORENO, A., and SILVA, A. (2002) Is the European Industry Moving toward Solving Requirements Engineering Problems? IEEE Software 19(6), 70-77.

KAINDL, H. (2000) Why Is It so Difficult to Introduce Requirements Engineering Research Results into Mainstream Requirements Engineering Practice? Proceedings of the Fourth International Conference on Requirements Engineering (ICRE'00), IEEE Computer Society, pp. 67-68.

KAINDL, H., BRINKKEMPER, S., BUBENKO, J., A., FARBEY, B., GREENSPAN, S. J., HEITMEYER, C., L., SAMPAIO DO PRADO LEITE, J., C., MEAD, N., R., MYLOPOULOS, J., and SIDDIQI, J. (2002) Requirements Engineering and Technology Transfer: Obstacles, Incentives, and Improvement Agenda. Requirements Engineering 7(3), 113-123.

KAMSTIES, E., HÖRMANN, K., and SCHLICH, M. (1998) Requirements Engineering in Small and Medium Enterprises. Requirements Engineering 3(2), 84-90.

KAUTZ, K., HANSEN, H., and THAYSEN, K. (2000) Applying and Adjusting a Software Process Improvement Model in Practice: The Use of the IDEAL Model in a Small Software Enterprise. Proceedings of the 22nd International Conference on Software Engineering, (ICSE 2000), pp. 626-633.

KEIL, M. and CARMEL, E. (1995) Customer-Developer Links in Software Development. Communications of the ACM 38(5), 33-44.

KOCK, N. (2003) Action Research: Lessons Learned from a Multi-Iteration Study of Computer-Mediated Communication in Groups. IEEE Transactions on Professional Communication 46(2), 105-128.

KOTONYA, G. and SOMMERVILLE, I. (1998) Requirements Engineering: Processes and Techniques. Chichester, England: John Wiley & Sons Ltd.

KRASNER, H., TERREL, J., LINEHAN, A., ARNOLD, P., and ETT, W. H. (1992) Lessons Learned from a Software Process Modeling System. Communications of the ACM 35(9), 91-100.

KRUCHTEN (2004) The Rational Unified Process: An Introduction, third edition. Boston: Addison-Wesley.

KUJALA, S. (2002) User Studies: A Practical Approach to User Involvement for Gathering User Needs and Requirements. [PhD Thesis], Helsinki University of Technology, Department of Computer Science and Engineering.

LEFFINGWELL, D. and WIDRIG, D. (2000) Managing Software Requirements: A Unified Approach. Boston: Addison-Wesley.

LUBARS, M., POTTS, C., and RICHTER, C. (1993) A Review of the State of the Practice in Requirements Modeling. Proceedings of the IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, pp. 2-14.

McFEELEY, B. (1996) IDEAL^SM: A User's Guide for Software Process Improvement. Handbook CMU/SEI-96-HB-001. Software Engineering Institute, Carnegie Mellon University, Pittsburgh, USA.

MILLER, S. (1997) Panel: How Can Requirements Engineering Research Become Requirements Engineering Practice. Proceedings of the 3rd IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, pp. 260.

MORRIS, P., MASERA, M., and WILIKENS, M. (1998) Requirements Engineering and Industrial Uptake. Proceedings of the 3rd International Conference on Requirements Engineering, IEEE Computer Society Press, pp. 130-137.

NEILL, C. J. and LAPLANTE (2003) Requirements Engineering: The State of the Practice. IEEE Software 20(6), 40-45.

NUSEIBEH, B. and EASTERBROOK, S. (2000) Requirements Engineering: A Roadmap. Proceedings of the Conference on the Future of Software Engineering, ACM Press, pp. 35-46.

O'HARA, F. (2000) European Experiences with Software Process Improvement. Proceedings of the 22nd International Conference on Software Engineering, (ICSE 2000), ACM Press pp. 635-640.

PATTON, M. Q. (1990) Qualitative Evaluation and Research Methods, second edition. Newbury Park, CA: Sage Publications.

PATTON, M. Q. (2002) Qualitative Research & Evaluation Methods, third edition. Thousand Oaks, CA: Sage Publications.

PAULK, M. C., WEBER, C. V., CURTIS, B., and CHRISSIS, M. (1997) The Capability Maturity Model: Guidelines for Improving the Software Process. Reading, MA: Addison-Wesley.

PFLEEGER, S. L. (1999) Understanding and Improving Technology Transfer in Software Engineering. The Journal of Systems and Software 47(2,3), 111-124.

PFLEEGER, S. L. and MENEZES W. (2000) Marketing Technology to Software Practitioners. IEEE Software 17(1), 27-33.

PINHEIRO, F. A. C., LEITE, J. C .S., and CASTRO, J. F. B. (2003) Requirements Engineering Technology Transfer: An Experience Report. Journal of Technology Transfer 28(2), 159-165.

POHL, K. (1996) Process-Centered Requirements Engineering. Taunton, England: Research Studies Press Ltd.

POTTS, C. (1993) Software-Engineering Research Revisited. IEEE Software 10(5), 19-28.

POTTS, C. (1995) Invented Requirements and Imagined Customers: Requirements Engineering for Off-the-Shelf Software. Proceedings of the Second IEEE International Symposium on Requirements Engineering (RE’95), IEEE Computer Society Press, pp. 128-130.

REGNELL, B. (1999) Requirements Engineering with Use Cases – a Basis for Software Development. [PhD Thesis], Lund University, Department of Communication Systems.

REGNELL, B., BEREMARK, B., and EKLUNDH, O. (1998) A Market-Driven Requirements Engineering Process – Results from an Industrial Process Improvement Programme. Requirements Engineering 3(2), 121-129.

ROBERTSON, S. and ROBERTSON, J. (1999) Mastering the Requirements Process. Harlow, England: Addison-Wesley.

ROMBACH, H. D. (1990) Software Specifications: A Framework. SEI Curriculum Module SEI-CM-11-2.1, Carnegie Mellon University, Software Engineering Institute.

RUMBAUGH, J. (1994) Getting Started – Using Use Cases to Capture Requirements. Journal of Object-Oriented Programming 7(5), 8-12,23.

SAKAMOTO, K., NAKAKOJI, K., TAKAGI, Y., and NIIHARA, N. (1998) Toward Computational Support for Software Process Improvement Activities. Proceedings of the 20th International Conference on Software Engineering, (ICSE 1998), IEEE Computer Society, pp. 22-31.

SALO, A. and KÄKÖLÄ, T. (1998) Requirements for Groupware-Supported Requirements Processes in New Product Development. Proceedings of the Fourth International Workshop on Requirements Engineering: Foundations of Software Quality, (REFSQ’98), Presses Universitaires de Namur, pp. 99-112.

SAWYER, P. (2004) Maturing Requirements Engineering Process Maturity Models. IN MATÉ, J. and SILVA, A. (Eds.) Requirements Engineering for Sociotechnical Systems, Idea Group Inc., in press.

SAWYER, P., SOMMERVILLE, I., and VILLER, S. (1997) Requirements Process Improvement through the Phased Introduction of Good Practice. Software Process - Improvement and Practice 3, 19-34.

SAWYER, P., SOMMERVILLE, I., and VILLER, S. (1998) Improving the Requirements Process. Proceedings of the Fourth International Workshop on Requirements Engineering: Foundations of for Software Quality, (REFSQ'98), Presses Universitaires de Namur, pp. 71-84.

SAWYER, P., SOMMERVILLE, I., and VILLER, S. (1999a) Capturing the Benefits of Requirements Engineering. IEEE Software 16(2), 78-85.

SAWYER, P., SOMMERVILLE, I., and KOTONYA, G. (1999b) Improving Market-Driven RE Processes. Proceedings of the International Conference on Product Focused Software Process Improvement (Profes´99), pp. 222-236.

SCHARER, L. (1981) Pinpointing Requirements, Datamation, Apr., 139-151. Reprinted (1997) IN THAYER, R. H. and DORFMAN, M. (Eds.) Software Requirements Engineering, second edition. Los Alamitos, CA: IEEE Computer Society Press, pp. 30-35.

SIDDIQI, J. and SHEKARAN, C. M. (1996) Requirements Engineering: The Emerging Wisdom. IEEE Software 13(2) 15-19.

SOMMERVILLE, I. (1996) Software Engineering, fifth edition. Wokingham, England: Addison-Wesley.

SOMMERVILLE, I. (2001) Software Engineering, sixth edition. Harlow, England: Addison-Wesley.

SOMMERVILLE, I. and RANSOM, J. (2005) An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. ACM Transactions on Software Engineering and Methodology 14(1), 85-117.

SOMMERVILLE, I. and SAWYER, P. (1997) Requirements Engineering: Good Practice Guide. Chichester, England: John Wiley & Sons Ltd.

STANDISH GROUP, The CHAOS Report (1994) **http://www.pm2go.com/sample_research/chaos_1994_1.php** (Accessed April 21, 2005).

STEVENS, R., BROOK, P., JACKSON, K., and ARNOLD, S. (1998) Systems Engineering: Coping with Complexity. London: Prentice Hall.

STRAUSS, A. and CORBIN, J. (1998) Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory, second edition. Thousand Oaks, CA: Sage Publications.

STRINGER, E. T. (1999) Action Research, second edition. Thousand Oaks, CA: Sage Publications.

TANAKA, T., SAKAMOTO, K., KUSUMOTO, S., MATSUMOTO, K., and KIKUNO, T. (1995) Improvement of Software Process by Process Description and Benefit Estimation. Proceedings of the 17th International Conference on Software Engineering, (ICSE 1995), ACM Press, pp. 123-132.

THAYER, R. H. and DORFMAN, M. (1997) Introduction to Tutorial: Software Requirements Engineering. IN THAYER, R. H. and DORFMAN, M. (Eds.) Software Requirements Engineering, second edition. Los Alamitos, CA: IEEE Computer Society Press, pp. 1-2.

THAYER, R. H. and THAYER, M. C. (1997) Software Requirements Engineering Glossary. IN THAYER, R. H. and DORFMAN, M. (Eds.) Software Requirements Engineering, second edition. Los Alamitos, CA: IEEE Computer Society Press, pp. 489-528.

TURNER, R. (2003) Seven Pitfalls to Avoid in the Hunt for Best Practices. IEEE Software 20(1), 67-69.

ULRICH, K. T. and EPPINGER, S. D. (2000) Product Design and Development, second edition. Boston: McGraw-Hill.

VAN LAMSWEERDE, A. (2000) Requirements Engineering in the Year 00: A Research Perspective. Proceedings of the 22nd International Conference on Software Engineering, (ICSE 2000), ACM Press, pp. 5-19.

VERNER, J. M., OVERMYER, S. P., and McCAIN, K. W. (1999) In the 25 Years since The Mythical Man-Month What Have We Learned about Project Management? Information and Software Technology 41(14), 1021-1026.

VOTTA, L. G. and PORTER A. (1995) Experimental Software Engineering: A Report on the State of the Art. Proceedings of the 17th International Conference on Software Engineering, (ICSE 1995), ACM Press pp. 277-279.

WARD, R. P., FAYAD, M. E., and LAITINEN, M. (2001) Thinking Objectively: Software Process Improvement in the Small. Communications of the ACM 44 (4), 105-107.

WEBER, M. and WEISBROD, J. (2003) Requirements Engineering in Automotive Development: Experiences and Challenges. IEEE Software 20(1), 16-24.

WEBSTER, J. and WATSON, R. T. (2002) Analyzing the Past to Prepare for the Future: Writing a Literature Review. MIS Quarterly 26(2), xiii-xxiii.

WEIDENHAUPT, K. POHL, K., JARKE, M. and HAUMER, P. (1998) Scenarios in System Development: Current Practice. IEEE Software 15(2), 34-45.

WIEGERS, K. (1999a) Software Requirements. Redmond, WA: Microsoft Press.

WIEGERS, K. (1999b) Software Process Improvement in Web Time. IEEE Software, 16(4), 78-86.

WIEGERS, K. (2003) Software Requirements, second edition. Redmond, WA: Microsoft Press.

WOHLIN, C., RUNESON, P., HÖST, M., OHLSSON, M. C., REGNELL, B., and WESSLÉN, A. (2000) Experimentation in Software Engineering – An Introduction. Norwell, MA: Kluwer Academic Publishers.

YEH, A. (1992) Requirements Engineering Support Technique (REQUEST): A Market Driven Requirements Management Process. Proceedings of the Second Symposium of Quality Software Development Tools, IEEE Computer Society Press, pp. 211-223.

YIN, R. K. (1994) Case Study Research – Design and Methods, second edition. Thousand Oaks, CA: Sage Publications.

ZAHRAN, S. (1998) Software Process Improvement: Practical Guidelines for Business Success. Harlow, England: Addison-Wesley.

ZAVE, P. (1997) Classification of Research Efforts in Requirements Engineering. ACM Computing Surveys 29(4), 315-321.

Appendix A. Glossary

Constraint (rajoite): In the context of a requirement, a restriction on the acceptable solution opportunities (a type of requirement itself) (Stevens et al., 1998).

Customer (asiakas): The person, or persons, who pay for the product and usually (but not necessarily) decide the requirements (IEEE Std 830:1998).

Functional requirement (toiminnallinen vaatimus): A requirement that specifies a function that a system or system component must be able to perform (IEEE Std 610.12-1990).

Non-functional requirement (ei-toiminnallinen vaatimus): A requirement that describes the required overall attribute of the system, including portability, reliability, efficiency, human engineering, testability, understandability, and modifiability (Davis, 1993).

Product (tuote): A product is something sold by an enterprise to its customers (Ulrich and Eppinger, 2000).

Product development (tuotekehitys): Product development is the set of activities beginning with the perception of a market opportunity and ending in the production, sale, and delivery of a product (Ulrich and Eppinger, 2000).

Property (ominaisuus, ei-toiminnallinen vaatimus): A requirement that describes the required overall attribute of the system, including reliability, performance, usability, availability, security. (See also non-functional requirement)

Requirement (vaatimus): A user need or a necessary feature, function, or attribute of a system that can be sensed from a position external to that system (Davis, 1993).

Requirements engineering (vaatimusten märittely ja hallinta):

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behaviour, and to their evolution over time and across software families. (Zave, 1997)

Requirements engineering covers all of the activities involved in discovering, documenting, and maintaining a set of requirements for a system. The term engineering implies that systematic and repeatable techniques should be used to ensure that system requirements are complete, consistent, relevant etc. (Sommmerville and Sawyer, 1997)

Stakeholder (sidosryhmä, asianosainen): Stakeholders are people or organizations who will be affected by the system and who have a direct or indirect influence on the system requirements (Kotonya and Sommerville, 1998).

Software engineering (ohjelmistotuotanto): Software engineering is the application of scientific principles to (1) the orderly transformation of a problem into a working software solution and (2) the subsequent maintenance of that software until the end of its useful life (Davis, 1993).

User (käyttäjä)**:** The person, or persons, who operate or interact directly with the product. The user(s) and the customer(s) are often not the same person(s) (IEEE Std 830:1998).

User need (käyttäjätarve): User needs refer to problems that hinder users in achieving their goals, or opportunities to improve the likelihood of users' achieving their goals (Kujala, 2002).

User requirement (käyttäjävaatimus): A user requirement is an externally visible function, property or constraint that the system must provide to fill the needs of the system's intended users.

Appendix B. Summary of definitions for different levels of requirements

Rombach (1990)	Stevens et al. (1998)	Wiegers (1999a, 2003)	Leffingwell and Widrig (2000)	Sommerville (2001)
	Business requirements Those requirements introduced for the benefit of the business that is developing a system.	**Business requirements** represent high-level objectives of the organization or customer requesting the system or product.		
Software needs are concerned with the questions: What demand exists? What needs should a proposed software product attempt to fulfil?			**Stakeholder need** A reflection of the business, personal, or operational problem or opportunity that must be addressed in order to justify consideration, purchase, or use of a new system.	
C-requirements are customer/user-oriented software requirements, which are concerned with the question: What functional and non-functional characteristics, from a customer's or user's point of view, must a product exhibit to meet those needs?	**User requirements** An expression of the needs of all stakeholders in the utilization domain. User requirements should be expressed in the terminology of the problem domain, defining what the users want to do with the system.	**User requirements** describe tasks the users must be able to accomplish with the product.	**Feature** A service the system provides to fulfil one or more stakeholder needs. With this definition, user's features can't be too far removed their needs. Features are simple descriptions, in the user's language, that are used as labels to communicate with the user how the system addresses the problem.	**User requirements** are statements, in a natural language plus diagrams, of what services the system is expected to provide and the constraints under which it must operate. They should only specify the external behaviour of the system, and should avoid system design characteristics.
D-requirements are developer-oriented software requirements are predominantly concerned with the question: What functional and non-functional characteristics, from a software developer's point of view, must a product exhibit to meet those needs?	**System requirements** An intermediate step between user requirements and system design; an internally consistent definition of what the system will have to do. They consist of a system architecture, functionality, control structure, material and information flows and dynamic behaviour of the system.	**Functional requirements** define the software functionality the developers must build into the product to enable users accomplish their tasks, thereby satisfying the business requirements. They specify system functions, behaviour, constraints, and interfaces. Non-functional requirements are an essential component of the system requirements.	**Software requirements** are those things that the software does behalf of the user or device or another system. A complete set of software requirements contains: ▪ Inputs of the system ▪ Outputs of the system ▪ Functions of the system ▪ Attributes of the system and system environment.	**System requirements** set out the system services and constraints in detail. They are used by software engineers as the starting point for the system design.

Appendix C. Summary of important factors for process improvement found in the RE literature

Important factors for RE process improvement	**Hutchings and Knox (1995)**	**Sommerville and Sawyer (1997)**	**Salo and Käkölä (1998)**	**Claus et al. (1999)**	**Jacobs (1999)**	**Wiegers (1999a, 2003)**	**Calvo-Manzano et al. (2002)**	**Damian et al. (2002, 2004)**	**Berander and Wohlin (2003)**	**Daneva (2003)**
	C	B	C	C	C	B	C	C	C	C
User involvement	(x)	x		x	x			x	x	
Benefits of RE processes	x	x		x	(x)		x	x		
Continuous improvement		x		x		x	x	(x)	x	
Pilot projects	x	x		x			x			
Training and education				x	x		x	x		
Management commitment and support				x			x	x	x	
Resistance to change	x			x				x		x
Simplicity of RE processes	x		x	x						
Cultural change	x			x	x					
Evolutionary improvement		x			x	x				
People-oriented improvement		x			x					
Problem-oriented improvement				x		x				
Time scale of the improvement		x		x						
Infrastructure for process improvement				(x)			x	(x)		
Measurement		(x)			(x)		x			
Change agent / evangelist		x						(x)		

B: Book

C: Case study

x: The reference classifies the factor as 1) an important factor for RE process improvement, 2) a critical problem, 3) a lesson learned or 4) a recommendation.

(x): The reference provides information related to the factor, but it does not explicitly point out the importance of the factor

Appendix D. Summary of the critical factors found in the SPI literature

Critical factors for software process improvement	**Humphrey 1989**	**Humphrey et al. 1991**	**Basili and Green 1994**	**Johnson 1994**	**Basili et al. 1995**	**Tanaka et al. 1995**	**Haley 1996**	**McFeeley 1996**	**Curtis 1997**	**Diaz and Sligo 1997**	**Paulk et al. 1997**	**Jakobsen 1998**	**Sakamoto et al. 1998**	**Zahran 1998**	**Wiegers 1999b**	**Jakobsen 2000**	**Kautz et al. 2000**	**O'Hara 2000**	**Basili et al. 2002**	**Conradi et al. 2002**
	B	C	C	C	C	C	C	TR	S	C	B	C	C	B	C	C	C	C	C	S
User involvement (bottom-up change)				x	x	x	x	x	x	x	x			x	x	x	x	x		x
Measurement / Monitoring	x		x	x	x	x	x	x	(x)				x	x			x	x	x	x
Management commitment and support		x		x			x	x	x	x	x			x	x		x	x	x	x
SPI goals and objectives						x	x	x		x	x		x	x				x		x
Resistance to change	x							x	x	x		x	x	x			x			x
Training and education	x			x			x				x		x	x	x		x	x		
Motivation, enthusiasm and pride		x				x		x				x		x	x	x		x		
Continuous process improvement	(x)			x	(x)		x				x			x					x	x
Evolutionary process improvement	x		x								x			x	x					x
Infrastructure for process improvement		x					x	x			x			x				x		
Change agent/process improvement expertise		x						x						x	x		x			(x)
Process improvement managed as a project									x				x	x			x	x		
Structured improvement approach	x		x								x									
Experimental improvement approach	x							x											x	
People-oriented improvement approach		x			x												x			

B: Book

C: Case study

S: Summary of experiences

TR: Technical report

x: The reference classifies the factor as 1) an important factor for software process improvement, 2) a critical problem, 3) a lesson learned or 4) a recommendation.

(x): The reference provides information related to the factor, but it does not explicitly point out the importance of the factor.

Appendix E. Summary of guidelines for RE process improvement

Success factor	Guideline
Human factors	Make all process users and managers aware of the RE process.
	Ensure management commitment.
	Cope with people's resistance against change.
	Support motivation, enthusiasm and pride of personnel.
	Prepare to make a cultural change.
	Support people in overcoming beliefs related to defining requirements from the customers' and users' points of view.
	Respect skills of personnel involved in requirements engineering.
Usefulness of the RE process	Provide benefits for all process users: ▪ Provide concrete and short-term benefits for all process users. ▪ Provide managers with measurable data on the benefits of the RE process.
	Use the experience of the pilot projects to show the usefulness of the RE process.
	Use the experience of the other companies to show the usefulness of the RE process.
Practicality of the RE process	Define a simple RE process.
	Define a flexible RE process.
	Integrate the RE process with the product development process.
Training	Train all process users: ▪ Give managers, product development teams and other process users an overview of the RE process through training. ▪ Use "Just-in-Time" training to get product development teams to apply the RE process in practice.
Support	Ensure support from all management levels.
	Offer simple templates and practical guidelines to process users.
	Offer an RE expert to help product development projects.
Implementation strategy	Use an evolutionary improvement strategy.
	Improve the process continuously based on feedback.
	Use a people-oriented strategy for RE process implementation.
Improvement activities	Involve process users in improvement work.
	Set goals for process improvement.
	Align process improvement goals with business objectives
	Test the RE process in pilot projects.
	Measure the impact of the improvement efforts.

ISBN 951-22-7922-3
ISBN 951-22-7923-1 (PDF)
ISSN 1795-2239
ISSN 1795-4584 (PDF)