Lea Hannola

CHALLENGES AND MEANS FOR THE FRONT END ACTIVITIES OF SOFTWARE DEVELOPMENT

Thesis for the degree of Doctor of Science (Technology) to be presented with due permission for public examination and criticism in the Auditorium of the Student Union House at Lappeenranta University of Technology, Lappeenranta, Finland, on the 16th of December, at noon.

Acta Universitatis Lappeenrantaensis
368
<table>
<thead>
<tr>
<th>Role</th>
<th>Name</th>
<th>Institution</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervisor</td>
<td>Professor Markku Tuominen</td>
<td>Department of Industrial Management</td>
<td>Finland</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Faculty of Technology Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lappeenranta University of Technology</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tampere University of Technology</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Faculty of Business and Technology Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Department of Business Information Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>and Logistics</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Finland</td>
<td></td>
</tr>
<tr>
<td>Reviewers</td>
<td>Professor Harri Haapasalo</td>
<td>Department of Industrial Engineering and Management</td>
<td>Finland</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Faculty of Technology</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>University of Oulu</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Professor Samuli Pekkola</td>
<td>Department of Business Information Management</td>
<td>Finland</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and Logistics</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Faculty of Business and Technology Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tampere University of Technology</td>
<td></td>
</tr>
<tr>
<td>Opponent</td>
<td>Professor Samuli Pekkola</td>
<td>Department of Business Information Management</td>
<td>Finland</td>
</tr>
<tr>
<td></td>
<td></td>
<td>and Logistics</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Faculty of Business and Technology Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Tampere University of Technology</td>
<td></td>
</tr>
</tbody>
</table>

ISSN 1456-4491
Lappeenranta University of Technology
Digipaino 2009
ABSTRACT

Lea Hannola

Challenges and Means for the Front End Activities of Software Development

Lappeenranta 2009

70 p.

Acta Universitatis Lappeenrantaensis 368
Diss. Lappeenranta University of Technology

The front end of innovation is regarded as one of the most important steps in building new software products or services, and the most significant benefits in software development can be achieved through improvements in the front end activities. Problems in the front end phase have an impact on customer dissatisfaction with delivered software, and on the effectiveness of the entire software development process. When these processes are improved, the likelihood of delivering high quality software and business success increases.

This thesis highlights the challenges and problems related to the early phases of software development, and provides new methods and tools for improving performance in the front end activities of software development. The theoretical framework of this study comprises two fields of research. The first section belongs to the field of innovation management, and especially to the management of the early phases of the innovation process, i.e. the front end of innovation. The second section of the framework is closely linked to the processes of software engineering, especially to the early phases of the software development process, i.e. the practice of requirements engineering. Thus, this study extends the theoretical knowledge and discloses the differences and similarities in these two fields of research. In addition, this study opens up a new strand for academic discussion by connecting these research directions.

Several qualitative business research methodologies have been utilized in the individual publications to solve the research questions. The theoretical and managerial contribution of the study can be divided into three areas: 1) processes and concepts, 2) challenges and development needs, and 3) means and methods for the front end activities of software development. First, the study discloses the difference and similarities between the concepts of the front end of innovation and requirements engineering, and proposes a new framework for managing the front end of the software innovation process, bringing business and innovation perspectives into software development. Furthermore, the study discloses managerial perceptions of the similarities and differences in the concept of the front end of innovation between the software industry and the traditional industrial sector. Second, the study highlights the challenges and development needs in the front end phase of software development, especially challenges in communication, such as linguistic problems, ineffective communication channels, a communication gap between users/customers and software developers, and participation of multiple persons in software development. Third, the study proposes new group methods for improving the front end activities of software development, especially customer need assessment, and the elicitation of software requirements.

Keywords: front end of innovation, requirements engineering, process improvement, software development, new product development, innovation management

UDC 004.414 : 001.895
ACKNOWLEDGEMENTS

First of all, I would like to express my gratitude to my supervisor Professor Markku Tuominen for the opportunity and circumstances to carry out the thesis work and postgraduate studies under his supportive guidance.

This study has been completed in cooperation with research colleagues and cooperating business partners. Therefore, I would like to thank the case companies and all the interviewees for their invaluable contribution for this study. Further, I express my warmest thanks to my co-authors Kalle Elfvengren, Samuli Kortelainen, Professor Hannu Kärkkäinen, Uolevi Nikula, Kari Leino, Petri Ononen and Professor Heikki Kälviäinen. I also thank all my colleagues in the Department of Industrial Management, and the members of the Management of Technology (MOT) research group.

Professors Harri Haapasalo and Samuli Pekkola were the pre-examiners of my thesis. I am very grateful for their constructive comments and suggestions on how to improve the manuscript.

I also wish to thank Sinikka Talonpoika for her professional help not just in improving my skills in English, but also in academic writing.

I gratefully acknowledge the financial support I have received from the Finnish Doctoral Program of Industrial Engineering and Management (Tuotantotalouden valtakunnallinen tohtorihelma), the Research Foundation of Lappeenranta University of Technology, Lauri and Lahja Hotinen Fund, and the Finnish Funding Agency for Technology and Innovation (TEKES).

Finally, I would like to express my deepest gratitude to my family and friends. Special thanks are due to my mother Marjatta and father Jorma, who have provided me with enthusiasm for seizing new opportunities. Furthermore, I would like to thank my husband Ami for sharing the joys and challenges during this research journey, and for tolerating my non-stop absent-mindedness. In addition, my dear sons Henri and Roope have encouraged me in a big way by providing the real empirical evidence on the fuzzy front end activities of life.

Lappeenranta, December, 2009

Lea Hannula
## TABLE OF CONTENTS

### PART I: OVERVIEW OF THE THESIS

1 INTRODUCTION ........................................................................................................ 15
1.1 BACKGROUND ........................................................................................................ 15
1.2 CHARACTERISTICS OF THE SOFTWARE INDUSTRY ........................................ 16
1.3 RESEARCH OBJECTIVES AND QUESTIONS ....................................................... 18
1.4 STRUCTURE OF THE STUDY .............................................................................. 19

2 INNOVATION PROCESS .......................................................................................... 21
2.1 INNOVATION PROCESS MODELS ........................................................................ 21
2.2 THE FRONT END OF INNOVATION ....................................................................... 22
2.3 SOFTWARE DEVELOPMENT PROCESS .................................................................... 24
2.4 REQUIREMENTS ENGINEERING ........................................................................... 25

3 SOFTWARE PROCESS IMPROVEMENT ............................................................... 29
3.1 PROBLEMS OF REQUIREMENTS ELICITATION .................................................. 29
3.2 TECHNIQUES FOR REQUIREMENTS ELICITATION .......................................... 30
3.2.1 Group methods ............................................................................................... 31
3.2.2 Electronic group systems ................................................................................ 32

4 RESEARCH METHODOLOGIES ......................................................................... 34
4.1 CLASSIFICATION OF RESEARCH APPROACHES .............................................. 34
4.2 RESEARCH PROCESS ......................................................................................... 35
4.3 SELECTED RESEARCH METHODOLOGIES AND METHODS .............................. 36
4.3.1 Publication 1 – conceptual approach ............................................................... 37
4.3.2 Publication 2 – nomothetical approach ............................................................. 38
4.3.3 Publication 3 – nomothetical approach ............................................................. 38
4.3.4 Publication 4 – action-oriented approach ......................................................... 39
4.3.5 Publications 5 and 6 – constructive approach ................................................ 40

5 SUMMARY OF THE MAIN RESULTS ................................................................... 42
5.1 COMPARISON BETWEEN THE FRONT END OF INNOVATION AND REQUIREMENTS ENGINEERING .......................................................... 42
5.2 PERCEPTIONS ABOUT THE FRONT END OF INNOVATION CONCEPTS .................... 44
5.3 PROBLEMS OF REQUIREMENTS ELICITATION ............................................... 47
5.4 ASSESSING AND IMPROVING THE FRONT END ACTIVITIES ............................... 48
5.5 A GROUP METHOD FOR REQUIREMENTS ELICITATION .................................. 50
5.6 A GSS PROCESS FOR REQUIREMENTS DEFINITION ..................................... 52

6 DISCUSSION AND CONCLUSIONS ...................................................................... 55
6.1 THEORETICAL CONTRIBUTION ....................................................................... 55
6.2 PRACTICAL AND MANAGERIAL IMPLICATIONS .............................................. 57
6.3 VALIDITY OF THE STUDY .................................................................................. 58
6.4 LIMITATIONS OF THE STUDY AND FURTHER RESEARCH ............................... 61

REFERENCES

### PART II: PUBLICATIONS
PART II: PUBLICATIONS


CONTRIBUTION OF THE AUTHOR IN THE PUBLICATIONS IN PART II

Publication 1
The author made the research plan, wrote most of the paper, coordinated the writing process, and presented the paper in the International Working Seminar on Production Economics.

Publication 2
The author coordinated the writing and publication process, made the research plan and wrote the paper together with the co-authors. The data was collected by the author, whereas the data analysis was done together with the co-authors. An earlier version of the paper was presented by the author in the annual ISPIM conference.

Publication 3
Sole author of the paper. The paper was presented by the author in the International Conference on Computers and Industrial Engineering.

Publication 4
The author coordinated the writing and publication process, made the research plan and wrote the paper together with the co-authors. The data was collected by the third author, and the data analysis was done by the present author. The paper was presented by the author in the annual ISPIM conference.

Publication 5
The author made the research plan together with the co-authors, wrote most of the paper, and coordinated the writing and publication process. The author worked as a facilitator in the group sessions. An earlier version of the paper was presented by the author in the Nordic Innovation Research conference.

Publication 6
The author coordinated the writing and publication process, made the research plan and wrote the paper together with the co-authors. The second author worked as a facilitator in the group sessions. An earlier version of the paper was presented by the author in the annual ISPIM conference.
LIST OF FIGURES

Figure 1: Theoretical framework and research focus .......................................................... 16
Figure 2: Contexts for professional software development ................................................ 18
Figure 3: Structure of the thesis......................................................................................... 20
Figure 4: New product development process ..................................................................... 21
Figure 5: New concept development model for organizing the core activities in the front end of innovation ................................................................................................................... 23
Figure 6: Research/practice emphasis: software and new product development ................. 24
Figure 7: Software life cycle model ................................................................................... 25
Figure 8: Requirements within the NPD process ............................................................... 26
Figure 9: Main activities of the requirements engineering domain ...................................... 27
Figure 10: Spiral model of the RE process ........................................................................ 28
Figure 11: Categories of business research, and positioning publications 1-6 ..................... 35
Figure 12: Combined causal map from experts of software organizations........................... 45
Figure 13: Combined causal map from experts of the FEI in the traditional industrial environment...................................................................................................................... 46

LIST OF TABLES

Table 1: Comparison of software markets in 2006-2008 .................................................... 17
Table 2: Levels of requirements ....................................................................................... 26
Table 3: Research strategy: approaches and methods for solving the research questions .... 37
Table 4: The activities and main tasks of the front end of innovation................................ 43
Table 5: The activities and main tasks of requirements engineering.................................... 44
Table 6: Problems and challenges in the elicitation of software requirements ..................... 47
Table 7: Currently used RE practices and recommendations for improvements ............... 48
Table 8: Group method for requirements elicitation......................................................... 51
Table 9: GSS-supported requirements definition process .................................................. 53
Table 10: Summary of the benefits of the developed GSS process ...................................... 54
Table 11: The review process of the publications in part II of the thesis ............................. 60

ABBREVIATIONS

ACRE Acquisition of requirements
B2B Business-to-business
CAT Computer-aided team
CMMI Capability maturity model integration
CSCW Computer-supported co-operative work
CSF Critical success factors
EMS Electronic meeting systems
FEI Front end of innovation
FESI Front end of software innovation
FFE Fuzzy front end
GDSS Group decision support system
GSS Group support system
ICT Information and communication technologies
ISIPI International Society for Professional Innovation Management
IT Information technology
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>JAD</td>
<td>Joint application development</td>
</tr>
<tr>
<td>KASRET</td>
<td>Knowledge-based approach for the selection of requirements engineering techniques</td>
</tr>
<tr>
<td>LUT</td>
<td>Lappeenranta University of Technology</td>
</tr>
<tr>
<td>NCD</td>
<td>New concept development</td>
</tr>
<tr>
<td>NGT</td>
<td>Nominal group technique</td>
</tr>
<tr>
<td>NPD</td>
<td>New product development</td>
</tr>
<tr>
<td>NPPD</td>
<td>New product and process development</td>
</tr>
<tr>
<td>QFD</td>
<td>Quality function deployment</td>
</tr>
<tr>
<td>R-CMM</td>
<td>Requirements capability maturity model</td>
</tr>
<tr>
<td>RE</td>
<td>Requirements engineering</td>
</tr>
<tr>
<td>REPM</td>
<td>Requirements engineering process maturity</td>
</tr>
<tr>
<td>RETKL</td>
<td>Requirements engineering techniques knowledge library</td>
</tr>
<tr>
<td>RM</td>
<td>Requirements management</td>
</tr>
<tr>
<td>SD</td>
<td>Software development</td>
</tr>
<tr>
<td>SEI</td>
<td>Software Engineering Institute</td>
</tr>
<tr>
<td>SME</td>
<td>Small and medium-sized enterprises</td>
</tr>
<tr>
<td>SPI</td>
<td>Software process improvement</td>
</tr>
<tr>
<td>SPICE</td>
<td>Software process improvement and capability determination</td>
</tr>
<tr>
<td>TBRC</td>
<td>Technology Business Research Center</td>
</tr>
<tr>
<td>XP</td>
<td>Extreme programming</td>
</tr>
</tbody>
</table>
PART I

Overview of the Thesis
1 Introduction

1.1 Background

The innovation process can be divided into three areas: the front end of innovation (FEI), the new product and process development (NPPD), and the commercialization phases (Koen et al., 2001). Kim and Wilemon (2002) define the front end as a period between when an opportunity is first considered and when it is judged ready for development. The actual product development is the process of transforming business opportunities into tangible products (Trott, 2005), and commercialization is a set of activities associated with preparing the market into which an innovation will be launched (Tidd et al., 2005). The front end of innovation is regarded as one of the most important steps in building new products or services, offering the greatest opportunities for overall innovation process improvements (Koen et al., 2001; Kim and Wilemon, 2002). This thesis examines the challenges and means related to the front end activities of the software development process. However, the term “front end of innovation” is not commonly used in the software industry, the early processes of software development are part of a practice called requirements engineering (RE) instead. Traditionally, RE has been seen as a front end activity that forms a solid basis for the other activities of product development (Kauppinen et al., 2007).

Requirements engineering is recognized as one of the most significant sources of customer dissatisfaction with delivered software products. “Getting the requirements right” is one of the most important activities in software development, because making a crucial misstep at this phase can easily lead to large amounts of rework when the customer simply cannot accept a system the way it was developed (Dieste et al., 2008). According to Ebert (2005), poor front end processes can result in insufficient project planning, continuous changes in the requirements and project scope, delays, configuration problems, defects, and overall customer dissatisfaction. A survey of Emam and Koru (2008) reveals that 15.5 percent of software projects were cancelled in 2005, and 11.5 percent were cancelled in 2007. The survey of Emam and Koru also summarizes the reasons for project cancellation, and the two most common reasons are requirements and scope changes, and lack of senior management involvement. These are followed by budget shortages and lack of project management skills.

The theoretical framework of this study comprises two fields of research. The first part of the theoretical framework belongs to the field of innovation management, and especially to the management of the early phases of the innovation process, i.e. the front end of innovation. Koen et al. (2001) define the front end of innovation as those activities that come before the formal and well structured NPPD or Stage Gate™ process. The second part of the framework is restricted to the processes of software engineering, especially to the early phases of software development process, i.e. the requirements engineering. Wiegers (2003) defines requirements engineering as the domain that includes all project life cycle activities associated with understanding a product’s necessary capabilities and attributes. According to Wiegers, RE also includes requirements development and management, and it is a sub discipline of system engineering and software engineering. The activities of the RE process include requirements elicitation, requirements analysis and negotiation, requirements documentation, and validation (Kotonya and Sommerville, 1998), whereas the activities of FEI can be summarized to include the following activities: opportunity identification, task definition, idea generation, idea screening and selection, concept development, concept testing, customer need assessment, technology verification, business analysis, and project
planning (Berg et al., 2009). The components of the theoretical framework and the research focus of this study are presented in Figure 1.

![Theoretical framework and research focus](chart.png)

**Figure 1: Theoretical framework and research focus**

This study has been carried out in the Department of Industrial Management, which combines research in business studies and technological engineering sciences. The business studies in this thesis are closely connected to technology and innovation management studies, whereas the requirements engineering represents technological engineering science. The individual publications have been created together with researchers in the Departments of Industrial Management and Information Technology, as well as with cooperating partners in software development organizations.

### 1.2 Characteristics of the software industry

The software industry is characterized by a high rate of process and product innovation, high knowledge intensity, rapidly shrinking product and technology life cycles, global market, intensive competition, and highly dispersed value chains (Nambisan, 2002). According to Tidd et al. (2005), software involves not only the design and manufacturing functions, but also administration, co-ordination, and distribution functions. Therefore it opens up technological opportunities in all sectors in both manufacturing and services.

The software industry has been evolving and growing quickly both worldwide and in Finland. According to the results of the eleventh annual Finnish software industry survey (Rönkkö et al., 2008), the total revenue of the software business in Finland was approximately 4B€ in 2007. Rönkkö et al. define the software industry to encompass all industries where software goods are developed and traded, and industries where services related to software development and systems implementation are provided. Whereas, the survey amounted Finnish software product business revenues to 1.52 B€ in 2007. Software
product business is based on selling software goods owned by the company (Rönkkö et al., 2008). A comparison of software markets during 2006-2008 is presented in Table 1, which summarizes the size and growth rate of software product business in Finland, the EU, the US and the world. The table indicates that Finland’s share of the European market in 2008 was 2 %, and the European and US software together formed almost 80 % of the global markets. In 2006, the Finnish IT industry consisted of approximately 8 000 firms with 46 000 employees, of which the contribution of the software industry was 33 000 employees (Ali-Yrkkö and Martikainen, 2008).

Table 1: Comparison of software markets in 2006-2008 (Rönkkö et al., 2008, p. 6)

<table>
<thead>
<tr>
<th></th>
<th>2006 (B€)</th>
<th>Forecast for 2008 (B€)</th>
<th>Mean Growth Rate 2006-2008</th>
</tr>
</thead>
<tbody>
<tr>
<td>World</td>
<td>208.8</td>
<td>238.5</td>
<td>7.40%</td>
</tr>
<tr>
<td>US</td>
<td>89.9</td>
<td>105.6</td>
<td>8.40%</td>
</tr>
<tr>
<td>EU</td>
<td>71.5</td>
<td>81.2</td>
<td>6.50%</td>
</tr>
<tr>
<td>Finland</td>
<td>1.4</td>
<td>1.65</td>
<td>8.60%</td>
</tr>
</tbody>
</table>

The annual survey of the Finnish software industry by Rönkkö et al. (2008) also reveals several characteristics about the software industry in Finland in 2007:

- there were 1556 software product companies in Finland (i.e. companies selling software goods owned by the company)
- the number of non-product software companies was 548 (i.e. companies providing software-related services to customers)
- almost 50 % of the software companies were small, having 2 to 10 employees, with one fifth having only one employee
- there were only a few (3 %) large software companies (more than 250 employees); the rest were small and medium-sized enterprises (SMEs)
- almost half (46 %) of software companies had a total revenue of less than 0.3 M€
- 69 % of the companies had a total revenue of less than 1 M€
- the size of a Finnish software company (i.e. total revenue) was quite small, 5 % had more than 10 M€ revenue
- over 90 % of the companies were located near technology centers, universities or affiliate university units
- more than 50 % of all the companies and over 80 % of the large companies were located in the capital area

Ali-Yrkkö and Martikainen (2008) point out that in addition to software firms, many other companies from different sectors utilize software in their products and services without the software itself being visible to the users. Software has a fundamental role e.g. in the function of elevators, mobile phones, trains, as well as paper machines. According to Rönkkö et al. (2008), the boundaries of the software industry are challenging to define, and they have developed a classification of software industry, shown in Figure 2. Furthermore, Lehtola et al. (2009) divide software companies roughly into two: 1) companies operating in the software project business (other terms used are ‘bespoke software’ and ‘software services’), and 2) the software product business (also called ‘market-driven software development). The study of Cusumano (2008) reveals that over the past few years, the revenues of software business have shifted to services, and traditional product sales and license fees have declined.
This thesis concentrates on customer-tailored software and in-house systems, where the degree of standardization of the software is low (see Figure 2). The software organizations in this study are companies that produce and sell their software products or services to other companies (B2B), which in turn use them for developing new products or as components for new products. Furthermore, the software can be used as supportive tools in the customers’ own business processes. These customized software products/services are systems commissioned by a particular customer. The software is developed specially for that customer by a software contractor. For customized products/services, the specification is usually developed and controlled by the organization buying the software.

1.3 Research objectives and questions

The main objective of this thesis is to explore the challenges and problems in the early phases of the software development process, and to improve these processes in order to reduce the costs and development time of software projects, and to improve software quality. The study puts emphasis on the practices and methods especially for collecting and analyzing customer needs and software requirements. Another objective is to exploit the knowledge and best practices presented in the front end of innovation and requirements engineering literature, and describe the similarities and differences between these two fields of research previously unconnected in literature. However, the front end of innovation research and requirements engineering in software development have both realized the opportunities for overall innovation process improvements by focusing on improving the front end activities.
Thus the objectives of this study can be summarized as the following research questions:

1. What are the similarities and differences between the concepts front end of innovation and requirements engineering? (Publication 1)
2. How perceptions about the front end of innovation concepts differ between product and software organizations? (Publication 2)
3. What are the challenges and problems of requirements elicitation in software development? (Publication 3)
4. What practices and techniques a software company uses to support their front end activities, and how these practices can be improved? (Publication 4)
5. How can group methods support the front end activities of software development? (Publications 5 and 6)

1.4 Structure of the study

This thesis is divided into two parts, as depicted in Figure 3. The purpose of the first part is to provide an introduction to the research area, describe the motives and research questions of the study, explain and describe the research process and methodological choices made during this research project, summarize the main findings of the individual publications presented in part two, and finally to discuss about the contribution of the study. The structure of the first part has been constructed after the main publications were published in international journals or conferences. The main objective of the first part is to cover the main topics related to the publications, and to bring out some additional viewpoints that were not concerned at the time of writing the articles.

The second part of the thesis consists of six (6) individual publications, which can be divided into three areas: 1) processes and concepts, 2) challenges and development needs, and 3) means and methods for the front end activities of software development. Publications 1 and 2 focus on the processes and concepts of FEI and RE, and explore the similarities and differences between these two fields of research. Publications 3 and 4 investigate the challenges and development needs related to the front end activities of the software development process. Finally, new group methods and processes for collecting customer needs and software requirements are presented in Publications 5 and 6.

All the publications have been written with colleagues in the Faculty of Technology Management at Lappeenranta University of Technology (LUT) and with cooperating partners in the software industry, except for Publication 3, which has been written solely by the author of this thesis. The contribution of the authors in the publications is summarized in the beginning of this thesis.
PART I: INTRODUCTION

PART II: PUBLICATIONS

<table>
<thead>
<tr>
<th>Front End of Innovation Processes and Concepts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research question 1</td>
</tr>
</tbody>
</table>

**Challenges and Development Needs**

<table>
<thead>
<tr>
<th>Research question 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. A Model for Aggregating the Challenges and Problems of Requirements Elicitation in Software Development</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Research question 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Assessing and Improving the Front End Activities of Software Development: Experiences in a Software House</td>
</tr>
</tbody>
</table>

**Means and Methods for the Front End of Software Innovation**

<table>
<thead>
<tr>
<th>Research question 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>5. The Front End of Innovation – A Group Method for the Elicitation of Software Requirements</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Research question 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>6. A Group Support System Process for the Definition of Software Requirements</td>
</tr>
</tbody>
</table>

Figure 3: Structure of the thesis
2 Innovation process

There exists several definitions about the term innovation in literature, and e.g. Tidd et al. (2005) define innovation as a process of turning opportunity into new ideas and of putting these into widely used practice. According to Trott (2005), the term innovation is a very broad concept that can be understood in a variety of ways, and he introduces the notion that innovation is a process with number of distinctive features that have to be managed. Thus, Trott (2005, p. 15) defines innovation as follows:

“Innovation is the management of all the activities involved in the process of idea generation, technology development, manufacturing and marketing of a new (or improved) product or manufacturing process or equipment”.

2.1 Innovation process models

The literature features numerous innovation process models and business decision models that describe how companies develop or should develop new products or services. According to Trott (2005), to many people new products are the outputs of the innovation process, where the new product development (NPD) process is a sub-process of innovation. Trott continues that managing innovation concerns the conditions that have to be in place to ensure that the organization as a whole is given the opportunity to develop new products, and the actual development of new products is the process of transforming business opportunities into tangible products. Koen et al. (2002) suggest that the innovation process can be divided into three areas: the fuzzy front end (FFE), the new product development process, and commercialization. Herstatt et al. (2006) have described a simplified process of product development, which demonstrates the stage at which the front end of innovation plays a role in the innovation process, as depicted in Figure 4. First, the idea is evaluated in an iterative process in Phase I. The tasks of the second phase are the development of a more detailed product concept and the initial project planning. The output of the front end is a detailed business plan, which is the basis for the decision on a business case. Then, if the concept passes the kill/go gate, the actual development of the product starts in Phase III, and is followed by prototype development and testing, production, market introduction and diffusion.

Figure 4: New product development process (modified from Herstatt et al., 2006)
In general, the innovation process involves activities that are essentially common to all companies (Tidd et al., 2005, p. 67):

1. **Searching** - scanning the internal and external environment for, and processing relevant signals about, threats and opportunities for change.
2. **Selecting** - deciding, on the basis of a strategic view of how the enterprise can best develop, which of these signals to respond to.
3. **Implementing** - translating the potential in the trigger idea into something new and launching it in an internal or external market.
4. **Learning** - companies have the opportunity to learn from progressing through this cycle so that they can build their knowledge base and improve the ways in which the process is managed.

However, according to Koivuniemi (2008), different types of innovation processes are required for different types of innovation. Innovations vary widely, e.g. in scale, by nature, and in the degree of novelty, and so do innovating organizations (Tidd et al., 2005). Tidd et al. classify four broad categories of innovation, i.e. the ‘4Ps’ of innovation:

- **Product innovation** – changes in the things (products/services) which an organization offers;
- **Process innovation** – changes in the ways in which they are created and delivered;
- **Position innovation** – changes in the context in which the products/services are introduced;
- **Paradigm innovation** – changes in the underlying mental models which frame what the organization does.

In addition, Trott (2005) points out that the management of the process is dependent on the type of the product being developed. This thesis focuses on supporting new software product innovations by introducing new methods and process innovations related to the early phases of the software development process.

### 2.2 The front end of innovation

The front end of innovation (FEI) is defined as those activities in the innovation process that take place before an actual, well structured product development process (Koen et al., 2001). The purpose of product development is to create an application, product, or service based on a concept delivered from the FEI. Generally, the FEI is regarded to be an important step in building new products or services, and by improving the FEI, there are great opportunities for overall innovation process improvements.

Koen et al. (2001) have studied the FEI, and they have realized the difficulty of comparing FEI practices across companies. They have developed a theoretical construct, the new concept development (NCD) model. The NCD model provides an insight, a common language and terminology to the key components of the front end of innovation. The NCD model, as illustrated in Figure 5, consists of three key parts (Koen et al., 2001):

1. The inner spoke area of the NCD model defines the five controllable activity elements (opportunity identification, opportunity analysis, idea generation, idea selection, and concept and technology development) of the front end of innovation.
2. The Engine drives the five front-end elements and is fuelled by the leadership and culture of the organization.
3. The Influencing Factors consist of organizational capabilities, business strategy, the outside world, and the enabling science that will be utilized.

The inner parts of the NCD model are specially designated as elements rather than processes. The circular shape of the model is meant to propose that ideas are expected to go around the circle and iterate between and among the five elements, in any order or combination. This approach is in contrast to the sequential new product development processes, in which “looping back” activities are associated with significant delays, added costs and poorly managed projects. (Koen et al., 2001)

Figure 5: New concept development model for organizing the core activities in the front end of innovation (Koen et al., 2001, p. 47)

In addition to the NCD model and rather extensive literature review of the FEI by Koen et al. (2001; 2002), the literature features several studies, which describe the main activities of the FEI. Khurana and Rosenthal (1998) define the core responsibilities of a development team in the front end, such as a) identifying customer needs and competitive situations; b) performing technology evaluation of current capabilities and requirements, as well as alignment with existing business and technology plans; c) identifying core product requirements; d) testing the concept; e) specifying the resources needed to complete the project; and f) identifying key risks and challenges. Furthermore, Poskela et al. (2004) have classified different front-end process models based on their formality and iterative nature, and Koivuniemi (2008) has studied different terms that have been used to describe the front end phase of innovation. Berg et al. (2009) summarize the front end activities at the operative level, such as: opportunity identification, task definition, idea generation, idea screening and selection, concept development, concept testing, customer need assessment, technology verification, business analysis, and project planning. In addition, Brem and Voigt (2009) have introduced an advanced front end of innovation approach, which integrates market pull and technology push in the corporate front end and innovation management.
2.3 Software development process

Within the software industry, the new product development (NPD) process is called software development (SD) in the case of developing software products, applications or services. According to a study of Nambisan and Wilemon (2000), the NPD and SD share several similarities and face many common challenges. The study of Nambisan and Wilemon reveals that much of the SD research has focused on development methodologies, techniques, and use of technology to support the development process. On the other hand, the NPD literature frequently emphasizes the organizational issues associated with the development process, and the focus of the NPD research has largely been on the interaction of people and the processes involved in the different phases. The different emphasis of SD and NDP research is presented in Figure 6.

![Figure 6: Research/practice emphasis: software and new product development (Nambisan and Wilemon, 2000, p. 213)](diagram)

As well as in the case of innovation process models, there exist several software development process models to explain the different approaches to software development. The first published model of the software development process, which was derived from other engineering processes, is presented in Figure 7. This model is known as the “waterfall model”, because of the cascade from one phase to another, or as the software life cycle model (Sommerville, 2001). According to Kess and Haapasalo (2002), there has been some justified criticism over the waterfall model. They point out that in practice it is hard to manage a software project in a linear manner, as detailed specifications are extremely difficult to make without any iterations and feedback to the process later, and the ‘time to market’ is too long, i.e. it takes too long before any significant results from the project can be presented to the (potential) customer. In addition to the waterfall model, other software development life-cycle models include e.g. evolutionary development, formal systems development, reuse-based development, incremental development and spiral development (Sommerville, 2001). Although there are many different software processes, there are four fundamental activities which are common to all software processes (Sommerville 2001, p. 43):
1. **Software specification.** The functionality of the software and the constraints on its operation must be defined.
2. **Software design and implementation.** The software to meet the specification must be produced.
3. **Software validation.** The software must be validated to ensure that it does what the customer wants.
4. **Software evolution.** The software must evolve to meet changing customer needs.

![Software life cycle model](Sommerville, 2001, p. 45)

Recently, Extreme Programming (XP) and agile methods in general have gained supporters among system development practitioners (Ovaska, 2009). According to Kettunen (2009), the basic premise in agile software development methods is that a small, co-located team working closely together with customer(s) can create a high-value product cost-effectively with frequent short iterations. The main characteristics of XP are short iterations with small releases and rapid feedback, customer participation, communication and coordination, continuous integration and testing, collective ownership of the code, limited documentation and pair programming (Abrahamsson et al., 2002). Other agile software development methods, according to Abrahamsson et al., include such methods as Scrum, Feature Driven Development, the Rational Unified Process (RUP) and Adaptive Software development.

### 2.4 Requirements engineering

According to Härkönen et al. (2009), requirements for products are set by customers, standards and technical constraints, and they illustrate the role of requirements within the NPD process, as presented in Figure 8. Kotonya and Sommerville (1997) determine that requirements are defined during the early stages of a system development as a specification of what should be implemented. The diverse definitions of the term requirement suggest that there is no universally accepted definition of what a requirement is (Kauppinen, 2005). On the one hand, Kauppinen discloses that RE researchers seem to agree relatively widely on the division of requirements into functional and non-functional ones, and many researchers also
classify constraints as one of the requirement types. On the other hand, few developers make a distinction between functional and non-functional requirements that are commonplace in the literature (Hansen et al., 2008).

In addition to the different requirement types, Kauppinen (2005) points out that there are different levels of requirements as well, as shown in Table 2. According to Kaupinen, requirements can be defined from business, user and development perspectives. Wiegers (2003) defines 1) business requirements as a high-level business objective of the organization that builds a product, or of a customer who produces it, 2) user requirements as user goals or tasks that users must be able to perform with a system, or statements of the user’s expectations of system quality, and 3) functional requirements as a statement of a piece of required functionality, or a behavior that a system will exhibit under specific conditions. According to Hansen et al. (2008), different levels of requirements may be discovered, specified and managed across stakeholders or organizations, and ensuring consistency across different levels creates a complex set of challenges.

### Table 2: Levels of requirements (Kaupinen, 2005, p. 14)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Business</td>
<td>Business</td>
<td>Business requirements</td>
<td>Stakeholder needs</td>
<td>User requirements</td>
</tr>
<tr>
<td>User perspective</td>
<td>requirements</td>
<td>requirements</td>
<td>requirements</td>
<td>Features</td>
<td></td>
</tr>
<tr>
<td>Development</td>
<td>Software</td>
<td>User requirements</td>
<td>Functional</td>
<td>Software requirements</td>
<td>System requirements</td>
</tr>
<tr>
<td>perspective</td>
<td>needs</td>
<td>requirements</td>
<td>requirements</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Customer/</td>
<td>System</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>user-oriented</td>
<td>requirements</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>requirements</td>
<td>System</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 8: Requirements within the NPD process (Härkönen et al., 2009, p. 547)
However, the term “front end of innovation” is not commonly used in the field of the software industry. Within software development, the front end of innovation activities are part of a practice called requirements engineering. RE is one of the early processes of software development, and traditionally RE has been seen as a front end activity that forms a solid basis for the other activities of new product development (Kauppinen et al., 2007). There is no single RE process which is suitable for all organizations. Each organization must develop its own process appropriate for the type of systems it develops, its organizational culture, and level of experience, and the ability of the people involved in requirements engineering (Kotonya and Sommerville, 1998). Wiegers (2003) divides RE activities into Requirements development and Requirements management, as illustrated in Figure 9. Requirements development can be further subdivided into elicitation, analysis, specification, and validation. The requirements development phase focuses on developing baselined requirements before the actual software development, and once the development is started, the requirements are managed through a requirements change process (Wiegers, 2003). However, the research literature provides several different frameworks for describing the tasks of the requirements process. E.g. Hansen et al. (2008) adopt a categorization of the requirement process into three facets: 1) discovery, 2) specification, and 3) validation & verification. Moreover, Sadraei et al. (2007) point out that RE process models used in practice differ from RE process models represented in literature.

Figure 9: Main activities of the requirements engineering domain (Wiegers, 2003, p. 13)

Requirements elicitation or requirements discovery is the first step in the requirements process. It is the process of identifying and understanding the needs and constraints of customers, different user classes and other stakeholders for a proposed software product. According to Laporti et al. (2009), the requirements elicitation phase has several activities, which include understanding the domain, capture and classification of the requirements, establishment of priorities, conflict resolution, verification of inconsistencies and ambiguities, and finally, a negotiation of the system requirements. The second step of Wiegers’ (2003) RE model is the requirements analysis, which can overlap with requirements elicitation, as some problems are obvious as soon as the requirements are expressed. Requirements are analyzed in detail, and different stakeholders negotiate to decide which requirements are to be accepted (Kotonya and Sommerville, 1998). In the third phase, in requirements documentation or specification, the agreed requirements are documented at an appropriate level of detail to be understandable to all system stakeholders. In the last phase, i.e. requirements validation & verification, the requirements are checked carefully for consistency and completeness. According to Belt (2009), the aim of validation
is to prove that the user is satisfied, and verification is widely understood as a method to prove the compliance with the specifications. This last phase is important because the requirements specifications will then be used as the basis for system development (Kotonya and Sommerville, 1998). Typically validation and verification are positioned at the end point of the requirements process, but in reality these activities begin simultaneously with requirements discovery and continues through the specification activity (Hansen et al., 2008). In addition, Requirements management (RM) is concerned with all of the processes involved in changing system requirements (Sommerville and Sawyer, 1997). RM is a process which supports other requirements engineering activities and is carried out in parallel with them.

The traditional “waterfall model” for software development, presented in Figure 7, suggests that requirements engineering is itself followed by a software design process (Kotonya and Sommerville, 1998). However, Kotonya and Sommerville point out that the reality of software development is much more complex than implied by this model, and there are no clear phases with well defined boundaries between them; the different documents describing the software system are not necessarily completed before the next stage of the process begins; there is a lot of feedback between process activities. Therefore, Kotonya and Sommerville offer an alternative way to present the requirements engineering process as a spiral model of RE, as depicted in Figure 10. The model illustrates that different activities in RE are repeated until a decision is made that the requirements document should be accepted, and if the draft of the requirements document is found to have problems, the elicitation, analysis, documentation, and validation spiral is re-entered. This continues until an acceptable document is produced or until external factors, such as schedule pressure or lack of resources force to end the requirements engineering process (Kotonya and Sommerville, 1998).

Figure 10: Spiral model of the RE process (Kotonya and Sommerville, 1998, p. 35)
3 Software process improvement

There has been a great deal of interest in software process improvement (SPI) when the importance of processes in producing high quality software has been acknowledged. SPI has become one of the major approaches to improve performances within the software industry (Napier et al., 2008). SPI aims to improve the effectiveness of the software development process by assessing and understanding the existing processes, and changing these processes to reduce the costs and development time of software products, and to improve the quality of the software. The first step in a process improvement activity is to assess the practices currently used in the organization, and to identify their strengths and shortcomings (Wiegers, 2003). The most comprehensive software process assessments are based on established process improvement frameworks, such as the Capability Maturity Model Integration (CMMI) (Software Engineering Institute (SEI), 2006) or SPICE (Emam et al., 1998). SPICE is a standard framework for assessing the software processes of an organization, either to improve those processes or determine process capability. The purpose of the CMMI for development is to help organizations to improve their processes for both products and services, and it consists of best practices that cover the product’s lifecycle from conception through delivery and maintenance (SEI, 2006). Furthermore, CMMI includes 22 process areas including such as requirements development and requirements management. In addition, other maturity models for RE include e.g. a RE process maturity approach by Sommerville and Sawyer (1997), the Requirements Engineering Process Maturity (REPM) model (Gorschek and Tejle, 2002), and the Requirements Capability Maturity Model (R-CMM) (Beecham et al., 2005).

Wiegers (2003) proposes three ways to initiate process improvement: (1) correcting problems faced on previous or current projects, (2) anticipating and preventing problems in future projects, and (3) adopting practices that are more efficient than the practices currently used. Even if fixing acute problems is very important, in the long run adoption of systematic method(s) is likely to provide wider benefits, as noted by Humphrey (2002, p. 66): “Method is critical and the disciplined practice of sound methods is the only way to learn to do consistently high quality work”. According to Sommerville (2001), processes may include outdated techniques or may not take advantage of the best practices in industrial software engineering, and many organizations still rely on ad hoc processes and do not utilize the methods offered by software engineering in their product development. Process improvement is sometimes seen as the introduction of new methods or techniques (Kotonya and Sommerville, 1998).

3.1 Problems of requirements elicitation

Problems in the front end activities of software development are widely acknowledged as having an essential impact on the effectiveness of the software development process (Niazi et al., 2008). Davey and Cope (2008) argue that it is clear that requirements elicitation has not been done well, and that this failure causes considerable problems. Some obstacles to a successful elicitation of requirements are summarized in the seminal paper by Davis (1982). They include the constraints of humans as information processors and problem solvers, the variety and complexity of information requirements, and the complex pattern of interaction among users and analysts in defining requirements. According to Christel and Kang (1992),
problems of requirements elicitation can be grouped into three categories, and they have classified ten elicitation problems to this framework as follows:

**Problems of scope**
- The boundaries of the system are ill-defined
- Unnecessary design information may be given

**Problems of understanding**
- Users have incomplete understanding of their needs
- Users have poor understanding of computer capabilities and limitations
- Analysts have poor knowledge of the problem domain
- Users and analysts speak a different language
- Ease of omitting “obvious” information
- Conflicting views of different users
- Requirements are often vague and not testable, e.g. “user friendly”, “robust”

**Problems of volatility**
- Requirements evolve over time

According to Coughlan and Macredie (2002), there have been numerous studies on the different types of problems in system design, and most of them are as a result of a breakdown in communication. Coughlan and Macredie continue that the overriding reason for the existence of communication problems lies in the fact that software design and development is very much a behavioral process, where human and organizational elements have an important impact on the design. Valenti et al. (1998) have discovered that several users may evaluate the same information need differently, or may see requirements as raising either conflicting or competing use of limited resources, according to their role in the organization, background or mindset. According to Pitts and Browne (2007), users and other stakeholders are often uncertain about their needs and/or are unable to articulate them clearly, requirements typically change during a project, and analysts are often poorly trained in requirements gathering techniques. Furthermore, Pitts and Browne argue that there are limited initiatives, in research or practice, aimed at providing methods or tools for improving requirements elicitation. In addition, few application developers have any training in any elicitation techniques (Leffingwell and Widrig, 1999), and there is lack of instructions for methods (Pöyhönen and Hukki, 2004).

### 3.2 Techniques for requirements elicitation

The field of all possible elicitation techniques that can be used during requirements elicitation is vast. It includes techniques borrowed from the fields of communication (e.g. consensus decision making), psychology (e.g. construct analysis), instructional design (e.g. task analysis), journalism (e.g. interviews), and anthropology (e.g. observations) (McGraw and Harbison, 1997). Traditional techniques encompass a variety of generic data gathering techniques. These include the use of questionnaires and surveys, interviews, introspections, and analysis of existing documentation, such as organizational charts, process models or standards, and user or other manuals of existing systems. Empirical studies (Davis et al., 2006) show that interviews appear to be one of the most effective, individual elicitation techniques in a wide range of domains and situations, and the most commonly used in practice. In addition to the traditional techniques, other techniques suitable for requirements elicitation are e.g. prototyping, group elicitation techniques, contextual and cognitive techniques, and model-driven techniques (Nuseibeh and Easterbook, 2000). Furthermore, Jiang et al. (2008) have analyzed 46 RE techniques, and information about these techniques,
their attributes, weaknesses and strengths were identified and stored in the RE Techniques Knowledge Library (RETKL), which can be used during selection of RE techniques.

The choice of a specific elicitation technique varies, based on the time and resources available to the requirements analyst, the kind of information that needs to be elicited, the type of application, the skill and sophistication of the development team and the customer, the scale of the problem, the technology used, and the criticality and uniqueness of the application. Careful technique selection ensures that the requirements analyst can select the right tool for the right need, and increases the probability of desirable results. Maiden and Rugg (1996) have presented a framework called ACRE (ACquisition of REquirements) that assists requirements analysts in choosing methods for requirements acquisition. Furthermore, Jiang et al. (2008) have developed a Knowledge-based Approach for the Selection of Requirements Engineering Techniques (KASRET), which helps during RE techniques selection, provides mechanisms for the management of knowledge about requirements techniques and support for RE process development. According to Koivuniemi (2008), it is not possible to name any single best system or method for all – best practices cannot be copied directly, but they need to be customized through organization-specific needs. In addition, Ovaska (2009) points out that industry is looking for practical and simple methods to be able to coordinate and guide the development work.

3.2.1 Group methods

According to Trott (2005), more recent innovations and scientific developments, such as significant discoveries like mobile phones or computer software and hardware developments, are associated with organizations rather than individuals. In addition, the creation, development and commercial success of new ideas require a great deal of input from a variety of specialist sources, and often vast amounts of money. Thus, today’s innovations are associated with groups of people or companies, and innovation is invariably a team game (Trott, 2005).

The Nominal Group Technique (NGT) is one of the earliest managerial methods specially designed to support group work. The NGT is a structured problem-solving or idea-generating strategy in which individuals’ ideas are gathered and combined in a face-to-face, non-threatening group situation. According to McGraw and Harbison (1997), the NGT technique is often used when a group needs to compare, select, or rank solutions or advantages and disadvantages of ideas. Furthermore, Quality Function Deployment (QFD) can be defined as an overall concept that provides a means for translating the voice of the customer into product features (Sharma et al., 2008). According to the comprehensive literature review by Sharma et al., the primary functional fields of QFD are product development, customer requirements analysis, and the quality management system. QFD is also a popular area in software development processes, and it is especially used when analyzing collected customer needs and translating the needs into software features.

In addition, Kärkkäinen et al. (2001) have developed ten need assessment tools for customer-driven product development, and for the management of development activities in industrial companies. The tools cover the most important phases in the industrial customer need assessment process. These include the planning of need assessment, organizing, analyzing and prioritizing customer needs, and ensuring that customer needs really direct the product development. Kärkkäinen et al. (2003) have also re-designed the Opera method, by Helin
to suit better for systematic assessment of customer needs in industrial markets. The resulting need assessment tool is called AVAIN. In addition, other group methods for need assessment and requirements elicitation include methods as such brainstorming, consensus decision making, focus groups and Joint Application Development (JAD). JAD is a generic term that describes a variety of methods for conducting workshops where stakeholders and developers work together on the system development phases, including requirements definition (Laporti et al., 2009).

3.2.2 Electronic group systems

The problem of gathering accurate requirements and the inefficiencies of user interviews and group meetings were some of the early driving forces for Group Support Systems (GSS). A GSS is a set of techniques, software and technology designed to focus and enhance the communication, deliberations, and decision making of a group. The GSS aids a group in cooperating and working effectively together to reach its goals by eliminating the barriers of communication, by offering tools for the group, and by leading the use of time and handling of items systematically. According to Turban et al. (2005), the aim of a GSS is to reduce the process losses associated with conventional group work, for instance disorganized and unfocused activity, the dominance of some members of the group, and social pressure. However, several different terms are used in this context along with the term GSS. One very similar term used in the same context is Group Decision Support Systems (GDSS). Today, according to Lindqvist (2009), it logical to use the term GDSS mainly in direct connection with the actual decision room, i.e. GDSS laboratory, and the correct term for computer-aided group support is GSS. Some generally, sometimes rather synonymously utilized other terms include groupware, Computer Supported Co-operative Work (CSCW) and Electronic Meeting Systems (EMS).

GSSs have been highly successful in improving the productivity and outcomes of group meetings. Wanninger and Dickson (1992) have used a GSS in the information systems development process, to facilitate group work and to support user involvement in the development process, especially in the requirements definition phase. The general benefits identified by Wanninger and Dickson were time savings, obtaintment of input in the form of ideas from quiet members, and also prevention of the domination of the meeting by a member due to their position or personality. Yihwa and Minder (1994) have used a GSS with JAD to facilitate the requirements specification process. The JAD session was supported by the GSS, and the results of the pilot study were promising; according to the interviews the GSS speeded the JAD process up significantly.

Further, the IBM Canada Laboratory has utilized a GSS in the collection of user feedback information and the application of this information to support the design and development of software products (Boshes, 1996). According to Boshes, the GSS has helped IBM Canada to understand the users and evaluate their software design. There have also been some cases in which a computer-supported cooperative environment has been used to provide support and teach asynchronous cooperation to groups carrying out requirements elicitation (Swigger et al., 1994; Swigger et al., 1995). In addition, a GSS has been successfully used for requirements engineering in an EasyWinWin project (Briggs and Gruenbacher, 2002). EasyWinWin is a requirements negotiation methodology supported by a GSS. According to the studies by Briggs and Gruenbacher, EasyWinWin helps to improve stakeholder involvement in the requirements engineering process and assists in managing the complexity
of requirements. Furthermore, Hengst et al. (2004) have presented a design for a repeatable process for eliciting user requirements for mobile information services based on the GSS.

Fjermestad and Hiltz (2001) have conducted a literature review summarizing 54 case and field studies in 79 published papers of GSS research, and provide an aggregate analysis of the methodology and results. An article by DeSanctis et al. (2008) summarizes the results of the Minnesota GDSS research project, which is a 20-year program of research that has generated more than 80 articles, chapters, dissertations, and proceedings publications. The study by DeSanctis et al. presents the basic theoretical framework, experimental strategy and design, field study design and results, along with a discussion of the significance and implications of the research project for theory and practice. In addition, Elfvengren (2008) has studied the use of GSS in the early phases of the innovation process in business-to-business organizations, and the results show that GSS processes support a company’s planning and decision tasks in the front end of innovation.

The present study focuses especially on face-to-face group techniques in collecting software requirements. The case experiences reported in this thesis, concerning requirements definition with a GSS, have been conducted in the GDSS laboratory of Lappeenranta University of Technology (LUT). The GDSS laboratory is used for research in the field of GSS and supportive processes, and for teaching purposes. Some of the main topics of the GSS research at LUT have been in the fields of strategic planning, technology management, and product innovation management, including group sessions supporting idea generation, customer need assessment and selection of product/service concepts (Pittrainen et al., 2007; Karhumaa, 2009).
4 Research methodologies

4.1 Classification of research approaches

Fields of science are generally categorized as natural science and social science. This thesis belongs to the field of social science, which comprises academic disciplines concerned with the study of human behavior and societies. The two main traditions of social science research are positivism and social constructionism (Easterby-Smith et al., 2008). Positivism represents the analytical research tradition, and considers the researcher as an objective outside observer who draws conclusions concerning behavioral patterns within natural phenomena (Vafidis, 2002). According to Easterby-Smith et al. (2008), social constructionism emphasizes the idea that ‘reality’ is determined by people rather than by objective and external factors, and the focus should be on what people, individually and collectively, think and feel, and attention should be paid to the ways they communicate with each other, whether verbally or non-verbally.

According to Vafidis (2002), business research is applied research, the aim of which is to offer a scientific contribution, and be of use to practitioners as well, who do not usually care of what kind of research approaches are used, as long as the results help to solve a practical problem. Applied research is intended to lead to the solution of specific problems, and usually involves working with customers to identify the important problems and decide how best to tackle them (Easterby-Smith et al., 2008). Neilimo and Näsi (1980) have developed a model for business research categorization in a two-dimensional framework: theoretical-empirical and descriptive-normative. In this model the categories of conceptual, nomothetical, decision-oriented and action-oriented approaches can be discovered. Lukka (1991) explains the variables of the model as follows:

- Theoretical means reasoning, i.e. theoretical knowledge is a priori knowledge that is observable without experimenting
- Empirical means that the data is collected on the field or in a laboratory
- Descriptive aims to describe “what is” and “how is”, i.e. emphasis is on describing, explaining and forecasting
- Normative is explicitly target oriented, i.e. it aims at recommending a way of acting in practical situations.

The conceptual approach is mainly based on previously formulated concepts and their analysis, and analysis and synthesis are used to create new concepts and frameworks. The goal of the nomothetical approach is to explain causal relations, where empirical evidence plays an important role. Further, the objective of the decision-oriented approach is to design a model or method that solves a certain type of problem under certain conditions. The emphasis of the action-oriented approach is on understanding or changing a certain situation or research object. (Neilimo and Näsi, 1980; Vafidis, 2002)

In addition, Kasanen et al. (1991) have added the constructive research methodology to the model of Neilimo and Näsi. The constructive approach focuses on real-world problems felt relevant to be solved in practice (Lukka, 2000). The categorization of business research and the positioning of publications 1-6 in Part II of this thesis are depicted in Figure 11.
Most research methods that can be used in research are based on either qualitative or quantitative methodologies (Silverman, 2005). Qualitative methods involve collecting data that is mainly in the form of words, and quantitative methods involve data which is either in the form of, or can be expressed as, numbers (Easterby-Smith et al., 2008). This study utilizes research methods based on the qualitative approach. According to Denzin and Lincoln (2005), qualitative researchers study things in their natural settings, attempting to make sense of, or interpret, phenomena in terms of the meanings people bring to them. However, Hirsjärvi et al. (2009) point out that quantitative and qualitative research approaches are difficult to distinguish accurately from each other, they can be seen as complementary approaches instead.

### 4.2 Research process

In any qualitative descriptive research, the researcher usually begins their study with one or several preliminary topics, questions, or problems driving the collection of empirical data, and new exciting questions often emerge (Eriksson and Kovalainen, 2008). First, the research process of this thesis was started with practical problems related to the elicitation of software requirements in a case company (Publication 6). The company had realized a need to improve the early processes of their software development, and they especially wanted to make the requirements elicitation process more effective, and to reduce the time required for requirements elicitation. Thus, the objective was to build a new systematic requirements elicitation method for practical application. Second, after the validation of the new method for requirements elicitation, some further improvement areas were discovered in the case company. These were e.g. manual documentation of requirements and the dominance of some team members during the requirements workshops. Therefore, as the challenges in previous methods were discovered, the focus of the study moved to the utilization of group
support systems as a supportive tool for requirements elicitation (Publication 5). The objective was to build a practical construct for GSS-supported requirements elicitation.

Third, during the collaboration with case companies, it was discovered that software organizations struggle with several challenges and problems related to the front end activities of software development, not merely with inefficient methods and techniques of requirements elicitation. However, it was discovered that previous studies (e.g. Valenti, Panti and Cucchiarelli, 1998; Sutton, 2000; Damian and Zowghi, 2003) focused mainly on describing some individual problem areas of requirements elicitation. An extensive model to aggregate the problems and challenges of requirements elicitation was lacking. Thus, a conceptual framework was developed, covering the challenges and problems related to the requirements elicitation phase of software development (Publication 3).

Fourth, as the challenges and problems of requirements elicitation were discovered in Publication 3, it was noticed that the problems perceived in the literature of software and requirements engineering were similar to those in the field of innovation management. However, the definitions and terms in these fields differed considerably. Therefore, it was decided to conduct a further investigation (Publication 1) on the differences and similarities in these fields, and the special focus was on the front end activities of software development, especially the concepts of the front end of innovation and requirements engineering.

Fifth, Publication 1 addresses the differences and similarities about the concepts of FEI and RE on the basis of a literature review. However, an empirical investigation about the managerial perceptions about the concepts was needed to investigate the differences and similarities of the concepts in software and product organizations in practice (Publication 2). The study applied a case study research, and cognitive causal maps were used as a tool to collect and analyze the data. Therefore, a further objective of the study was to get experience about the usage of causal cognitive maps in the collection and analysis of managerial perceptions about the concepts of FEI and RE.

Finally, in parallel with Publication 2, the study was extended to the assessment of all RE activities, i.e. requirements elicitation, analysis, documentation, validation, and management of a small software company. The objective was to further validate the model created in Publication 3, to recognize the RE practices, problems, and development needs, and to outline an RE process improvement agenda for the company, based on the existing literature.

4.3 Selected research methodologies and methods

The individual publications of this thesis follow the principles of several qualitative business research methodologies for solving the research questions, as presented in Table 3. According to Easterby-Smith et al. (2008), authors and researchers who work in organizations and with managers should attempt to mix methods to some extent, because it provides more perspectives on the phenomena being investigated. The following paragraphs describe the used methodologies, and states reasons the selection of a particular research methodology. The order of the publications is organized with relation to the stated research questions, not chronologically.
Table 3: Research strategy: approaches and methods for solving the research questions

<table>
<thead>
<tr>
<th>Research questions</th>
<th>Publications</th>
<th>Research approaches and methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>What are the similarities and differences between the concepts of <strong>front end of innovation</strong> and <strong>requirements engineering</strong>?</td>
<td>Publication 1</td>
<td>Conceptual approach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Literature review</td>
</tr>
<tr>
<td>How perceptions about the front end of innovation concepts differ between product and software organizations?</td>
<td>Publication 2</td>
<td>Nomothetical approach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Semi-structured interviews</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Causal cognitive maps</td>
</tr>
<tr>
<td>What are the challenges and problems of requirements elicitation in software development?</td>
<td>Publication 3</td>
<td>Nomothetical approach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Grounded theory</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Theme interview</td>
</tr>
<tr>
<td>What practices and techniques a software company uses to support their front end activities, and how these practices can be improved?</td>
<td>Publication 4</td>
<td>Action-oriented approach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Case study</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Structured interviews</td>
</tr>
<tr>
<td>How can group methods support the front end activities of software development?</td>
<td>Publications 5 and 6</td>
<td>Constructive approach</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Case study</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Questionnaires</td>
</tr>
</tbody>
</table>

4.3.1 **Publication 1 – conceptual approach**

Publication 1 addresses the research question “What are the similarities and differences between the concepts of **front end of innovation** and **requirements engineering**?” The objective was to clarify the activities, characteristics and meanings of the concepts, and to compare their similarities and differences on the basis of a literature review in the fields of innovation management and software engineering. Based on the analysis of the special characteristics of the software industry, and the differences and similarities between the FEI and RE concepts, a new framework for managing the front end of software innovation was formulated.

The research approach of Publication 1 can be categorized as conceptual. Neilimo and Näsi (1980) suggest that the doctrine of the research is mainly based on previously formulated concepts and their analysis, but it can also include some empirical evidence and research findings. The main method in the conceptual approach is the “method of reasoning”, i.e. analysis and synthesis are used to create new concepts and frameworks, and there are no clear rules or formulas to guide how the research should be performed. Puusa (2008) defines concept analysis as a non-empiric research method that can be used as a basis for empirical research or as an independent research approach. According to Puusa, concept analysis helps to understand what kinds of meanings are related to the concept at hand, how the concept differs in terms of content in comparison to its related concepts, and thereby, what features can be determined to be critical characteristics of the concept.
4.3.2 Publication 2 – nomothetical approach

The research question motivating Publication 2 was “How perceptions about the front end of innovation concepts differ between product and software organizations?” The objective of the study was to investigate the prevailing perceptions about the front end of innovation concepts between the software industry and the traditional industrial sector.

The research category of Publication 2 belongs to the nomothetical approach, which is closely linked to the modernist (positivist) research approach. The aim of the nomothetical approach is to show the causal interdependencies between different phenomena of the economic life, and the empirical part plays an essential role (Neilimo and Näsi, 1980). In Publication 2, causal cognitive maps were used as graphic tools for collecting and analyzing the perceptions of selected experts about the concepts of FEI, and for recognizing the perceived interrelationships between these concepts. Causal cognitive maps are a commonly used cognitive mapping technique, and for instance according to Swan (1997), causal mapping methodologies have been the most commonly used by researchers investigating cognitions of decision-makers in organizations. They seem to have brought illustrative and useful results in managerial studies; for instance, according to Huff (1990), causal maps have been commonly utilized in strategic planning and organizational studies, among others.

The data for Publication 2 was collected with semi-structured interviews in organizations in the software industry and the traditional industrial sector. In total, 9 interviews were conducted in three organizations for the data material. Organization A is an ICT company providing telecommunication services with long experiences in in-house software development and subcontracting their software projects. In this company, the interviews were conducted with a senior verification manager and an IT developer. Organization B is a service and IT-company specializing in project-oriented business, and their customers are organizations that either buy solutions with bundled products and services, or deliver those to their own customers. The interviews with company B were conducted with the managing director of the company and with an IT system specialist. Organization C is an international research institute, the Technology Business Research Center (TBRC) at LUT. In this organization, the interviews were conducted with a researcher specialized in requirements engineering in software development, and 4 researchers focusing on the front end of innovation concepts in the traditional industrial environment.

All the interviewees were given the task of creating an individual causal map to describe their different views and causal beliefs concerning the concepts of the front end of innovation. The concepts represent in this study the critical success factors of the FEI, as well as the outcome measures of a successful product or software development project. After the interviews, the causal maps were combined as two separate causal maps, the software map and the FEI map. The analysis was done by comparing the causal maps, the critical success factors, the outcome measures for project success, and their relationships between the concepts in these two industries.

4.3.3 Publication 3 – nomothetical approach

In order to make improvements into requirements development practices and processes, it is necessary to describe and evaluate existing challenges and problems in the area of requirements elicitation. The principal research question of Publication 3 is “What are the
The research approach of this study can be categorized as *nomothetical*, and *grounded theory* was used as the research method. According to Eriksson and Kovalainen (2008), the grounded theory approach is not a mere method or technique for qualitative data; rather its application in research has more ambitions towards theory development than empirical analysis. Grounded theory has been found suitable for the study of software and information systems. For instance, Damian and Zowghi (2003) have used grounded theory successfully for investigating requirements engineering challenges introduced by stakeholders’ geographical distribution in multi-site organizations, and Ovaska (2009) has used grounded theory in her studies on the coordination of the systems development process.

The methodology commonly known as grounded theory was originally developed by two sociologists, Barney Glaser and Anselm Strauss (Strauss and Corbin, 1990). According to Strauss and Corbin, grounded theory can be divided into three main phases: 1) open coding, 2) axial coding, and 3) selective coding. During open coding the data is broken down into discrete parts, closely examined, and compared for similarities and differences. Events, happenings, objects, and actions/interactions that are found to be conceptually similar by nature or related in meaning are grouped under more abstract concepts termed “categories”. Examining the data closely for differences and similarities allows for fine discrimination and differentiation among categories. In later analytic steps, such as axial and selective coding, the data are reassembled through statements about the nature of relationships among the various categories and their subcategories. These statements of relationship are commonly referred to as “hypotheses”. The theoretical structure that ensues makes it possible to form new explanations about the nature of phenomena. (Strauss and Corbin, 1990)

The study in Publication 3 was started with open coding, by studying the literature and breaking the data into discrete parts. The identified problems of requirements elicitation were captured and recorded as codes. The data sources were recorded for each code in order to maintain traceability. The codes were closely examined and compared for similarities and differences. The problems of requirements elicitation that were found to be conceptually similar by nature were grouped under categories, and these categories were given a descriptive title. In order to saturate the theory, the open coding was continued until there were no new codes or categories discovered in the literature. In later analytic steps, such as axial and selective coding, the data were reassembled through statements about the nature of relationships among the various categories and their subcategories. As a conclusion of the study, a new model for aggregating the challenges and problems related to requirements elicitation was created. The developed model was validated through an initial theme interview with a requirements specialist in a software development department of an ICT company in order to gain insight into the developed model. The further validation of the model is reported in Publication 4.

### 4.3.4 Publication 4 – action-oriented approach

The main purpose of the Publication 4 was to assess the current RE practices in a software company, and to outline an improvement agenda for the company based on interviews, existing literature, and experience of the researchers. Thus, the objectives of this publication can be summarized as the following research question: “What practices and techniques a software company uses to support their front end activities, and how these practices can be improved?”
The research approach of Publication 4 can be categorized as action-oriented, and it applies a case study research. Yin (1994) defines a case study as an empirical inquiry that investigates a contemporary phenomenon within its real-life contexts, especially when the boundaries between phenomenon and context are not clearly evident. According to Punch (1998), the basic idea of a case study is that one or perhaps a small number of cases will be studied in detail, using whatever methods seem appropriate. Furthermore, according to Punch, the general objective is to develop as full an understanding of a case as possible, although there may be a variety of specific research questions.

The study was started with key informant discussions to gain preliminary understanding of the company situation. As the current RE practices were found to be at an initial level, an interview-based approach was chosen to clarify the current practices and development needs. The data was collected by interviewing all the employees with designer responsibilities (6) in a small software company, and a meeting with unit managers (3) was conducted thereafter. The meeting with the design, development, and test managers was conducted to complement the designer viewpoints with a management viewpoint on the company’s RE practices, how the practices fit the company’s whole software development process, and what kind of problems and development needs the managers had observed. The developed assessment framework is described in detail in Nikula et al. (2009). The collected data was analyzed first by reviewing the interview summaries to find common RE practices and problems, and the found competences were tabulated. Second, all the interview recordings were listened to and discussions of closer interest, especially about how people performed specific RE activities, were transcribed. Third, any patterns rising from the data were studied closer to clarify the findings. Finally, the interview findings were compared with the manager meeting notes to identify similarities and differences between the two viewpoints.

4.3.5 Publications 5 and 6 – constructive approach

Publications 5 and 6 address the research question: “How can group methods support the front end activities of software development?” Methodologically, Publications 5 and 6 use the constructive approach. Constructive research designs start from the assumptions that there is no absolute truth, and the work of the researcher should be to establish how various arguments for truth and reality become constructed in everyday life (Easterby-Smith et al., 2008). The logic in the constructive approach is to design a new construct and test its applicability in real-life cases. According to Lukka (2000), the constructions can be of almost any type, ranging from simple models in merely technical terms to complex management system designs covering both technical and socio-technical elements, and to manifestations of new ways of approaching and doing things in an organization. The results of the research are evaluated according to their newness and applicability in the progress of scientific knowledge. The validation of the practical usability of the results is also important in the evaluation (Kasanen et al., 1993).

In Publication 5, a new group method for the elicitation of software requirements was created. Three group sessions for requirements elicitation were arranged in the beginning of software development projects in a Finnish telecommunications company in order to test and improve the developed requirements elicitation method. The first group session was a pilot where the goal was to get experience of the method, and obtain comments on further improvement areas. The other case studies were derived from ongoing software projects, and they were divided into two different sessions. In the first parts of the sessions, the goal was to
elicit all possible customer needs and requirements, and in the second parts, the collected requirements were analyzed and prioritized by the session participants. At the end of each session, a questionnaire was handed to the participants in order to get some feedback about the ease of the process and the effectiveness of the session. Also information about benefits, drawbacks and improvement areas was collected.

The objective of Publication 6 was to improve the process of requirements elicitation and analysis in order to capture and document the requirements needed in the later phases of software development. This study presents a practical group support system (GSS) process for requirements definition, clarifies the advantages and drawbacks of the developed process, and summarizes the features of the developed GSS process which can reduce the problems related to requirements engineering. The GSS process was designed together with the company representatives and a GSS facilitator. The developed process aimed at supporting the gathering, organizing and analyzing of requirements information effectively, and helping the participants to concentrate on the essential tasks when considering the objectives of the session. The process consisted of separate phases that included clarification of the objectives for the session, brainstorming of the requirements for the software, grouping and prioritization of the clarified requirements, and a more detailed analysis of the most important or most diverging requirements. The validation of the proposed GSS process for requirements definition was made through a software development project in a Finnish telecommunications company, and a GSS-supported group session was conducted in the GDSS laboratory at LUT. At the end of the group session, the participants were asked to evaluate the developed GSS process by answering a questionnaire.
5 Summary of the main results

5.1 Comparison between the front end of innovation and requirements engineering

Publication 1 concentrates on the front end of innovation and requirements engineering in the software industry, and discloses the differences and similarities between these two fields of research. The results of the study reveal that the literature of the FEI and RE features several similarities and differences. First of all, the definitions and vocabulary differ from each other considerably, although the activities and tasks have a certain analogy. Both the FEI and RE are defined as those activities that take place before an actual new product or software development process. The term “front end of innovation” is not commonly used in the field of requirements engineering. Other terms for RE used in the literature are e.g. requirements management, requirements definition, or software specification. The front end of innovation is also called the fuzzy front end (FFE), the pre-development phase or phase 0. Table 4 indicates the diversity of the phases, definitions and tasks used in the field of the front end of innovation based on the literature of innovation management, and Table 5 describes some commonly used activities, as well as the main tasks in the requirements and software engineering literature.

The study indicates that some of the phases of the FEI are not included in RE and vice versa. According to Kauppinen et al. (2007), the idea generation activity is not included in RE, because ideas are typically regarded as inputs and are generated outside requirements engineering activities. This often leads to a problem, where new ideas and concepts from e.g. the marketing department never reach software development, especially if they come after the final set of requirements are accepted for software production. RE focuses mainly on collecting, analyzing and documenting requirements based on the ideas generated before RE. Furthermore, the deliverables of the FEI and RE also differ from each other. According to Wiegars (2003), the output of requirements engineering is called a requirements document, functional specification or product specification. The requirements are documented in a formal document which is used to communicate the requirements to customers, system and software engineers and managers of the software engineering process (Wiegars, 2003). The outputs of the FEI are a business plan and/or formal project proposal (Koen et al., 2001). In addition, the nature of work, both in the FEI and RE, is often considered to be chaotic, unstructured, ambiguous and difficult to plan. Both the FEI and RE are claimed to be one of the hardest tasks in the innovation process. However, these two fields of research have realized the opportunities for overall innovation process improvements by focusing on improving the front end activities.

Based on the special characteristics, differences and analogies in these two fields of research, Publication 1 presents a new framework for managing the front end of innovation activities in software development, i.e. for managing the front end of software innovation (FESI). The elements of the model are mainly based on the new concept development (NCD) model (Koen et al., 2002) and the spiral model (Wiegars, 2003) of the RE process. Also the concept of requirements management during the whole software innovation process is included into this approach. The model is comprised of six phases, which can be entered in any order or combination. The phases of the model are 1) Opportunity identification and analysis, 2) Collection of needs and requirements, 3) Requirements analysis and selection, 4) Documentation and concept development, and 5) Requirements and concept testing. In addition, in parallel with all the above activities, 6) Requirements management is concerned...
with managing changes of any kind. The new approach suggests new concepts and activities from the NCD model that could be applied in the requirements engineering processes in order to support and improve software development. Especially the business plan should be one important deliverable from the FEI in order to control the changes in business priorities better and to satisfy customer needs.

Table 4: The activities and main tasks of the front end of innovation

<table>
<thead>
<tr>
<th>Activities/Phases</th>
<th>Main tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Opportunity identification</strong> (Koen et al., 2001, 2002; Poskela et al., 2004; Berg et al., 2009)</td>
<td>- An organization identifies the business and technological opportunities that the company might want to pursue.</td>
</tr>
<tr>
<td><strong>Task definition</strong> (Poskela et al., 2004; Berg et al., 2009) <strong>Mission statement</strong> (Poskela et al., 2004)</td>
<td>- Defines the identified opportunity and the forthcoming task, integrates the opportunity with the strategic goals, and forms the development team.</td>
</tr>
<tr>
<td><strong>Opportunity analysis</strong> (Koen et al., 2001, 2002) <strong>Market and technology analysis</strong> (Khuarana and Rosenthal, 1998)</td>
<td>- An opportunity is assessed to confirm that it is worth pursuing. - Technology and market assessments. - Business capability and competency are assessed. - Sponsorship for further work is determined.</td>
</tr>
<tr>
<td><strong>Idea generation and enrichment</strong> (Koen et al., 2001) <strong>Idea genesis</strong> (Koen et al., 2002) <strong>Idea generation and assessment</strong> (Herstatt and Verworn, 2004)</td>
<td>- The birth, development, and maturation of a concrete idea. - Direct contact with customers and users, linkages with other cross-functional teams, collaboration with other companies and institutions.</td>
</tr>
<tr>
<td><strong>Customer need assessment</strong> (Kärkkäinen, 2002; Poskela et al., 2004; Berg et al., 2009)</td>
<td>- The recognition, gathering, and clarification of customer and user needs and their importance to create useful information about customer needs for new product development.</td>
</tr>
<tr>
<td><strong>Technology verification</strong> (Poskela et al., 2004; Berg et al., 2009) <strong>Technical investigation</strong> (Cooper, 1998)</td>
<td>- Translating the customer needs into technically and economically feasible solutions, including the assessment of the potential technical solution, risks, legal requirements and patent issues.</td>
</tr>
<tr>
<td><strong>Idea selection</strong> (Koen et al., 2001, 2002; Berg et al., 2009) <strong>Idea screening</strong> (Poskela et al., 2004; Berg et al., 2009)</td>
<td>- Identifying and selecting the most potential ideas for further development in order to achieve the best business value.</td>
</tr>
<tr>
<td><strong>Concept and technology development</strong> (Koen et al., 2002) <strong>Concept development</strong> (Poskela et al., 2004; Berg et al., 2009) <strong>Business analysis</strong> (Poskela et al., 2004; Berg et al., 2009)</td>
<td>- Development of a business plan and/or a formal project proposal for the new concept. - Development of a business case is based on estimates of market potential, customer needs, investment requirements, competitor assessment, technology unknowns, and overall project risk.</td>
</tr>
<tr>
<td><strong>Concept testing</strong> (Poskela et al., 2004; Berg et al., 2009) <strong>Project planning</strong> (Herstatt and Verworn, 2004; Poskela et al., 2004; Berg et al., 2009)</td>
<td>- The viability of a new product concept is tested. - Integration of the front-end results by creating a coherent plan.</td>
</tr>
</tbody>
</table>
Table 5: The activities and main tasks of requirements engineering

<table>
<thead>
<tr>
<th>Activities/Phases</th>
<th>Main tasks</th>
</tr>
</thead>
</table>
| Feasibility study (Sommerville, 2001; Sommerville and Sawyer, 1997) | - Business objectives are considered.  
- Current technology is evaluated.  
- Integration with current way of working is assessed.  
- Checking adequate returns of investment. |
| Problem analysis (Leffingwell and Widrig, 1999) | - The process of understanding real-world problems and users’ needs and proposing solutions to meet these needs. |
| Requirements elicitation (Kotonya and Sommerville, 1998; Pressman, 2000; Sommerville, 2001; Wiegers, 2003) Requirements discovery (Kotonya and Sommerville, 1998; Hansen et al., 2008) | - The requirements are discovered through consultation with stakeholders, from system documents, domain knowledge and market studies. |
| Requirements analysis and negotiation (Kotonya and Sommerville, 1998; Sommerville and Sawyer, 1997; Pressman, 2000; Sommerville, 2001; Wiegers 2003) | - The requirements are analyzed in detail and different stakeholders negotiate on which requirements are accepted.  
- Requirements are analyzed for conflicts, overlaps, omissions and inconsistencies.  
- Conflicts are resolved and requirements are prioritized. |
| Requirements documentation (Kotonya and Sommerville, 1998) Requirements specification (Pressman, 2000; Wiegers, 2003; Hansen et al., 2008) | - The agreed requirements are documented at an appropriate level of detail. |
| Requirements validation (Pressman, 2000; Sommerville, 2001; Wiegers, 2003; Hansen et al., 2008) Requirements verification (Thayer and Dorfman, 1997; Hansen et al., 2008) | - Requirements are checked for consistency and completeness.  
- Problems in the requirements document are detected before it is used as a basis for system development. |

5.2 Perceptions about the front end of innovation concepts

The objective of the second publication was to investigate how the perceptions about the front end of innovation concepts between the traditional industrial sector and the software industry differ. Investigating how and to what extent the perceptions and the significance of the front end activities between product and software companies differ, makes it possible to transmit the knowledge and best practices from the traditional industrial environment into software development and vice versa.

Causal cognitive maps were used as a tool to organize and study the perceptions of selected experts about the concepts of FEI and for recognizing their perceptions about the interrelationships between these concepts. In this research article, the concepts represent the critical success factors of the product definition phase (Khurana and Rosenthal, 1998), and
classical generic basic outcome measures of a successful product or software development project (i.e. time, costs and quality). The causal cognitive maps created in the product and software organizations are presented in Figures 12 and 13.

![Combined causal map from experts of software organizations](image)

Figure 12: Combined causal map from experts of software organizations

The analysis of the perceptions between the two industries brings out some interesting notions. The results suggest that both groups believed that quality was the most important outcome measure for project success. Also, both groups seemed to believe that minimizing delays in product or software development projects was more important for project success than the costs. Further, the most similarly assessed concept between the groups was detailed customer needs analysis, and the link to quality that was appraised as 100% by both groups. This means that the quality of a product or software development project can potentially be influenced by successful execution of detailed customer needs analysis. The biggest difference in the appraisals could be found in the relation between priorities for product features and delay. These links were weak in the FEI interviews, but they were among the strongest links in the software interviews. This indicates that the prioritization of software requirements is perceived to have a strong effect on the schedule of software projects, whereas in the traditional industrial environment, the number of product features is often smaller and more stable than in the case of software requirements, and the production time is easier to predict up front. Another difference can be seen between early sharp definition and priorities for product features. This link was relatively strong in software development and mostly ignored in the FEI interviews. In software development, the early definition of
software is a prerequisite of a successful prioritization of software requirements, which partly explains the relatively strong link in the software map.

At conceptual level, the most differing concept linkages involved the role of preliminary technology assessment. For example, the link from preliminary technology assessment to quality was stronger in software development than in traditional product development. In this study, quality was measured by whether the final product or software satisfied the customer needs. By highlighting the importance of technology assessment in software development, the data confirms existing literature (Ojiako et al., 2008). According to Ojiako et al., the ability to deliver IT project solutions with newer technology influences the perception of customers on whether a project is successful or not. On the basis of the findings, the significance of preliminary technology assessment varies strongly between these industries. The interviewees’ background, perceptions and experience have a strong influence on how technology assessment and the relationships between the concepts are understood. E.g. in traditional product development, technology assessment can be understood as assessing the technology to enable processes and production instead of evaluating the technology in a final product.

![Diagram](https://via.placeholder.com/150)

Figure 13: Combined causal map from experts of the FEI in the traditional industrial environment
5.3 Problems of requirements elicitation

The objective of Publication 3 was to investigate the challenges and problems in the elicitation of software requirements, and what kinds of effects these problems might have on the entire software development project. Publication 3 presents a model for aggregating these challenges and problems faced by requirements elicitation, which are depicted in Table 6. As a result of the study, 8 key problem categories were formulated, with altogether 28 subcategories. The key categories constructed during axial and selective coding were: 1) human constraints, 2) poor communication, 3) inadequate processes, 4) volatility, 5) lack of resources, 6) management, 7) distributed software development, and 8) challenges in cooperation. The problems under the above categories include, among others, problems in requirements engineering processes, shortage of resources, cognitive and behavioral problems, inadequate methods and techniques, and changing understanding of requirements.

Table 6: Problems and challenges in the elicitation of software requirements

<table>
<thead>
<tr>
<th>Human constraints</th>
<th>Cognitive and behavioral constraints</th>
<th>Stakeholders’ understanding</th>
<th>Analysts’ understanding</th>
<th>Poor communication</th>
<th>Linguistic problems</th>
<th>Communication channels</th>
<th>The “user and developer” syndrome</th>
<th>Participation of multiple persons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inadequate processes</td>
<td>Defining the scope</td>
<td>Methods and techniques</td>
<td>Documentation</td>
<td>Complexity of requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Volatility</td>
<td>New/changing requirements</td>
<td>Changing understanding</td>
<td>Changing priorities</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lack of resources</td>
<td>Budget</td>
<td>Schedule</td>
<td>Staff members</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Management</td>
<td>Requirements management</td>
<td>Project management</td>
<td>Support of executive management</td>
<td>Organizational issues</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distributed software development</td>
<td>Cultural diversity</td>
<td>Inadequate communication</td>
<td>Knowledge management</td>
<td>Time difference</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Challenges in cooperation</td>
<td>Partnering</td>
<td>Subcontracting</td>
<td>Outsourcing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Problems with poor communication were strongly emphasized during the literature review. During the open coding phase of the study, altogether 36 codes or elements concerning communication obstacles between users, customers, analysts or other system stakeholders were discovered. The emphasis on communication problems facing the requirements elicitation was a predictable result, because there have been numerous studies (e.g. Valenti et al., 1998; Coughlan and Macredie, 2002; Davey and Cope, 2008; Ovaska, 2009) on different types of problems in software development, and most of them are a result of failure in communication. The principal reason for communication problems lies in the fact that requirements elicitation is very much a behavioral process, where human and organizational factors have an important impact on the design (Coughlan and Macredie, 2002). In addition, the study revealed that another major challenge with requirements is constant change. The software development team has to be prepared for changes in the scope of the software
project, requirements, priorities, and/or understanding of requirements at any time. By responding rapidly to modifications of any kind, the team can provide timely customer value for new software products.

The problems and challenges of the requirements elicitation presented in the model may lead to poor requirements, insufficient quality of requirements documents, or unsatisfactory or unacceptable software by users and/or customers. Further, poor requirements elicitation can cause project failures, e.g. cancellation of software development, overruns on budget and schedule, or development of a system that has high maintenance costs, or undergoes frequent changes. Hence, it is necessary to describe and evaluate existing challenges and problems in the area of requirements elicitation in order to make improvements into requirements development practices and processes.

5.4 Assessing and improving the front end activities

Publication 4 reports an initial state assessment of requirements engineering practices in a small software company, along with an outline to improve them. The study started from a company’s need to establish a solid basis for process improvement, and to introduce systematic working practices in the area of requirements engineering. To address the need to introduce tangible improvements as quickly as possible, the company started to evaluate the used practices, techniques and tools to identify problems, development needs, and alternatives for the current way of practice. The requirement practices were approached by discussing them as five different activities: requirements elicitation, analysis, documentation, validation, and management. The perceived RE practices in the company are summarized in Table 7, which also includes some recommendations for suitable basic techniques and practices, based on the literature, to complement the current practices used in the company.

Table 7: Currently used RE practices and recommendations for improvements

<table>
<thead>
<tr>
<th>RE activity</th>
<th>Currently used RE practices</th>
<th>Suitable basic RE techniques and practices</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td>Interviewing</td>
<td>Identify and consult system stakeholders (Sommerville and Sawyer, 1997; Wiegars, 2003)</td>
</tr>
<tr>
<td>elicitation</td>
<td>Discussing with end-users</td>
<td>Interviews (Christel and Kang, 1992; Goguen and Linde, 1993; Kotonya and Sommerville, 1998; Maiden and Rugg, 1996)</td>
</tr>
<tr>
<td></td>
<td>Prototyping</td>
<td>Prototyping (Christel and Kang, 1992; Kotonya and Sommerville, 1998; Maiden and Rugg, 1996; Sommerville and Sawyer, 1997; Wiegars, 2003)</td>
</tr>
<tr>
<td></td>
<td>Meetings with project team</td>
<td>Requirements workshops (JAD, GSS) (Christel and Kang, 1992; Goguen and Linde, 1993; Leffingwell and Widrig, 1999; Maiden and Rugg, 1996; Turban et al., 2005; Wiegars, 2003)</td>
</tr>
<tr>
<td></td>
<td>Meetings with customer representatives</td>
<td>Brainstorming (Koen et al., 2001; Leffingwell and Widrig, 1999; Maiden and Rugg, 1996)</td>
</tr>
<tr>
<td></td>
<td>Receiving feedback and wishes</td>
<td>Questionnaires, surveys (Christel and Kang, 1992; Goguen and Linde, 1993)</td>
</tr>
<tr>
<td></td>
<td>Making demos</td>
<td>Scenarios (Kotonya and Sommerville, 1998; Maiden and Rugg, 1996; Sommerville and Sawyer, 1997)</td>
</tr>
<tr>
<td>Requirements analysis</td>
<td>Unstructured interviews</td>
<td>Discussions</td>
</tr>
<tr>
<td>----------------------</td>
<td>--------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td></td>
<td>Analysis during project meetings</td>
<td>Continuous analysis</td>
</tr>
<tr>
<td></td>
<td>Data filtering</td>
<td>Use checklists (Kotonya and Sommerville, 1998; Sommerville and Sawyer, 1997)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requirements documentation</th>
<th>Modified document templates</th>
<th>Memos of project meetings</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Define a standard document structure (Sommerville and Sawyer, 1997)</td>
<td>Use checklists (Kotonya and Sommerville, 1998; Sommerville and Sawyer, 1997)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requirements validation</th>
<th>Using pictures of user interfaces</th>
<th>Reviews of design documents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Testing prototypes</td>
<td>Feedback from colleagues</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Requirements management</th>
<th>Recording changes from customer service</th>
<th>Change management group</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Consulting customers</td>
<td>Internal decisions</td>
</tr>
<tr>
<td></td>
<td>Define change management policies (Kotonya and Sommerville, 1998; Sommerville and Sawyer, 1997)</td>
<td>Define requirements management policies (Sommerville and Sawyer, 1997)</td>
</tr>
</tbody>
</table>

The results indicated that the designers used several practices and techniques to support their RE activities in their daily work, although most of the designers were not aware of and did not have any training in RE techniques and concepts to support their work. The absence of a process was evident, the used RE practices were very person-dependent, and the designers did not have any systematic way of doing RE. First of all, the designers utilized requirements
elicitation techniques such as interviews, discussions, prototypes and project meetings to discover system requirements. Secondly, there was no shared or systematic way of analyzing requirements among the designers, but the requirements were analyzed during discussions, during project meetings, or individually before documentation. Thirdly, all the designers had utilized the design document template for requirements documentation, and they all had made some modifications to it in order to adopt it for their own purposes. In addition, the designers used several requirements validation techniques, such as prototyping, document reviews, and pictures of user interfaces. Finally, practices concerning requirements and/or change management were lacking in the company. Other problems mentioned were e.g. lack of normal meeting practices, unclear ownership of the design document, variations in design document quality and readiness, and communication issues between the design and development units.

The findings of the study suggest a need to improve the practices in all the key RE areas – elicitation, analysis, documentation, validation, and management – starting with introducing and training the practices and techniques considered suitable for the case company. Collaboration and team spirit between the development units were also found to be an essential part of the improvement effort. In addition, the extensive experience and best practices of each designer could be shared within the development team. The meeting with unit managers confirmed many of the findings in the designer interviews but also introduced a higher-level viewpoint on the process improvement effort, covering both collaboration between organizational units and the company goals for the process improvement effort.

5.5 A group method for requirements elicitation

The main goal of Publication 5 was to improve the requirements capturing process in a Finnish telecommunications company. The objective was to utilize the existing requirements elicitation and need assessment techniques, and to develop a new systematic requirements elicitation method for practical application. The new group method for requirements elicitation is presented in Table 8. The method created for the requirements elicitation has been derived from several techniques used in the fields of requirements engineering and innovation management. A document review (e.g. Turban et al., 1996) by the facilitator was used as a starting point for gathering the main requirements before the group sessions were conducted. Unstructured interviews were also conducted in order to gain insight into the system. To elicit all possible needs and software requirements, group sessions were arranged with different system stakeholders. The original AVAIN method (Kärkkäinen et al., 2003) was re-designed to suit the software requirements elicitation better. During the brainstorming phase of the group sessions, the requirements were written down on a card based on the Volere Snow Card (Robertson and Robertson, 1999), modified to suit the defined requirements elicitation sessions. Some items on the card were removed (e.g. customer satisfaction, dependencies, history), because filling in too many items might be harmful to the brainstorming phase and decrease the number of elicited requirements. The last phase of the requirements elicitation process was to produce a requirements document. The Concept of Operation document (ConOps) (IEEE Standard 1362, 1998) was used to help in determining the outputs of the requirements elicitation method.
Table 8: Group method for requirements elicitation

<table>
<thead>
<tr>
<th>INPUT</th>
<th>PROCESS PHASE</th>
<th>OUTPUT</th>
</tr>
</thead>
</table>
| - Document review  
- Unstructured interviews  
- Goals and schedule of the session  
- Planning of the session | 1. PREPARATION | - Defined goals and schedule  
- Elicitation process and tools  
- Participants |
| - Introduction of the participants  
- Purpose of the session  
- Description of the new system  
- Elicitation method | 2. INTRODUCTION | - Knowledge  
- Motivation of the participants |
| - Potential user classes of the new system | 3. IDENTIFICATION OF USER CLASSES | - The main user classes  
- Shared roles within the group |
| - System functions and application areas  
- Knowledge of experts  
- Different roles | 4. BRAINSTORMING REQUIREMENTS | - List of requirements |
| - Importance of each requirement | 5. PRIORORIZATION OF REQUIREMENTS | - Prioritized requirements (High, low, later) |
| - Other personnel involved in the system development | 6. IDENTIFICATION OF STAKEHOLDERS | - List of system stakeholders |
| - Different operating modes of the system | 7. IDENTIFICATION OF MODES OF OPERATION | - List of different modes of operation |
| - Goals of the session | 8. CONCLUSION OF THE SESSION | - Reached goals  
- Elicited requirements  
- Next phases of the process |
| - Questionnaire about the session  
- Evaluation of the process | 9. FEEDBACK | - Benefits and problems of the process  
- Further improvement areas |
| - Requirements  
- User classes  
- Other stakeholders  
- Modes of operation  

The new method was validated with three empirical case studies, and the results of the empirical requirements elicitation sessions show that the information gathering from the different stakeholders was expedited and effective with this requirements elicitation method. The method helped to form a general view of the new system, and made it more effective to analyze and prioritize the requirements. The discussion assisted in clarifying dissenting requirements, and focusing on a certain problem was considered valuable. The method also gave everyone an equal opportunity to participate. In addition, the assignment of various
roles based on different user classes and branches was considered positive. The participants felt that focusing on a certain role forced them to think of the requirements from the point of view of that role, which expanded the variety of the requirements. They also felt that the method helped to concentrate on relevant issues, and the session corresponded with the invested effort. The participants of the sessions also considered that they would use the developed method again in similar tasks, and they would also recommend it to others in their organization.

5.6 A GSS process for requirements definition

The objective of Publication 6 was to improve the requirements process in order to capture and analyze the requirements needed in the later phases of software development. The study presents a practical group support system process for requirements definition, clarifies the advantages and drawbacks of the developed process, and summarizes the features of the developed GSS process which can reduce the problems related to requirements engineering.

The findings reported in Publication 6 are based on the experiences of a software development case carried out in the GDSS laboratory at Lappeenranta University of Technology. The problem owner was a software development team in a Finnish telecommunications company, who wanted to make their requirements definition process more efficient and systematic. The team had utilized some traditional requirements elicitation techniques, such as interviews and requirements workshops, and they now wanted to test the possibility of the GSS to support requirements elicitation and analysis. The authors of the article designed a GSS process together with the company representatives and a GSS facilitator. The developed process aims at supporting the gathering, organizing and analyzing of requirements information effectively, and helping the participants to concentrate on the essential tasks. The process consists of separate phases that include clarification of the objectives for the session, brainstorming of the requirements for the software, grouping and prioritization of the clarified requirements, and a more detailed analysis of the most important or most diverging requirements. The designed GSS process for requirements definition is presented in Table 9.

The results of the study indicate that the GSS-supported requirements definition was expedited and more effective compared with the traditional requirements workshops conducted earlier in the case company. The GSS process helped to collect and prioritize a large number of requirements simultaneously from the development team efficiently. The process also supported the recognition of requirements with high priorities or dissenting requirements, which drove the discussion towards the most relevant issues. All the respondents agreed that the GSS was successful in providing every participant with an equal opportunity to participate and express their expertise and opinions during the GSS process. In addition, the company participants considered the session to have been successful and expressed satisfaction with the results, and felt that the objectives of the session had been successfully achieved. All the participants agreed that the results of the session were worth the effort spent. As a conclusion, it can be stated that the GSS offers some benefits which can reduce the common problems in requirements engineering. Table 10 summarizes the features of the developed GSS process which can reduce the problems related to RE.
Table 9: GSS-supported requirements definition process

<table>
<thead>
<tr>
<th>INPUT</th>
<th>PROCESS PHASE</th>
<th>OUTPUT</th>
</tr>
</thead>
</table>
| - Goals of the meeting  
- Planning of the process  
- Participant candidates | 1. PRE-PLANNING | - Adjusted goals and the agenda  
- Participants  
- Pre-session information |
| - Purpose of the meeting  
- Introduction of the agenda | 2. INTRODUCTION | - Activation of the participants  
- Assigned user roles |
| - The goal: obtaining all the important requirements | 3. REQUIREMENTS BRAINSTORMING | - List of requirements |
| - Additional and clarifying comments on the requirements from the specialists who contributed the requirements  
- Verbal discussions | 4. REQUIREMENTS CLARIFICATION & SPECIFICATION | - List of requirements in a well-specified form  
- A basis for mutual understanding of the requirements |
| - Specialists’ verbal and written opinions and arguments  
- A suggestion of categories for the participants | 5. CATEGORIZING & COMMENTING THE REQUIREMENTS | - Categorized list of requirements  
- Additional comments on the requirements |
| - Five lists of requirements  
- Verbal discussion on voting results; re-voting if necessary | 6. PRIORITIZATION OF THE REQUIREMENTS | - Five prioritized lists of categorized requirements |
| - Participants’ verbal opinions about the results  
- Agreement on further action  
- Evaluation of the GSS process | 7. EVALUATION OF THE RESULTS | - Document about the results and action plan  
- Benefits, problems and development needs of the process |
| - The results submitted to selected managers for evaluation | 8. POST-SESSION EVALUATION OF THE RESULTS | - Managerial level opinions on the results |
Table 10: Summary of the benefits of the developed GSS process

<table>
<thead>
<tr>
<th>Problems of RE</th>
<th>Reducing the problems with GSS</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Problems in defining the scope of the system</td>
<td>- Recognizing important and dissenting opinions quickly</td>
</tr>
<tr>
<td></td>
<td>- Voting focuses the discussion</td>
</tr>
<tr>
<td>- Lack of a systematic process</td>
<td>- The GSS process keeps the group on track and helps them to avoid diversions</td>
</tr>
<tr>
<td></td>
<td>- Facilitation of groups is easier with the GSS tools</td>
</tr>
<tr>
<td>- Poor communication between people</td>
<td>- More open communication with the GSS</td>
</tr>
<tr>
<td></td>
<td>- Focus on the content rather than the contributor</td>
</tr>
<tr>
<td>- Lack of appropriate knowledge and shared understanding</td>
<td>- Enhances the sharing of knowledge</td>
</tr>
<tr>
<td></td>
<td>- Finding out the reasons for conflicting views</td>
</tr>
<tr>
<td></td>
<td>- Access to external information and databases</td>
</tr>
<tr>
<td>- Complex pattern of interaction among users and analysts in defining requirements</td>
<td>- An equal opportunity to participate and express opinions</td>
</tr>
<tr>
<td></td>
<td>- Rapid recognition of conflicting and important opinions</td>
</tr>
<tr>
<td>- Inappropriate, incomplete or inaccurate documentation</td>
<td>- Better documentation: records comments and votes automatically</td>
</tr>
<tr>
<td></td>
<td>- Instantly available meeting records, and records of past RE-meetings</td>
</tr>
<tr>
<td>- Problems in dealing with the volatile nature of requirements</td>
<td>- The GSS process speeds up the requirements definition</td>
</tr>
<tr>
<td></td>
<td>- The short development circle decreases the requirement changes</td>
</tr>
<tr>
<td>- Poor management of people or resources in the context of continuously changing requirements</td>
<td>- More efficient RE-phase; simultaneous, parallel collection and structuring of requirements from different experts</td>
</tr>
</tbody>
</table>
6 Discussion and conclusions

This section discusses the theoretical contribution, and practical and managerial implications of the study. The overall contribution of this doctoral thesis is formulated in the individual publications and by combining the results. In addition, limitations and future research directions are disclosed.

6.1 Theoretical contribution

According to Easterby-Smith et al. (2008), theoretical contributions may include the discovery of new ideas, the invention of new procedures and methods, the replication of existing studies in new contexts, or the application of new theoretical perspectives to existing research questions. The theoretical contribution of this study can be divided into three areas: 1) processes and concepts, 2) challenges and development needs, and 3) means and methods for the front end activities of software development.

Processes and concepts

The research of the front end of innovation and requirements engineering in the field of software development has realized the opportunities for overall innovation process improvements by focusing on improving the front end activities of the innovation process. This thesis extends the theoretical knowledge between these two fields of research. Further, this study opens up a new strand for academic discussion by connecting these two approaches - RE and FEI. Previously, Wallin et al. (2002) have studied the possibilities to integrate business decision models and software development life-cycle models, by combining Cooper’s State-Gate model (Cooper, 2001) and three software development life-cycle models. Gassman et al. (2006) have investigated whether analogies from the Extreme Programming (XP) approach for software development can be drawn for managing the FEI to enable successful integration of the customer. In addition, Nambisan and Wilemon (2000) have studied the potentials for cross-domain knowledge sharing between software development and new product development. However, the earlier studies have mainly concentrated on describing and comparing innovation process models as a whole, whereas this study explores the potential for learning between the front end activities of product and software innovation processes.

Publication 1 addresses the first research question: “What are the similarities and differences between the concepts front end of innovation and requirements engineering?” The study reveals that there exist several similarities and differences between the FEI and RE. The definitions and vocabulary differ from each other considerably, although the activities and tasks have a certain analogy. In addition, Publication 1 presents a new framework for managing the front end of software innovation. The framework is mainly based on the new concept development model (Koen et al., 2002), and the spiral model of requirements engineering (Kotonya and Sommerville, 1998). The new approach suggests new concepts and activities from the FEI that could be applied in the RE processes in order to support and improve the early phases of software development. In addition, this approach brings business and innovation perspectives into software development by integrating the elements of the FEI into the processes of RE.
As some of the similarities and differences between the FEI and RE were identified in Publication 1, the second research question was formulated as “How perceptions about the front end of innovation concepts differ between product and software organizations?” The objective was to investigate managerial perceptions about the FEI concepts in product and software organizations. Thus, Publication 2 highlights the managerial perceptions of the similarities and differences in the concept of the front end of innovation between the software industry and the traditional industrial sector. As a result, the most similarly assessed concept was that the quality of a product or software development project can be most often influenced by successful execution of detailed customer needs analysis. By investigating how and to what extent the perceptions and significance of the front end activities between product and software organizations differ makes it possible to transfer the knowledge and best practices from the FEI into RE and vice versa. In addition, the intersection between the FEI and RE, and its handling by causal cognitive mapping is a novel territory in academic research. Therefore, this study also expands the understanding of the methodology, and makes a contribution to the applications of the methodology, as well as the actual topic at hand.

Challenges and development needs
Publication 3 answers the third research question “What are the challenges and problems of requirements elicitation in software development?” Previous research has focused on describing some individual problem areas of requirements elicitation (e.g. Damian and Zowghi, 2003; Sutton, 2000; Valenti et al., 1998). However, an extensive model to aggregate the problems and challenges which affect requirements elicitation in software development has been lacking, considering that the consequences of poor requirements elicitation are commonly recognized. This study discloses the challenges and problems faced by requirements elicitation, and presents a model of how to aggregate these influencing factors. The new model highlights the problems in communication, such as linguistic problems, ineffective communication channels, a communication gap between users/customers and software developers, and participation of multiple persons in software development. In addition, the model reveals that another major challenge with requirements is constant change, and three different kinds of volatility exist when struggling with requirements. First, the requirements change, new requirements will occur or will be removed at any phase of the software development lifecycle. Secondly, the understanding of the requirements changes during the development (Ovaska et al., 2005). The third type of volatility is the changing priorities of requirements. The importance of a particular requirement may change during the development, as people often find it difficult to assign priorities during the requirements elicitation phase because they do not have a complete picture of the software requirements at that time.

The emphasis of the RE problems varies strongly between software organizations. The size of the company, the experience of the software developers, the type of the software product, the level of co-operation, and the maturity of the requirements processes influence the problems that software organizations have to struggle with. According to Hansen et al. (2008), a relatively small amount of the literature has focused on advancing a theoretical or empirical understanding of how requirements are discovered, defined, negotiated, and managed within organizations and why these processes are so difficult. Thus, Publication 4 extends the approach, presented in Publication 3, by investigating used practices and development needs in a small software company, covering all the RE activities, i.e. requirements elicitation, analysis, documentation, validation, and management.
Means and methods
A number of techniques and methods (e.g. Goguen and Linde, 1993; Kotonya and Sommerville, 1998; Robertson and Robertson, 1999; Kärkkäinen et al., 2001, 2003; Davis et al., 2006,) have been developed to assist the difficult process of requirements elicitation, and to overcome many of the problems described in Publications 3 and 4. However, many earlier studies have focused on describing and comparing single requirements elicitation methods, but there is no systematic process which would bring together the benefits and best practices of different techniques used in the fields of innovation management and software engineering. This study focuses especially on group methods for supporting requirements elicitation, as the importance of team work in generating innovations has been widely recognized (e.g. Trott, 2005; Elfvengren, 2008). Thus, Publications 5 and 6 address the fifth research question “How can group methods support the front end activities of software development?”

Publication 5 presents a new systematic group method for requirements elicitation, developed on the basis of evaluations on customer need assessment and requirements elicitation techniques utilized in the fields of innovation management and RE. The new requirements elicitation method applies several techniques and tools, such as document reviews (e.g. Turban et al., 1996), unstructured interviews, group sessions, the AVAIN method (Kärkkäinen et al., 2003), brainstorming, the Volere Snow Card (Robertson and Robertson, 1999), and the Concept of Operation document (ConOps) (IEEE Standard 1362-1998).

Publication 6 introduces a new practical GSS process for requirements elicitation and analysis by utilizing the characteristics and benefits of group support systems. The study contributes to the earlier GSS and requirements engineering literature, as rather few practical industrial experiences have been reported on the utilization of the GSS in the early phases of software development. The developed GSS process aims at supporting the gathering, organizing and analyzing of customer needs and software requirements effectively. The process consists of separate phases that include clarification of the objectives for the session, brainstorming of the requirements for the software, grouping and prioritization of the clarified requirements, and a more detailed analysis of the most important or most diverging requirements. In addition, Publication 6 expands the knowledge about the benefits and restrictions of the GSS, and the features which can reduce the problems related to RE.

6.2 Practical and managerial implications
Publication 1 presents a new approach for managing the front end of software innovation. The new approach suggests new concepts and activities from the FEI that software teams can apply in software development projects in order to support and improve the requirements engineering processes. Especially the business plan should be one important deliverable from the FEI controlling the changes in business priorities and satisfying customer needs. Furthermore, the idea generation and assessment activities should be an integral part of RE processes in order to guarantee that all ideas and concepts reach software development. In addition, the study encourages software organizations to learn from other disciplines, besides requirements engineering. For instance, the traditional industrial sector could provide several viewpoints and new practices to the current ways of action in the front end activities of software development.
Publication 2 provides viewpoints to managers and team members of software development projects about the most important factors in their front end activities that influence the success of their product or software development projects. The study emphasizes particularly the importance of detailed customer needs analysis in both product and software development organizations. In addition, Publication 2 highlights the fact that even in the same organization or project team, the views of different experts can differ concerning the features of a successful product or software development project. Thus, managers may highlight the importance of costs as the major outcome measure for project success, but developers or other team members may be mainly concerned with quality issues. Even though this is merely a difference in perceptions, these differences are often realized in a very concrete way in unsuccessful communication of project members about the important factors affecting the project outcomes and finally the end results of the project. Thus, as a result, to succeed in development projects, the importance of defining and sharing the most important outcome measures for each project is emphasized.

Publication 4 addresses the fourth research question “What practices and techniques a software company uses to support their front end activities, and how these practices can be improved?” The study suggest that the first step when starting the requirements process improvements in a small software company is to assess the practices currently used in the organization, and to identify their strengths and shortcomings. As the current practices were found to be at the initial level in the case company, an interview-based approach was chosen to clarify the current practices and development needs. In the small software company, the comprehensive process assessment frameworks (e.g. CMMI or SPICE) were not considered to be usable, as they were found to be complicated and expensive to use. As a conclusion, the study provides recommendations for requirements process improvements in a small software company. The findings suggest a broad need for improving the practices in all key RE areas, starting with introducing and training the techniques and practices available to improve RE activities in the case company. In addition, the extensive experience and best practices of each designer could be shared within the development team, the collaboration and team spirit within the units needed improvement, and design quality variance needed to be reduced.

Finally, many software organizations, including the one in Publication 4, still rely on ad hoc processes and do not utilize the methods offered by software engineering in their software development, and the processes may include outdated techniques or may not take advantage of the best practices in industrial software engineering (Sommerville, 2001). According to Kotonya and Sommerville (1997), process improvement is sometimes seen simply as the introduction of new methods and techniques. Publications 5 and 6 offer practical tools and processes for managers and development teams for managing and improving the challenging activities in the front end of the software development process, especially in the collection and analysis of customer needs and software requirements.

6.3 Validity of the study

Such criteria as credibility, transferability, dependability, and confirmability are used for the evaluation of qualitative research (Hoepfl, 1997; Denzin and Lincoln, 2005), and these attributes are also applied to evaluate the validity of this study. According to Denzin and Lincoln (2005), constructivist terms, such as credibility, transferability, dependability, and confirmability replace the usual positivist criteria of internal and external validity, reliability, and objectivity.
Credibility
According to Silverman (2005), credibility means the extent to which any research claim has been shown to be based on evidence. Thus, the results of Publications 1 and 3 are mainly based on a literature review and the conceptual approach, and these publications have been used as the basis for empirical research. The results of the other publications are based on empirical data. In Publications 2 and 4, the main data sources are interviews conducted in the case companies. In Publication 2, cognitive causal maps are used to collect and analyze the data and perceptions of the interviewees. In Publication 4, the data was collected by interviewing all the employees with designer responsibilities, and a manager meeting was conducted afterwards. All the interviews were audio-recorded, and all the interview recordings were listened to and discussions of closer interest, i.e. RE practices, were transcribed. In Publications 5 and 6, the main data sources are interviews, observations of requirements sessions, post-session feedback surveys and group discussions. The requirements sessions were developed by the researchers and company representatives together, and the meetings were recorded in minutes. The facilities of the GDSS laboratory enabled electronic documentation of the session progress, requirements and post-session feedback for Publication 6, whereas these items were recorded manually for Publication 5.

Transferability
As the research findings are based on a sample of quite few case situations, the results cannot be generalized. The conclusions must be viewed more as probable as an absolute truth. In qualitative research, studies are normally based on a limited amount of research data, for example on one or a few studied case situations. The reason for this is usually limited time and resources, because understanding one case situation profoundly is very laborious and time consuming. In the constructive approach, in Publications 5 and 6, the starting point of the research was a well specified real problem situation, and the research environment was a real situation. This sets some restrictions to the sample size. The great challenge of this kind of research is to make conclusions which can also be reliably generalized to other situations in a similar environment. Such generalization of research results is hard and seldom done without any gaps. However, in this study there was no intention to generalize the research results to any company or industry setting. In this case it is not possible or necessary to prove that the research findings presented in the study are general and suitable to any possible settings. The findings and conclusions are treated as probable or recommendable on the basis of the case experiences in this thesis.

Dependability
Dependability can be seen as a quality control of the research that evaluates the quality of the research process. According to Eriksson and Kovalainen (2008), dependability is concerned with researcher’s responsibility for offering information that the research process has been logical, traceable and documented. This study is based on various research articles published in international journals and conferences. The review process and the publication channels of these articles are described in detail in Table 11.

Confirmability
The confirmability of a research evaluates the sufficiency of the research process, and whether the research findings flow from the data. This study is comprised of six separate publications, each having distinctive objectives and data material within the scope of this thesis. First, the research process was started with problems in a case company related to the difficult process of requirements elicitation, and several methods and tools were evaluated to support this phase. Second, as the challenges in traditional and manual elicitation techniques were discovered, the focus of this thesis moved to the utilization of group support systems as
a supportive tool for requirement elicitation and analysis. Third, a conceptual framework was developed, covering the challenges and problems related to the requirements elicitation phase of software development. Fourth, in order to meet the challenges related to the early phases of software development, and to discover new ideas, practices and concepts to support the difficult RE process, the terms, activities, processes and managerial perceptions between the concepts FEI and RE were evaluated. Fifth, the study was extended to the assessment of all the RE activities, i.e. requirements elicitation, analysis, documentation, validation, and management in a small software company.

Table 11: The review process of the publications in part II of the thesis

<table>
<thead>
<tr>
<th>Publication</th>
<th>Review process and channels</th>
</tr>
</thead>
</table>
| 1. The Front End of Innovation in Software Product Development               | 1) Accepted for the International Working Seminar on Production Economics on the basis of the abstract  
2) First version presented in the seminar by the author                      
3) Final version improved on the basis of the feedback in the seminar         |
| 2. Utilizing Front-End-of-Innovation Concepts in Software Development        | 1) Accepted for the international ISPIM conference (International Society for Professional Innovation Management) on the basis of the abstract  
2) Presented in the ISPIM conference by the author                           
3) Review process for the journal                                             
4) Final version published in the Journal of Industrial Management & Data Systems |
| 3. A Model for Aggregating the Challenges and Problems of Requirements Elicitation in Software Development | 1) Preliminarily accepted for the international conference on Computers and Industrial Engineering (CIE) on the basis of the abstract  
2) Review process of the full paper                                           
3) Accepted for the CIE conference on the basis of the full paper             
4) Presented in the CIE conference by the author                             |
| 4. Assessing and Improving the Front End Activities of Software Development: Experiences in a Software House | 1) Accepted for the ISPIM conference on the basis of the abstract  
2) Presented in the ISPIM conference by the author                           
3) Review process for the journal                                             
4) A revised version accepted for publication in the International Journal of Business Information Systems |
| 5. The Front End of Innovation – A Group Method for the Elicitation of Software Requirements | 1) Accepted for the Nordic Innovation Research (NIR) conference on the basis of the full paper  
2) Presented in the NIR conference by the author                            
3) Rewarded for one of the two best papers in the conference                  
4) Review process for the journal                                             
5) Final version accepted for publication in the International Journal of Innovation and Learning |
| 6. A Group Support System Process for the Definition of Software Requirements | 1) Accepted for the ISPIM conference on the basis of the abstract  
2) Presented in the ISPIM conference by the author                           
3) Review process for the journal                                             
4) Final version accepted for publication in the International Journal of Innovation and Learning |
6.4 Limitations of the study and further research

Limitations
As this thesis comprises several individual publications, and utilizes different methodological approaches and methods, the limitations of the study vary between the publications. In Publication 1, a new theoretical framework for managing the front end of software innovation is presented. However, a limited section of the literature is covered, and the framework has been developed on the basis of theoretical and analytical work, influenced by the opinions and perceptions of the authors. Publications 2 and 4 have some limitations inherent in the case study approach. The study is based on a limited number of interviews, and therefore the results cannot be fully generalized to other companies or fields of industry. Furthermore, in Publication 2, the objective was to study the beliefs and perceptions of selected experts about the concepts of the FEI, and therefore the conclusions must be viewed more as a probable than an absolute truth. In addition, the selection of a purely academic interview group to represent the traditional industrial sector has some limitations. In Publication 3, the problems and challenges of requirement elicitation are aggregated into the model, and the validation of the model has been done in two software organizations. Therefore, further validation of the model in other software organizations is needed in order to verify these results. Publications 5 and 6 present new constructs to improve the process of requirements elicitation and analysis. As the research findings are based on a limited number of case situations, the results cannot be generalized.

Further research
The developed processes for requirements definition need further validation of practical usability in different software companies and organizations, and with other software applications. Practical experiences provide the best guidance for further development of these processes. The author has also considered further studies on how the front end activities of software development can be supported by scenario processes utilizing GSS. Scenarios are increasingly used in the software industry throughout all phases of development, including requirements definition.

Today, the Finnish universities enjoy a dynamic phase, where processes and supportive tools are changing, and new ones are introduced. Especially, several new software systems are implemented to support research and education. However, some of these systems have been found to be challenging from the users’ and customers’ point of view. Therefore, an interesting subject for further research would be the utilization of the methods, processes and tools presented in this thesis, to collect and analyze the needs of the actual end users of these and future systems.

There exists considerable potential for knowledge sharing between the FEI and RE, especially by benchmarking the activities, techniques, tools and technologies used in these fields. For instance, the activities of the front end of innovation do not include requirements management. However, customers’ needs and requirements for a new product change, especially in the front end phase of innovation. The practices and tools of requirements management in software development could provide viewpoints on managing the ideas and customers’ needs in the FEI. In addition, benchmarking the concept ‘requirements management’ with product data management (PDM) may provide several similarities and lessons to learn between RM and PDM. PDM focuses on managing and tracking the creation, change and archives of all product-related information.
This thesis concentrates on comparing the front end of innovation concepts with the software engineering approach, especially with requirements engineering. However, there exist several other approaches to software development, e.g. agile methods such as XP, and Soft Systems Methodology (Checkland and Scholes, 1990). Therefore, an interesting area for further research would be to investigate how and to what extent these methodologies are connected to the concepts of the front end of innovation.
REFERENCES


