OULU 2013
A 618

ACTA UNIVERSITATIS OULUENSIS


*Pilar Rodríguez*

# COMBINING LEAN THINKING AND AGILE SOFTWARE DEVELOPMENT

## *HOW DO SOFTWARE-INTENSIVE COMPANIES USE THEM IN PRACTICE?*

*UNIVERSITY OF OULU GRADUATE SCHOOL;*
*UNIVERSITY OF OULU,*
*FACULTY OF SCIENCE,*
*DEPARTMENT OF INFORMATION PROCESSING SCIENCE*

ACTA UNIVERSITATIS OULUENSIS
A Scientiae Rerum Naturalium 618

*PILAR RODRÍGUEZ*

# COMBINING LEAN THINKING AND AGILE SOFTWARE DEVELOPMENT

How do software-intensive companies use them in practice?

Academic dissertation to be presented with the assent of the Doctoral Training Committee of Technology and Natural Sciences of the University of Oulu for public defence in Auditorium IT116, Linnanmaa, on 16 December 2013, at 12 noon

UNIVERSITY OF OULU, OULU 2013

Acta Univ. Oul. A 618, 2013

Supervised by
Professor Markku Oivo
Professor Juan Garbajosa

Reviewed by
Professor Pekka Abrahamsson
Doctor Hakan Erdogmus

Opponent
Professor Kari Smolander

ISBN 978-952-62-0331-7 (Paperback)
ISBN 978-952-62-0332-4 (PDF)

ISSN 0355-3191 (Printed)
ISSN 1796-220X (Online)

Cover Design
Raimo Ahonen

JUVENES PRINT
TAMPERE 2013

**Rodríguez, Pilar, Combining Lean thinking and Agile Software Development. How do software-intensive companies use them in practice?**

University of Oulu Graduate School; University of Oulu, Faculty of Science, Department of Information Processing Science

*Acta Univ. Oul. A 618, 2013*

University of Oulu, P.O. Box 8000, FI-90014 University of Oulu, Finland

***Abstract***

Software engineering is advancing according to market needs. Consequently, software development methods that initially caused controversies such as Agile, and more recently Lean, are increasingly being adopted by the software industry. Particularly, Lean Software Development, which was initially regarded as one of the Agile methods, is acquiring an identity of its own as a means to scale Agile. However, Lean thinking is still open to interpretation in the domain of software development, which differs fundamentally from the manufacturing domain where Lean originally emerged. Specific issues such as the essence of Lean Software Development, the compatibility of Lean and Agile and the best combination of them are not properly understood.

This dissertation addresses Lean thinking and its combination with Agile in the field of software development, by providing empirical evidence on how software-intensive organisations use them in practice. The research was performed in four phases. First, the relevant literature was analysed to identify research opportunities. Second, a survey strategy was used to investigate status and trends in the adoption of Agile and Lean. The third phase explored in detail how Agile and Lean are combined in practice, by conducting case studies on two large-scale, industry-leading companies that were transforming their processes from Agile Software Development into Lean Software Development. Finally, in the fourth phase, the results of the previous research phases were synthetized to draw conclusions and outline implications.

The results of the study confirmed the interest of practitioners in using a combination of Agile and Lean. Unlike in manufacturing, the borders of Agile and Lean are not clearly defined in the software domain. The results provided evidence of numerous compatibilities between Agile and Lean in software development. Generally, the use of Agile methods at a prescriptive level is guided by Lean principles. However, Lean thinking also brings new practical elements to software development processes, such as Kanban, work-in-progress limits, a ‘pull’ and ‘less waste’-oriented culture and an extended emphasis on transparency and collaborative development. The results showed the fundamental importance of practices that enable quick feedback, fast learning and adaptation.

*Keywords:* agile software development, case study, content analysis, exploratory research, le-agile, lean software development, lean thinking, organizing vision, software development, survey

**Rodríguez, Pilar, Lean-ajattelun ja ketterien ohjelmistokehitysmenetelmien yhdistäminen. Kuinka ohjelmistoalan yritykset käyttävät niitä käytännössä?**
Oulun yliopiston tutkijakoulu; Oulun yliopisto, Luonnontieteellinen tiedekunta, Tietojenkäsittelytieteiden laitos
*Acta Univ. Oul. A 618, 2013*
Oulun yliopisto, PL 8000, 90014 Oulun yliopisto

### *Tiivistelmä*

Ohjelmistotuotanto kehittyy markkinoiden tarpeiden mukaisesti. Aiemmin kiisteltyjä ketteriä menetelmiä, ja nykyään myös Lean-menetelmiä sovelletaan yhä useammin ohjelmistoteollisuudessa. Lean-menetelmiin perustuva Lean-ohjelmistokehitys erottuu selkeämmin välineenä laajentaa ketterien menetelmien käyttöä. Lean on yhä monitulkintainen ohjelmistotuotannossa, joka poikkeaa teollisuustuotannosta, josta Lean on peräisin. Lean-ohjelmistokehitystä, Lean- ja ketterien menetelmien yhteensopivuutta ja niiden parasta yhdistelmää ei vielä ymmärretä riittävän hyvin.

Tämä väitöskirja käsittelee Lean-menetelmien yhdistämistä ketteriin menetelmiin ohjelmistotuotannossa. Tutkimus esittää kokemusperäistä tietoa, kuinka näitä menetelmiä käytetään ohjelmisto-alan organisaatioissa. Tutkimus oli nelivaiheinen. Aluksi tutkimusmahdollisuudet kartoitettiin tutkimalla aiheeseen liittyvää kirjallisuutta. Seuraavaksi tutkittiin kyselytutkimuksen avulla Lean- ja ketterien menetelmien käyttämisen nykytilaa ja kehitystä. Kolmannessa vaiheessa tapaustutkimuksilla selvitettiin Lean- ja ketterien menetelmien yhdistämistä käytännössä. Tapaustutkimuksia tehtiin kahdessa suuressa yrityksessä, jotka olivat muuttamassa prosessejaan ketteristä menetelmistä kohti Lean-ohjelmistokehitystä. Lopuksi aiemmat tutkimusvaiheet yhdistettiin johtopäätöksiä ja vaikutusten hahmottamista varten.

Tutkimuksen tulokset vahvistavat Lean- ja ketterien menetelmien yhdistämisen kiinnostavan ohjelmistotuotannonharjoittajia. Lean- ja ketterien menetelmien rajat eivät ole selkeästi määriteltyjä ohjelmistotuotannossa. Tulokset tukevat käsitystä Lean- ja ketterien menetelmien yhteensopivuudesta. Lean ohjaa yleisellä tasolla ketterien menetelmien käyttöä. Lean tuo kuitenkin myös uusia elementtejä ohjelmistotuotantoon, kuten Kanban-menetelmän, keskeneräisen työn rajoittamisen, kysyntään perustuvan ’pull’-menetelmän ja turhan työn vähentämistä tavoittelevan ’less-waste’-työkulttuurin. Lean-ajattelu myös lisää painotusta läpinäkyvyyteen ja yhteistyöhön.

*Asiasanat:* : lean-ohjelmistokehitys, eksploratiivinen tutkimus, ketterät menetelmät, lean-ajattelu, ohjelmistokehitys, sisältöanalyysi, tapaustutkimus

*A mis padres, porque siempre estáis cerca independientemente de los kilómetros que nos separen.*

# Preface

After four years of extensive work, I am sitting in front of my computer, writing the preface to my PhD thesis. It is amazing how quickly time flies. I feel as if it was only yesterday that I moved to Finland to begin my PhD studies. My mind is filled with many memories. Moving to a new country is a life-changing experience that extends beyond the pursuit of a PhD thesis. In Oulu, I have had the privilege of learning from leading researchers, working with world-renowned companies, meeting lovely people who have become close friends and enjoying the uniqueness of the Finnish nature.

First, I would like to sincerely thank my supervisors, Professors Markku Oivo and Juan Garbajosa, who assisted and motivated me to complete this thesis. It all started when I knocked on the door to Prof. Garbajosa's office after completing my bachelor degree. After being a member of his research group during my master studies, he advised me to visit Oulu to continue my work on the Flexi project. I moved to Oulu in March 2009, and what was initially to be a period of six months led to the beginning of this PhD journey. I am thankful to Prof. Garbajosa for giving me the opportunity to enter the academic world and for encouraging me to make such a transcendental decision. In Oulu, I met Prof. Markku Oivo, who has, from the very beginning, been very supportive during this journey. It has been a pleasure to work with someone who has taught me patiently, enjoyed my successes and encouraged me to face my challenges positively. I thank you both for your excellent guidance during this process, for seeing to it that my research remained focused, for ensuring that I was headed in the right direction and for giving me the freedom and autonomy to learn by myself.

The Department of Information Processing Science has provided a first-class environment that has enabled me to learn and grow as a researcher. I truly appreciate the department's efforts to remain up to date with regard to its courses and to invite leading researchers from around the world to deliver lectures. It has been a privilege to undertake PhD studies in such an environment.

I would also like to thank all my wonderful colleagues, who made me feel at home during this journey. In particular, I would like to thank all my colleagues and friends of the M-Group, and especially my co-workers in the Cloud Software Program: Burak, Harri, Kari, Markku, Mika, Ovais, Pasi, Teemu and Vladimir. Research is a teamwork activity, and this thesis would not exist without you. I hope to continue our collaboration in the future. Thanks, Pasi, for letting me learn

from your experience in software assessments; I could not have had a better master for this topic. Thanks, Sanja and Anna; you have been my guardian angels in Oulu. Thanks for being such good friends and for inviting me into your families. Thanks, Jouni, for patiently supporting my crises regarding how to properly apply research methods. This thesis would not have been possible without your support in that area and the 'sisu' recipe. Thanks, Burak, Vladimir, Harri, Ayse, Nebojsa Davide and Markus, for all the academic discussions we had and the social activities outside the working place that enabled me to learn more about you and your countries. I am also very thankful to my colleagues and friends from UPM. Agustín, Jennifer, Jessica, Pedro and many others who have been always interested in my research and have constantly animated me in this journey.

The Cloud Software Program provided an excellent environment for collaborating with the Finnish software development industry. During the course of the project, I had the opportunity to collaborate with expert practitioners and researchers. I would like to thank all my Cloud Software colleagues for the discussions, debates and shared understandings, which had a significant and definitive impact on my work. Conducting empirical research in industrial settings is an exceedingly demanding activity for both researchers and practitioners. Therefore, I would especially like to thank Kirsi Mikkonen from Ericsson R&D Finland and Jari Partanen from Elektrobit, as well as their teams, for kindly opening the doors of their companies and supporting our studies.

I would respectfully like to thank Prof. Pekka Abrahamsson from the Free University of Bolzano and Dr Hakan Erdogmus from Carnegie Mellon University, who dedicatedly pre-examined my thesis. Their comments and suggestions not only encouraged me to improve the quality of the thesis, but will definitely impact my future research on the topic.

I would not have been able to conduct this research without financial support. I would like to acknowledge the research project Cloud Software Program of DIGILE (Finnish Strategic Centre for Science, Technology and Innovation in the field of ICT and digital business) and the one-year funded position at the Graduate School of Software Engineering (both funded by TEKES), which extensively supported my research. Moreover, I would like to acknowledge the financial support that I received from the University of Oulu Graduate School's travel grants, as well as the Tauno Tönning Research Foundation, Oulu University Scholarship Foundation and Nokia Foundation grants.

During this journey I have been very lucky to have a large support group of friends. There being so many, it is not possible for me to name the individuals here but I am deeply thankful to all of them for showing me other aspects of the concept of friendship that are difficult to know if you have not had the opportunity to live abroad. I guess foreigners can more easily understand what I mean. Living abroad helped me also to realize even more the importance of my friends back home. I would like to thank them all for having trust in me and listening with enthusiasm my stories from Finland.

Finally, my warmest thanks go to my family and my parents. Muchas gracias porque vuestro permanente esfuerzo me ha permitido llegar hasta aquí. Gracias por entenderme, aconsejarme y apoyarme en mis decisiones. Gracias por vuestros ánimos, por estar siempre pendientes de mí y por hacer que los fríos inviernos finlandeses fueran mucho más cálidos después de nuestras sesiones en Skype. Y sobre todo gracias por vuestro cariño.

Thank you all for contributing to making all this possible. I feel ready to take the next step forward!

Oulu, November 2013 Pilar Rodríguez

# Abbreviations and Terminology

This section presents a list of abbreviations and terminology that are used in this dissertation. For deeper understanding of the concepts behind these terms, it is suggested to consult the References section at the end of the thesis.

| | |
|---|---|
| ASD | Agile Software Development |
| CMMI | Capability Maturity Model Integration |
| DEV | Development |
| FIPA | The Finnish Information Processing Association |
| ICT | Information and Communication Technologies |
| IS | Information Systems |
| ISO | International Standard Organisation |
| JIT | Just-In-Time |
| OICA | International Organisation of Motor Vehicle Manufacturers |
| QUAL | Qualitative |
| QUAN | Quantitative |
| RQ | Research Question |
| SW | Software |
| Software Process | Goal-oriented set of interrelated or interacting activities which transforms inputs into outputs in the context of engineering–style software development (ISO/IEC 12207) |
| Software Process Model | Abstraction and simplified representation of a software process |
| SPICE | Software Process Improvement and Capability dEtermination |
| TDD | Test Driven Development |
| TPS | Toyota Production System |
| VSM | Value Stream Mapping |
| WIP | Work in Progress |

# List of Publications

This thesis is based on the following original papers, which are referred to in the text by their Roman numerals.

I Rohunen A, Rodríguez P, Kuvaja P, Krzanik L & Markkula J (2010) Approaches to agile adoption in large settings: a comparison of the results from a literature analysis and an industrial inventory. In: Product-Focused Software Process Improvement. Springer: 77–91.

II Rodríguez P, Markkula J, Oivo M & Turula K (2012) Survey on agile and lean usage in Finnish software industry. Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement. ACM: 139–148.

III Rodríguez P, Markkula J, Oivo M & Garbajosa J (2012) Analysing the drivers of the combination of lean and agile in software development companies. In: Product-Focused Software Process Improvement. Springer: 145–159.

IV Lawrence C & Rodríguez P (2012) The interpretation and legitimization of values in Agile's organizing vision. Proceedings of the European Conference on Information Systems (ECIS). Barcelona, Spain, 10–13 June 2012.

V Rodríguez P, Mikkonen K, Kuvaja P, Oivo M & Garbajosa J (2013) Building lean thinking in a telecom software development organisation: strengths and challenges. Proceedings of the 2013 International Conference on Software and System Process. ACM: 98–107.

VI Rodríguez P, Partanen J, Kuvaja P & Oivo M (2014) Combining lean thinking and agile methods for software development. A case study of a Finnish provider of wireless embedded systems. Proceedings of the 47th Hawaii International Conference on Systems Sciences (HICSS 2014). In press.

# Table of Contents

**Abstract**
**Tiivistelmä**
**Preface** **9**
**Abbreviations and Terminology** **13**
**List of Publications** **15**
**Table of Contents** **17**
**1 Introduction** **21**
1.1 Motivation ... 23
1.2 Research Opportunities ... 27
1.3 Objective and Research Questions ... 30
1.4 Scope of the Research ... 32
1.5 Overview of the Research Design ... 33
1.6 Structure of the Thesis ... 34
**2 Background and Related Work** **37**
2.1 Evolution of Software Development Methodologies ... 37
2.2 Origins of Lean and Agile ... 39
2.3 Agile Software Development ... 45
2.4 Lean Software Development ... 48
2.4.1 Lean Software Development before the Formulation of the Agile Manifesto ... 50
2.4.2 Lean Software Development after the Formulation of the Agile Manifesto ... 50
2.4.3 Lean Thinking and its Combination with Agile Software Development ... 56
2.5 Identified Research Gaps ... 57
**3 Research Design** **61**
3.1 Overall Research Design ... 62
3.2 Phase I: Defining the Research Problem ... 63
3.2.1 Literature Analysis and Industrial Inventory ... 64
3.2.2 Cloud Software Program ... 67
3.3 Phase II: Status and Trends in Lean and Agile Software Development ... 68
3.3.1 Survey on Agile and Lean Usage in the Finnish Software Industry ... 68
3.3.2 Organising Vision of Agile Software Development ... 71

3.4 Phase III: Analysing Lean and Agile Software Development in Specific Company Cases ..... 74
3.4.1 Cases Selection Strategy ..... 75
3.4.1.1. Case A: Ericsson ..... 76
3.4.1.2. Case B: Elektrobit ..... 77
3.4.2 Data collection ..... 77
3.4.3 Data analysis ..... 79
3.5 Phase IV: Synthesising the Results ..... 80
**4 Original Contributions 85**
4.1 Paper I: Approaches to Agile adoption in large settings: a comparison of the results from a literature analysis and an industrial inventory ..... 86
4.2 Paper II: Survey on Agile and Lean usage in the Finnish software industry ..... 87
4.3 Paper III: Analysing the drivers of the combination of Lean and Agile in software development companies ..... 89
4.4 Paper IV: The interpretation and legitimisation of values in the Agile's organizing vision ..... 90
4.5 Paper V: Building Lean thinking in a Telecom software development organisation: Strengths and challenges ..... 92
4.6 Paper VI: Combining Lean thinking and Agile Methods for software development. A case study of a Finnish provider of wireless embedded systems ..... 94
**5 Findings and Discussion 97**
5.1 Research Question 1: How is Lean thinking combined with Agile methods in software development? ..... 97
5.2 Research Question 2: What are the key factors that influence the successful transformation of software organisations towards Lean and Agile methods? ..... 106
5.3 Research Question 3: What are the impacts perceived by practitioners when using a combination of Lean thinking and ASD? ..... 111
5.4 Implications for Practice ..... 112
5.5 Implications for Research ..... 113
**6 Conclusions 115**
6.1 Main Contributions ..... 115
6.2 Validity and Limitations of the Study ..... 119

6.2.1 External Validity ..... 119
6.2.2 Construct Validity ..... 120
6.2.3 Reliability ..... 121
6.2.4 Internal Validity ..... 122
6.3 Future Work ..... 123
**References** **125**
**Appendices** **133**
**Original papers** **141**

# 1 Introduction

The latest trends towards a more customer-centric, responsive, iterative and human-oriented software development are bringing up new paradigms in the field of software development processes (Boehm 2006). Software engineering has progressed according to market needs. Therefore, methods that initially caused some controversy, such as Agile[1] and more recently Lean Software Development, are increasingly being adopted by industrial practitioners (West *et al.* 2010). Accordingly, they are also receiving more and more attention from the research community, as clearly reflected in the increasing number of journals that include special issues on Agile and Lean software development[2], and in the conferences that have emerged around the topic[3] or which contain papers that are focused on these methods in their proceedings[4].

Over the past 15 years, the software development industry has been seeking for lighter weight methods to respond to changing technological, societal and environmental needs. As a prominent solution, Agile Software Development (ASD) was born. The family of ASD methods relies on a set of four values and twelve principles firstly formulated in the *Agile Manifesto* (Agile Manifesto 2001). ASD methods are based on iterative development, where solutions evolve through collaboration between customers and self-organizing cross-functional teams. In essence, ASD recognises the limitations of anticipating external business changes and provides means oriented to implement the right products quickly as a key competitive advantage (Abrahamsson *et al.* 2002). Agile methods in small software teams have been proven to be beneficial (Dybå and Dingsøyr 2008). However, scaling Agile up beyond team practices in order to achieve the same benefits in large-scale software development has been found to

[1] *Agile* when used as an adjective or as a noun in the scope of this thesis refers to Agile Software Development.

[2] E.g.: European Journal on Information Systems. Special Issue: Agile Processes in Software Development. Volume 18, Issue 4, August 2009.
Journal of Systems and Software. Special Issue: Agile Development. Volume 85, Issue 6, June 2012.
IEEE Software Magazine. Special issue: Lean Software Development. Volume 29, Issue 5, September-October 2012.

[3] E.g.: Lean Software and Systems Conference series, http://www.leanssc.org/conferences.
Conference on Lean Enterprise Software and Systems (LESS), http://less2013.org.
International Conference on Agile Software Development (XP).

[4] E. g.: International Conference on Software Engineering (ICSE).
International Symposium on Empirical Software Engineering and Measurement (ESEM).
International Conference on Software and Systems Process (ICSSP).

be challenging (Maples 2009; Turk, France and Rumpe 2002; Abrahamsson, Conboy and Wang 2009). Lean thinking (often simply Lean), which is in alignment with many Agile principles but considers a more holistic enterprise perspective, has been discerned as a means to overtake limitations when scaling ASD in large projects (Poppendieck and Poppendieck 2003; Larman and Vodde 2008; Vilkki 2010; Laanti 2012). Lean ideas initially emerged in the manufacturing industry (i.e. the automotive industry) and were one of the main success drivers of the Japanese industry after the World War II (Middleton and Sutton 2005). Lean is steeped in a philosophy of maximizing value and minimizing waste, with the purpose of "*doing more with less*" (Agarwal *et al.* 2006). Important concepts of Lean include customer value and value stream, waste reduction, workflow analysis and continuous improvement. The interest from the software-intensive industry in applying Lean has significantly grown in recent years, and Lean Software Development, which was initially regarded as one of the ASD methods (Dybå and Dingsøyr 2008), is progressively acquiring an identity of its own. Particularly in Finland, the interest in adopting Lean Software Development is evidenced by initiatives such as the Cloud Software Program (2010)[5], in which Lean Software Development constitutes the main initiative to accomplish *cheaper-faster-better* software.

However, what some have denoted as '*Leagile*' software development (Wang *et al.* 2012), in reference to the combination of Agile and Lean methods, is still an ambiguous and quite unexplored phenomenon. The essence of both approaches, whether Agile and Lean are really compatible in the software domain and, if so, how to combine the best of them are not properly understood. For example, the guest editors' introduction of a recent special issue on Lean Software Development in the IEEE Software Magazine (Ebert, Abrahamsson and Oza 2012) states that although there is a need across the ICT and software industry to learn more about Lean thinking, '*lean conferences are born, lean software books are selling, and organisations are keenly adopting lean principles', 'we still lack a coherent set of features applicable to Lean Software Development* […] *If everything is called 'Lean', and different methods from agile to project management are mixed ad hoc, confusion results both in science and practice,*

[5] *Cloud Software Program* (2010-2013, http://www.cloudsoftwareprogram.org/) is a Finnish industry-driven research program that includes 22 industrial and eight research participants. Cloud Software Program has the largest volume in terms of budget and companies involvement in the history of the IT research in Finland.

[creating] *no sustainable improvement in software development*'. In the same line, the works by Jonsson (2012), who conducted a systematic literature review in 2012, and Pernstål *et al.* (2013), who conducted a systematic mapping study regarding the topic in 2013, indicate that there is a research gap in Lean Software Development. While Lean Software Development is a promising approach, the small number of available empirical studies and the dominance of some authors make it difficult to draw reliable conclusions.

This thesis addresses Lean thinking in the field of software development and specifically its combination with ASD methods, by providing empirical evidence on how software-intensive organisations combine them in practice. A precise definition of Lean Software Development is not of foremost importance within the scope of this thesis. Lean and Agile software development are being used in the software-intensive industry, and their boundaries can be debated in an academic setting. However, in order for Lean and Agile to serve the software development business effectively, it is important (i) to understand their main elements, their relationship and their connection to other well-established concepts in software engineering, (ii) to understand what the fundamental principles underlying Lean thinking in a software development context are, as well as the principles underlying its combination with ASD and (iii) to identify the actual manifestations of the fundamentals in the form of concrete practices and tools suited to the software development domain. Once (i), (ii) and (iii) are embraced, Lean thinking in the context of software development will be better understood. Moreover, we will much better able to explain what Lean thinking and its combination with ASD have to offer to software development processes.

The rest of the chapter is structured as follows. Sections 1.1 and 1.2 delve into the reasons why Lean Software Development and its combination with Agile have potential for being an important research topic and specify the research opportunities that the dissertation will address. Next, the objective and research questions are formulated and the scope of the research is specified. Finally, the research design is introduced and the structure of the thesis is outlined.

## 1.1 Motivation

From the more general to the more concrete, the three main reasons that motivated the research are as follows: 1. the relevance of software processes to achieving quality in software products, 2. the prominence of ASD methods to face the change in which the software development industry is caught up and 3. the

potential of Lean thinking as a means for scaling ASD and enhancing the software development processes.

1. *Relevance of software processes*. One may legitimately question the importance of software development methodologies by wondering what development processes were used behind successful innovations such as Facebook, which emerged in 2004 as a student's project at Harvard University (Wasserman 2013). However, as companies grow, business and technical risks also increase. More customers become dependent on the developed products, projects become larger, the development process involves more people and work coordination becomes essential (Münch *et al.* 2012). Using the example of Facebook, what was initially composed of a group of students had become today one of the *Fortune 500* companies with a revenue of $5.1 billion in 2012[6]. Facebook has around 4900 employees and 655 million daily active users[7]. Obviously, nowadays, Facebook needs to coordinate its work through a more sophisticated development process, as shown on its engineering web page[8].

   The importance of software development processes and their influence on product quality has been highlighted by the software engineering community for decades, under the premise that more appropriate software development processes will result in better products (Oivo *et al.* 1999; Münch *et al.* 2012). Consequently, efforts for enhancing software development processes have made different contributions, from process models such as the waterfall model (Royce 1970), the spiral model (Boehm 1988) and more recently the Agile methods (Agile Manifesto 2001), to process standards that are widely used in the industry, such as the ISO/IEC 12207:2008 (2008) and ISO/IEC 90003:2004(E) (2004), and software process improvement frameworks such as CMMI (2010), SPICE (ISO/IEC 15504:1998) and Bootstrap (Kuvaja and Bicego 1994).

2. *Prominence of Agile Software Development methods*. However, the environment in which the software development industry works today is

[6] Fortune 500 companies 2012 (accessed November 13, 2013). http://money.cnn.com/magazines/fortune/fortune500/2013/snapshots/160.html?iid=F500_sp_list

[7] http://newsroom.fb.com/Key-Facts (accessed November 13, 2013).

[8] *Scaling Facebook to 500 Million Users and Beyond,* Facebook engineering web page, http://www.facebook.com/note.php?note_id=409881258919 (accessed November 13, 2013).

fundamentally different from the environment in which it worked ten years ago. The *Age of Information*[9], which is strongly based on the Internet, has shaped a digital economy and a knowledge based society. Digital resources are constantly available for everyone, enabling new business opportunities. Information flows are accelerated, global communication and networking are faster and individuals can explore their personal needs easier. As a consequence, globalisation and market dynamics, characterised by rapid and unpredictable change, have shifted the competitive landscape. These market features pressure software-intensive organisations to develop what Eisenhardt and Martin (2000) call '*dynamic capabilities*'. Software development organisations need to be creative, innovative and flexible with business changes, while working with incomplete information and pursuing economic efficiency. This situation is especially relevant for organisations that develop in markets such as telecommunications, web applications or services, where applications are frequently developed in a matter of weeks or months, rather than years, as it was common when using traditional methods such as Waterfall (Wasserman 2013). Following the example of Facebook, its engineering webpage highlights the importance of moving fast and being flexible by trying different options, making small and incremental changes, constantly measuring their effects and working with small and independent teams composed of responsible engineers.

Facing these challenges, ASD emerged in 2001 through the *Agile Manifesto* (Agile Manifesto 2001) and provoked a change in the way in which the software engineering community was addressing software development processes. As opposed to the traditional stage-based models, iterative development, continuous delivery and short feedback cycles were advocated to adapt to the market or customer fluctuations (Abrahamsson *et al.* 2002). Moreover, ASD emphasized the people-side as a primary driver of project success and positioned software development as a socio-technical activity (Cockburn and Highsmith 2001). Although ASD caused some initial controversies (Rakitin 2001; Boehm 2002; Beck and Boehm 2003; Rosenberg and Stephens 2003), what was initially considered a fad progressively

[9] *Information Age* refers to the age that began in the 1970s in which society started to work in networks through a constant flow of information based on technology. In this age, company competitiveness is dependent on its knowledge of technology, information and network access (Castells 2011).

became mainstream. For example, the survey conducted by Forrester Research in 2009 on the state of ASD indicated that it was the methodology that most closely reflected the development process of 35% of IT organisations (West *et al.* 2010).

In summary, during recent years, it has been learnt that software development processes cannot be isolated from the market that they serve (Bowers 2002; Münch *et al.* 2012; Wasserman 2013). Predictability, stability and determinism premises on which software development methods have traditionally been built can no longer be assumed in many software development market areas (Boehm 2006; Cleland-Huang 2013). Thus, ASD methods have become prominent in the markets that demand high degrees of flexibility and creativity. Moreover, the '*softer'* side of software development has been stressed in recent years (Conboy *et al.* 2011; Cockburn and Highsmith 2001), appearing in studies that affirm that socio-organisational factors have a stronger impact on software development failures than technological factors (Ewusi-Mensah 2003).

3. *Potential of Lean Software Development*. Despite the advantages that supporters of Agile attribute to Agile methods, ASD has been found to be insufficient for large software development and the operation of the whole organisation (Turk, France and Rumpe 2002; Abrahamsson, Conboy and Wang 2009; Maples 2009; Vilkki 2010). Lean Software Development has been discerned as a promising approach for scaling Agile. Many ideas behind the Agile Manifesto were inspired by Lean thinking (Highsmith 2002; Conboy 2009). For example, Conboy (2009), who studied the origins of Agile in IS development, defined agility as the sum of flexibility and leanness. Hence, it is not surprising that Agile and Lean share many similarities in the software development domain (Vilkki and Erdogmus 2012). However, contrary to Agile, the focus of Lean on the organisation as a whole makes it a promising solution for addressing the limitations of ASD methods. Moreover, the potential shown by Lean in different domains have generated a greater demand for knowledge in Lean thinking, not only as a way of scaling Agile, but also as a way for enhancing software development processes from a wider perspective (Maglyas *et al.* 2012). Table 1 shows some examples of these benefits.

**Table 1. Examples of benefits attributed to Lean by companies from different domains.**

| Topic | Benefits |
|---|---|
| Profitability and productivity | Business Week reported that Toyota [10] cut $2.6 billion out of its $113 billion manufacturing costs, without closing a single plant in 2002 (Bremner *et al.* 2003). Although Toyota recently suffered what may be regarded as 'the most challenging crisis in its history' (Cusumano 2011), the company led global automobile sales until the March 2011 earthquake in Japan, which brought the production to a halt (OICA). |
| Time to market | Zara, operating in the retail clothing industry, has reduced its lead time via a business model based on the collecting and sharing of input from customers daily and using Lean inventories. Shorter lead times have enabled Zara to deliver new items to stores twice a week (as much as 12 times faster than its competitors) and to bring in almost 30000 designs each year, as opposed to the 2000–4000 new items introduced by its competitors (Tokatli 2008).<br>In healthcare, the application of Lean has been reported to cause significant improvements in reducing patient waiting lists, floor space utilisation and lead-time in laboratorial tests (de Souza 2009). |
| Product quality | Toyota is recognised as an icon in quality control (Cusumano 2011); for example, the Toyota Lexus CT200h recently received the maximum rating under the Japanese overall safety assessment (2011 Japan New Car Assessment Program). |

Success stories of companies such as the Boeing Corporation (Venables 2005), General Electric (Byrne and Womack 2012), Zara (Tokatli 2007) and the University of Michigan Health System (Kim *et al.* 2009), have undoubtedly aroused the interest of the software development industry as well.

## 1.2 Research Opportunities

The early stage of the research on the topic of Lean Software Development at the beginning of the dissertation provided a wide space for research contributions. It also drove it towards an explorative approach. In the scope of this work, four research opportunities, introduced below and further developed in *Chapter 2*, have been considered.

[10] Toyota and its Toyota Production Systems are well-known as an icon of Lean thinking (for more information see *Chapter 2.2 Origins of Lean and Agile*).

*Research opportunity 1: To clarify Lean thinking in the domain of software development and its combination with Agile methods*. Four years ago, when this work began, Lean thinking and its combination with ASD appeared as emerging and ambiguous phenomena. The terms Agile and Lean were often found to be inconsistent in software development literature (Conboy 2009). One reason behind this situation is that in recent years the progress toward Lean Software Development has mainly been driven by industry pioneers who were familiar to some extent with ASD. Consequently, Lean Software Development itself has not been extensively researched, and there is a lack of understanding about which of its elements are beneficially applied in practice, as well as about the ways to combine Lean thinking with ASD. Although Lean thinking has penetrated many industries, there is no common definition of Lean (Shah and Ward 2007). This deficit in specification is even greater in Lean Software Development due to the freshness of the topic (Ebert *et al.* 2012). Moreover, knowledge from other disciplines only has limited applicability in software (Münch *et al.* 2012). Therefore, Lean is open to interpretation in a software development domain (Ebert *et al.* 2012). On the other hand, Lean is a multifaceted topic that is intricately intertwined with concepts that are also of growing importance in the software development industry, such as Agile methods, customer centred development, learning organisation, etc., which makes it complicated to analyse. Confusion, although natural during the initial stages in any field of knowledge, is a clear risk, in that it may lead to a meaningless buzz and produce disillusionment. For example, teams that superficially adopt Lean Software Development because they do not understand it properly will probably not achieve the expected outcomes, resulting in frustration (Ebert *et al.* 2012).

*Research opportunity 2: Identifying how the principles of Lean thinking are interpreted in software development, as well as the concrete practices and tools for implementing the fundamentals in practice*. Lean and Agile rest on a set of principles (i.e. the five principles of value, value stream, flow pull and perfection in the case of Lean (Womack and Jones 1996) and the ASD's values and principles espoused in the Agile Manifesto (Agile Manifesto 2001)). However, proper practices and tools are important to actually implement the fundamentals in practice. In the case of ASD, twelve principles, together with four values, were purposely defined from a software development perspective, as espoused in the Agile Manifesto (Agile Manifesto 2011). Practices to implement those fundamentals, such as Test Driven Development (TDD), pair programming,

continuous integration and daily stand-up meetings, have been studied in the latter years (Dybå and Dingsøyr 2008). However, in the case of Lean Software Development, difficulties appear because Lean principles have been defined in a general way. Consequently, while Lean principles could be virtually applied to any domain (Staats *et al.* 2011; Poppendieck and Cusumano 2012), specific practices and tools are context dependent and, therefore, need to be adapted. Research investigating the concrete manifestations of Lean fundamentals in the form of more measurable practices is growing, and techniques such as Kanban (Anderson 2010) or 5 Whys (Parnell-Klabo 2006), are starting to become sound in software development literature. Nevertheless, research in the topic is still scarce. Thus, some authors have demanded higher attention to the practices and tools for implementing Lean thinking in the software development domain. For example, Erdogmus (Vilkki and Erdogmus 2012) claims that '*the real value of Lean thinking is in concrete tools that development teams and managers can employ on the ground every day*'. In order to clarify Lean thinking and its combination with Agile methods in a software domain, this thesis focuses on identifying concrete manifestations in the form of practices and tools that software-intensive companies use when they advocate the use of Lean thinking in combination with ASD.

*Research opportunity 3: Exploring the phenomenon of Lean Software Development and its combination with Agile methods in its natural context by using empirical research*. As many other software engineering innovations, Lean Software Development is mainly led by the industry. Companies have their own research centres, where they investigate ways to extract the maximum benefit from Lean and Agile ideas in the context of software development. Thus, through a process of learning by experimentation, companies come up with their own interpretation of Lean Software Development. In this environment, empirical research offers an opportunity to provide evidence on how Lean thinking and its combination with Agile is happening in the real world. This work, mainly carried out in Finland, took advantage of the following opportunities: i) the Finnish software industry constitutes a convenient population for the study. Finland is well-known to have one of the most successful IT industries in the world; Finland traditionally takes the top positions in international rankings such as the Global Information Technology report 2013 of the World Economic Forum (Bilbao-Osorio *et al.* 2013), which was topped by Finland, and the IT Industry Competitiveness Index (2011) of the BSA/Economist's report, where Finland

took the second position. Moreover, Finland can be considered as one of the pioneers in adopting Agile and Lean methods for software development, being one of the countries with the widest contribution of scientific articles in the topic (Dingsøyr *et al.* 2012). ii) In addition, Cloud Software Program provided participants who were willing to serve in the study. The project, which lasted four years, offered the possibility of collecting data over a sustained period of time. Moreover, it provided companies that were committed with their transformation towards Agile and Lean and motivated for collaborating in the research. iii) Finally, the work also took advantage of the services of *The Finnish Information Processing Association* (FIPA)[11], an independent association of Finnish ICT professionals and companies. *Tietotekniinan Liitto* (in Finnish) helped in conducting some of the studies by using its membership registry, which is comprised of about 16000 professionals as personal members and over 500 institutional members.

*Research opportunity 4: Investigating the impacts of applying a combination of Lean thinking and Agile methods for software development*. Understanding the consequences of using a combination of Agile and Lean in software development is one of the most interesting research topics, if not the most. However, adopting Lean is usually a complex process that leads to the transformation of the organisation impacting many organisational aspects and functions (Womack *et al.* 1990). Therefore, it takes time to notice its impact in a quantifiable way. Using a qualitative approach, this thesis develops an experience base that includes benefits and disadvantages of applying a combination of Lean and Agile in software development. The close cooperation with organisations enabled the identification of consequences as perceived by experts leading the transformation and by practitioners directly impacted by Lean and Agile software development. Strengths and challenges when transforming towards Lean thinking are also analysed, which are more feasibly observable during the transformation process.

## 1.3 Objective and Research Questions

The objective of the dissertation can be formulated as follows,

[11] http://www.ttlry.fi/english, (accessed November 13, 2013).

*Studying Lean Software Development and its combination with Agile methods as it is happening in practice, by analysing: i) the main elements of the combination of Lean thinking and Agile Software Development in the form of principles guiding its application, and practices and tools used in its implementation, ii) the key factors influencing the transformation of software-intensive organisations towards Lean and Agile and iii) the impact of its application as perceived by practitioners.*

Three complementary research questions (RQ) drove the research.

*RQ1: How is Lean thinking combined with Agile methods in software development?*

- RQ1.1: Why are Lean thinking and Agile methods combined in software development?
- RQ1.2: What main elements characterise Lean Software Development and its combination with Agile Software Development?
- RQ1.3: What elements have been brought by Lean thinking on top of those predating the Lean Software Development movement?

*RQ2: What are the key factors that influence the successful transformation of software organisations towards a combination of Lean and Agile methods?*

- RQ2.1: What challenges are potentially faced when combining Lean thinking and Agile Software Development?
- RQ2.2: What are the strengths of software-intensive companies in combining Lean thinking and Agile Software Development?

*RQ3: What are the impacts perceived by practitioners when using a combination of Lean thinking and Agile Software Development?*

Table 2 associates the research questions to the research opportunities that were identified in the previous section. RQ1 and its three sub-research questions aim to take advantage of research opportunities 1, 2 and 3. Thus, RQ1 investigates the combined use of Lean and Agile methods in software development with the aim of understanding the process of application (research opportunity 1) at fundamental and practical levels (research opportunity 2) using empirical research (research opportunity 3). In addition, RQ2 and RQ3 focus on factors that influence the transformation towards a combination of Lean and Agile methods and impacts of using such a combination respectively (research opportunity 4), again investigating it through empirical research (research opportunity 3).

**Table 2. Research questions and research opportunities matrix.**

| Research question | Research opportunity | Description |
|---|---|---|
| RQ1<br>RQ1.1<br>RQ1.2<br>RQ1.3 | Research opportunities 1, 2 and 3 | RQ1 focuses on clarifying how Lean thinking is interpreted in the software development context and how it is combined with Agile methods in practice (studied through empirical research methods). RQ1 focuses on the reasons for adopting a combination of Lean thinking and ASD and on the elements that characterise the combination of both methods, considering both principles at fundamental level and practices and tools for implementing the fundamentals in practice. |
| RQ2<br>RQ2.1<br>RQ2.2 | Research opportunities 3 and 4 | RQ2 focuses on the strengths and the challenges that companies face when transforming from following the basic Agile principles to complementing them with Lean thinking. Again, the research is conducted following an empirical approach. |
| RQ3 | Research opportunities 3 and 4 | RQ3 focuses on the consequences of using a combination of Lean thinking and Agile methods. The impacts of the transformation are investigated as they are perceived by practitioners. |

## 1.4 Scope of the Research

Defining the scope of the research has been challenging due to the multidisciplinary nature of the investigated topic (Abrahamsson *et al.* 2002; Conboy 2009), which makes it difficult to define absolute limits. However, in order to make the work manageable, some main focus points were established. On one hand, the phenomenon of Lean thinking and its combination with Agile was analysed from a software development point of view. Thus, the epicentre of the research was on software products and the software development processes around it, and not on the product development in general. However, according to Lean thinking, Lean companies should apply Lean in whatever they do (Womack *et al.* 1990), thus the limit in software development processes was not strict and the surrounding areas were also considered when they emerged as important through empirical investigations.

In addition, the focus of the work was on elements that were considered relevant from a Lean and Agile software development point of view, including

principles, practices and tools. A precise definition of Lean Software Development and its main elements, considering semantic and epistemological concerns were out of the scope of this dissertation.

Finally, changing the way of working towards Lean Software Development is a complex process that requires time to show specific impact. Moreover, transformations that implicate a cultural change, as is the case, are especially hard due to the high level of learning that they require (Alder and Shenhar 1990). People need enough time to internalise and to adapt themselves to a new way of working. Considering the limited timeframe of a PhD dissertation, the main focus of the work was not on measuring the benefits or the negative consequences of using Lean thinking and Agile methods in software development in a quantifiable way. Therefore, quantitative business metrics were out of the scope of this dissertation. However, the close contact with practitioners allowed for the identification of impacts as perceived by them, which has been reported as part of this dissertation.

## 1.5 Overview of the Research Design

This section introduces the research design followed in the dissertation, which will be further developed in *Chapter 3*. The research design is based on empirical software engineering and is exploratory in nature. As indicated by Creswell, '[exploratory research] *may be needed because the topic is new, the topic has never been addressed with a certain sample or group of people, and existing theories do not apply with the particular sample or group under study*' (Creswell 2009, pp.18). A mixed methods procedure (Creswell 2009), combining both quantitative and qualitative forms, was used with the purpose of addressing the research problem from multiple perspectives and strengthening the overall contribution. The research was carried out in four phases as follows:

- *Phase 1: Literature analysis*. First, the literature on the topic was analysed to frame the problem and to identify the research opportunities.
- *Phase 2: Extensive surveys*. Then, two surveys were conducted to explore the phenomenon from a wide perspective. The first survey investigated the status and trends of Agile and Lean adoption in software development, whilst the second focused on analysing the interpretation and the legitimisation of values in Agile's organising vision (community discourse on ASD).

- *Phase 3: Case studies*. The survey studies were followed up by two case studies conducted in cooperation with Ericsson and Elektrobit in order to analyse how Lean and Agile are actually combined in practice more deeply.
- *Stage 4: Research synthesis*. Finally, the results from previous phases were synthesised in order to draw conclusions and outline the implications.

## 1.6 Structure of the Thesis

This thesis collects the published contributions of the author in the above-mentioned topic. The thesis consists of six peer-reviewed original research publications whose scope and contribution are summarised in Figure 1.

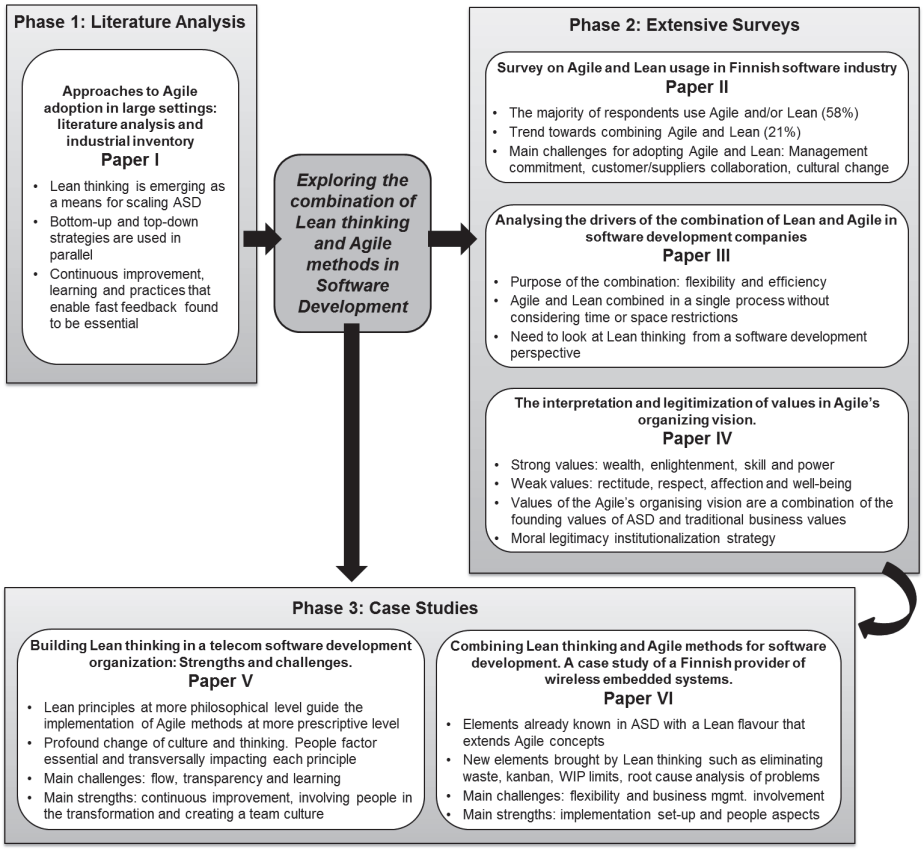


**Fig. 1. The scope and contribution of the original publications of the dissertation.**

Paper I constitutes a literature analysis and an industrial inventory on the topic of large scale ASD. Among other findings, Paper I pointed to the potential of Lean Software Development for scaling ASD, which motivated the rest of the studies in the thesis.

Papers II, III and IV focus on studying the phenomenon of Lean and Agile software development from a wide perspective. Papers II and III confirmed the tendency to use Lean thinking in combination with ASD by surveying their adoption in the Finnish software industry. Moreover, they pointed out trends in using specific principles and practices and revealed a certain lack of clarity in the application of these methods. Paper IV analysed the value foundations that can facilitate or hinder the adoption of Agile methods.

Finally, Papers V and IV present the findings of a more detailed analysis of how software-intensive companies are actually implementing Lean and Agile in practice through a case study strategy. Each paper presents a case study, conducted with Ericsson and Elektrobit respectively. The main elements characterising the combination of Lean thinking and Agile methods in a software domain are analysed in these studies as well as the strengths and challenges of the companies when transforming towards this way of working.

Besides the original contributions, the thesis includes an introduction composed of 6 chapters, as depicted in Figure 2. These chapters constitute a summary based on the set of publications to guide the reader in the dissertation.

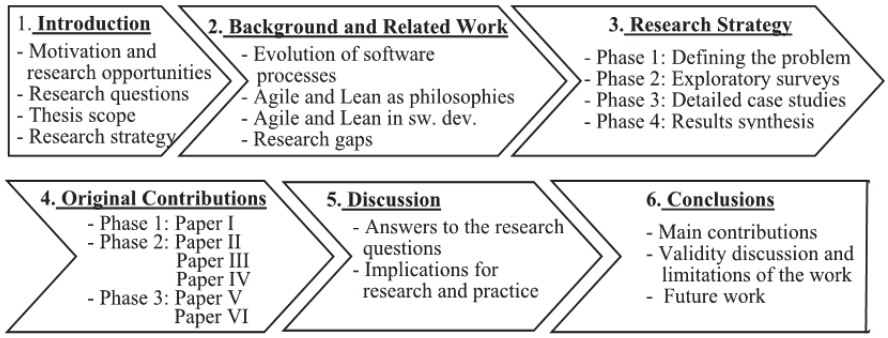


**Fig. 2. The contents of the dissertation.**

Besides this introductory chapter (Chapter 1), Chapter 2 reviews the literature on the topic to provide the foundations that guided the empirical investigations. The purposes of the chapter are to place the reader in the research area, to position the

dissertation within the larger body of research and to show the research gaps that the dissertation will address. Chapter 2 is structured in three main areas of knowledge: software development methodologies, Lean and Agile as philosophies and Lean and Agile applied to software development.

Chapter 3 presents the research framework, including a detailed description of each of the four phases that composed the research. The purpose of each specific phase, the research methods applied in both data collection and data analysis and the outcomes of each phase are described in detail.

In Chapter 4 the six publications that compose the thesis are introduced. Each publication is separately summarised and the way in which it contributes to the global research goal of the dissertation is explained. Also, the author involvement in each publication is clarified.

In Chapter 5 research questions are answered one by one and the main findings of the study are discussed and compared with related work. Implications for research and practice are also examined.

Finally, Chapter 6 concludes the dissertation by exposing the main contributions of the work, discussing validity aspects and limitations, and outlining opportunities for future research.

# 2 Background and Related Work

The work presented in this dissertation is positioned in the areas of software development processes and Agile and Lean methodologies. Section 2.1 presents an overview of software development methodologies in order to show their evolution during the last decades. Then, Agile and Lean as initially developed in manufacturing are reviewed in section 2.2 in order to understand the theoretical differences on the basis of how they have evolved. This section includes a description of relevant concepts, which due to space limitations could not be included in the original publications. Sections 2.3 and 2.4 present an overview of Agile and Lean in the field of software development. Finally, section 2.5 enumerates the research gaps that motivated the dissertation.

## 2.1 Evolution of Software Development Methodologies

Software development models have gradually evolved from the classical Waterfall model of the 1970s (Royce 1970), built on the assumption that requirements are relatively stable, to the Spiral model introduced by Boehm in 1986 (Boehm 1988) and more recent Agile and Lean methodologies. Figure 3 depicts a timeline with some important contributions to software development processes[12]. It is based on the comprehensible review of process models as conducted by Münch *et al.* (2012). As illustrated in Figure 3, software development methodologies have evolved during last 30 years from heavy-weight processes, such as the Waterfall model, to lighter processes based on Agile methods. Over time, process-centric models have given way to more people oriented approaches, in which the importance of breaking down the software process into smaller and more manageable units has been evidenced, as well as the importance of increasing customer involvement. Although the iterative pattern had already been initiated by the Iterative Enhancement Model in 1975 (Basili and Turner 1975), a tendency from plan-driven, deterministic and repeatable processes, that Osterweil called '*software processes are software too*' (Osterweil

[12] The purpose of Figure 3 is to show the trends in the software development processes. Therefore, representative contributions with illustrative purposes have been included. Classical software engineering books such as (Sommerville 2010) and guides such as the *Software Engineering Body Of Knowledge* (SWEBOK) provide a more complete inventory on software engineering processes and related topics. Similarly, detailed descriptions of each approach are not included. More detailed descriptions can be found in the referenced work.

1987), to software development methods that praise project variability, evolutionary development, flexibility and human aspects such as creativity and interactions among individuals is noticeable.

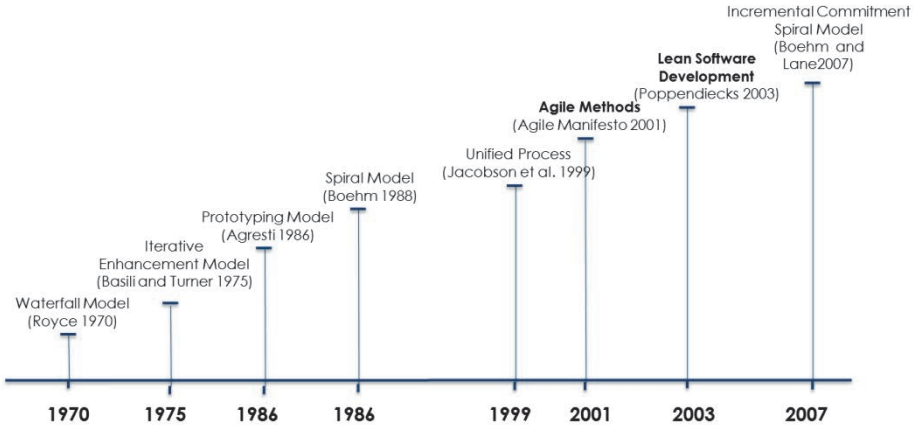


**Fig. 3. Software development methods timeline.**

In this context, the human side of software development is especially relevant, since it makes software processes less deterministic and repeatable as previously considered by traditional prescriptive methods (Osterweil 1987). Kruchten (2011) explained it using the party metaphor: '*Consider you organize a party this Saturday at your home. You invite a group of people, order the food, move the furniture, and you record carefully: who comes, when, who eats what, drinks what, etc., and everyone leaves having had a good time. The next Saturday, to have another great time, you decide to have the very same people, the very same food and drinks, you ask them to dress the same way, arrive at the same time, you introduce them to the same people, bring them the same glasses with the same drinks, make the same jokes,... will you be guaranteed to have a nice party again...?*'. In the same line, Boehm stressed that: '*one of the most significant contributions of the agile methods community has been to put to rest the mistaken belief that there could be a one-size-fits-all software process by which all software systems could be developed*' (Münch *et al.* 2012, pp. v).

Agile and more recently Lean are in the forefront of new software development methods following these trends. Next, Agile and Lean in a production domain, where they were born, are revised, which serve as basis to understand how Agile and Lean are being interpreted in software development.

## 2.2 Origins of Lean and Agile

*Lean and Lean thinking* - Lean was born as part of the industrial renaissance in Japanese manufacturing after the *Second World War* in the 1940s. Based on fundamental industrial engineering principles, Lean thinking is steeped in a philosophy of maximizing value and minimizing waste (Womack *et al.* 1990). Traditional mass production, based on producing large batches of products through continuous assembly lines of interchangeable parts, was found to be unsuitable in a Japanese market lashed by the war and characterised by low volumes. Thus, Japanese manufactures found themselves surrounded by obstacles that pushed them to develop new methods. Japanese new production ideas led to what is known today as Lean (or Lean thinking). However, it was not called Lean until researchers from MIT [13] began research under the *International Motor Vehicle Program* (*IMVP*) [14] in 1986, with the purpose of investigating the differences among worldwide automotive industries. IMVP researchers discovered that the Japanese auto industry was far ahead when compared with the auto industry in America. By carefully studying Japanese methods, particularly those of Toyota under its Toyota Production System (TPS), they conceived an entirely different production system, which they called Lean manufacturing (Womack *et al.* 1990). Defined succinctly, '*Lean is about doing more with less*' by ideally producing '*the right things, at the right time and in the right place*'. Concepts, such as customer centric production, value stream, waste reduction, workflow analysis and continuous improvement characterise Lean. Although some voices have questioned the validity of IMPV's studies (Dybå and Sharp 2012), they have vastly influenced the shaping of Lean as understood nowadays[15]. A detailed description of the story of Lean can be found in the book *'The machine that changed the world*' (Womack *et al.* 1990).

From the TPS described by Taiichi Ohno (1988) and the MIT studies, Lean incrementally evolved into being described differently by various authors (Ohno

[13] Massachusetts Institute of Technology.

[14] *International Motor Vehicle Program* (IMVP, http://www.imvpnet.org/) is an international network analysing the challenges faced by the global automotive industry. IMVP, founded at the Massachusetts Institute of Technology (MIT) in 1979, has mapped Lean methodologies, established benchmarking standards and proved the entire automotive value chain.

[15] As an indicator "*The machine that changed the world*" (Womack *et al.* 1990) is one of the most widely cited references in operations management (referenced by 10237 sources in *Google Scholar*, last accessed November 13, 2013).

1988; Womack and Jones 1996; Liker 2004; Morgan and Liker 2006). Table 3 summarises the main interpretations of Lean thinking.

**Table 3. The sources of Lean thinking in a production context.**

| Source | Description |
|---|---|
| Ohno (1988) | Toyota Production System (TPS). The basis of the TPS is the absolute elimination of waste. Seven classes of waste: overproduction, waiting, transportation, over-processing, inventory, movement and defects. Two pillars support this system, JIT and *autonomation*. |
| Womack and Jones (1996) | According to MIT's researchers five principles guide Lean thinking: value, value stream, flow, pull and perfection. This dissertation takes these principles as theoretical basis. |
| Liker (2004) | Liker proposed that fourteen principles guide the TPS as follows :<br>Base your management decisions on a long-term philosophy, even at the expense of short-term financial goals.<br>Create a continuous process flow to bring problems to the surface.<br>Use 'pull' systems to avoid overproduction.<br>Level out the workload (*heijunka*).<br>Build a culture of stopping to fix problems, to get quality right the first time.<br>Standardized tasks and processes are the foundation for continuous improvement and employee empowerment.<br>Use visual control so no problems are hidden.<br>Use only reliable, thoroughly tested technology that serves your people and process.<br>Grow leaders who thoroughly understand the work, live the philosophy, and teach it to others.<br>Develop exceptional people and teams who follow your company's philosophy.<br>Respect your extended network of partners and suppliers by challenging them and helping them improve.<br>Go and see for yourself to thoroughly understand the situation (*genchi genbutsu*).<br>Make decisions slowly by consensus, thoroughly considering all options; implement decisions rapidly (*nemawashi*).<br>Become a learning organisation through relentless reflection (*hansei*) and continuous improvement (*kaizen*). |

| Source | Description |
|---|---|
| Morgan and Liker (2006) | Toyota Product Development Process - Thirteen principles structured into three categories:<br>Process<br>Establish customer-defined value to separate value-added from waste.<br>Front-load the product development process to explore thoroughly alternative solutions while there is maximum design space.<br>Create a levelled product development process flow.<br>Utilize rigorous standardization to reduce variation, and create flexibility and predictable outcomes.<br>Skilled people<br>Develop a chief engineer system to integrate development from start to finish.<br>Organize to balance functional expertise and cross-functional integration.<br>Develop towering technical competence in all engineers.<br>Fully integrate suppliers into the product development system.<br>Build in learning and continuous improvement.<br>Build a culture to support excellence and relentless improvement.<br>Tools & Technology<br>Adapt technology to fit your people and process.<br>Align your organisation through simple, visual communication.<br>Use powerful tools for standardization and organisational learning. |

One of the main challenges when studying Lean is that although it has been widely discussed and used for more than three decades, there is no common definition of Lean but a variety of interpretations as summarized in Table 3. Recently, Cusumano indicated that: '*The authors* [authors of The machine that changed the word (Womack *et al.* 1990)] *used the term 'Lean' to describe any efficient management practice that minimized waste*' (Poppendieck and Cusumano 2012). Shah and Ward (2007), aware of this problem, aimed to provide a more concrete definition. They defined Lean production as: '*an integrated socio-technical system whose main objective is to eliminate waste by concurrently reducing or minimizing supplier, customer and internal variability*'. However, these definitions only capture some of the facets of Lean. For example, the concept of customer value, which should guide each activity in a Lean organisation, is not explicitly considered. To face this challenge, this dissertation took the five principles of Lean, as originally introduced by MIT researchers, as the theoretical framework for guiding the work. The rationale was that since there is no standardised definition of Lean thinking, exploring Lean Software Development through the lens of the Lean principles as originally defined can

help to reduce the analysis to the roots of Lean and avoid secondary interpretation bias. Table 4 describes these principles in more detail.

**Table 4. The core five principles of Lean thinking according to MIT's researchers (Womack and Jones 1996).**

| Principle | Description |
|---|---|
| Value | Value is everything that a customer is willing to pay. Defining and understanding value from the perspective of the customer is the central focus of Lean thinking. The goal is to make organisations deliver as much customer value as possible. Its counterpart, waste, or '*muda*' in Japanese, is everything that absorbs resources but outputs no value. |
| Value Stream | Value stream is the optimised end-to-end collection of actions required to bring a product/service from customer order to customer care, ensuring that each activity adds customer value. |
| Flow | Flow means that activities are organised as a continuous 'flow', eliminating discontinuities in the value stream and enabling smooth delivery. Flow requires that the unnecessary steps are eliminated (waste in Lean terminology). Opposite to mass production, the products are made using 'single piece flows'. |
| Pull | Pull implies producing products only when they are really needed, by making customer needs and the market the primary decision-drivers. Accordingly, in a pull system, an upstream process only produces when a downstream process is ready and 'pulls' some more work from the upstream process. The goal is to minimise the inventories that do not produce customer value but are sources of waste. |
| Perfection | Perfection maintains the enterprise-level improvement and learning continuously on-going. The aim of Lean is to achieve zero waste and defects based on the concept that there is no end to the strive for perfection. |

Although Lean has no formal practices, a toolkit of recommended practices, from which to choose from, has emerged to implement the fundamentals. Table 5 summarises those that are more relevant from the perspective of this dissertation, because they were found to be relevant in a software development context too. The definitions are based on those proposed by Womack and Jones in their book *Lean Thinking* (Womack and Jones 1996).

**Table 5. Elements of Lean thinking.**

| Practice/Tool/Technique | Description |
|---|---|
| Autonomation (Jidoka) | Providing machines and operators the ability to stop the work immediately whenever a problem or defect is detected. Related to the concept of 'Stop-the-line'. |
| A3 Report | Problem solving practice consistent on getting problem, analysis, corrective actions, and action plan down on a single sheet of large (A3) paper. |
| Cell | Location of processing steps for a product immediately adjacent to each other so that parts can be processed in very nearly continuous flow. |
| Chief Engineer (Shusa) | Manager with total responsibility for the development of a product. |
| Just-in-Time (JIT) | System for producing the right items at the right time in the right amounts. |
| Kaikaku | Process improvement through a radical change. |
| Kaizen | Continuous process improvement through small and frequent steps. |
| Kanban | Method based on signals for implementing the principle of pull by signalling upstream production and delivery. |
| Lead time | Metric defining the total time that the customer must wait in order to receive a product after placing the order. |
| Level schedule (Heijunka) | Levelling schedule by sequencing orders in a repetitive pattern and smoothing the day-to-day variations in total orders to correspond to longer-term demand. |
| Mistake-proofing (Poka-yoke) | Method that helps operators to prevent quality errors in production by choosing the wrong part. |
| Big room (Obeya) | Project leaders room containing visual charts to enhance effective and timely communications, and to shorten the Plan-Do-Check-Act cycle. |
| Set-Based Concurrent Engineering | Approach to designing products and services by considering sets of ideas rather than a single idea, and delaying selecting the final design until the team knows enough to make a good decision. |
| Self-reflection (Hansei) | Continuous improvement practice of looking back and thinking about how a process can be improved. |
| Six Sigma | Management system focused on improving quality by using mathematical and statistical tools to minimise variability. |
| Standardization | Precise procedures for each activity, including working on a sequence of tasks, minimum inventory, cycle time (the time required to complete an operation) and takt time (available production time divided by customer demand). |
| Usable knowledge | Capturing the knowledge to be applied in other projects. |
| Value Stream Mapping (VSM) | Tool for analysing value and waste in the production process. VSM provides a standardised language for identifying all specific activities that occur along the value stream. |
| Visual control | Tools to show the status of the system so that it can be understood at a glance by everyone involved. |
| Work-In-Progress(WIP) | Items of work between processing steps. |
| 5-Whys | Method for analysing/finding the root cause of the problems consisting in asking 'why' five times whenever a problem is discovered. |

A more thorough listing of Lean terms, with examples and illustrations, can be found in *The Lean Lexicon* created by *The Lean Enterprise Institute* (Marchwinski and Shook 2008).

*Agility*- ASD also has its roots in a manufacturing context. Agile manufacturing appeared at the beginning of the 1990s, with the publication of a report by the Iacocca Institute (Nagel and Dove 1991), which defined it as a solution to satisfy fluctuant demand through flexible production. Specifically, Nagel and Dove defined the concept of *Agile manufacturing* as '*a manufacturing system with extraordinary capability to meet the rapidly changing needs of the marketplace, a system that can shift quickly among product modes or between product lines, ideally in real-time response to customer demand*'. Christopher and Towill (2000) indicated that '*the origins of agility as a business concept lie in flexible manufacturing systems',* characterising it as *'a business-wide capability that embraces organisational structures, information systems, logistics processes and in particular mind-set*'. Similarly on the concept of leanness, Agility seems to be '*highly polymorphous and not amenable to simple definition*' (Conboy 2009). However, two aspects are usually stressed in the literature, responding to change and exploiting changes to take advantage of them, and four capabilities, *responsiveness*, *competency*, *flexibility* and *quickness* (Sharifi and Zhang 1999). A more comprehensible description of the main attributes of Agility can be found in (Sherehiy *et al.* 2007).

*Combining Lean and Agile in production* – As it is happening nowadays in software development, the combination of Lean and Agile has been considered in manufacturing (Ben Naylor *et al.* 1999). However, in a manufacturing context, the migration occurred from Lean and functional systems to Agile and customised production (Christopher and Towill 2000). Accordingly, the shift was characterised by a strong influence of Lean thinking, understood as efficiency and waste reduction. Specifically, the theory of *Le-agility* (Ben Naylor *et al.* 1999; van Hoek 2000) says that whilst the combination of Agile and Lean might work well, since Lean capabilities can contribute to Agile performance, the combination has to be carefully planned to prevent risks that Agile's demands may cause on Lean capabilities (Ben Naylor *et al.* 1999). Specifically, time and space restrictions are considered. Thus, the theory of Le-agility says that the combination of Agile and Lean in a production context is limited to three scenarios, i) different value streams, that is different products, ii) the same product but at different points in time or iii) using both paradigms at different points in the value stream by using de-coupling strategies.

The principles underlying Lean and Agile manufacturing have not been confined to manufacturing processes, but have also been applied to other disciplines. When it comes to the software development domain, the figure becomes complex due to the fundamental differences between the domains. Next, section 2.3 briefly summarises ASD. Then, Lean Software Development is analysed in more detail since it is the core focus of the dissertation.

## 2.3 Agile Software Development

In software development, Agile was officially established through the formulation of the Agile Manifesto (Agile Manifesto 2001), under the slogan '*We are uncovering better ways of developing software by doing it and helping others do it*'. Agile was a reaction against traditional methods that were considered as unable to meet market dynamics. Based on the original ideas of Agility, and also taking Lean thinking as one of its inspiring sources (Highsmith 2002), the Agile Manifesto was formulated from a software development angle. Four values were highlighted, 1) *individuals and interactions over processes and tools*, 2) *working software over comprehensive documentation*, 3) *customer collaboration over contract negotiation*, and 4) *responding to change over following a plan*. Together with a set of twelve principles, the Agile Manifesto offered an alternative to documentation driven, heavyweight software development processes (Cohen, Lindvall and Costa 2004). It also caused some initial controversy, especially in those who perceived *Agile* as an easy excuse for irresponsibility with no regard for the engineering side of the software discipline (Rakitin 2001). However, many others saw that agility and discipline were not so opposed, and hybrid approaches that combine characteristics of Agile and plan-driven methods were seen as feasible (Boehm 2002; Beck and Boehm 2003; Cobb 2011). Indeed, the seventeen proponents that formulated the Agile Manifesto stated that: '*the Agile movement is not anti-methodology, in fact, many of us want to restore credibility to the word methodology. We want to restore a balance. We embrace modelling, but not in order to file some diagram in a dusty corporate repository. We embrace documentation, but not hundreds of pages of never-maintained and rarely used tomes. We plan, but recognize the limits of planning in a turbulent environment*' (Cohen, Lindvall and Costa 2004).

Thus, the Agile Manifesto established the fundamentals of ASD. Several methods emerged to implement Agile in practice such as *eXtreme Programming* (*XP*) (Beck and Andres 2004), *Scrum* (Schwaber and Beedle 2001), *Dynamic*

*Systems Development Method* (Stapleton 2003) and *Feature-Driven Development* (Palmer and Felsing 2001). Next, Scrum is described in more detail since it appeared as the most popular method in the empirical studies conducted in this dissertation (Papers II, V and VI).

Scrum (Schwaber and Beedle 2001) can be considered a method for managing the software development process. Scrum focuses on how the software development team should work to produce software in a flexible way, so that it is able to respond to changes. Figure 4 depicts the Scrum process. The product is defined through user stories and features, which describe its functionally from a customer perspective. The development is carried out through increments or time-boxes, called sprints. Each sprint has a sprint goal that is defined during the sprint planning meeting, including the user stories that have to be undertaken during that sprint. The sprint backlog selected from the product backlog is frozen and remains unchangeable during the sprint. After every sprint, two meetings are held. The sprint review that focuses on analysing the project progress and demonstrates the current state, and a retrospective meeting to reflect about the way of working. The main roles in Scrum are *Product Owner*, *Scrum Team* and *Scrum Master*. The Product Owner is the voice of the customer to the team and is responsible for defining and prioritising features/user stories in the product backlog and accepting them as done at the end of each sprint. The Scrum team is self-organised and has the authority to decide on the necessary actions to complete the sprint goal. Usually, it is cross functional so that it has all the skills that are needed to meet the goal. Finally, the Scrum Master has the responsibility to ensure that the rules of Scrum are followed and is in charge to remove the possible impediments in the work of the Scrum team.

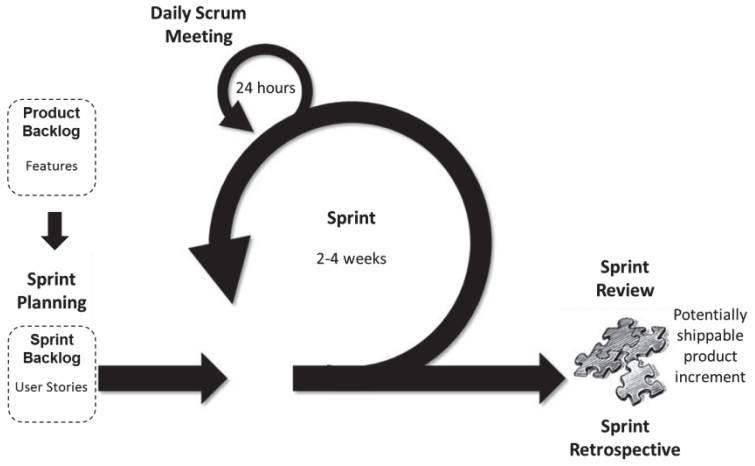


**Fig. 4. The Scrum process.**

Although the exact meaning of ASD is under debate in academic circles (Conboy 2009; Iivari and Iivari 2011), methods under the umbrella of ASD are used widely in the industry (West *et al.* 2010; Version One 2011). This industrial interest has caused many research efforts to focus on ASD. Dybå and Dingsøyr (2008) presented an excellent review of the empirical studies of ASD up to and including 2005. They found that most studies at that moment were conducted in small teams applying XP. Evidence of improvements in customer satisfaction, job satisfaction, productivity and product quality were found. However, the deficit of studies in the application of Agile in large-scale software development, using mature Agile teams and the lack of rigour in some studies, made the authors to request '*more and better empirical studies*' on the topic.

In spite of the encouraging benefits claimed by Agile practitioners (Dybå and Dingsøyr 2008), challenges when using ASD have also been identified in the last years as a result of a better understanding of ASD itself. For example, Boehm (2011) questioned the simplicity of the principles of the Agile Manifesto, indicating that: '*today's and tomorrow's complex, interdependent, dynamic systems require richer process principles than the simplistic principles in the Agile Manifesto*'. Other studies have also found the limitations of ASD, especially when intending to scale it for large software development. For example, Turk, France and Rumpe (2002) found limited support for developing large and complex software and limitations when involving large teams and distributed

environments. Communication mechanisms provided by Agile were also found to be insufficient by Pikkarainen *et al.* (2008), who claimed that: '*in larger development situations involving multiple external stakeholders, a mismatch of adequate communication mechanisms can sometimes even hinder the communication*.' Maples's research at Borland[16] (Maples 2009) also revealed difficulties when trying to involve the enterprise's areas outside of the scope of Scrum's prescriptions, due to the cultural frictions between Agile teams and other more traditional parts of the value chain. In fact, research on organisational culture and the deployment of Agile methods is a nascent area, where preliminary findings indicate that compatibilities and incompatibilities between Agile methods and organisational culture are key aspects that facilitate or hinder its adoption (Tolfo and Wazlawich 2008; Iivari and Iivari 2011). In the same line, Abrahamsson, Conboy and Wang (2009) demanded more research efforts for finding '*strategies for organisational level implementation of agility*' in their analysis of the state of Agile systems development research in 2009.

## 2.4 Lean Software Development

More recently, Lean Thinking has been discerned as a way to scale ASD. However, the universal application of Lean principles, as defined by Womack and Jones (1996), in knowledge work, such as software development, is under debate (Staats *et al.* 2011). Knowledge from other domains, such as manufacturing, has limited applicability in the software development domain (Münch *et al.* 2012 pp. ix; Mandic *et al.* 2011). For example, the easiness of modifying software products (code) over time is a dominant factor in the overall value of software products (Poppendieck and Cusumano 2012). Software products are malleable and their value is not limited to a single time-bound effort (Poppendieck and Cusumano 2012). Thus, the concept of value is not straightforward in software development, having a whole research field on its own, value-based software engineering (Biffl *et al.* 2005). Waste is also a controversial matter. Sources of waste in software development do not have to necessarily follow the path of the original seven forms of waste as identified by Taiichi Ohno (Ohno 1988). In manufacturing, most sources of waste can be directly detected by observing the physical material

[16] Borland was one of pioneers who applied the ideas behind Scrum. Sutherland and Schwaber (2007) describe how the Borland Quattro Pro project triggered the ideas that gave rise to Scrum in 1993.

flow and machine/worker activities. However, work items in software development are much more intangible, which challenge waste recognition and process improvement. The principles of value stream and flow are also challenged by the fact that software development is a process that bases mainly on information. Thus, what is flowing in software development has been questioned in the software development literature (Mandic *el al.* 2011). Moreover, the whole software development process outputs a single copy of a final product (excepting the case of software product lines where variants of the same product are developed). In this process, characterised by a certain lack of task repetition, the human factor is a dominant factor. Whilst in a manufacturing environment human presence is mainly required to operate automated machines, in software development creativity, knowledge and experience are essential (Mandic *el al.* 2011).

Despite the fundamental differences between software development and car manufacturing, it is recognised that software companies use customised interpretations of Lean principles (Mehta *et al.* 2008). Poppendieck and Cusumano (2012) recently claimed that: '*if lean is thought of as a set of principles rather than practices, then applying lean concepts to product development and software engineering makes more sense and can lead to process and quality improvements*'. Maglyas *et al.* (2012) also suggest that Lean principles may help in well-known software development management issues, such as shorter release cycles and decreasing time-to-market by using flow, defining key performance indicators based on value, improving collaboration with customers by using pull and avoiding short term thinking by using perfection and continuous improvement. In this context, Lean should be seen as a way of working, rather than as a prescriptive solution (Liker 2004). Lean principles could be applied to software development, but tools and practices supporting the principles in practice would need to be adapted.

Next, the path followed by Lean Thinking within software development is analysed by reviewing related works before and after the formulation of the Agile Manifesto, which constituted a turning point in the understanding of Lean thinking in software development.

### 2.4.1 Lean Software Development before the Formulation of the Agile Manifesto

The adoption of Lean within software development started as early as the 1990s, with concepts such as *lean software production* and *mistake proofing* (Freeman 1992; Tierney 1993). At that time, Lean thinking was understood as a way of making software development processes more efficient and improving their quality. For example, Freeman (1992) viewed Lean Software Development as a system of '*achieving ends with minimal means*' by striving to have '*the minimum amount of tools, people, procedures, and so on*'. In 1998, Raman (1998) analysed the feasibility of Lean in software development from a wider perspective, considering the five principles of value, value stream, flow, pull and perfection. Raman concluded that Lean could be implemented through contemporary concepts, such as reusability, rapid prototyping, spiral model and object-oriented technologies. As explained next, the concept of Lean Software Development has considerably evolved after the formulation of the Agile Manifesto.

### 2.4.2 Lean Software Development after the Formulation of the Agile Manifesto

Right after the formulation of the *Agile Manifesto*, Lean Software Development was considered as one of the Agile methods. See Table 1 in the systematic literature review conducted by Dybå and Dingsøyr (2008) on empirical studies of ASD as an example. However, as Lean thinking was analysed more deeply in a software context, Lean Software Development incrementally acquired an identity in itself (Wang *et al.* 2012). Nowadays, Lean thinking is seen as a way of scaling Agile, based on the closeness of ideas between Agile and Lean, and the unique focus of Lean thinking in the organisation as a whole (Poppendieck and Poppendieck 2003; Larman and Vodde 2008; Petersen 2010; Vilkki 2010; Laanti 2012). However, as in the case of manufacturing, there is no common understanding of what Lean Software Development actually means, resulting in different interpretations of what it is and how it should be applied, as follows.

*Interpretations of Lean thinking in software development* - Lean Software Development was initially mainstreamed with an interpretation of Lean thinking led by Poppendiecks' work (Poppendieck and Poppendieck 2003). This interpretation took the view on software development from the Lean (manufacturing) angle (Mandic *et al.* 2010). Later, more diversity has been

introduced. For example, Coplien and Bjørnvig (2011) focused on Lean architectures and Anderson (2010) on adopting Kanban to bring Lean thinking into software development. The Poppendiecks have also evolved their initial interpretation, emphasising the Lean mind-set (Poppendieck 2013). Table 6 summarises the main interpretations of Lean thinking in software development.

**Table 6. Interpretations of Lean thinking in software development.**

| Author | Description |
|---|---|
| Poppendieck and Poppendieck (2003, 2006, 2009) | There are seven principles that guide Lean Software Development as follows:<br>1. Eliminate waste, understanding first what value is.<br>2. Build quality in, by testing as soon as possible, automation and refactoring.<br>3. Create knowledge, through rapid feedback and continuous improvement.<br>4. Defer commitment, by maintaining options and making irreversible decisions in the last responsible moment when most information is available.<br>5. Deliver fast, through small batches and limiting WIP.<br>6. Respect people, the people doing the work.<br>7. Optimise the whole, by implementing Lean across an entire value stream.<br>Seven sources of waste in software development: partially done work, extra features, relearning, handoffs, task switching, delays and defects. |
| Middleton and Sutton (2005) | Interpretation based on Womack and Jones's (1996) five Lean principles: value, value stream, flow, pull and perfection. |
| Larman and Vodde (2008) | Two pillars: respect for people and continuous improvement, and 14 principles: management decisions based on a long-term philosophy, flow, pull, less variability and overburden, stop and fix, master norms, simple visual management, good technology, leader-teachers from within, develop exceptional people, help partners be lean, Go-See, consensus, reflection and kaizen. |
| Anderson (2010) | Kanban to bring Lean thinking into a software development organisation. Kanban uses five core properties: visualise workflow, limit WIP, make policies explicit, measure and manage flow and use models to recognise improvement opportunities. |
| Reinertsen (2009) | Set of principles of product development flow, including managing queues, reducing batch size, applying WIP constraints, etc. |

Kanban has frequently appeared in the empirical studies of this dissertation (Papers II, V and VI). Kanban in manufacturing was based on a signals system in order to implement the principle of pull by signalling upstream production and delivery. In software development, Kanban has been interpreted in a form of a

board, as depicted in Figure 5. Each activity in the development cycle is represented in a column of the board. The number of items under each activity is limited by establishing WIP limits (the numbers in red shown in the figure). Items from the backlog are moved from column to column when they are being developed. An item can be moved to the next development step only if there is room for the new item according to the WIP limits. The board, which has the same purpose that the signals in a manufacturing context have, helps to visualise the workflow by providing visibility to the entire software development process and enables to identify problems, such as bottlenecks in the workflow. Moreover, WIP limits help reduce inventory and focus on the feature that is being developed (Anderson 2010).

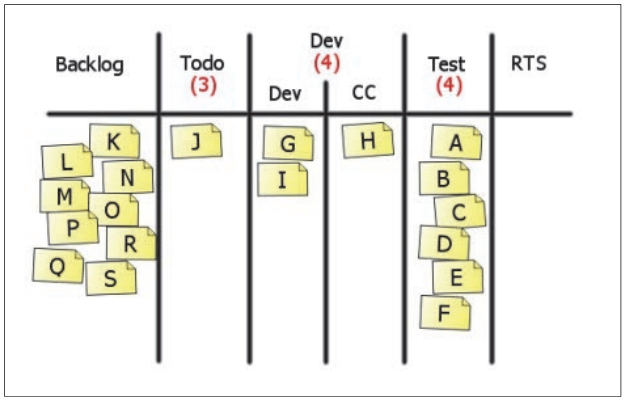


**Fig. 5. Example of a Kanban board.**[17]

*Empirical studies on Lean Software Development* – The interpretations presented above can be considered in practice as the body of knowledge of Lean Software Development. Although they have their merits, as demonstrated through the examples included in the books in which they are presented (Poppendieck and Poppendieck 2003, 2006 and 2009; Middleton and Sutton 2005; Larman and Vodde 2008; Anderson 2010; Reinertsen 2009), empirical evidence that prove that these interpretations are appropriate and can produce the expected results are scarce (Pernstål *et al.* 2013). Some early publications have analysed Lean

[17] http://blog.jaffamonkey.com/files/2012/01/Kanban-board-2.jpg

experiences, a majority in combination with Agile. Although, to date most of them are in the form of experience reports (Wang *et al.* 2012), there is also a growing body of knowledge, not only documenting case studies, but also investigating specific elements of Lean, such as waste (Reinertsen 2009; Ikonen *et al.* 2010; Nord *et al.* 2012), flow (Petersen 2010; Mandic *et al.* 2010) and Kanban (Ahmad *et al.* 2013; Sjoberg *et al.* 2012). Next, case studies similar to this work that address the '*how*' of Lean Software Development are reviewed in more detail. A summary of the main results of these case studies is presented in Table 1 of Paper VI.

Middleton and his colleagues conducted three case studies in which they investigated how Lean thinking was applied in three software-intensive companies (Middleton 2001; Middleton *et al.* 2005; Middleton and Joyce 2012). In Middleton (2001) two development teams at the IS Department in an organisation employing 7000 people were examined. The teams were applying the Lean techniques of reducing WIP (inventories) and continuous problem correction. The results of the study indicated that although starting Lean may be frustrating, due to the need of stopping-the-line continuously to solve problems, Lean Thinking helps to go to the root of the problems. Middleton found that aligning the entire organisation around Lean, which may request a deep organisational change, and considering human aspects fairly were of fundamental importance. Middleton's study was restricted to two teams composed by three members, which limited the generalisation of the results.

Middleton *et al.* (2005) also conducted a case study at Timberline Inc., an American software company employing 160 software developers. The study provided a wide set of techniques for implementing Lean, such as continuous flow of small development batches, focusing on understanding customer needs to eliminate rework using techniques, such as Kano's model (Sauerwein *et al.* 1996) and the Design Structure Matrix (Browning 2001), workload balance to ensure the allocation of resources and to avoid bottlenecks, cross-functional teams, standardised procedures, WIP limits, transparency and data driven decisions. However, the certain lack of detail in the descriptions made it hard to understand how these techniques were really applied.

More recently, Middleton and Joyce (2012) conducted another exploratory case study at the BBC Worldwide (London). The study focused on a development team composed by nine members, which applied Lean concepts, mainly Kanban, for one year. Despite its narrow scope, Middleton and Joyce provided interesting conclusions. They showed how the use of Kanban boards with WIP limits and

information radiators can help to ensure progress transparency, support teams to estimate their own capacity and benefit self-organisation. Moreover, daily stand-up meetings were found to be vital to ensure a smooth workflow and to identify bottlenecks. As a result of using a Kanban system, the team decreased the lead and development time, fixed the bugs more quickly and was able to deliver business value more frequently.

Mehta *et al.* (2008) also conducted a case study in order to identify relevant Lean principles in software development. An IT department supporting the web-based sales system of a large financial services firm was the focus of the study. The results suggest that Lean principles are aligned with best practices of software engineering, such as using coding standards, system decomposition or *'forethought in design'*. The factors identified as critical were to organise the work according to integrated product teams, to encourage employees to try out new ideas by promoting a '*failure tolerant culture'*, to make the development process transparent, to design a process champion who promotes Lean, to spend a significant amount of time on upfront planning, analysis and architectural work, to have a value based strategy for prioritising working on new features and fixing defects, to use cadence and more frequent and smaller builds.

Staats *et al.*'s (2011) case study is especially interesting, because the case is focused on a company that was applying Waterfall before introducing Lean. The authors describe how the introduction of Lean led into a more iterative development. The main elements characterising Lean in this case included a more detailed specification of tasks that use more standardised approaches, streamlined communication by using tools such as visual control boards, Design Structure Matrix (Browning 2001), value stream mapping (VSM) and videoconferencing such as WebEX to connect engineers at different locations, and hypothesis-driven problem solving supported by iterative development, continuous integration and periodic code reviews. Staats *et al.* concluded that Lean can improve the performance of software development process and creates opportunities to effectively deploy an iterative development model. However, they also found the lack of task repetition in knowledge work, such as software development, to be an important challenge when applying Lean, due to difficulties in specifying and standardising tasks.

Trimble and Webster (2013) presented their experiences in the transformation of a development team at NASA Ames Research Center from traditional to Lean and Agile development. Trimbel and Webster reported that the team, which was developing a user-centric software platform for mission control, decreased

delivery cycles and increased customer and team satisfaction as a result of the transformation. What the authors described as Lean and Agile was composed of aspects like shorter delivery cycles, increased customer collaboration, stand-up meetings, automated unit testing and continuous integration.

Although in these studies there is no unified model for the use of Lean and Agile, some patterns are observable. More studies converge in the importance of aspects like frequent builds through iterative development, reducing WIP, using cross-functional teams and increasing transparency. However, there are also more conflictive aspects, such as flexibility and standardisation. For example, while flexibility is essential in most studies, Staats *et al.* (2011) highlight the forethought design and the ability of Lean to help project managers plan better and monitor more closely. Regarding standardization, Middleton *et al.* (2005) and Staats *et al.* (2011) stress the importance of standardised procedures, which may challenge the application of Lean in a software development context. However, it does not appear as essential in the rest of the studies. On the other hand, while Middleton (2001) and Staats *et al.* (2011) emphasise the importance of fixing defects as soon as they appear (stop-the-line), Mehta *et al.* (2008) recommend '*a common prioritization code for errors and new features based on the ultimate customer value*' to decide whether working on new features or fixing defects.

*Literature reviews on Lean* Software *Development* – Finally, some studies have also reviewed the current status of Lean in software development (Wang *et al.* 2012; Jonsson 2012; Ahmad *et al.* 2013; Pernstål *et al.* 2013). These studies clearly reflect on the freshness of the topic. First, based on the analysis of 30 experience reports published in past agile software conferences, Wang *et al.* (2012) examined the purposes of applying Lean in ASD, by identifying six strategies of application: non-purposeful combination of Agile and Lean, using Lean to interact with other business areas while keeping Agile in software development, using Lean directly in software development processes for facilitating Agile adoption, using Lean in software development to improve Agile processes, transforming from Agile to Lean, and synchronising Agile and Lean. Although, Wang et al.'s study offers remarkable insights into the reasons why Lean is applied in combination with ASD, it does not explain on how the combination is really implemented. Second, Jonsson (2012) reviewed how Lean principles are interpreted in software development. Jonsson found Lean Software Development to be a promising approach for developing software. However, the low number of studies, as well as the anecdotal evidence, made it difficult to get reliable conclusions. Thus, Jonsson determined that further studies were needed in

order to advance in the field. Ahmad *et al.* (2013) focused their systematic literature review on studies applying Kanban in software development. They found promising benefits when using Kanban, such as the better understanding of the whole process by all stakeholders, especially developers, improved team communication and coordination, and increased customer satisfaction. However, challenges were also found, which included integrating Kanban with supporting practices of Agile, changing the organisational culture according to Lean principles and applying Kanban in distributed environments. Finally, Pernstål *et al.* (2013) recently conducted a systematic mapping study to identify opportunities for future research needs in Lean Software Development. The study was limited to investigating articles published between 1999 and 2010 and, therefore, papers that compose this dissertation were not considered in the review. Pernstål *et al.* explain that the decision to conduct systematic mapping instead of a systematic literature review was based on the fact that '*initial searches in database showed that there were relative few relevant and high-quality studies on the topic of interest'*. They concluded that *'the research in the much hyped field of lean software development is in its nascent state when it comes to large scale development. There is very little support available for practitioners who want to apply lean approaches for improving large/scale software development'*.

### 2.4.3 Lean Thinking and its Combination with Agile Software Development

As described in the previous section, during the last several years the transition from Agile to Lean processes, or a combination of both, is apparent. However, whilst some of the Agile Manifesto principles resemble Lean thinking others seem to be antagonistic. For example, '*simplicity – the art of maximizing the amount of work not done – is essential*', '*at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly*', further, '*our highest priority is to satisfy the customer through early and continuous delivery of valuable software*' are consistent with Lean thinking. However, '*welcome changing requirements, even late in development*' does not seem aligned with Lean. Practitioners and scholars have also pointed out the intertwining evolution of Agile and Lean approaches in software development and

argued their differences. For example, Wang and Conboy[18] (2011) question whether Agile and Lean are just two different names for the same thing, or whether they are actually different and, therefore, the challenges and issues faced by Agile processes could be addressed by Lean approaches. On the other hand, Petersen (2010) analysed the overlap between Lean Software Development, as interpreted by Poppendiecks (2003, 2006, 2009), and Agile. He concluded that both paradigms share very much the same principles in aspects, such as managing people and the continuous attention to quality and technical excellence. Moreover, they may complement each other with regard to aspects, such as flexibility, priority of customer needs/value and learning. However, the stress on the end-to-end focus (considering the whole) and flow are unique to Lean. However, the discussion is on-going and empirically grounded studies on the differences between Agile and Lean software development are needed (Wang and Conboy 2011; Petersen 2010). For example, Petersen recognises that the lack of empirical evidence for Lean Software Development made his comparison based on the generic descriptions provided in books.

## 2.5 Identified Research Gaps

Based on the literature review, the research gaps presented in Table 7 have been identified. Next, each research gap is described in more detail.

1. Scaling ASD continues to be a challenge in software development (e.g.: Turk, France and Rumpe 2002; Pikkarainen *et al.* 2008; Maples 2009;). Many studies highlight the benefits of using Agile at the development/team level (Dybå and Dingsøyr 2008). However, the literature also shows that achieving the same benefits in a wider organisational scope is still a challenge. More research efforts for finding strategies for organisational level implementation of Agile are required by scholars in the topic (Abrahamsson, Conboy and Wang 2009).

[18] Conboy deeply studied the origins of the concept of Agility in Information System Development and defined it based on the concepts of flexibility and leanness as "*the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment*" (Conboy 2009).

**Table 7. Research gaps.**

| Research gap | Studies supporting the research gap |
|---|---|
| 1. Scaling ASD continues to be a challenge in software development. | Turk *et al.* 2002; Pikkarainen *et al.* 2008; Maples 2009; Abrahamsson *et al.* 2009. |
| There is little empirical evidence on how Lean thinking is implemented in software development. Specifically, there is little empirical evidence on: | Wang and Conboy 2011; Petersen 2010; |
| 2. Lean Software Development and its relationship with ASD. | Jonsson 2012; Pernstål *et al.* 2013. |
| 3. Fundamentals of Lean thinking in software development. | Petersen 2010; Pernstål *et al.* 2013. |
| 4. Practices for implementing Lean principles in software development. | Vilkki and Erdogmus 2012. |
| 5. Benefits or negative consequences of applying Lean Software Development. | Jonsson 2012; Pernstål *et al.* 2013. |
| 6. Lean Software Development in a large industrial scale. | Dybå *et al.* 2008; Pernstål *et al.* 2013. |
| 7. There is a certain lack of rigour in previous studies on Lean Software Development | Dybå *et al.* 2008; Pernstål *et al.* 2013. |

2. Lean Software Development is acquiring more and more relevance as a worthy approach that may address challenges faced by ASD, and helps to improve the software development processes (e.g.: Poppendieck and Poppendieck 2003; Larman and Vodde 2008; Vilkki 2010; Laanti 2012; Poppendieck and Cusumano 2012; Wang *et al.* 2012). However, whether Lean thinking provides something new or it is equivalent to Agile is debated in the literature (Wang and Conboy 2011; Petersen 2010).
3. Lean is open to interpretation in a software development domain. Most of the current knowledge on the topic is provided in generic discussions in books, where authors, usually practitioners, make their own interpretation of what Lean Software Development is and how it should be used (e.g.: Poppendieck and Poppendieck 2003, 2006, 2009; Middleton and Sutton 2005; Larman and Vodde 2008; Anderson 2010; Coplien and Bjørnvig 2011). Progress toward Lean Software Development has mainly been driven by industry pioneers who are familiar to some extent with ASD. Little empirical evidence is available in which elements of Lean thinking are applied in practice, as well as ways to combine the Lean principles with ASD (Petersen 2010; Pernstål *et al.* 2013). Consequently, there is no unified understanding in essential aspects, such as flexibility, standardisation or continuous improvement.

4. How to implement the fundamentals in practice appears especially relevant, due to the fundamental differences between manufacturing and the software development domains (Vilkki and Erdogmus 2012). Lean thinking states principles and provides a toolkit to guide companies in applying the principles in practice. There is an apparent agreement that whilst principles may be universally applicable, practices and tools have to be adapted for the specific domain and chosen in accordance to the specific company context (Liker 2004; Poppendieck and Cusumano 2012). Again, empirical evidence in the practices and tools that support Lean thinking in the software development context is scarce.
5. Moreover, there is little empirical evidence on the consequences of applying Lean thinking in the software development domain. Studies reviewing the literature on the topic such as Jonsson (2012) and Pernstål *et al.* (2013) claim that '*The empirical evidence from lean applications in software development found is encouraging in terms of reduced lead time, increased productivity and higher team and customer satisfaction, but the limited number of studies and the dominance of a single author (Middleton), in the report calls for more studies*' (Jonsson 2012) and '*the implication for future research is that there is a strong need for more rigorous studies on the benefits of LPD* [Lean Product Development]' Pernstål *et al.* (2013).
6. As evidenced by the literature reviews conducted in the topic, especially Dybå and Dingsøyr (2008) and Pernstål *et al.* (2013), most empirical studies are conducted in small, collocated and quite often insufficiently mature teams in the usage of these methods. For example, from the 38 studies found as relevant by Pernstål *et al.* (2013) in their mapping study on Lean Software Development only 16 clearly dealt with software development in large-scale settings (See Figure 6, the number of studies on large-scale are placed in brackets). Therefore, there is a need to increase the amount of empirical studies that investigate Lean Software Development in a large industrial scale context.
7. Finally, Dybå and Dingsøyr (2008) and Pernstål *et al.* (2013) also detected a certain lack of rigour when assessing the quality of research studies in the topic. Specifically, as presented in Figure 6, Pernstål *et al.* (2013) found that '*even though the quality assessment shows that the relevance of the studies is relatively high, the lack of rigor reduces the possibility to judge the capability of the state of the art to be contributing for practitioners seeking to adopt new best practices, and limits replications of the studies*'.

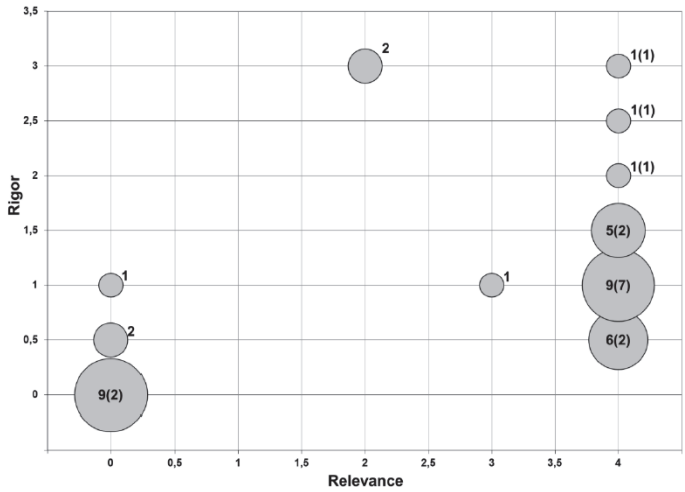


**Fig. 6. The distribution of rigor and relevance in the studies analysed in the systematic mapping on Lean Software Development conducted by Pernstål *et al.* 2013, published by permission of Elsevier.**

These research gaps gave way to the four research opportunities that motivated the thesis as follows:

- *Research opportunity 1:* To clarify Lean thinking in the software development domain and its combination with Agile methods.
- *Research opportunity 2:* Identifying how the principles of Lean thinking are interpreted in software development, as well as the concrete practices and tools for implementing the fundamentals in practice.
- *Research opportunity 3:* Exploring the phenomenon in its natural context using empirical research.
- *Research opportunity 4:* Investigating the impacts of applying a combination of Lean thinking and Agile methods for software development.

These research opportunities established the foundations of the research questions that guided the thesis (see *Chapter 1*, *Table 2. Research questions and research opportunities matrix*).

# 3 Research Design

The work has been conducted following an exploratory empirical research approach based on mixed research methods. Empirical research refers to the way of investigating a phenomenon by the means of direct or indirect observation or experience (Shull *et al.* 2008). Three main reasons motivated this design:

- First, exploratory studies are needed when the phenomenon of study has not been clearly defined or there is no understanding of the important variables to examine for theory generation (Creswell 2009, Shull *et al.* 2008), as it is for the case of Lean thinking in software development (Pernstål *et al.* 2013).
- Second, aligned with the thoughts by Kruchten (2011) in '*A plea for lean software process models*', the work aims to identify what software development practitioners do when they apply Lean thinking in combination with Agile methods. It was well known that software-intensive companies were investigating ways in which to extract the maximum benefit from Lean ideas in the context of ASD. Thus, through a process of learning by experimentation, software development companies were coming up with their own interpretation of Lean Software Development. This thesis takes advantage of empirical research in order to understand the phenomenon as it is happening in its natural context.
- Third, technical and non-technical aspects are both important in the topic of study. As recently highlighted by Münch *et al.* (2012), '*the relevance of cognitive laws for human-based processes* [such as software development processes] *is nondeterministic and can only be determined empirically […] through (combination of) empirical studies of different types (e.g., qualitative or quantitative studies, controlled experiments or case studies, real studies or simulation)*'.

The rest of the section describes the research design of the dissertation. First, the overall research approach is presented, including its four phases, as introduced in *Chapter 1*. Then, each phase is further elaborated, along with a comprehensible description of the applied research methods, including details that due to space limitations could not be included in the original publications. Strategies to minimise validity threats and limitations of the work will also be discussed later in Chapter *6.2 Validity discussion and limitations of the work*.

## 3.1 Overall Research Design

The research process was composed of four phases using a mixed method research model. The overall research design was designed following the guidelines as defined by Creswell (2009) in designing research studies, and Shull *et al.*'s (2008) guidelines for empirical studies in software engineering.

Mixed methods is a relatively new methodology in research that has aroused a great deal of attention in different fields of social and human sciences (Creswell 2009). With origin around the early 1990s, it involves the collection of both quantitative and qualitative data in order to provide a more complete understanding of a research problem than either approach alone (Creswell 2009). Both forms of data, quantitative and qualitative, provide different types of information and have their own limitations and strengths. The main advantage of mixed methods is its ability to combine the strengths of both in order to overcome the limitations of each and to develop a stronger understanding of the research problem (Creswell 2009). The main challenges are the need for extensive data collection, time for analysing both quantitative and qualitative data, the need for the researcher to be familiar with both forms of research and challenges of synthesizing findings from both approaches to draw meaningful conclusions.

The strategy in this work follows in what Creswell calls *explanatory sequential mixed methods design* (Creswell 2009). Explanatory sequential mixed methods involve two phases, in which the quantitative data is first collected and then the qualitative data helps to explain the initial quantitative results in more detail. Particularly, a combination of quantitative exploratory survey research and qualitative case studies formed the core of the research. The exploratory surveys contributed to understanding the trends and main elements of ASD and Lean thinking in a software development context. Afterwards, the case studies helped to get a more in-depth understanding of the survey results. Figure 7 depicts an overview of the research design. The study began with a literature analysis to define the research problem (Phase I). Although Phase 1 represents the initial literature review, the literature in the topic was continuously reviewed along the rest of the work in order to keep up to date with related work. Then, two quantitative studies were conducted in order to analyse trends and the status of Lean and Agile software development (Phase II). Extensive surveys were followed by a qualitative case study method involving a detailed exploration of Lean thinking in two specific companies following basic Agile methods, and transforming towards Lean Software Development, Ericsson and Elektrobit

(Phase III). Finally, the results from the previous phases were synthesised in phase 4 to outline the conclusions and implications to research and practice.

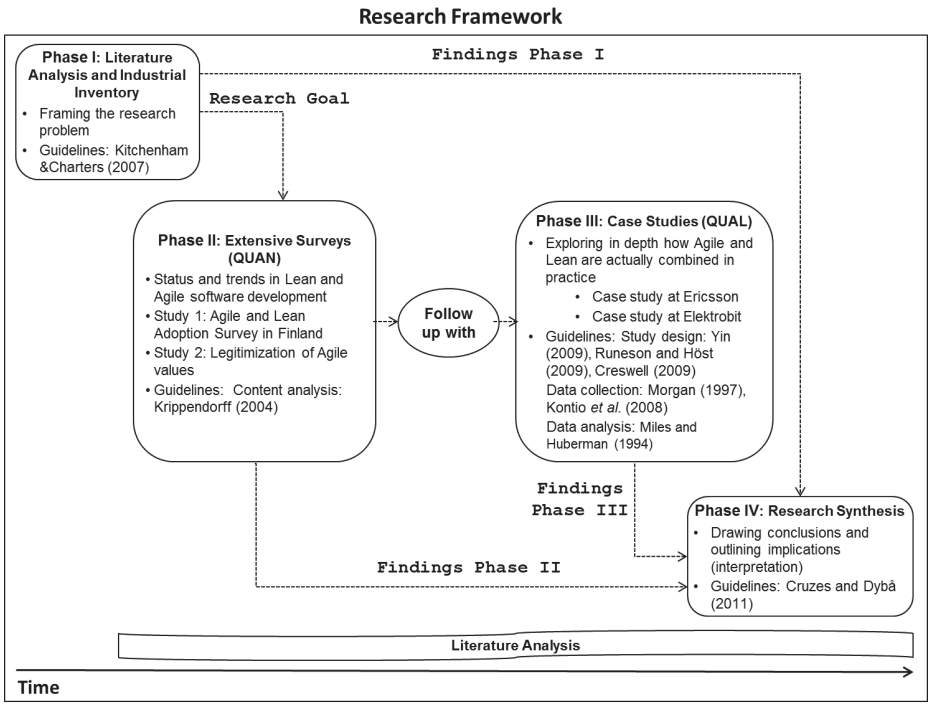


**Fig. 7. Research framework.**

Next, each of the four phases is described in more detail. Each individual phase has been driven by a subset of research questions, particular to each focal study phase. In order to facilitate their understanding and contribution to the general objective of the thesis, Figure 11, at the end of the chapter, presents a map where the research gaps, research objective and research questions (thesis level and paper level) are shown.

## 3.2 Phase I: Defining the Research Problem

According to Creswell (2009), the first step in any research is to identify the research problem and to reflect on whether it is practical and useful to undertake it as the topic of study. Accordingly, the research process was initiated by identifying the research opportunities and by defining the research problem to be

studied. This process implied not just the reviewing of relevant literature, but also inquiring industry in order to identify the most relevant issues that needed further research from a practitioner's perspective. Phase I's outcome is the first paper of the thesis (Paper I) and motivated the research problem. Two elements took part in this phase, as explained in subsections 3.2.1 and 3.2.2.

### *3.2.1 Literature Analysis and Industrial Inventory*

The work on the thesis started at the end of the FLEXI project[19], in which the author of the thesis was working during the last two years of the project (2008–2009). FLEXI, which focused on strategies for using Agile in software development, demonstrated concrete impact of Agile methods in product innovation and reducing lead times (FLEXI leaflet 2009). However, it also revealed difficulties to understand the implementation of Agile at an organisational level (Abrahamsson, Conboy and Wang 2009). Motivated by FLEXI's results, the initial idea was to focus on Agile adoption strategies for large companies. Hence, the work started with an analysis of related literature and an industrial inventory among the FLEXI industrial partners in Agile adoption frameworks, emphasising the *in-the-large* aspect. The purpose was to provide a general overview of Agile adoption topics, from both an academic and an industry perspective, in order to identify issues when scaling Agile and needs for future research.

The review process, both in the literature analysis and in the industry inventory, was based on Kitchenham and Charters (2007) guidelines for conducting systematic literature reviews. A team of three researchers accomplished the research. A review protocol was designed to guide the work composed by three phases: i) planning the review, in which objectives were established and the review protocol was designed, ii) conducting the review, in which both the literature analysis and the industrial inventory were carried out and iii) reporting the review in the three outcome reports: one that resulted from the literature analysis, another that resulted from the industrial inventory and the last one, which compared results and synthesised the main findings, serving as a basis for writing Paper I.

[19] ITEA2 06022 FLEXI (2007-2009), *Flexible global product development and integration: From idea to product in 6 months*. http://www.itea2.org/project/index/view?project=187 (accessed November 13, 2013).

Regarding the literature review, the analysis presented in Paper I focused on structured frameworks for adopting Agile in large organisations through the research question PI.RQ1[20] *What are currently the strategies to adopt Agile methods that are used in the software market?*'. Figure 8 shows the process for selecting primary studies that was composed of five stages and was based on a set of acceptance criteria as showed in Table 9.

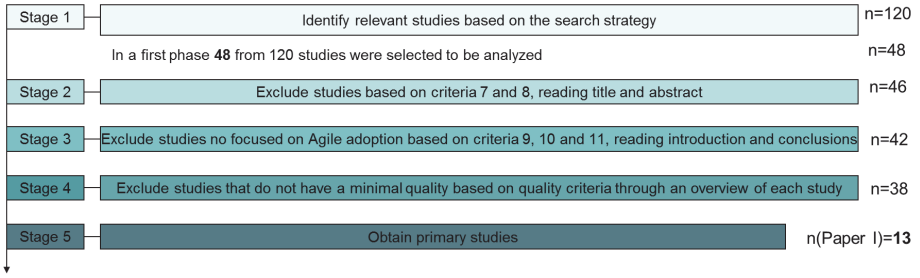


**Fig. 8. Literature analysis – studies selection process.**

Table 8 shows the search terms and the electronic databases used in Stage 1.

**Table 8. Literature analysis – Search terms and databases.**

| Search terms | Databases |
|---|---|
| agile AND adoption | ABI/Inform (ProQuest) |
| extreme programming AND adoption | Academic Search Premier (EBSCO) |
| xp AND adoption | Emerald Journals (Emerald) |
| scrum AND adoption | Science Direct (Elsevier) |
| crystal AND adoption AND (clear OR orange OR red OR blue) | ACM |
| dsdm AND adoption | IEEE Xplore - IEEE/IEE Electronic |
| dynamic systems development method AND adoption | Library |
| fdd AND adoption | |
| feature driven development AND adoption | |
| lean AND adoption AND software | |
| rational unified process AND adoption | |
| rup AND adoption | |
| adaptive software development AND adoption | |
| asd AND adoption | |

[20] In order to distinguish between dissertation research questions and phase specific research questions the acronym P(x) has been included at the beginning of each phase specific research question, in which x refers to the specific phase I, II or III.

Once the relevant literature was identified, the acceptance criteria were used to determine what studies were included or excluded in the literature review. Criteria and the stage where each criterion was applied (see Figure 8) are presented in Table 9.

**Table 9. Acceptance criteria applied in the literature analysis.**

| Criterion | Stage of application |
|---|---|
| *Inclusion criteria* | |
| 1. Studies of both students and professional software developers | Stage 1 |
| 2. Studies published between 2000–2009 | Stage 1 |
| 3. Studies that focus on Agile adoption or directly answer the research questions | Stage 3 |
| 4. Studies of both qualitative or quantitative research | Stage 3 |
| 5. Studies that pass the minimum quality threshold (see Appendix A, stage 4) | Stage 4 |
| *Exclusion criteria* | |
| 6. Studies not in English | Stage 1 |
| 7. Studies that are not research studies or reports with experiences or lessons learned such as prefaces, article summaries, overhead presentations, interviews, short-papers, introductions to special issues, tutorials or mini-tracks | Stage 2 |
| 8. Duplicate reports of the same study (only the most complete version of the study will be included in the review) | Stage 2 |
| 9. Studies not related to any of the research questions or external to Agile adoption | Stage 3 |
| 10. Studies without a rationale for why the study was undertaken | Stage 3 |
| 11. Studies whose findings or contributions are unclear or ambiguous | Stage 3 |

The review considered empirical studies (quantitative and qualitative), non-empirical studies and experience reports. Quality criteria for the quality assessment were defined differently for each kind of study, since their aims are different too. The quality criteria of empirical studies were more focused on study design and rigor of data collection and analysis. In non-empirical studies, study design, along with collection and analysis of findings were the most important. Finally, in the experience reports quality was measured through the quality of the reporting, credibility and relevance. Appendix A shows the data extraction form used to evaluate the quality of the studies (Appendix A, stage 4).

In this process, 120 studies were identified as relevant, based on the search strategy. Because of the limited resources, a random sample of the whole material repository composed of 48 studies was analysed in a first phase of the literature review. Using the acceptance and quality criteria, 13 studies from the sample were found to be relevant for the purpose of Paper I and were included as primary

studies in the analysis. For each primary study, the data extraction form was answered (see Appendix A, stage 5). Then, data synthesis was executed and the results were reported.

The findings of the literature analysis were validated with an industrial inventory, which was carried out among FLEXI industrial partners, in which the same topics were investigated. Materials coming from the FLEXI project's internal databases that could include some knowledge about the adoption of any Agile method were considered in the industrial inventory. Particularly, the following materials were used: i) Materials from the FLEXI publications, such as conference and journal publications. The publications were extracted from the FLEXI project's internal publication database. As there were 121 publications at the time of review, the list was manually reviewed, checking the acceptance criteria and possible duplications regarding to the literature analysis materials, ii) industrial trials performed in line with project goals that provide an experimental basis for the theoretical developments, iii) any available materials from the FLEXI meetings, such as presentations or work documents, were reviewed looking for experiences that could provide information about how Agile was being adopted by FLEXI organisations, iv) other materials such as PhD theses, Master theses, deliverables or internal documents. Acceptance and quality criteria were the same as those used in the literature analysis. The materials coming from the industry were fully analysed. The results of the first phase of the literature analysis, together with the results of the industrial inventory, clearly pointed out Lean thinking as an interesting and worthy topic for research. Therefore, it was decided to focus the work on studying Lean Software Development without continuing with the literature analysis.

### *3.2.2 Cloud Software Program*

The direction of the research was also highly influenced by the research project *Cloud Software Program* (2010), which began at the start of the thesis. The project proposal espoused Lean Software Development as a highly relevant approach for practitioners. Thus, its 22 industrial participants showed their interest in learning more about Lean thinking in a software development context. The webpage of the project[21] states:

[21] http://www.cloudsoftwareprogram.org/lean-software-enterprise (last accessed November 13, 2013).

*'It is the dream of every enterprise to have software developed cheaper, faster, and better […]. Release cycles are reduced from months to weeks and days, which requires novel ways of assuring quality, organizing the development and management. Agile approaches have been used to accomplish that dream partly. Recently, there has been serious thinking on adopting lean principles to accomplish cheaper-faster-better software. Through Cloud SW programme and particularly through Lean related research work, we will closely examine this very issue in more detail […].The aim is to go beyond agile to take into account the whole value chain. This requires lean enterprise thinking as an approach.'*

The Cloud Software Program helped to define the research problem and had a significant impact on the work. Industry partners in Cloud Software Program exhibited significant interest on learning more about Lean Software Development as well as ways to combine it with ASD. In addition, the Cloud Software Program provided an excellent framework for conducting the research by giving opportunities for having participants who were willing to serve in the study and resources for collecting data over a sustained period of time. Accordingly, the research questions of the thesis were aligned with key project goals such as '*Crystallising, analysing and providing concrete means to transform to Lean Software Enterprise*'.

## 3.3 Phase II: Status and Trends in Lean and Agile Software Development

Phase II explored the phenomenon of Lean and Agile software development from a broad perspective through a quantitative exploratory survey strategy. Two studies were conducted in this phase, which had an outcome of three papers (Papers II, III and IV). Subsections 3.3.1 and 3.3.2 describe the research setting of each study respectively.

### *3.3.1 Survey on Agile and Lean Usage in the Finnish Software Industry*

First, the software development industry was surveyed to identify the trends and the current status of the adoption of Agile and Lean methods. The main advantage of the survey studies is the economy of their design and their rapid turnaround in

data collection, which allows for the rapid first understanding of what the phenomena looks like. Specifically, this study helped to shape and limit the scope of the research, not just by improving the global understanding of the phenomenon, but also providing new insights that guided the work in Phase 3. Next, the research questions, as well as data collection and data analysis procedures, are described in detail.

*Research questions:* With the goal of understanding the phenomenon from different perspectives, five research questions were considered:

PII-RQ1. *What is the current state of adoption and usage of ASD and Lean Software Development methods and practices in the software industry?*

PII-RQ2. *What are the reasons why ASD and Lean Software Development are being adopted in some software development organisations?*

PII-RQ3. *What are the impacts, in terms of benefits, of using ASD and Lean Software Development?*

PII-RQ4. *What are the limitations and factors that can challenge the usage of ASD and Lean Software Development?*

PII-RQ5. *What are the reasons for some organisations for not using ASD and Lean Software Development?*

*Data collection - sampling strategy:* The goal of survey studies is to identify attributes of a large population from a small group of individuals. The target population of the study was the Finnish software development industry. Usually, identifying individuals inside the population is one of the challenges when conducting survey studies, due to the need of individuals who have the experience, time and motivation for properly answering the survey (Creswell 2009). In this study, the *Finnish Information Processing Association* (FIPA) constituted a good sampling frame, since a large part of software professionals in Finland are members of it. Moreover, in order to motivate individuals, a prize and an exclusive report with the results were offered to those who filled out the survey. FIPA provided the e-mail addresses of a subset of 4950 professionals, whose backgrounds were relevant to software development. This can be considered as a very large and representative sample of the population of software professionals and companies in Finland. 408 valid responses were collected, which represented 200 different organisations and constituted a response rate of 9%.

*Data collection - instrument:* The data was collected by means of an extensive explorative questionnaire containing almost fifty questions. The survey was developed by three researchers. Predefined response options in closed-ended questions, when used, were based on the literature. Figure 9 shows how the data collection connects with the research questions by relating the items on the survey instrument and the research questions. Besides some questions regarding the background information, the survey was composed of five sections to cover each of the research questions. Each section was composed of a number of items and measure in a scale, as indicated in Figure 9. Some sample items from the instrument are included in Appendix B, so that readers can see some examples of the actual items used.

*Data collection – timeline:* The survey was administered online using Webropol tool [22]. First, the survey was piloted with a selected group of practitioners for checking consistency and legibility issues in the questionnaire. Then, the Internet survey request was e-mailed to the sample provided by FIPA and the survey was open for two weeks. Three reminders were sent during the second week to those respondents who had not yet filled out the survey.

*Data analysis:* IBM SPSS Statistics software[23] was used to analyse the data. Papers II, III were the outcomes of this study. Paper II reports a descriptive analysis organised according to the sections of the survey and comparing with the results of earlier studies in the topic. The unit of analysis was company organisational unit. This design was considered to be more appropriate, since organisations, especially large ones, may have different units that may be at different levels of Agile and Lean usage. Therefore, it would have been impossible for a single person to answer for the whole company. In Paper III, further analysis was carried out with the group of Agile and/or Lean methods users. The goal of this analysis was to identify the reasons why Agile, Lean or a combination of both methods is being adopted in some software development organisations. For that, goals which importance varies depending on the adopted method were identified by analysing the association between the goal importance and the adopted method with Chi Square test of independence. For example, it was investigated whether the adoption of Lean Software Development and ASD depends on goals such as removing waste from the software development process

[22] http://w3.webropol.com/int/ (accessed November 12, 2013).
[23] http://www-01.ibm.com/software/analytics/spss/ (accessed November 12, 2013).

or reducing time-to-market. Goals that had a statistically significant association were selected in order to interpret which goals were more important in the function of the adopted method.

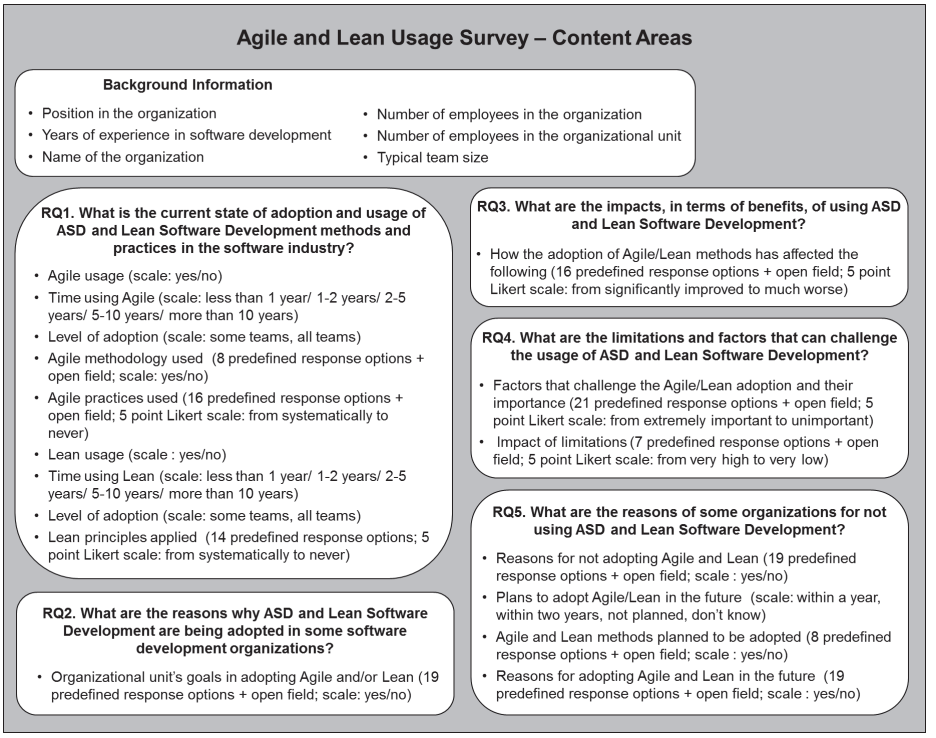


**Fig. 9. Agile and Lean Usage Survey structure.**

### *3.3.2 Organising Vision of Agile Software Development*

Following the survey, the vendor-sponsored white papers and IT and business magazine articles on ASD were surveyed to analyse the value foundations of Agile's organising vision, which conceptualises the community discourse on Agile. As stated in *Chapter 2*, Agile has already acquired considerable attention in the IT community. Thus, it can be said that Agile's organising vision has already been able to successfully mobilise the IT community. Consequently, it makes sense to study its organising vision. However, this is still not the status of Lean Software Development. So, it was decided to focus the study on ASD only.

*Motivation and research questions:* Previous research on IS and culture has exposed the impact of values on the adoption and diffusion of IT innovations, such as Agile (Baskerville and Myers 2009; Swanson and Ramiller 1997). Value compatibility has been proposed as a key barrier to adoption and diffusion of innovations (Leidner and Kayworth 2006). The results of the Agile and Lean usage survey, presented in the previous section, also evidence the cultural aspects as one of the top challenges when software-intensive organisations try to adopt Agile and Lean. The purpose of studying Agile organizing vision was to go deeper into the topic of ASD's value foundations by investigating how they have been publicised in the IT community discourse. Usually, innovators engage with the media to learn about the IT innovation potential in their organisation. Thus, research in IT discourse can help us to understand the impact of the company culture, as represented by values, in the adoption or diffusion of Agile methods. One research question drove the study as follows:

PII-RQ6. *How have Agile values been interpreted and legitimized in the IS community discourse by its contributors?*

*Theoretical foundations:* Drawing from a set of four values and twelve principles, a value orientation approach was already initiated by the *Agile Manifesto* (See *Chapter 2.3 Agile Software Development*). However these explicit values are not easily translated into an objective and accepted value framework. This study used the theory of organising vision, as introduced by Swanson and Ramiller (1997), as the theoretical framework to conceptualise the community discourse of Agile. Swanson and Ramiller theorised that IS community discourses do not develop randomly, but are the products of loosely coupled collaborations called *organising visions*. Organising visions provide the functions of interpretation, legitimatisation and mobilisation that shape how an IS innovation will be adopted and diffused. Although values were not specially addressed in the initial development of the theory, its base on the institutional theory, in which institutions are a collection of values, norms, beliefs and taken-for-granted assumptions, support the importance of values in the theory of the organising vision.

*Data collection:* Using a prestige sampling strategy, the data was collected from two sources of Agile's organising vision. The *Techrepublic.com* research database was used to get vendor-sponsored whitepapers on Agile. Business magazines articles published by *Businessweek*, *Informationweek*, *eWeek* and *CIO*

*magazine* were also collected from the ESBCO database. Both, vendor-sponsored whitepapers and articles found in IT and business magazines are key resources used by organisations and practitioners to learn about and promote Agile. For each source, 100 articles were selected so that a total of 200 articles were analysed.

*Data analysis:* Content analysis, as described by Stone *et al.* (1966) and the Lasswell value dictionary[24] were combined to analyse the encoded values in Agile's organising vision. Content analysis is a systematic and objective research method for making inferences from text. Values are normative elements encoded in media by organising vision contributors and are decoded by potential adopters. Computer-supported content analysis was used to make inferences about the interpreted espoused values encoded in the text and the moral legitimacy strategy of their contributors. Driven by the need for objectivity when comparing values across different sources, the Lasswell value dictionary, based on Lasswell's value theory, was used to code the data sources. The dictionary has eight value categories divided into two groups, as shown in Table 10.

**Table 10. Categories of Lasswell value dictionary.**

| Group | Category | Description |
|---|---|---|
| Welfare values | Wealth | Desire of income or services of goods |
| | Skill | Desire of proficiency in any practice whatever, trade or profession |
| | Enlightenment | Desire of knowledge, insight and information concerning personal and cultural relations |
| | Well-being | Desire of health (physical and psychological) |
| Deference values | Respect | Desires of status, honour, prestige and recognition |
| | Rectitude | Moral values, such as virtue, goodness and righteousness |
| | Power | Influence, particularly on the actions of others |
| | Affection | Values of love and friendship |

The General Inquirer[25] software program, developed by Harvard University, was used to code the text of the data sources. Two assumptions were made when analysing the texts. First, the frequency of the value category is an indication of the strength of the category. Second, the strength of the value category is an appropriate measure of the relative priority that the value has in the total value schema of each and all of the documents. The split-half technique was used to test

[24] http://www.wjh.harvard.edu/~inquirer/lasswell.htm (accessed November 13, 2013).
[25] http://www.wjh.harvard.edu/~inquirer/ (accessed November 13, 2013).

the sampling. Each sample was split into two sets and analyse them separately. The analysis provided the same results for each half sample indicating significance of the categories in the datasets. This study resulted in Paper IV of the dissertation.

## 3.4 Phase III: Analysing Lean and Agile Software Development in Specific Company Cases

Phase II showed a clear tendency towards using Lean thinking combined with Agile methods. It also exposed certain confusions in the main elements that characterise both approaches. In order to further investigate how Lean thinking and ASD were actually being performed in practice, two case studies were conducted in Phase III. The guidelines for conducting case studies by Yin (2009) and their adaptation to software engineering by Runeson and Höst (2009) were used for designing the research in this phase. Phase III was qualitative in nature. The intent of the qualitative research is to gather extensive and in-depth information from a small sample of individuals. Seaman (1999) explains that '*the principal advantage of using qualitative methods is that they force the researcher to develop into the complexity of the problem rather than abstract it away*'. Case studies are appropriated when focusing on contemporary events (Yin 2009) and are commonly used to study complex phenomena, which are hard to study in isolation (Runeson and Höst 2009). Both case studies in this dissertation aimed to answer similar research questions as follows:

PIII-RQ1. *How are Lean principles interpreted and implemented in a software development domain? What elements characterise the combination of Lean thinking and Agile methods in software development?*

PIII-RQ2. *What challenges are potentially faced when combining ASD and Lean thinking?*

PIII-RQ3. *What are the strengths of software-intensive companies in combining Lean thinking and Agile Software Development*?

The case studies were driven by the companies' desire to assess their transition to Lean and to design an assessment instrument for that purpose. Thus, companies had their own motivation for taking part in the study. The same research framework, composed by two sub-phases, was applied in both cases as depicted in Figure 10. First a material walkthrough workshop was conducted to introduce the researchers to the specific company Lean and Agile software development

processes. Then, focus group sessions were conducted with company representatives in order to design an assessment instrument, which composed by a set of statements defining the company's Lean and Agile processes, was used later to assess the company's transformation towards Lean and Agile software development. From a research perspective, discussions during focus groups enabled to explore the company interpretation and implementation of Lean thinking and ASD. In the second sub-phase, the assessment instrument resulting from sub-phase 1 was used among representative units inside the company to investigate challenges and strengths of the company when applying Lean and Agile software development. Thus, the first sub-phase focused on answering *PIII-RQ1* and the second on *PIII-RQ2* and *PIII-RQ3* respectively. This process resulted in Papers V and VI respectively.

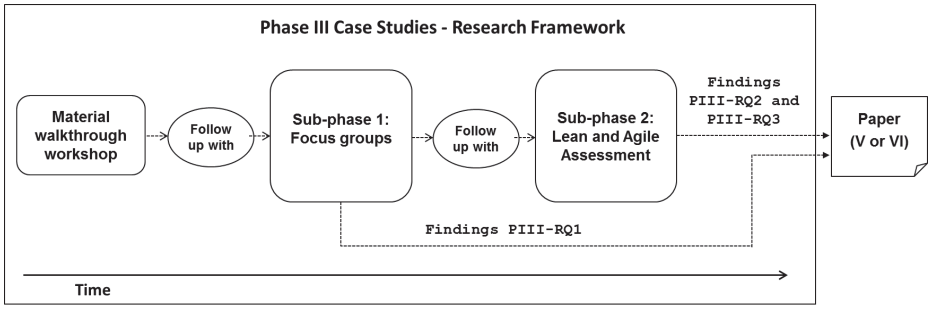


**Fig. 10. Case studies research framework.**

### *3.4.1 Cases Selection Strategy*

As advised by Creswell (2009) in conducting mixed methods research, a purposive case selection strategy[26] was followed in Phase III. According to Creswell, results from previous phase (quantitative phase) should inform the

[26] Creswell conceptualises this strategy as "*purposeful sampling*" (Creswell 2009). However, this dissertation uses the term "cases selection" instead of "sampling" to avoid confusion with the understanding of sampling in quantitative research, where the intent is to make generalisations, which implies the randomised selection of a subset of individuals from a statistical population with the objective of making inferences from the sample to the whole population. The term "purposive cases selection strategy" is used to refer to a strategy, where the subjects of the study are determined so that it is assured that the phenomena of interest exists and there also exist "good" informants who have particular knowledge of the topic, experience and are willing to articulate, reflect and share their knowledge about the topic.

types of participants to be purposefully selected in the qualitative phase and the types of questions that the participants will be asked. In this work, cases were selected considering: 1. the aims of the dissertation, 2. the findings of Phase II that indicated that most of the companies applying Lean Software Development also apply ASD, and 3. the likelihood of offering insight into the topic. Cloud Software Project (2010) constituted the framework for selecting the company cases. Thus, Cloud Software Project supported the research by providing access to representative software-intensive companies. Two companies that could be considered pioneers in the usage of Lean and Agile in software development were selected, Ericsson and Elektrobit. Subsections 3.4.1.1 and 3.4.1.2 describe each case context.

#### *3.4.1.1. Case A: Ericsson*

The first case study was conducted in cooperation with Ericsson[27]. Ericsson is a world-leading provider of telecommunications equipment and related services to mobile and fixed network operators. Ericsson is well known to be one of the most advanced companies in the usage of Lean thinking for software development (Poppendieck 2013). Ericsson R&D Finland, and its cooperating sites in Hungary and China, took part in the study. These units are developing a complex mobile network product using RoseRT/RSA-RTE, C++ and Java. The product is mature (10 years old) and engineers are highly experienced. Some teams are distributed across multiple sites. Specifically, Ericsson R&D Finland is a pioneer within the company in the adoption of Agile and Lean on large scale. The unit adopted Scrum in 2009 and started its transition towards Lean in 2010, involving a total of 400 people in the transformation. Its cooperating sites in Hungary and China were following the path of Ericsson R&D Finland. The organisation is committed with the transformation. Transformation leaders, team members and managers actively investigate ways to extract the maximum benefit from Lean ideas in the context of software development, creating their own interpretation of Lean Software Development. This case study is reported in Paper V.

[27] http://www.ericsson.com/ (accessed November 13, 2013).

#### 3.4.1.2. Case B: Elektrobit

The second case study was conducted in cooperation with Elektrobit[28]. Elektrobit is a large Finnish provider of automotive software products and wireless embedded systems. Specifically, the Elektrobit's Wireless Segment, which employs approximately 600 people who are distributed mainly in Finland and the United States, took part in the study. The segment offers wireless solutions for customer-specified devices in a no mass-manufacturing scale. The systems are developed using C++. Elektrobit's Wireless Segment has used Agile since 2007 and began to adopt Lean Software Development in 2010. As in the previous case, the Elektrobit Wireless Segment is also very committed to the transformation into Lean Software Development. Currently, all software development teams use Scrum or Kanban as the primary method. This case study is reported in Paper VI.

### 3.4.2 Data collection

The data was collected in two phases as follows:

- *Sub-phase 1 – Focus groups:* The data sources in the first phase were company documents describing their processes and focus groups conducted with experts guiding the transformation and employees applying Lean and Agile software development in different roles. There were two main reasons that motivated the use of focus groups. First, focus groups have the ability to proceed flexibly as an explorative research method, enabling discussions between experts, who query and explain to each other (Kontio *et al.* 2008). Both Ericsson and Elektrobit have thorough experience in these kinds of discussions. Thus, the experience of the companies minimised the typical weaknesses of this research method, such as unfocused discussions or dominant personalities hindering debates. Moreover, the outcomes of interactions between the participants offer information that it is difficult to obtain through to the sum of separate individual interviews (Morgan 1997). The intention was to get an overall picture of the phenomenon from different perspectives, rather than to learn from each individual. Thus, instead of carrying out a post individual analysis, as typically done when using interviews, participants had the responsibility of conducting discussions, until

[28] http://elektrobit.com/ (accessed November 13, 2013).

reaching an agreement about the most relevant attributes of their company's Lean/Agile software development processes. Secondly, the goal was to gather data from a wide range of experiences and perspectives, but still with enough technical substance for enriching our understanding. Focus groups are able to provide more in-depth information than methods such as surveys, still covering a relatively wide range of topics and participants.

Morgan (1997) and Kontio *et al.*'s (2008) guidelines were followed when conducting the focus groups. The focus group discussions were directed by the question '*What are the essential elements defining our company's Agile/Lean way of working that should be included in the tool to assess our transformation?*'. From a company perspective, the goal of the sessions was to define a set of statements that represent the most important aspects of their Lean and Agile processes, in order to create an instrument for assessing their transformation process. From a research perspective, statements defined during the focus groups helped understand how Lean principles were being interpreted and implemented in practice by espousing the main elements that characterise the companies' combination of Lean thinking and ASD. The number of focus group sessions was not limited at the beginning of the research. Sessions were conducted until everyone in the group was satisfied with the set of statements. In the end, 17 sessions involving 21 company representatives were conducted in the Ericsson case, and 5 sessions involving 6 company representatives in the case of Elektrobit. Session dynamics were quite similar in both cases. Each session lasted between 2 and 3 hours. They were based on discussions, in which the topics were recorded in form of statements using Excel spreadsheets. During the sessions, the Excel document was displayed on a big screen, so that it was visible to everyone participating in the session. Statements were created from scratch in the case of Ericsson. However, a guide was used to focus the discussions in Elektrobit case. The guide, containing the statements that could be a part of the assessment instrument, was created by the researchers using company's internal material and literature on the topic. The resulting assessment instruments were composed by 114 statements in the case of Ericsson, and 97 in the case of Elektrobit. Due to confidentiality, the final set of statements cannot be published. However, Papers V and VI contain a sample of statements with illustrative purposes. More details of the focus groups, such as participants profile for each case, can be found in respective Tables 2 of papers V and VI.

- *Sub-phase 2 – Lean and Agile Assessment:* In the second sub-phase, the statements resulting from sub-phase 1 were used to assess the transformation of the company towards the Lean and Agile software development. Different variants were used in this sub-phase. In the Ericsson case, the instrument, which was called *Team Amplifier*, was applied in team sessions, not only for identifying strengths and challenges, but also for further coaching Lean concepts during the sessions. 16 selected teams with different levels of Lean adoption maturity took part in the sessions. Participants scored each statement using the following scale: 0= *Not applicable/Do not understand/Do not see*; 1= *Not yet demonstrated*; 2= *Basic knowledge and skills*; 3= *Good knowledge and skills* and 4= *Highly developed knowledge and skills*. In the case of Elektrobit, the assessment was conducted in the form of a survey distributed to 226 staff from the Elektrobit Wireless Segment. In this case, the statements were evaluated by asking the respondents to rate the extent to which he/she agrees with each of the 97 statements, again using a five-point Likert scale, from *strongly agree* (5), *agree* (4), *neutral* (3), *disagree* (2) *to strongly disagree* (1), including also an option *'I cannot answer/not applicable*' (0).

### 3.4.3 Data analysis

The five core principles as defined by MIT researchers (Womack and Jones 1996) were used as the theoretical framework for analysing collected data. Therefore, the analysis was framed in the principles of value, value stream, flow, pull and perfection. Statements resulting from the focus group sessions (sub-phase 1) were analysed using coding techniques, as suggested by Miles and Huberman (1994). During the coding process, descriptive codes emerged as important concepts were identified (e.g. 'customer value', 'prioritisation', 'tool', 'communication', 'teamwork'). Descriptive codes were subsequently clustered into themes, and according to the five principles of Lean when possible, under the codes 'value/waste', 'value stream', 'flow', 'pull' and 'perfection', using interpretative codes. Afterwards, the codes were analysed to answer *PIII.RQ1* and a company representative reviewed the primary findings for validation.

The results from the sub-phase 2 (assessment) were analysed by calculating the average of each statement and identifying top challenges and strengths. In the case of Ericsson, in which the assessment was based on team sessions, the top three challenges and strengths for each team were identified according to the

scoring results, and patterns across teams were analysed. In the case of Elektrobit, using a survey based assessment, statements were shortened by lower averages (mean scores on the five points Likert scale) and those with an average lower than 3 (where most of the respondents disagreed or strongly disagreed) were considered challenges. In the case that more than one statement belonged to a specific topic, the average of the group was calculated. Statements with an average higher than 3.5 (where most of the respondents agreed or strongly agreed), were similarly analysed to identify the strengths of the company.

## 3.5 Phase IV: Synthesising the Results

The results from Phases I, II and III were synthesised in Phase IV using the guidelines by Cruzes and Dybå (2011) and Crewell (2009). A deductive approach was used to synthesise the data from the original publications. In particular, the six original publications were coded using a list of codes based on the research questions of the dissertation as presented in Table 11. Then, pieces of text under the same code were compared to identify the similarities and differences between findings. The quantitative results from the survey were explained in more detail through the qualitative data from the case studies.

**Table 11. Codes used for coding the original publications.**

| Code | Definition |
|---|---|
| [RQ1] | Discussion on how Lean and Agile are combined |
| Driver | Reasons why a combination of Agile and Lean has been adopted |
| Element/Principle | Rule that guides the application of the usage of Lean thinking in ASD |
| Element/Practice | Element that says how the fundamental are implemented in practice |
| Element/Agile | Element already established in ASD |
| Element/Lean | Element brought by Lean thinking |
| Element/[Lean principle] | Element supporting the principle of [value/value stream/flow/pull/perfection] |
| [RQ2] | Discussion on factors impacting the transformation towards Lean and ASD |
| Challenge | Factor that challenges the usage of Lean and ASD |
| Strength | Factor that was identified as being implemented in an exemplary or noteworthy way |
| [RQ3] | Discussion on impacts when using a combination of Lean and ASD |

**Table 12. Research questions overview and papers' contribution.**

| Thesis research questions | Specific phase research questions | Paper |
|---|---|---|
| RQ1: How is Lean thinking combined with Agile methods in software development? | Phase I. Literature analysis and industrial inventory | |
| | PI-RQ1: What are currently the strategies to adopt Agile methods that are used in the software market? | Paper I |
| | Phase II. Survey on Agile and Lean Usage | |
| | PII-RQ1: What is the current state of adoption and usage of ASD and Lean Software Development methods and practices in the software industry? | Paper II |
| | PII-RQ2: What are the reasons why ASD and Lean Software Development are being adopted in some software development organisations? | Papers II and III |
| | Phase III Case studies | |
| | PIII-RQ1: What elements characterise the combination of Lean thinking and Agile methods in software development? | Papers V and VI |
| RQ2: What are the key factors that influence the successful transformation of software organisations towards Lean and Agile methods? | Phase II. Survey on Agile and Lean usage and analysis of the Agile's organizing vision | Paper II |
| | PII-RQ4: What are the limitations and factors that can challenge the usage of ASD and Lean Software Development? | Paper II |
| | PII-RQ5: What are the reasons for some organisations for not using ASD and Lean Software Development? | |
| | PII-RQ6: How have Agile values been interpreted and legitimised in the IS community discourse by its contributors? | Paper IV |
| | Phase III Case studies | |
| | PIII-RQ2: What challenges are potentially faced when combining ASD and Lean thinking? | Papers V and VI |
| | PIII-RQ3: What are the strengths of software-intensive companies in combining Lean thinking and ASD? | Papers V and VI |
| RQ3: What are the impacts perceived by practitioners when using a combination of Lean thinking and Agile Software Development? | Phase II. Survey on Agile and Lean Usage | |
| | PII-RQ3: What are the impacts, in terms of benefits, of using ASD and Lean Software Development? | Paper II |
| | Phase III Case studies | |
| | Close collaboration with companies and practitioners enabled identifying some perceived benefits. | Papers V and VI |

As presented in Table 12, the manner in which Lean is combined with ASD (RQ1) was analysed by using the empirical evidence as provided in Papers I, II, III, V and VI. Specifically, the sources of evidence were i) the knowledge acquired in the literature analysis and industrial inventory on strategies to adopt Agile in large scale software development (PI-RQ1); ii) empirical evidence from the Agile and Lean usage survey, at the level of adoption of specific methods and

practices (PII-RQ1) and adoption drivers (PII-RQ2) and iii) elements characterising the combination of Lean thinking and Agile methods as identified in the case studies (PIII-RQ1).

The second research question of the dissertation focused on the factors that influence the successful transformation towards Lean and Agile methods (RQ2). Papers II, IV, V and VI contributed to answer this research question. Specifically, i) the empirical evidence from the Agile and Lean usage survey with respect to limitations and challenges when using Lean and Agile (PII-RQ4) and reasons for not using these methods (PII-RQ5); ii) the results of the assessment conducted in the case studies (PIII-RQ2 and PIII-RQ3) and iii) the findings from the analysis of the values that benefit or hinder the adoption of Agile methods (PII-RQ6).

Finally, the consequences of using a software development process based on a combination of Lean and Agile (RQ3 of the dissertation) were answered using Papers II, V and VI. Particularly, i) the empirical evidence from the Agile and Lean usage survey on the impacts of using ASD and Lean Software Development (PII-RQ3); and ii) the impacts perceived by practitioners in the case studies (PIII).

An important aspect when designing the research and how results from different sources of evidence will be integrated and synthesised is to consider the particular context information of each of the sources, specially the '*information that will help in understanding and interpreting the findings of the study*' (Cruzes and Dybå 2011). In the specific case of this dissertation, the Agile and Lean usage survey provided an overview about the usage of these methods. Then, these findings were explored in more detail through two case studies, which were selected following a replication logic strategy. Thus, Ericsson and Elektrobit share common characteristics which make them comparable. Units studied in both cases were focused on software development and had a medium size. Although developed products were different in both companies, they can be considered complex, in the domain of telecommunications and using similar programming languages such as C++. Both companies were committed in transforming their processes from following basic Agile principles to complement them with Lean principles. Their development process before introducing Lean thinking was based in ASD. Thus, both Ericsson and Elektrobit had mature Agile development teams, composed by highly experienced engineers, using Agile for more than three years. Moreover, both cases show success in their respective transformations as it is reported in *Section 5.3*. In the same line, the same research framework was followed in both cases. The same research questions guided the

work and similar procedures were used for data collection and analysis, which benefits the later comparisons of evidence.

Table 13 summarises the research design of the dissertation, including the research phases, the goal of each research phase, the specific research method applied in each phase and the final outcome in paper form or, in the case of Phase IV, a chapter in this dissertation.

**Table 13. Research methods overview.**

| Research Phase | Goal | Research Methods | Outcome |
|---|---|---|---|
| I. Literature analysis and industrial inventory | To determine the research problem | Systematic literature review | Paper I |
| II. Quantitative surveys | To analyse the status and trends in ASD and its combination with Lean Thinking | Exploratory survey<br>Descriptive statistics<br>Statistical tests | Paper II<br>Paper III |
| | To analyse the Agile's organising vision | Content analysis | Paper IV |
| III. Qualitative case studies | To deeper analyse how the combination of ASD and Lean is happening in practice | Focus groups<br>Coding Techniques | Papers V and VI |
| | To find strengths and challenges when combining of Lean thinking and ASD | Exploratory survey<br>Descriptive statistics | |
| IV. Synthesis | To draw conclusions and implications | Thematic Synthesis | Chapter 5 |

In addition, Figure 11 presents a map where the research gaps, research objective, research questions (thesis level) and research questions (paper level) are shown.

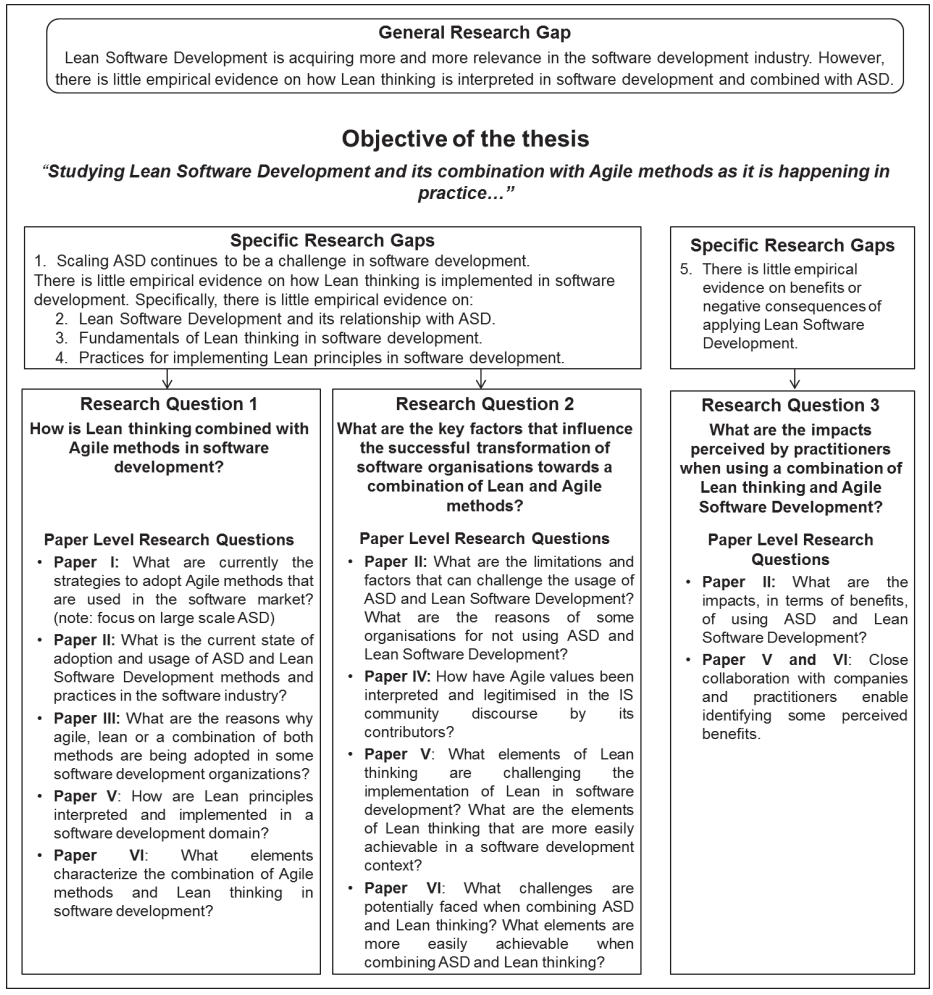


**Fig. 11. The relationship between research gaps, research objective, research questions (thesis level) and research questions (paper level).**

# 4 Original Contributions

This section presents the publications that compose the dissertation. The dissertation consists of 6 papers published in peer-reviewed international conferences in the fields of software engineering and information systems as follows: *Product-Focused Software Process Improvement (PROFES) 2010 and 2012, ACM-IEEE international symposium on Empirical Software Engineering and Measurement (ESEM) 2012, European Conference on Information Systems (ECIS) 2012, International Conference on Software and Systems Process (ICSSP) 2013 and Hawaii International Conference on System Sciences (HICSS) 2014*. ESEM, ECIS, ICSSP and HICSS are A level conferences according to CORE ERA 2008 Ranking[29], whilst PROFES is B level. Table 14 summarises the publications' scope and contribution to the thesis, and the author's involvement in them. Each paper is elaborated in more detail in the following subsections.

**Table 14. A summary of the publications that compose the dissertation.**

| Publication | Purpose of the study | Highlights | Author's contribution |
|---|---|---|---|
| I: Rohunen A, Rodríguez P, Kuvaja P, Krzanik L, Markkula J. 'Approaches to agile adoption in large settings: a comparison of the results...' PROFES (2010) | To analyse the state of the art on existing large-scale Agile adoption frameworks to identify needs and opportunities for future research. | Most studies view the adoption process from somewhat high-level perspective. Lean is emerging as a promising approach to effect top-down transformation. | Major involvement in all phases of the research. |
| II: Rodríguez P, Markkula J, Oivo M, Turula K. 'Survey on agile and lean usage in Finnish software industry'. ESEM (2012) | To investigate the status and trends in Agile and Lean usage in the software development industry. | Tendency towards combining Agile and Lean. Organisational culture and top management support are the main challenges in the adoption of Agile and Lean. | Major involvement in all phases of the research. Main author of the paper. |

[29] The ERA Conference Ranking Exercise, 2008. The Computing Research and Education Association of Australasia (CORE). http://core.edu.au/index.php/categories/conference%20rankings/1 (accessed November 13, 2013).

| Publication | Purpose of the study | Highlights | Author's contribution |
|---|---|---|---|
| III: Rodríguez P, Markkula J, Oivo M, Garbajosa J. 'Analyzing the drivers of the combination of lean and agile…' PROFES (2012) | To characterise the combination of Lean and Agile in SW development and to analyse the drivers that motivate it. | Drivers: flexibility and economical efficiency. Agile and Lean combined without time and space restrictions. | Major involvement in all phases of the research. Main author of the paper. |
| IV: Lawrence C, Rodriguez P. 'The interpretation and legitimization of values in Agile's organizing vision' ECIS (2012) | To analyse the organising vision of Agile methods to find its value foundations. | A combination of methodological and business foundations. Wealth, enlightenment, skill and power values are found as strong values in ASD. | Major involvement in study design and write-up. Supporting data collection and analysis. |
| V: Rodríguez P, Mikkonen K, Kuvaja P, Oivo M, Garbajosa J. 'Building lean thinking…' ICSSP (2013) | To analyse how Lean thinking and ASD are combined in practice, and the factors impacting its usage. | Lean principles guide the implementation of Agile methods. Profound culture and mindset change. | Major involvement in all phases of the research. Main author of the paper. |
| VI: Rodríguez P, Partanen J, Kuvaja P, Oivo M. 'Combining Lean thinking and Agile methods…' HICSS (2013) | To analyse how Lean thinking and ASD are combined in practice, and the factors impacting its usage. | Incremental improvement with the adoption of Lean thinking. Transparency essential and supported by tools such as Jira. | Major involvement in all phases of the research. Main author of the paper. |

## 4.1 Paper I: Approaches to Agile adoption in large settings: a comparison of the results from a literature analysis and an industrial inventory

The work on the thesis started with this study, which was conducted in 2009 and published in 2010. Paper I aimed to answer the research question: *What are currently the strategies to adopt Agile methods that are used in the software market?*. The results of the study provide extensive knowledge in what academia proposed for the adoption of Agile, especially in large-scale software development, and what the status of Agile adoption in the industry was at that moment. Through a literature analysis, it was found that Agile adoption

frameworks usually consider an incremental transformation, as opposed to a wholesale strategy. Continuous improvement was found to play a fundamental role for the incremental transformation. Moreover, three elements characterised the frameworks: stages, measurement models for guiding the adoption and dependences between practices (i.e. the order in which the practices should be adopted in order to have a successful transformation). However, with very few exceptions, frameworks found in the literature were considered to have very high-level strategies, lacking technical substance and failing to provide clear empirical evidence and indications on how the framework should be used, and how the adoption should be actually carried out.

The industrial inventory added important insights into the literature analysis. The importance of using both bottom-up and top-down strategies was emphasized in the materials coming from the industry. Moreover, Lean Software Development was pointed out as a worthy approach for effecting top-down adoption and extending Agile beyond the team boundaries. Continuous learning was also found to be essential for inspection and adaptation purposes. Moreover, the multidimensional nature of the concept of agility and the importance of defining agility goals and enabling factors in accordance to the level of agility aimed by the company were stressed.

From the point of view of the dissertation, one of the main contributions of Paper I was that it highlighted the industry interest in Lean thinking as a means for achieving enterprise agility, and obtaining the benefits of Agile in the whole organisation (Vilkki 2008). Thus, it was decided to focus the thesis on Lean Software Development as the central idea to explore.

## 4.2 Paper II: Survey on Agile and Lean usage in the Finnish software industry

The second paper of the thesis reports the results of an extensive survey conducted in Finland in order to explore trends in Agile and Lean adoption in software-intensive companies. Due to the lack of clarity in the phenomenon, it was decided to conduct an exploratory survey study with the goal of providing up-to-date results of the real-world status of the usage of these methods. It was conducted in 2011 and 408 responses from 200 software-intensive organisations were collected.

Figure 12 summarises the main findings of the study. The figure is structured according to the five research questions that guided the study (see *Section 3.3.1*).

Boxes 1 and 3 highlight the main results related to the current state of ASD and Lean Software Development adoption (PII-RQ1). The reasons for adopting ASD and Lean Software Development (PII-RQ2) are presented in the box number 2. The impacts, in terms of benefits, of using ASD and Lean Software Development (PII-RQ3), the limitations and factors that can challenge the usage of ASD and Lean Software Development (PII-RQ4) and the reasons for some organisations for not using ASD and Lean Software Development (PII-RQ5) are presented in boxes 5, 4 and 6 respectively.

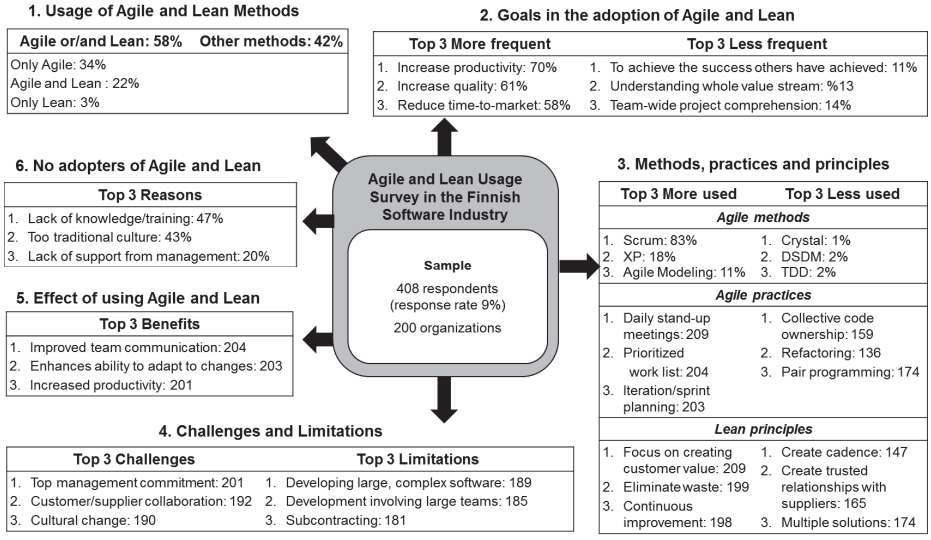


**Fig. 12. Agile and Lean usage survey results.**

According to the results a majority of respondents' organisational units used Agile and/or Lean methods (58%). Agile seemed to be more popular than Lean. However, a tendency towards combining both paradigms was also emerging (22%). Whilst Scrum was by far the most applied method, Agile practices were generally used without big differences between them. Regarding the Lean principles, focusing on creating customer value and eliminating waste and excess activities appeared as the most popularly applied principles. As highlights, the results of the survey showed important implications for management, since management support appeared as a limitation to adopt these methods. The company culture was also found to limit the successful adoption of Agile and Lean. In addition, the results of the survey evidenced a lack of clarity in some

concepts. For example, Kanban and TDD were pointed out as Agile methods by some respondents. However, Kanban is a Lean technique, whilst TDD is a practice of XP rather than an Agile method itself.

This study extends previous work that was carried out in similar surveys on Agile adoption (e.g., VersionOne 2011; West *et al.* 2010) by also analysing trends in Lean Software Development. The work is particularly relevant because most of the earlier surveys were conducted by IT consultants, who may have had a vested interest in the results. From the point of view of the dissertation, this study motivated and provided the basis for the case studies reported in Papers V and VI. The identified challenges and limitations create significant opportunities for further research. Moreover, the study provides up-to-date results that can be used by organisations implementing or planning to implement Agile and/or Lean methods through a better understanding of the state of practice.

### 4.3 Paper III: Analysing the drivers of the combination of Lean and Agile in software development companies

Paper III further elaborates the drivers behind the combination of Lean and Agile in software development companies in order to answer the research question *RQ1.1 Why are Lean thinking and Agile methods combined in software development?*. Supported by the results of the survey presented in Paper II, the relationship between Agile and Lean in software development is analysed. The paper makes two main contributions as follows:

First, the study identifies the drivers behind the adoption of Agile and/or Lean, where importance of the driver is varying depending on the adopted method. The results suggest that software development companies typically use a combination of Agile and Lean to achieve the best of the two worlds, namely flexibility and economical efficiency. Agile is not thought of as '*another fad*', but it is deliberately kept when moving towards Lean.

In addition, the study reflects on original work from manufacturing, a field in which the combination of Agile and Lean is more mature, to characterise the combination of Agile and Lean in a software development context. Particularly, the theory of *Le-agility* (Ben Naylor *et al.* 1999), which takes into account the space and time dimensions when combining Agile and Lean, was examined from a software development perspective. The manufacturing domain has traditionally understood that Agile and Lean could work well together, since Lean's capabilities can contribute to Agile's performance. However, according to the *Le-*

*agility* theory, Agile and Lean cannot be employed in integrated processes in the supply chain, because the aims of responsiveness of Agile (flexibility) can sacrifice the foundations of Lean (efficiency). The main reason behind this is the unstable demand that characterises environments where Agile is used. Unstable demand requires flexibility, which is difficult to satisfy with a levelled schedule and upfront planning as requested by Lean. Thus, the theory of Le-agility proposes to combine Agile and Lean according to three strategies: 1. in different value streams, 2. in the same product but at different points in time or 3. at different points of the value stream (different points in space) using de-coupling strategies. However, as empirically analysed in this dissertation, Agile and Lean are combined in software development without considering the time and space restrictions, mixing Agile and Lean techniques in the same product and at the same time. The paper reflects on how the particularities of software products open different possibilities for combining Agile and Lean methods. For example, whilst Agile in manufacturing emphasises flexibility in both volume and variety of products, in software development flexibility mainly impacts variability (changing requirements in software products) but not volume, since the entire development process produces a single copy of the software product that can be easily replicated. In conclusion, the importance of a careful selection of appropriate aspects of both paradigms, appertaining to the particular organisations strategy in terms of flexibility and economical efficiency, is pointed out in the study. Moreover, researchers and practitioners should look at Lean thinking from a software development angle, considering software development and software product particularities, and avoiding simply unmodified adoption of elements that work in the manufacturing domain.

## 4.4 Paper IV: The interpretation and legitimisation of values in the Agile's organizing vision

The results of the survey on Agile and Lean usage in the Finnish software industry suggest that the cultural change that the adoption of ASD and Lean thinking implies is one of the main challenges for adopting this kind of methods. According to Tolfo *et al.* (2011), '*the organizational culture corresponds to the values and beliefs of the company's staff, and it guides their behavior*'. Therefore, how value foundations impact the adoption of ASD merited further study, which was conducted in Paper IV.

While the foundations of Agile methods were portrayed in the *Agile Manifesto* in the form of a set of values and principles, how they are actually being represented in the Agile community discourse has not been studied. Paper IV analyses the organising vision of Agile in order to understand its value foundations and find values that will potentially impact the adoption and diffusion of ASD. The research question *PII-RQ6 how have Agile values been interpreted and legitimized in the IS community discourse by its contributors?* guided the study. Two samples of the Agile's organising vision were evaluated, a sample composed by 100 vendor-sponsored white papers and another composed by 100 IT and business magazine articles.

The results reveal Agile's value patterns across the two samples. Specifically, the values[30] of enlightenment, power and, to a lesser degree, wealth and skill appeared as strong values in the Agile's organising vision. Conversely, values such as rectitude, respect, affection and well-being showed significantly lower occurrences in both datasets. Comparing these results with values in the Agile manifesto, similarities can be found. Competence and responsibility of individuals are valued in Agile. Valuing enlightenment and skill supports the view of software development as a creative endeavour, supporting values of the Agile Manifesto such as '*working software over comprehensive documentation*' and '*continuous attention to technical excellence and good design enhances agility*'. However, wealth, as a value expressing desire for income, and power, referring to influence on the actions of others, are not explicitly considered in the Agile Manifesto. Thus, the results suggest a combination of the Agile value foundations as espoused through the Agile Manifesto and practical organizational desires (i.e. values wealth and power, not explicitly included in the Agile Manifesto, express values related to practical business needs in Agile's community discourse).

Papers II, III and IV provided insights into the trends in the adoption of ASD and Lean thinking. In addition, two case studies were conducted to get a more in-depth understanding of how the combination of ASD and Lean thinking actually happens in practice. The main findings of these case studies, which were reported in Papers V and VI, are summarized in the next two sections.

[30] A description of these value categories is presented in Table 10 (Chapter 3, Section *3.3.2 Organizing Vision of Agile Software Development).*

## 4.5 Paper V: Building Lean thinking in a Telecom software development organisation: Strengths and challenges

Paper V presents the case study conducted with Ericsson R&D Finland and its cooperative sites in Hungary and China. As explained in Chapter 3, the case studies were conducted in two phases. The first phase, based on focus group sessions, focused on answering the first research question of Phase III,

PIII-RQ1. *How are Lean principles interpreted and implemented in a software development domain? What elements characterise the combination of Lean thinking and Agile methods in software development?*

The data collected during the focus group sessions indicates that the transformation at Ericsson has impacted the whole development chain, inculcating a profound change of culture and thinking. As illustrated in Figure 13, Lean principles at a more philosophical level guide the implementation of Agile methods at a more prescriptive level. The purpose is to create a learning organization that is able to adapt to fluctuant customer demands.

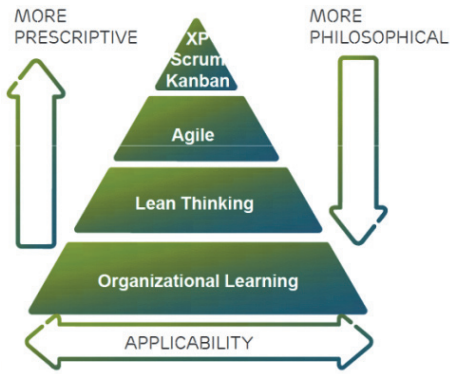


**Fig. 13. Lean and Agile in context at Ericsson R&D (Source: Paper V).**

Lean principles are implemented in the context of ASD under a flexible Scrum framework and, therefore, many of the fundamental elements of Ericsson R&D Finland's processes come from Agile methods. '*Everything that is done in the organisation is adding value*' is the key principle that guides each activity. Elements that were found to be fundamental in order to support Ericsson in their

Lean and Agile processes are summarised in Table 15. They are structured according to the five core principles of Lean thinking (Womack and Jones 1996)[31].

**Table 15. Main elements of the combination of Lean and Agile methods in Ericsson.**

| Lean principle | Elements supporting the principle implementation in practice |
|---|---|
| Value and Value Stream | Flexible releases composed of clear and inspiring feature/user stories following techniques such as INVEST (Independent, Negotiable, Valuable, Estimable, Small and Testable) and MMF (Minimum Marketable Feature).<br>Network of product owners.<br>Focus on quality for offering the best possible solution to the customer.<br>Managing products as a whole by focusing on the big picture of the product.<br>Transparency supported by R&D team areas, which improves communication, cooperation and faster feedback loops, and go-and-see principle for managers.<br>Value Stream Mapping. |
| Flow and Pull | Flatter organisation with a maximum of three organisational layers.<br>Scrum and Kanban.<br>Continuous integration.<br>Work-In-Progress limits.<br>Avoiding extra-standardization. |
| Perfection | Continuous improvement and learning through:<br>Team retrospectives.<br>Communities of practice and open spaces.<br>Cross-functional teams to create a collective knowledge culture.<br>Team experiments.<br>Stop-the-line principle.<br>Network of Scrum masters and coaches. |

Besides the elements supporting the implementation of the five principles of Lean, the people factor was found to be essential and transversally impacting each principle. Thus, it was found that '*Respect people*' is a central aspect in the company. The company believes that speeding up the development process is only possible when people have a personal initiative, and when team work, motivation, self-organisational and empowerment are in place.

[31] Elements usually have synergies so that an element may contribute to implement more than one principle. However, in order to structure the findings in a meaningful way, elements have been structured in accordance to the most relevant principle that they contribute.

In the second phase of the study, 16 teams were evaluated in order to identify their status with respect to their usage of Lean and Agile. The research questions in this phase were *PIII-RQ2 What challenges are potentially faced when combining ASD and Lean thinking? and PIII-RQ3 What are the strengths of software-intensive companies in combining Lean thinking and ASD?*. Achieving flow, making the development process transparent and creating a culture of continuous learning were found to be the main challenges. Creating a culture of continuous improvement, involving people in the transformation and creating a team culture were the main strengths of the company in its Lean and Agile processes.

Regarding the consequences of the Agile and Lean way of working, the experience of the company so far has been positive. Transformation leaders informed that the company has made important improvements in the quality of its products, customer satisfaction and transparency within the organisation. Moreover, build times have been reduced more than ten times, and the number of commits per day has increased by roughly five times.

## 4.6 Paper VI: Combining Lean thinking and Agile Methods for software development. A case study of a Finnish provider of wireless embedded systems

Paper VI presents the second case study conducted in collaboration with the Elektrobit Wireless Segment. Again, the first phase of the study focused on analysing how Elektrobit Wireless Segment was interpreting and implementing Lean thinking in its software development (PIII-RQ1). As in the case study conducted with Ericsson, it was found that elements characterising their Agile and Lean processes concentrated on aspects already known in ASD, with a Lean flavour that extend Agile concepts. Thus, rather than a radical transformation, an incremental transformation was applied. Figure 14 summarizes the findings in the first phase of the study. The main elements that characterize the combination of Lean thinking and ASD at Elektrobit Wireless Segment are structured according to the five principles of Lean. Moreover, elements have been clustered in two groups. One group contains elements that belong to ASD (on the left side of the figure) and the other group is composed by elements that have been brought in the top of Agile by the adoption of Lean thinking (on the right side of the figure).

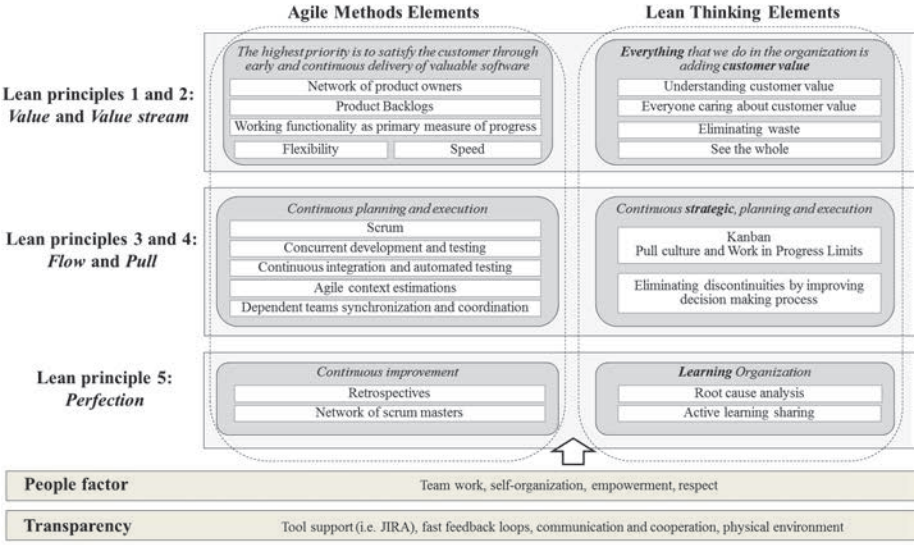


**Fig. 14. Main elements of the Elektrobit Wireless Segment's Agile and Lean way of working (Source: Paper VI).**

In addition to well-established practices in Agile, Lean thinking has brought new elements to Elektrobit Wireless Segment's processes, such as extending the responsibility of caring about customer value from product owners to everyone in the organisation, continuous planning and execution not just in development teams but also in the organisational strategy, focus on eliminating waste to make the development process more efficient, managing products as a whole by focusing on the big picture, a pull culture supported by Kanban and WIP limits, promoting a learning organisation through continuous improvement, root cause analysis of problems, and encouraging the sharing of learning. As depicted in Figure 14, transparency and people oriented development were especially stressed. Transparency refers to the ability to increase visibility of customer value, plans and roadmaps, and development status. Tool support was found to be fundamental to achieve transparency and shorter feedback cycles. Moreover, respect for people, team work, self-organisation and empowerment characterised the Elektrobit Wireless Segment's people orientation.

Table 16 presents the results of the second phase of the study, which analysed the challenges faced when combining ASD and Lean thinking and elements that were strengths in the company case (PIII-RQ2 and PIII-RQ3). People related aspects were usually found to be strengths in the company. Moreover, focusing on

customer value and achieving transparency also scored high in the survey. However, accomplishing company level agility, which means the ability of the organisation to include changes in products during development, and management support were found to be main challenges.

**Table 16. Top five strengths and challenges at EB Wireless Segment.**

| Strengths | Avg. | Challenges | Avg. |
|---|---|---|---|
| Implementation set-up | 4.16 | Flexibility | 2.69 |
| Respect people | 3.90 | Business management tasks | 2.72 |
| Self-organisation | 3.71 | Waste reduction | 2.82 |
| Focus on customer value | 3.68 | Synchronization and coordination | 2.86 |
| Transparency | 3.63 | Short feedback loops | 2.87 |

Regarding the impacts of using a combination of Lean thinking and ASD, company's internal metrics evidence that Elektrobit Wireless Segment has perceived improvements in three areas: i) productivity has increased, more than 30% in some areas, ii) customer satisfaction has been improved and iii) the work environment has been improved to better encourage and support the generation of new ideas through better information and expertise sharing.

# 5 Findings and Discussion

This chapter synthetizes the findings of the empirical studies introduced in Chapter 4. The synthesis was conducted as described in *Section 3.5*. Eight general findings were identified as summarized in Figure 15. These general findings are further developed in Sections 5.1, 5.2 and 5.3, where the three research questions of the thesis are answered. The findings are examined in the light of the current knowledge in order to determinate whether they are supported by the extant research. Their implications for both practice and research are summarized in Sections 5.4 and 5.5.

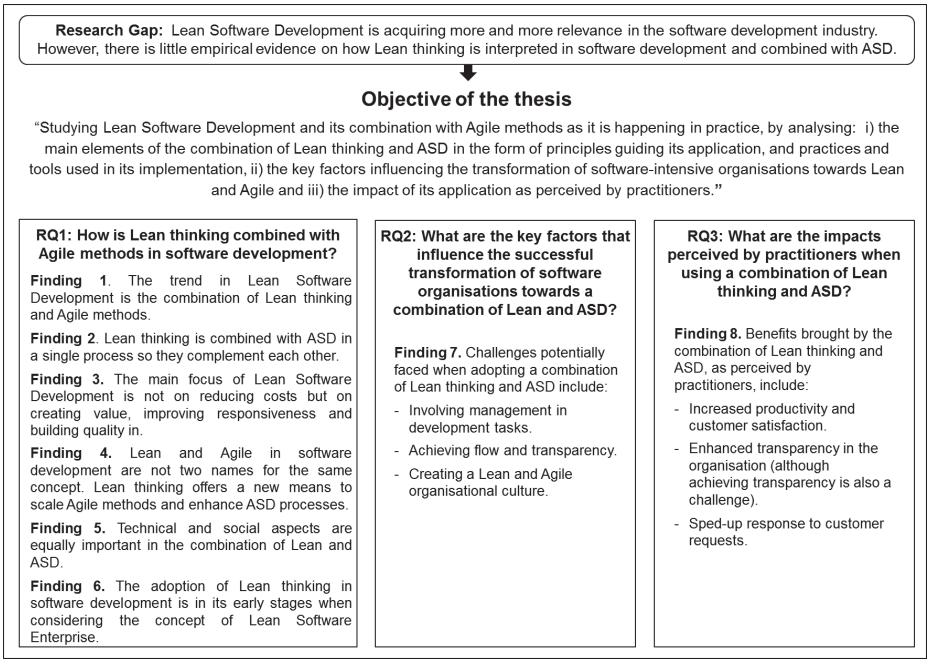


**Fig. 15. Summary of the main findings of the thesis.**

## 5.1 Research Question 1: How is Lean thinking combined with Agile methods in software development?

The first finding of the dissertation (Figure 15, Finding 1) is that software-intensive companies in Finland seriously consider the usage of Lean thinking in

their development processes. The survey study, which was reported in Papers II and III, revealed that 58% of respondents used Agile and/or Lean methods, and 24% said they used Lean. The rate of Agile adoption was slightly lower than previously reported by IT consultants (Ambler 2008; West *et al.* 2010; Version One 2011) which, however, may have been because of vested interests in their reports. Nevertheless, Agile seems to have a strong position in the Finnish software industry. On the other hand, Lean thinking has been adopted much more frequently than previous studies indicated (Version One 2011). Based on the survey data, the trend in Lean Software Development is towards combining it with ASD (21.6% of the respondents indicated using a combination of Lean thinking and ASD). Thus, the use of Lean in isolation is quite uncommon (2.7% of respondents indicated this option). The combination of Lean and Agile is supported by the studies conducted at Ericsson and Elektrobit (Papers V and VI).

The case studies evidenced numerous compatibilities between Lean and Agile. Agile, which has been adopted earlier by the two companies, was not abandoned when Lean thinking was considered. Instead, Lean was incorporated into a process that combines elements of both approaches according to what works better for the company. Previous studies also support the combination of Agile and Lean (Wang *et al.* 2012; Pernstål *et al.* 2013; Ahmad *et al.* 2013; Trimble and Webster 2013). In both case studies, the adoption of Lean in ASD followed category E in the classification established by Wang *et al.* (2012): '*purposeful application of Lean approaches in ASD – Comprehensible application of Lean approaches to transform Agile processes*'. Therefore, Lean and Agile seem complementary in software development (Figure 15, Finding 2). Next, how the combination of Lean thinking and ASD is actually implemented in practice is exanimated through the findings related to the sub-research questions RQ1.1, RQ1.2 and RQ1.3.

> *RQ1.1: Why are Lean thinking and Agile methods combined in software development?*

Drivers of using Agile and Lean appeared well aligned with previous research (Version One 2011; Vijayasarathy and Turk 2008). Productivity, quality and time-to-market were the most significant according to the survey on Agile and Lean usage in the Finnish software industry (see Paper II). The industrial inventory conducted in Paper I suggested that companies felt the need to use both bottom-up and top-down strategies in parallel in order to scale Agile to the organisational level. Paper I presented Lean Software Development as a strategy to define and

operate top-down transformation (Vilkki 2008). The analysis in Paper III suggested that achieving both flexibility and efficiency motivates the combination of Agile and Lean. Flexibility is maintained when Lean thinking is adopted. In fact, flexibility is seen as necessary in order to provide customer value. In the specific case of Ericsson (Paper V), the adoption of Lean thinking was prompted as an improvement activity. In the company's opinion, their processes, which were based on Agile methods, were satisfactory for the present but deemed insufficient for the future. Creating the most value, improving responsiveness, building in quality and empowering people were the specific drivers for adopting Lean thinking at Ericsson R&D Finland. Thus, it can be concluded that Lean Software Development appears an incremental improvement with regard to ASD, with the purpose of not only scaling Agile methods up but also enhancing software development processes from a wider perspective. In addition, although the literature on Lean thinking suggested that the spur of a previous crisis precedes the adoption of Lean (Womack and Jones 1996), no empirical evidence supporting this assertion was found in this work.

Next, the main elements that characterise the combination of Lean and Agile in a software context are analysed, particularly those elements that the adoption of Lean thinking has brought on top of practices that pre-date the Lean Software Development movement.

*RQ1.2: What main elements characterise Lean Software Development and its combination with Agile Software Development?*

The main elements characterizing the combination of Lean thinking and Agile methods in a software domain are shown in Tables 17, 18, 19 and 20. The original five principles of Lean thinking defined by IMVP researchers (Womack and Jones 1996) were used as a lens to frame the analysis in the original publications. They are also used in the synthesis presented in this section. Lean principles and the practices that support the implementation of those principles are synergistic. For example, removing waste is the counterpart of the principle of value, and it is closely connected to the principle of perfection and continuous improvement. Similarly, Kanban, which favours the principles of flow and pull, is also a tool used to detect bottlenecks and improve the development process. To make a comprehensible synthesis and structure the findings in a meaningful way, elements that fit multiple Lean principles have been classified in accordance to the most relevant principle that they contribute. Elements that characterize the combination of Lean Software Development and ASD were classified in two

levels: 1) principles and 2) practices and tools. Principles refer to rules that guide the application of the usage of Lean thinking in ASD. Practices and tools include elements that determine how the fundamentals are implemented in practice. The main sources used to answer this research question were the case studies conducted with Ericsson and Elektrobit (Papers V and VI), which explained in more detail the responses collected by the Agile and Lean usage survey (Paper II). Instead of an entire inventory of principles and practices, this section collects those that the empirical evidence showed are the most important in the context of software development. When a source of evidence does not support a specific element, it does not mean that that element is not considered in that source, but that it did not appear as essential (e.g., in the companies' focus group discussions). The tables provided in this chapter summarise the findings; for more details, please refer to the original publications.

Table 17 shows the main elements regarding the Lean principles of value and value stream. The survey reported in Paper II found that the Lean principles of creating customer value and eliminating waste and excess activities were applied the most (n = 209 and mean = 3.9, and n = 199 and mean = 3.4, respectively). These principles were also strongly present in the case studies.

**Table 17. Main elements regarding to the Lean principles of Value and Value Stream.**

| Value and Value Stream | Sources of evidence |
|---|---|
| Principles | |
| Everything that we do in the organisation is adding customer value | Paper II, Paper V and Paper VI |
| Eliminating waste and excess activities (connected to perfection) | Paper II, Paper V and Paper VI |
| Seeing-the-whole | Paper V and Paper VI |
| Practices and Tools | |
| Network of product owners | Paper V and Paper VI |
| Feature oriented development and flexible releases | Paper V and Paper VI |
| Scrum masters acting as firewalls to protect the team and focus it on producing customer value | Paper V and Paper VI |
| Managing the product as a whole | Paper V and Paper VI |
| Continuously challenge stakeholders/customers' needs (bells and whistles vs. bare bones solutions) | Paper V |
| Product owners paint the product big picture for development teams | Paper V |
| MMF (Minimum Marketable Features) and INVEST (Independent, Negotiable, Valuable, Estimable, Small, Testable) user stories | Paper V |
| Value Stream Mapping (VSM) | Paper V |
| Making decisions at the last responsible moment (defer commitment) | Paper V |

The companies stressed that all staff should care about providing customer value, not just the product owners. To implement the principle of value in practice, practices proceeding from Agile methods, such as networks of product owners and feature-oriented development, are complemented with Lean techniques, such as VSM and making decisions at the last responsible moment when most information is known. Thus, companies such as Ericsson manage uncertainty by learning from experimenting with different solutions and keeping options open until the last responsible moment to make the decision (Poppendieck and Poppendieck (2003) Principle 4 – *Defer commitment*).

It is remarkable that in both case studies, a network of product owners was found to make the voice of the customer heard by everyone. Some studies have indicated that it may be harmful because '*the implementation of [product owner roles] has frequently led to violate the principle of optimizing the whole*' (Poppendieck and Cusumano 2012). The *chief engineer* is the closest role to product owner in the Toyota Production System. However, in the TPS, the chief engineer is a unique person, who has total responsibility for managing the development of the product. In contrast, in software development, a team has that responsibility (team of product owners). It has been recognized that '*unlike manufacturing, where "front-loaded"decisions are possible [15, p.38], the product development environments, such as software development, are continuously feeding-in new information that require new decisions*' (Mandic *et al.* 2010). In that sense, it may be hard for a single person to manage the amount of (new) information involved in the software development process. Issues related to the risk of having networks of product owners from the perspective of violating the principle of optimizing the whole were not found in the empirical material.

Regarding the concept of waste, it was found that the focus groups stressed the concept of value and building in quality more than eliminating waste (see Ericsson case in Paper V). Eliminating waste, although it is promoted, is understood in the context of continuous improvement without a concrete focus on just reducing cost (Figure 15, Finding 3). For example, practices oriented to provide flexibility such as refactoring, which usually produce also higher costs, are not considered a source of waste but necessary in meeting customer expectations. Still, sources of waste in software development were also identified, which included delayed decisions, excess WIP and unused features, which were the most relevant according to the Elektrobit case study (Paper VI).

Table 18 shows the elements that were found important from the perspective of the principles of flow and pull. It was found that both Ericsson and Elektrobit base their software development in small batches of working software, thus creating a continuous flow. Moreover, both Ericsson and Elektrobit flattened their organisational structure in order to reduce feedback times and facilitate information flow. In addition to well established practices in Agile methods, such as continuous integration and test automation, all sources of evidence (the Agile and Lean usage survey in Paper II, and the case studies reported in Papers V and VI) pointed to Kanban, and its WIP limits, as a practice that is being increasingly adopted in software development to apply the principles of flow and pull. According to the case studies, Kanban is mainly used in product maintenance, where the nature of the work consists of a constant flow of customer service requests, but also in the development of new products.

**Table 18. Main elements regarding to the Lean principles of Flow and Pull.**

| Flow and Pull | Source of evidence |
|---|---|
| Principles | |
| Continuous flow of small batches of working software | Paper II, Paper V and Paper VI |
| Practices and Tools | |
| Leaner organisational structure (Hierarchy not deeper than 3 levels) | Paper V and Paper VI |
| From Scrum to Kanban (pull culture and WIP limits) | Paper II, Paper V and Paper VI |
| Concurrent development and testing | Paper V and Paper VI |
| Continuous integration and automatic testing | Paper V and Paper VI |
| Synchronization and coordination by Scrum of Scrum meetings and common iteration schedules | Paper VI |
| Continuous strategic planning & execution | Paper VI |

A key element of Lean thinking at Elektrobit, which is used to facilitate flow, is the continuous reconsideration of long-term targets to reduce the elapsed time between making a decision and realizing the consequences. Moreover, continuous long-term targets reconsideration supports flexibility and learning. Ericsson highlights the importance of having teams approximately equal in competence and permitting any team to pull a top priority item from the backlog for achieving flow.

Finally, an important but also conflictive aspect of achieving flow is the role of standardization in Lean Software Development. Some authors suggested Lean thinking might be not suitable in a software development domain because of difficulties in standardizing processes in creative industries (Staats *et al.* 2011).

Poppendieck and Poppendieck (2006) praised also the importance of using standards. Several aspects, such as coding standards, semi-standardized definition of done and a flexible Scrum/Kanban framework appeared as standardized in the case studies. However, Ericsson also stressed the importance of avoiding extra standardization. In their experience, over-standardizing processes and extensive mechanisms to ensure predictability and control prevent flexibility and closeness to the customer, resulting in organisational silos with multiple handover-related challenges. Thus, development teams at Ericsson have the authority to select the practices that work the best for them inside the bounds of a flexible Scrum/Kanban framework (e.g. teams can decide whether to use Scrum or Kanban depending on their own needs. They can also decide on practices such as pair-programming or when and how to conduct experiments. However, technical aspect such as continuous integration and unit testing are mandatory, as well as Scrum ceremonies, if the team decides to use Scrum, or working in open offices to facilitate collaboration).

Table 19 shows elements related to the principle of perfection. Continuous improvement, which is represented in ASD by the principle '*at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly*', is also considered in Lean thinking within the concept of Kaizen. The case studies at Ericsson and Elektrobit suggested that continuous improvement extends to the concept of the learning organisation as a result of adopting Lean Software Development. The purpose of a learning organization is to maintain knowledge and share lessons learnt within the company (usable knowledge in Lean thinking).

**Table 19. Main elements regarding to the Lean principle of Perfection.**

| Perfection | Source of evidence |
|---|---|
| Principles | |
| Creating a learning organisation | Paper II, Paper V and Paper VI |
| Practices and Tools | |
| Team retrospectives | Paper V and Paper VI |
| Network of coaches and scrum masters eliminating impediments | Paper V and Paper VI |
| Root-cause-analysis | Paper V and Paper VI |
| Cross-functional teams (collective knowledge) | Paper V and Paper VI |
| Communities of Practice and open spaces | Paper V |
| Team experiments | Paper V |
| Stop-the-line | Paper V |

In addition to retrospectives (also considered in Lean thinking through the *Hansei* practice), several tools from Lean thinking have been incorporated into software development, such as root-cause-analysis and stop-the-line for implementing the principle of perfection in practice. One lesson learned by Ericson related to the technique of stop-the-line is that solving impediments within the proper scope and at the right time brings benefits in terms of avoiding making unnecessary major improvements upfront. This finding aligned with Staats *et al.* (2011), who highlighted the need to keep problems and solutions together in time, space, and personnel. *Kaikaku*, which is process improvement through radical change, was not found in the studies.

In addition to the five core principles of Lean thinking, two elements appeared particularly important in the case studies, transparency and people (see Table 20). They transversally supported the companies' Lean and Agile processes.

**Table 20. Other essential elements of the combination of Lean thinking and ASD.**

| Other essential elements | Source of evidence |
|---|---|
| Transparency | |
| Fluent communication and collaboration between levels and teams | Paper V and Paper VI |
| Tools support such as JIRA, wikis, burn-down charts, information radiators (visual control) | Paper V and Paper VI |
| R&D Team area | Paper V |
| Go-and-see principle for managers | Paper V |
| *People* | |
| Team work | Paper V and Paper VI |
| Empowerment, personal initiative and self-organisation | Paper V and Paper VI |
| Respect people | Paper V and Paper VI |

Transparency was one of the aspects that were stressed the most, particularly in the Elektrobit case. Tool support (such as JIRA and wikis) for sharing knowledge, spreading management's top-down vision and strategy in all directions and at all levels, as well as obtaining rapid feedback on product development status were found essential in Elektrobit's Lean and Agile processes. A very remarkable change to facilitate transparency in the case of Ericsson was the enormous physical transformation that the site in Finland underwent, in which individual offices gave way to R&D team areas, resembling cells and Obeyas in Lean manufacturing. Both cells and Obeyas are used in Lean manufacturing to enhance flow and effective and timely communication. Cells are used at production level to facilitate flow. The idea of cells is to locate processing steps '*immediately*

*adjacent to each other so that parts, documents, etc., can be processed in very nearly continuous flow, either one at a time or in small batch sizes that are maintained through the complete sequence of processing steps*' (Marchwinski and Shook 2008). Obeya is the Japanese term used to name *big room*. Obeyas are commonly used in Toyota as a project management tool (Marchwinski and Shook 2008). Thus, project leaders have their desks in the obeya, which is set up with visual charts and graphs depicting time-line, progress, milestones, etc. to see the status of the product in a glance and facilitate its management.

In addition, people are considered to be the core of the Lean transformation. People oriented-culture based on people initiative and self-organisation, as opposed to earlier top-down control, was praised in both companies. Moreover, there was evidence for elevated consideration for people, such as when metrics were defined or teams were set up.

> *RQ1.3: What elements have been brought by Lean thinking on top of those predating the Lean Software Development movement?*

According to the empirical evidence of the thesis, Lean thinking is implemented in the context of ASD. Thus, many fundamental elements of the combination were already well established in Agile methods, such as Scrum. Petersen (2010) pointed out '*The high overlap between lean and agile also means that companies having adopted agile practices are not too far away from having a lean software process, given that they add the flow and E2E perspective on-top of their current practices*'. However, this thesis suggests that Lean thinking has brought other new elements to software development processes as follows:

- *Value and value stream*: the concept of customer value is extended with regard to ASD. Thus, a value-oriented company culture is attempted in which everyone cares about customer value. Furthermore, new techniques coming from Lean thinking, such as VSM, are now used in a software development context. On the other hand, the principle 'see-the-whole' (e.g., the E2E perspective suggested by Petersen) was also emphasized in both case studies. In addition, although waste was already considered in ASD by, for example, reducing documentation that do not add customer value or focusing on working software, with the introduction of Lean thinking, waste has acquired an identity in itself. Thus, development teams are trained to look for sources of waste in their specific contexts from the perspective of customer value.

- *Flow and pull*: Kanban and WIP limits are increasingly used to achieve flow. Aligned with Wang *et al.*'s (2010) findings, the tendency is to move from time-boxed Agile processes to flow-based Lean processes.
- *Perfection*: the concept of continuous improvement is also extended to the concept of the learning organisation. Techniques of Lean such as stop-the-line and root-cause analysis appear to be used in software development.
- Finally, greater emphasis on end-to-end transparency and collaborative development is also appreciable when companies move from ASD to a combination of Agile methods and Lean thinking.

Therefore, Lean and Agile in software development are not two names for the same concept. Lean thinking offers a new means to scale Agile methods and enhance ASD processes (Figure 15, Finding 4). When reviewing the elements identified as important in the studies of this thesis, it can be recognized that technical and social aspects are equally important in the combination of Lean and ASD (Figure 15, Finding 5). However, elements identified as important in the combination of Lean thinking and ASD were limited to the software development boundaries. A Lean Software Enterprise is expected to apply Lean in whatever they do, including even its way of working with suppliers (Womack *et al.* 1990). Therefore, it can be concluded that the adoption of Lean thinking in software development is still in its early stages when considering the concept of Lean Software Enterprise (Figure 15, Finding 6).

## 5.2 Research Question 2: What are the key factors that influence the successful transformation of software organisations towards Lean and Agile methods?

Research question 2 focuses on analysing factors that are important in the transformation to a combination of Lean thinking and ASD. When organisations change their way of working, which is referred to organisational change[32], changes can happen at different levels, from automating manual tasks to rethinking the nature of the organisation. Risks increase as the scope of the transformation broadens. One reason that complicates the transformation towards

[32] Organisational change is a specific field of knowledge and refers to "*the process by which organisations move from their present state to some desired future state to increase their effectiveness*" (Jones 2010, p. 88).

Lean thinking is that Lean emphasizes the importance of considering the whole enterprise. A Lean enterprise should use Lean not only in some areas but in everything the organisation does (Womack *et al.* 1990). When software-intensive companies commit to Lean and Agile, they start a process of change that will affect the entire development, from the earliest product release phases to maintenance (see e.g., Paper V on Ericsson's case). Understanding the factors that facilitate or hinder this process, in the form of potential challenges and strengths in implementing a combination of Lean thinking and ASD, will help other organisations in the process of adopting Lean thinking and provide insights into future research. The main evidence used to answer RQ2 was provided in Papers II, V and VI. Paper IV also provided insights into the topic of organisational culture. The synthesis of the results is presented in the following two subsections.

*RQ2.1: What challenges are potentially faced when combining Lean thinking and Agile Software Development?*

According to the empirical studies conducted during the dissertation, three main factors challenge the usage of Lean and Agile methods: management support, the development of large software/complex software that hinders flow and transparency, and creating a proper organizational culture (Figure 15, Finding 7). Table 21 synthetizes these challenges, including the specific issues involved in each challenge as evidenced in the cases studies.

**Table 21. Main challenges when using a combination of Lean thinking and ASD.**

| Main challenges | Sources of evidence | | |
|---|---|---|---|
| | Paper II<br>Adoption Survey | Paper V<br>Ericsson Case | Paper VI<br>Elektrobit Case |
| Management commitment | ✓ | | ✓<br>Involvement of mgmt. in development tasks |
| Cultural change | ✓ | ✓<br>Learning culture | ✓<br>Waste reduction culture |
| Developing large/complex software | ✓ | ✓<br>Transparency<br>Flow | ✓<br>Scaling flexibility<br>Synchronization and coordination<br>Short feedback |
| Measuring success of using Lean/Agile | ✓ | | |
| Customer/supplier collaboration | ✓ | | |
| Subcontracting | ✓ | | |

The Agile and Lean usage survey in the Finnish software industry and the case study conducted at Elektrobit (Papers II and VI, respectively) found that involving management in the Lean and Agile way of working was a central challenge. In particular, the survey's respondents identified it as the most important challenge to their Agile/Lean adoption (201 respondents selected it with a mean of 4.0 in a five point scale from unimportant [1] to extremely important [5]). The case study of Elektrobit provided a deeper understanding of the topic. In Elektrobit's experience (see Paper VI), the involvement of management in development task such as prioritizing the backlog, defining feature contents, and accepting features as done is difficult to implement in practical set-ups. Thus, statements of the Elektrobit's assessment survey such as "*business managers are available to product backlog prioritization*", "*business managers define the feature/release contents for implementation teams on regular basis*" and "*business managers validate and accept features as done in release reviews and demos*" scored 2.93, 2.47 and 2.78 respectively (scale from strongly disagree [1] to strongly agree [5]). In addition, in Elektrobit's experience, it negatively affects the cooperation between teams and business levels, causing feedback loops that were longer than expected and hindering customer value transparency. Although some scholars have rejected management commitment as a critical success factor in Agile software projects (Chow and Cao 2008), most early studies are well aligned with this finding (Pikkarainen *el al.* 2008; Vijayasarathy and Turk 2008).

The involvement of business management clearly connects to the culture of the company, understanding culture as the set of beliefs, values and norms that guide organisational behaviour. Both the survey and the case studies found that cultural change was a challenge. In particular, overly traditional company culture was the third highest challenge in the survey study (Paper II). The case of Ericsson (Paper V) found that the transformation towards Lean implied a profound change in the company's culture and thinking, which went far beyond only process and tools alone. Creating a culture of learning appeared particularly challenging in this case. Elektrobit's case (Paper VI) showed that eliminating waste was a challenge because the high pressure to deliver to customers limited the resources and time allotted for simultaneous waste removal, improvement and learning activities. Thus, a company culture that embraces learning as a high priority does not appear immediately achievable. It can be concluded that aspects of cultural change, such as creating a learning culture, require time and constitute a significant challenge during the first years of the transformation.

Paper IV offers more insights into the topic of company culture and its influence in the adoption of Agile methods. The analysis of value patterns in Agile's organizing vision revealed that contributors to Agile's community discourse shared strong values related to methodological ASD's values and business interests. Paper IV suggests that companies that are more easily able to adopt Agile have foundational values regarding methodological aspects such as gaining and keeping knowledge and retaining skilful professionals. The roots of these foundations are in aspects of the Agile Manifesto (2001) such as keeping knowledge through close and preferable face-to-face collaboration between business people and developers, and attention to technical excellence and skilful individuals. Values that, although not endemic to Agile, represent practical commercial applications, such as generating income and influencing employees were found also strong. Similar foundations of method-oriented values were stressed by authors, such as Iivari and Iivari (2011), Strode *et al.* (2009) and Misra *et al.* (2009). However, commercial (business) values that also have a strong influence in the adoption of Agile, such as power and wealth, have not attracted much attention in the research so far.

Lean thinking has been seen as a means for scaling Agile methods. The case studies showed that the application of principles of Lean thinking, such as considering the whole value stream, is not an easy task. One reason was the number of decision points that take place in software development, which challenges the principle of flow. Ericsson indicated the need for further developing decision points inside the company to continue its Lean transformation. Elektrobit stressed the importance of on-time decision making because delays in making decisions result in significant waste in the form of unnecessary handover during the development process.

Cooperation between teams was found limited, which also challenges flow and transparency, and posed a risk for local optimisations. Bottlenecks between hardware and software teams were also found one of the most important challenges that Elektrobit currently faces. Lead times may vary from minutes or hours in software development up to weeks in hardware development. The best way of coordinating software and hardware teams is not yet well understood.

*RQ2.2: What are the strengths of software-intensive companies in combining Lean thinking and Agile Software Development?*

Research question 2.2 was answered through the assessments conducted at Ericsson and Elektrobit (Papers V and VI). Specifically, the strengths of the

companies when transforming to Lean thinking in an ASD context were the object of the study. The findings are shown in Table 22.

**Table 22. Main strengths of Ericsson and Elektrobit in using a combination of Lean thinking and ASD.**

| Main strengths | Sources of evidence | |
|---|---|---|
| | Paper V - Ericsson Case | Paper VI - Elektrobit Case |
| Set-up at implementation level | ✓ | ✓ |
| People oriented development | ✓<br>Involving people<br>Creating a team culture | ✓<br>Respect people<br>Self-organisation |
| Continuous improvement activities | ✓ | |
| Focus on customer value | | ✓ |
| Transparency | | ✓ |

Strengths refer mainly to elements already considered in ASD, where the companies were more experienced. The set-up at the implementation level, which refers to elements of Agile methods, such as roles, Scrum ceremonies and practices such as continuous integration and automated testing, appeared to work well in both companies. Both Ericsson and Elektrobit considered people-related aspects strengths. Involving people in the transformation and creating a team culture in the case of Ericsson and respecting people and self-organisation in the case of Elektrobit scored very high in the assessments (see Papers V and VI, section 4.2).

Practices oriented to continuous improvement, such as retrospectives, were well regarded by Ericsson, not only by mature Agile teams that had already interiorized their dynamics quite well, but also in teams that had less experience in using Agile and Lean. However, Ericsson's case also revealed that although many opportunities for improvement may be discovered, careful focus is important in selecting the improvements that have a highest priority. Finally, the case study of Elektrobit found strengths regarding better focus on customer value and achieving transparent development. Thus, statements such as "*All features delivered are relevant to our customers*" and "*Projects complete with a satisfied customer*", scored 3.67 and 3.71 respectively. The use of tools such as JIRA was stressed in this case.

## 5.3 Research Question 3: What are the impacts perceived by practitioners when using a combination of Lean thinking and ASD?

The effects perceived by practitioners were analysed through the data collected by the *Agile and Lean Usage Survey* (Paper II). Close cooperation with the companies also offered insights in this regard (Papers V and VI). Table 23 synthesizes the findings.

**Table 23. Effects of using a combination of Lean thinking and ASD.**

| Effect | Sources of evidence | | |
|---|---|---|---|
| | Paper II<br>Adoption Survey | Paper V<br>Ericsson Case | Paper VI<br>Elektrobit Case |
| Improved team communication | ✓ | | |
| Enhanced ability to adapt to changes | ✓ | | Contradicting finding: However, scaling flexibility is a challenge |
| Increased productivity | ✓ | | ✓<br>Increase of 30% in some areas |
| Improved quality of products | ✓ | ✓ | |
| Increased customer satisfaction | | ✓ | ✓ |
| Increased transparency in the organisation | | ✓ | ✓<br>Information more effectively shared |
| Accelerated time-to-market/ development cycle time | | ✓<br>Build times reduced over ten times<br>Commits/day increased five times<br>Speeded up response to fault claims | |
| Improved flow | | ✓<br>Decreased unnecessary handovers | |

The effects in the survey were measured by a five-point scale (from '*significantly improved*' [5] to '*much worse*' [1]). The respondents were asked to indicate how

the adoption of Agile/Lean methods had affected them (See Appendix B, question 25). In Table 23, the column headed Paper II Adoption Survey reports the number of respondents to the survey that considered the effect as well as the mean of the particular effect (average rating). The empty cells in the table indicate that the particular effect was not present in that source. Increased productivity and customer satisfaction, enhanced transparency in the organisation and sped-up response to customer requests were the most reported benefits brought by the combination of Lean thinking and ASD (Figure 15, Finding 8). It is interesting to note that Elektrobit's experience was that although Agile provided means to increase flexibility, in order to realize flexibility in a practical manner, flexibility needs to be implemented throughout the whole value stream, which Elektrobit found challenging.

## 5.4 Implications for Practice

This dissertation provides one view of how Lean Software Development is combined with Agile methods in practice. The results of this dissertation can benefit other organisations that pursue similar endeavours by providing a better understanding of how their peers undertook Lean transformation. Furthermore, this research could provide companies with the means to compare themselves with companies that have succeeded in the transformation. They might also find insights into improving their existing Lean/Agile processes. Based on the findings of the study, the following suggestions are put forward to guide practitioners:

- The question is not Lean or Agile but the judicious integration of the appropriate aspects of both paradigms in relation to the particular company's strategy on flexibility and efficiency. This work points out the elements which characterise the combination of Lean thinking and ASD in practice. The journey towards such a combination can be viewed as a continuous improvement process, in which each company needs to explore what works best for its specific context.
- An important aspect of Lean thinking is the principle of 'Seeing the whole'. The decision to adopt Lean thinking is often a part of the organisational strategy, which leads to transform the company and thus affects diverse disciplines such as product development, sales, human resources, etc. However, the empirical evidence found in this work indicates that Lean as currently applied in software development does not fully meet this principle.

Lean thinking is applied within the boundaries of software development and not in the entire value chain, which includes, for example, software and hardware teams, as Lean thinking requires. Cusumano (2011) attributes the recent problems which Toyota encountered with the unsecured driver-side floor mat that got under the accelerator pedal in a 2007 Toyota Lexus ES 350 to Toyota's violation of this principle. Therefore, software-intensive companies should make an effort to extend Lean thinking from software development teams to the entire value chain.

- Decision points are essential to achieve flow. Both Ericsson and Elektrobit show that one of the main factors which prevent flow in software development is delayed decisions. Although it was found that the principle of '*making decisions at the last responsible moment*' is applied to irreversible and costly decisions (see Ericsson case, Paper V), the number of decisions involved in the process can create handovers, preventing readiness and flow. Therefore, decisions points need to be particularly considered when the value stream is analysed to eliminate bottlenecks and enable flow.
- Finally, people are an essential aspect of the transformation process. The adoption of Lean thinking implies a deep change in company culture and people's mind-set. Moreover, in Ericsson's experience, the more freedom was given to the teams, the more responsibility these teams were prepared to take. When team members are given responsibilities, they make decisions fast, so the development process is also fast. However, people-focus does not necessarily mean neglecting technical practices. Technical aspects such as coding standards, refactoring, unit testing, TDD, continuous integration, automatic testing, frequent reviews, simple design and coding/testing dojo, also need to be a key focus.

## 5.5 Implications for Research

Lean is a worthwhile research topic, as shown by the attention that it has received in the Finnish software development industry. Opportunities for research on this topic will be further developed in *Chapter 6.3 Future work*. However, it is important to highlight two groups of implications that, according to the findings of this work, should be considered.

The first group refers to new ways of interpreting the combination of Agile and Lean in software development. The combination of Lean thinking and ASD requires a different interpretation in the software context than it does in the

manufacturing context since the domains have important differences. For example, the concept of flexibility is fundamentally different in software and in manufacturing. In software development, flexibility refers to changing customer preferences, whereas in manufacturing it refers also to changing production volumes that commonly imply significant costs. Moreover, software is more malleable than hard products, and the concept of value is fundamentally different.

The second group refers to conflicting aspects of Agile and Lean. This work suggests that we are still in the early stages of the development of Lean thinking in software development. Software development companies have selected elements of Lean thinking that are well aligned with ASD. However, there are also some theoretical differences in how the two paradigms have evolved. Examples are the role of standardization and the front-loading product development emphasized by Lean thinking versus the continuous reconsidering of goals, incremental delivery and refactoring proposed by Agile. Deeper analysis of these differences will provide useful insights on the best way to combine both paradigms in the software development domain.

# 6 Conclusions

This study contributes to the understanding of how Lean thinking is interpreted and implemented in practice in the context of ASD by providing empirical evidence on the following: i) the main elements which characterise the combination of Lean thinking and Agile methods in software development, ii) the challenges and strengths when software-intensive companies transform and adopt a combination of both Agile and Lean and iii) impacts, in terms of benefits, which practitioners perceive when they combine Lean thinking and ASD. Thus far, the body of knowledge on the topic comprises insights of consultants, who have provided a variety of interpretations of Lean Software Development (e.g., Poppendieck and Poppendieck 2003, 2006, 2009; Middleton and Sutton 2005; Larman and Vodde 2008; Anderson 2010; Coplien and Bjørnvig 2011). However, research contributions from this dissertation suggest that the area has yet to be fully developed. This dissertation analysed Lean thinking and ASD in depth by using an exploratory research approach based on mixed methods, including literature analysis, survey research and case studies. Section 6.1 summarises the main conclusions of this dissertation. Section 6.2 discusses the validity and limitations, and Section 6.3 describes future research directions based on the findings of this work.

## 6.1 Main Contributions

The synthesis of the studies produced overall conclusions that are summarised in the first six numbered items, and a number of more detailed results associated with the research questions that are summarised in the rest of the section.

1. *The current trend in Lean Software Development is the combination of Lean thinking with ASD*. The results of the survey in Papers II and III along with the effort that companies in Finland, such as Ericsson and Elektrobit (Papers V and VI), are doing to transform themselves into Lean organisations, confirm the relevance that Lean thinking is progressively acquiring in the software development industry. Although the current study involves some limitations on generalising the findings, which will be discussed further in *Section 6.2*, it can be confirmed that the attention that the combination of Lean thinking and ASD has gained in one of the highest-ranked software development industries in the world (IT Industry Competitiveness Index

2011; Bilbao-Osorio *et al.* 2013) is a clear indicator of the expectations that practitioners have in the ability of these methods to enhance today's software development processes.

2. *Lean thinking is combined with ASD in a single process so they complement each other*. Lean thinking is considered an evolution of Agile to achieve an improved paradigm, in which the line separating Agile and Lean is blurred. The empirical evidence of this dissertation indicates that the use of Lean thinking alone is quite uncommon. Regarding how the combination is implemented, the studies of the dissertation show a clear overlap between the two paradigms. In general, Lean principles guide the implementation of Agile methods at a prescriptive level. It was found that many elements which define the combination of Lean thinking and ASD belong to Agile methods such as Scrum (e.g. network of products owners, backlogs of user stories and features, continuous integration, test automations, self-organised and empowered cross-functional teams, retrospectives, etc.). In addition, Lean principles such as creating a customer value culture in which everyone cares about providing customer value, seeing-the-whole value stream, providing continuous flow through small batches of working software and creating a learning organization to adapt to business and market changes guide team level activities. Lean practices at the implementation level are also found, such as stop-the-line, Kanban and value stream mapping. The case studies suggest that companies select the elements from Lean thinking which fit the ASD context and align with the Agile Manifesto. Time or space restrictions, which prevent mixing of techniques from Lean and Agile paradigms in manufacturing according to the theory of *le-agility* (ben Naylor *et al.* 1999; van Hoek 2000), are not applicable in the combination of Lean and Agile in software development. Thus, a combination of techniques is implemented in a single process. As a result, a hybrid approach is created.
3. *The main focus of Lean Software Development is not on reducing costs but on creating value, improving responsiveness and building quality in*. Some studies have interpreted the main purpose of Lean thinking as reducing costs (see, for example, Christopher and Towill 2000). This assumption is understandable as suggested by Ohno (1988) who highlights the importance of eliminating waste in his description of the TPS. However, flexibility and leanness, which are seen as antagonists in a manufacturing context (van Hoek 2000), can be pursued together in software development. The empirical evidence of this dissertation indicates that increasing customer value, which

requires flexibility to adapt to changing customer needs, and building quality in are emphasised more than simply reducing costs. Although the concept of waste is important in Lean Software Development to achieve efficiency, the costs associated with flexibility were not identified as sources of waste during the focus group discussions.

4. *Lean and Agile are not two names for the same concept. Lean thinking offers a new means to scale Agile methods and enhance ASD processes*. Some researchers have speculated whether Agile and Lean are just two names for the same concept (Wang and Conboy 2011). Based on the results obtained in this dissertation, it is possible to conclude that Lean and Agile in software development are not synonyms and they are conceptually different. Lean brings new elements on top of ASD, which are oriented to i) scaling Agile beyond team boundaries and across the entire organisation, such as a flow and end-to-end focus in product development, and ii) enhancing software development processes by encouraging a learning organisation which can operate under uncertainty and respond to changes, focuses on creating the highest value to the customer, adheres to a 'pull' and 'less waste' culture and emphasises transparency and collaborative development. Therefore, Agile is not abandoned when Lean thinking is adopted, as scaling agility/flexibility is considered essential, and Lean thinking complements ASD.
5. *Both technical and social aspects are equally important in the combination of Lean thinking and ASD*. In recent years, the social side of ASD has attracted increased attention. Some scholars have criticised Agile because of its disregard for the engineering side of software development (Rakitin 2001). When companies were asked to identify the main elements which characterised their Lean and Agile way of working, they pointed to the balance between technical and social aspects of the software development process. Technical competence is valued across a comprehensive spectrum of tools such as continuous integration, test automation, coding standards, refactoring, unit testing, TDD and static code analysis. In addition, social aspects of the software development process such as fluent and frequent communication, teamwork, empowerment, personal initiative, self-organisation, etc. were also highlighted during the focus group sessions.
6. *The adoption of Lean thinking in software development is in its early stages when considering the concept of Lean Software Enterprise*. One of the key aspects in Lean thinking is the principle of seeing-the-whole value stream. Although the studies conducted in this dissertation show that software

companies apply this principle in developing their software products, they do so only partially because surrounding areas such as sales and human resources, or suppliers do not appear to use Lean thinking. Therefore, extending Lean thinking to the entire enterprise remains a challenge. In fact, Elektrobit indicated that one of the bottlenecks which it is currently facing is synchronising its hardware and software teams. Meanwhile, from the *toolkit*[33] which Lean thinking offers (Marchwinski and Shook 2008), only a subset of tools was found to be used in a software development context. Whether other Lean elements have not been considered in Lean Software Development because they are not useful in a software development context or because Lean Software Development is in its infancy stage and software development practitioners have not still discovered them needs further research.

7. *Involving management in development tasks, achieving flow and transparency, and creating a Lean and Agile organisational culture are found as potential challenges when adopting a combination of Lean thinking and ASD*. Involving business management in the Agile and Lean way of working, which has been also highlighted in previous research works, continues to be a challenge according to the empirical evidence of this dissertation. Furthermore, new challenges continue to emerge, particularly at the organisational level, such as *achieving flow* and *transparency*. In both case studies on Ericsson and Elektrobit, transparency was enhanced through the adoption of Lean thinking. However, the case study at Ericsson indicated a tendency for development teams to limit collaboration outside the team, which limits also transparency. Finally, transforming the company culture towards a Lean thinking also takes time and entails aspects such as *creating a learning culture*, which are not readily achievable. However, involving people at the implementation level in the Agile and Lean way of working and setting-up teams was found to be more easily achievable.
8. I*ncreased productivity and customer satisfaction, enhanced transparency in the organisation and sped-up response to customer requests* were indicated as the main benefits brought on by the combination of Lean thinking and ASD in the companies studied in this dissertation.

[33] *Lean thinking toolkit* is used in the context of this thesis as the set of tools, practices or techniques that belong to the umbrella of Lean thinking.

## 6.2 Validity and Limitations of the Study

Empirical research involves different threats to validity which need to be appropriately addressed to ensure the quality of a study. Creswell's (2009), Yin's (2009) and Wohlin *et al.*'s (2012) guidelines were considered to ensure the validity of the current study. In the following sections, the validity and limitations of this work are discussed in terms of external validity, construct validity, reliability and internal validity. External validity refers to the extent to which findings can be generalised outside the investigated sample or cases. Construct validity refers to the extent to which operational measures represent the concepts being studied. Reliability refers to the extent to which the study can be repeated with the same results. Finally, internal validity refers to causal relationships and the extent to which the cause can lead to the effect. A study specific discussion of validity is included in each publication.

### *6.2.1 External Validity*

External validity refers to whether the findings of a study are transferable to other contexts or can be generalised. The studies included in this dissertation were mainly conducted in the Finnish software-intensive industry, with the exception of the case study conducted in collaboration with Ericsson R&D Finland, which included its cooperative sites in Hungary and China (Paper V) and the analysis of Agile's organizing vision, which included whitepapers and magazines worldwide (Paper IV). Therefore, the major concern on the generalisability of the findings is scaling them beyond the borders of Finland. Whether the Finnish software industry is representative of the international software development industry is questionable. For example, the findings on adoption levels obtained from the Agile and Lean usage survey are likely influenced by the characteristics of the Finnish software development industry, which is considered a pioneer in the use of Agile and Lean methods (Dybå and Dingsøyr 2008). However, the findings of this dissertation are especially relevant because Finland is well known to have one of the most efficient software development industries in the world (Bilbao-Osorio *et al.* 2013; IT Industry Competitiveness Index 2011).

Another concern is the external validity of the results within the Finnish software industry. The sample provided by FIPA to conduct the Agile and Lean usage survey was composed of 4950 software professionals. Because most software professionals in Finland are members of FIPA, the sample can be

considered representative of the population. More than 400 responses, which represent a 9%, were collected and can be said to largely represent the population. However, because of FIPA's confidentiality policies, which prohibited from providing the e-mail addresses of the respondents to the researchers, non-response bias could not be post-analysed. Therefore, the findings of the survey can be affected if, for example, the respondents who were closer to Agile and Lean methods were more willing to respond to the survey that those who were not. Regarding the case studies, the general concerns on the inability to generalise from individual cases also applies to this dissertation. Industrial studies are conducted in the context of a particular company. Therefore, the results are affected by factors which are specific to the company, such as its application area, culture and other contextual factors. In this dissertation, the case studies involved two companies, which were selected according to replication logic and in consideration that both (Ericsson and Elektrobit) are representative of the software-intensive industry. Replication of cases which share common characteristics and are representative of the software development industry improves the generalisation of the findings. Nevertheless, because of the lack of clarity in the phenomenon of study and the low number of empirical studies on the topic, the purpose of this dissertation was to achieve in-depth understanding of how Lean thinking is combined with ASD in its natural context rather than providing generalisations outside of the company-specific context.

Regarding the generalisability of the findings from the analysis of Agile's organising vision, which was presented in Paper IV, the sample consisted of commercial publications, which are commonly accepted and used to represent organising visions. Furthermore, a split-half technique was applied to test the sampling adequacy and the same results for each half sample were obtained.

### 6.2.2 Construct Validity

*Construct validity* occurs during data collection and data analysis (Yin 2009). It refers to the extent to which operation measures represent the concepts which researchers have in mind, so no misunderstandings and misconceptions arise when the constructs are measured (Yin 2009). The phenomenon involved in the present study is ambiguous, and the backgrounds, knowledge and experiences of the participants, especially with regard to traditional, Agile and Lean software development methods, may have likely affected the participants' interpretation of

the questions in the surveys and discussions during the focus groups. Therefore, several countermeasures were considered to reduce this limitation.

In the different survey tools (Agile and Lean usage survey and the survey instrument used to assess the transformation at Elektrobit) used, the option 'I don't know' was included, where applicable, to avoid erroneous responses caused by the lack of knowledge of the respondents. Moreover, the categories of the scales were labelled with words instead of only numbers to avoid interpretations which were unanticipated by the researchers. The surveys were also pilot-tested to a group of practitioners to check the consistency and readability of the questions. Particularly in the Agile and Lean usage survey, in which the researchers had less control, open fields were also added to allow the respondents to add elements which were different from those included in the predefined options, which were based on the literature (see Appendix B).

Moreover, a half day '*material walkthrough*' workshop was conducted at the beginning of each case study to introduce the researchers to company-specific terminologies and practices. In the Elektrobit case study, the guide used to direct the focus groups was adapted to the specific company terminology instead of using general terms from the literature (e.g., roles and practices). The statements of the assessment instruments were defined by the focus groups, which also contributed to enhancing the construct validity of the instrument. In the Ericsson case study (Paper V), the assessment sessions were facilitated by a company representative, who explained the meaning of the statements in case of confusion. Finally, the results were returned to the company representative who helped validate the findings.

Finally, a question which may arise is whether the Lean implementation of the companies involved in the research conformed strictly to Lean thinking. The study did not focus on epistemological concerns mainly because as in other domains in which Lean is applied, there is no universally accepted definition of Lean in software development. However, both companies involved in this research are part of the large initiative to investigate ways of applying Lean thinking in software development, Cloud Software Program (2010), and are consciously trying to adopt Lean Software Development.

### *6.2.3 Reliability*

The preconceptions of researchers in the data collection and analysis stages can affect the reliability of a study. In the current research, investigator triangulation

was used in data collection and data analysis as a countermeasure for the potential subjectivity of the researchers. Furthermore, the participants in the focus groups were the ones who created the concrete set of statements which best defined their Lean and Agile way of working to minimise the bias of the researchers in the data collection process. During the focus group sessions, the researchers were limited to performing tasks which were related to the logistics of the meeting, recording the data with the use of a shared Excel spread-sheet and ensuring that each participant in the sessions was given the opportunity to express his or her own opinions.

Finally, the research relies on working directly with currently active software development practitioners who are applying Agile and Lean methods provides the best opportunity to elicit the main elements that characterise their way of working. However, as in any research which involves practitioners, the data collected were affected by the subjective opinions, knowledge and attitudes of the individual participants. Although knowledgeable participants were selected in all phases of the research, controlling their subjectivity, which may favoured certain practices, values and principles was impossible.

### *6.2.4 Internal Validity*

According to Yin (2009) internal validity is mainly a concern '*when an investigator is trying to explain how and why event x led to event y*'. Therefore, it primarily affects to explanatory studies. Exploratory studies, as it is the case in the current research, do not have their main focus on defining cause-effect relationships, but on exploring a specific phenomenon (Yin 2009). However, internal validity is also related to inferences made when drawing findings and conclusions from empirical evidence. In that sense, counter measures to ensure internal validity need to be considered in exploratory studies. In this thesis, the results were strengthened through triangulation (i.e., conducting different types of studies, observing both quantitative and qualitative data and selecting participants from different profiles). For example, the impacts of the combination of Lean thinking and ASD were explored from the perspective of experts who lead the transformation and practitioners who are directly affected by the Lean and Agile way of working. Although the main focus was not on establishing causal relationships but developing an experience base, whether the application of Lean thinking combined with ASD has led to the benefits and challenges highlighted by the participants or there were some other factors that accidently led the

participants to this conclusion could be questioned. Participants and data triangulation were used as countermeasures to limit the impact of this threat. Knowledgeable participants with deep understanding of the phenomena and in different positions in the organization were selected as subjects of study (experts who lead the transformation and practitioners who are directly affected by the Lean and Agile way of working). Moreover, internal documents were used to support the interpretation and description of the findings. In addition, data analysis was based on theoretical foundations as long as they exist. Collected data was analysed from the perspective of the five principles of Lean as originally proposed by MIT's researchers in order to avoid secondary interpretations bias. As described in Papers V and VI, a systematic coding strategy was used in the analysis for inferring the main elements that characterize the combination of Lean thinking and ASD in the case studies. In both of the specific studies and the later research synthesis, the initial list of codes was designed according to the initial literature analysis (theoretical foundations), research design and research questions (see *Section 3.5 Phase V: Synthesising the results*). Thus, coding techniques supported the inference of meaningful patterns.

## 6.3 Future Work

From a research perspective, an in-depth understanding of the combination of Lean thinking and ASD opens up a wide variety of opportunities for further research on the topic in the coming years. The current trend points towards the combination of Lean thinking and ASD. However, the body of knowledge on this topic is scarce, and practitioners who want to adopt Lean thinking to scale their Agile processes or enhance their software development have little scientific support available. Studies similar to those conducted in this dissertation are needed to confirm or refute the findings and thus determine how broadly applicable they may be, as well as to enable analytical generalisation by extending results from cases that have common characteristics.

Moreover, this dissertation provides an overall picture of how Lean thinking is implemented and combined with ASD in software-intensive organisations. However, each of the relevant elements identified in this dissertation is a candidate to be studied in greater detail. The challenges identified when transforming towards a combination of Lean thinking and ASD particularly merit further study. For example, the concept of transparency in the software development domain needs further investigation. Software development is a

process which involves large amounts of information. The purpose of transparency is to make the product value stream transparent to everyone, so that decisions can be made quickly, feedback loops are sped-up and problems can be easily identified. However, different roles require different types of information for tasks to be performed properly. Therefore, which kind of information needs to be available to everyone through information radiators, wikis, and the like, as well as which information is role-specific, so that people are not overloaded with valueless information, needs to be examined in detail. Similarly, how value is created in software development, where it is injected and what decisions are involved in the process to support readiness and enable flow also need to be investigated in depth. The important elements identified in this dissertation were derived from the opinions of the practitioners who participated in the study. Whilst similar studies which obtain practitioners' opinions are needed, additional effort is required to validate them via other research methods, such as direct observations of practitioners or experiments.

The empirical evidence of this thesis suggests that Lean Software Development and ASD are combined in a single process. The characteristics of software products and software development processes open up new possibilities that are different from those offered in other domains to achieve leanness and flexibility. Whilst Lean principles are universal, a further understanding of the techniques required to apply such principles from a software development angle is needed. For example, Mandic *et al.* (2010) reflect on the meaning of flow from a software development perspective. How flexibility and leanness can be combined in software development is especially interesting.

Furthermore, although Lean thinking was initially proposed as a way to scale Agile methods, this dissertation shows that Lean thinking in an ASD context facilitates additional means to enhance software development processes. The potential of Lean Software Development should be further analysed. Moreover, the collaboration between practitioners and scholars is needed to continue learning about the implications of Lean thinking and its combination with Agile methods in software development.

# References


Abrahamsson P, Conboy K & Wang X (2009) 'Lots done, more to do': the current state of agile systems development research. European Journal of Information Systems 18(4): 281–28.

Abrahamsson P, Salo O, Ronkainen J & Warsta J (2002) Agile Software Development Methods: Review and Analysis. VTT Publications 478.

Adler PS & Shenhar A (1990) Adapting your technological base: the organizational challenge. Sloan Manage Review 32: 25–37.

Agarwal A, Shankar R, Tiwari MK (2006) Modeling the metrics of lean, agile and leagile supply chain: An ANP-based approach. European Journal of Operational Research 173(1):211–225.

Agile Manifesto (2001) Beck K, Beedle M, van Bennekum A, Cockburn A, Cunningha, W, Fowler M, Grenning J, Highsmith J, Hunt A, Jeffries R, Kern J, Marick B, Martin RC, Mellor S, Schwaber K, Sutherland J & Thomas D Manifesto for Agile Software Development, http://www.agilemanifesto.org/.

Agresti WW (1986) New paradigms for software development: tutorial. IEEE Computer Society Press.

Ahmad MO, Markkula J & Oivo M (2013) Kanban in software development: A systematic literature review. Proceedings of the 39th Euromicro Conference series on Software Engineering and Advanced Applications (SEAA) Santander, Spain.

Ambler S (2008) Results from Scott Ambler's February 2008 Agile Adoption Survey (available: http://www.ambysoft.com/surveys/ )

Anderson DJ (2010) Kanban. Blue Hole Press.

Basili VR & Turner AJ (1975) Iterative enhancement: A practical technique for software development. IEEE Transactions on Software Engineering (4): 390–396.

Baskerville RL & Myers MD (2009) Fashion waves in information systems research and practice. MIS Q 33(4): 647–662.

Beck K & Andres C (2004) Extreme programming explained: embrace change. Addison-Wesley Professional.

Beck K & Boehm B (2003) Agility through discipline: A debate. Computer 36(6): 44–46.

Ben Naylor J, Naim MM & Berry D (1999) Leagility: integrating the lean and agile manufacturing paradigms in the total supply chain. International Journal of Production Economics 62(1): 107–118.

Bilbao-Osorio B, Dutta S & Lanvin B (2013) The global information technology report 2013. Growth and jobs in a hyper-connected world. Wold economic forum. http://www.weforum.org/reports/global-information-technology-report-2013, accessed 17.0.2013

Biffl S, Aurum A & Boehm B (Eds.) (2005) Value-based software engineering. Springer.

Boehm B (1988) A spiral model of software development and enhancement. Computer 21(5): 61–72.

Boehm B (2002) Get ready for agile methods, with care. Computer 35(1): 64–69.

Boehm B (2006) Some future trends and implications for systems and software engineering processes. Systems Engineering 9(1): 1–19.
Boehm B & Lane J (2007) Using the incremental commitment model to integrate system acquisition, systems engineering, and software engineering. CrossTalk 20(10): 4–9.
Boehm B (2011) Towards richer process principles. Proceedings of the 2011 International Conference on Software and Systems Process, ACM: 234–234.
Bowers P (2002) Highpoints from the agile software development forum. Crosstalk: 26–27.
Bremner B, Dawson C, Kerwin K, Palmeri C & Magnusson P (2003) Can anything stop Toyota? Bus Week 117.
Browning TR (2001) Applying the design structure matrix to system decomposition and integration problems: a review and new directions. IEEE Transactions on Engineering Management 48(3): 292–306.
Byrne A, Womack J P (2012) The Lean turnaround: How business leaders use Lean principles to create value and transform their company. McGraw-Hill.
Castells M (2011) The rise of the network society: The information age: Economy, society, and culture. Wiley.
Chow T & Cao DB (2008) A survey study of critical success factors in agile software projects. Journal of Systems and Software 81(6): 961–971.
Christopher M & Towill DR (2000) Supply chain migration from lean and functional to agile and customised. Supply Chain Management: An International Journal 5(4): 206–213.
Cleland-Huang J (2013) Are Requirements Alive and Kicking? Software. IEEE 30(3):13–15.
Cloud Software Program (2010) http://www.cloudsoftwareprogram.org/
CMMI (2010) Capability maturity model integration 1.3, Carnegie Mellon Software Engineering Institute, http://www.sei.cmu.edu/cmmi/, Accessed 9 Jun 2013.
Cobb CG (2011) Making sense of agile project management: balancing control and agility. Wiley.
Cockburn A & Highsmith J (2001) Agile software development, the people factor. Computer 34(11): 131–133.
Cohen D, Lindvall M & Costa P (2004) An introduction to agile methods. Advances in Computers 62: 1–66.
Conboy K (2009) Agility from first principles: Reconstructing the concept of agility in information systems development. Information Systems Research 20(3): 329–354.
Conboy K, Coyle S, Wang X & Pikkarainen M (2011) People over Process: Key Challenges in Agile Development. IEEE Software 28(4): 48–57.
Coplien JO & Bjørnvig G (2011) Lean architecture: for agile software development. Wiley.
Creswell JW (2009) Research design: Qualitative, quantitative, and mixed methods approaches. 3rd ed. SAGE Publications, Incorporated.
Cruzes DS & Dybå T (2011) Recommended steps for thematic synthesis in software engineering. International Symposium on Empirical Software Engineering and Measurement (ESEM), 2011, IEEE: 275–284.
Cusumano MA (2011) Reflections on the Toyota debacle. Commun ACM 54(1): 33–35.

de Souza LB (2009) Trends and approaches in lean healthcare. Leadership in Health Services 22(2): 121–139.

Dingsøyr T, Nerur S, Balijepally V & Moe NB (2012) A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software 85(6): 1213–1221.

Dybå T & Dingsøyr T (2008) Empirical studies of agile software development: A systematic review. Information and software technology 50(9): 833–859.

Dybå T & Sharp H (2012) What's the Evidence for Lean? IEEE Software 29(5): 19–21.

Ebert C, Abrahamsson P & Oza N (2012) Lean Software Development. IEEE Software 29(5): 22–25.

Eisenhardt KM & Martin JA (2000) Dynamic capabilities: what are they? Strategic Manage Journal 21(10–11): 1105–1121.

Ewusi-Mensah K (2003) Software development failures: anatomy of abandoned projects. The MIT press.

FLEXI leaflet (2009) ITEA 2 Project results. Speeding change in large companies. Scaling-up agile development technology for embedded systems software. http://www.itea2.org/project/result/download?result=5402&file=06022_FLEXI_Project_Leaflet_results_oct_10.pdf (last accessed 29.8.2013)

Freeman P (1992) Lean concepts in software engineering. IPSS-Europe International Conference on Lean Software Development. Stuttgart, Germany: 1–8.

Highsmith JA (2002) Agile software development ecosystems. Addison-Wesley Professional.

Iivari J & Iivari N (2011) The relationship between organisational culture and the deployment of agile methods. Information and Software Technology 53(5): 509–520.

Ikonen M, Kettunen P, Oza N & Abrahamsson P (2010) Exploring the sources of waste in kanban software development projects. 36th Euromicro conference on software engineering and advances applications: 376–381.

ISO/IEC 122007:2008(E) (2008) Systems and software engineering – Systems life cycle processes. second edition, ISO/IEC, Geneva, Switzerland.

ISO/IEC 90003:2004(E) (2004) Software engineering - Guidelines for the application of ISO 9001:2000 to computer software, ISO/IEC 90093:2004(E).

ISO/IEC 15504-1:1998 (1998) Information technology – Process assessment – Part1: Concepts and introductory guide, WG10N222.

IT Industry Competitiveness Index (2011) Investment for the future. Benchmarking IT industry competitiveness 2011. Business software alliance, Economist intelligence.

Jacobson I, Booch G & Rumbaugh JE (1999) The unified software development process-the complete guide to the unified process from the original designers. Addison-Wesley.

Jones GR (2010) Organizational theory, design, and change. Prentice Hall.

Jonsson H (2012) Lean Software Development: A Systematic Review. IDT Mini-conference on Interesting Results in Computer Science and Engineering (IRCSE).

Kim CS, Spahlinger DA, Kin JM, Coffey RJ & Billi JE (2009) Implementation of lean thinking: one health system's journey. Joint Commission Journal on Quality and Patient Safety 35(8): 406–413.

Kitchenham BA & Charters S (2007) Guidelines for performing systematic literature reviews in software engineering. No. EBSE-2007-01.

Kontio J, Bragge J & Lehtola L (2008) The focus group method as an empirical tool in software engineering. In: Guide to advanced empirical software engineering. Springer: 93–116.

Krippendorff K (2004) Reliability in content analysis. Human communication research, 30(3): 411–433

Kruchten P (2011) A plea for lean software process models. Proceedings of the 2011 International Conference on Software and Systems Process. New York, NY, USA, ACM: 235–236.

Kuvaja P & Bicego A (1994) BOOTSTRAP—a European assessment methodology. Software Quality Journal 3(3): 117–127.

Laanti M (2012) Agile methods in large-scale software development organisations. Applicability and model for adoption. Doctoral thesis. University of Oulu.

Larman C & Vodde B (2008) Scaling lean & agile development: Thinking and organisational tools for large-scale Scrum. Addison-Wesley Professional.

Leidner DE & Kayworth T (2006) A Review of Culture in Information Systems Research:

Towards a Theory of Information Technology Culture Conflict. MIS Quarterly 30(2): 357–399.

Liker JK (2004) The Toyota way. Esensi.

Maglyas A, Nikula U & Smolander K (2012) Lean solutions to software product management problems. IEEE Software 29(5): 40–46.

Mandic V, Oivo M, Rodriguez P, Kuvaja P, Kaikkonen H & Turhan B (2010) What is flowing in Lean Software Development? In Proceedings of the 1st International Conference on Lean Enterprise Software and Systems (LESS): 72–84.

Maples C (2009) Enterprise agile transformation: the two-year wall. Agile Conference, 2009. AGILE'09. IEEE: 90–95.

Marchwinski C & Shook J (2008) Lean lexicon: a graphical glossary for lean thinkers. 4th edition, Lean Enterprise Institute.

Mehta M, Anderson D & Raffo D (2008) Providing value to customers in software development through lean principles. Software Process: Improvement and Practice 13(1): 101–109.

Middleton P (2001) Lean software development: two case studies. Software Quality Journal 9(4): 241–252.

Middleton P, Flaxel A & Cookson A (2005) Lean software management case study: Timberline inc. In: Anonymous Extreme Programming and Agile Processes in Software Engineering, Springer: 1–9.

Middleton P & Sutton J (2005) Lean software strategies: proven techniques for managers and developers. Productivity Pr.

Middleton P & Joyce D (2012) Lean software management: BBC Worldwide case study. IEEE Transactions on Engineering Management 59(1): 20–32.

Miles MB & Huberman AM (1994) Qualitative data analysis: An expanded sourcebook. Sage.

Misra SD, KumarV & Kumar U (2009) Identifying some important success factors in adopting agile software development practices. The Journal of Systems and Software 82: 1869–1890.

Morgan DL (1997) Focus groups as qualitative research. Sage.

Morgan JM & Liker JK (2006) The Toyota product development system. Productivity press.

Münch J, Armbrust O, Kowalczyk M & Soto M (2012) Software Process Definition and Management. Springer.

Nagel RN & Dove R (1991) 21st century manufacturing enterprise strategy: An industry-led view. DIANE Publishing.

Nord RL, Ozkaya I & Sangwan RS (2012) Making architecture visible to improve flow management in lean software development. IEEE Software 29(5): 33–39.

OICA, The International Organisation of Motor Vehicle Manufacturers. World Ranking of Manufactures 2010. http://oica.net/wp-content/uploads/ranking-2010.pdf

Ohno T (1988) Toyota production system: beyond large-scale production. Productivity press.

Oivo M, Birk A, Komi-Sirviö S, Kuvaja P & Solingen RV (1999) Establishing product process dependencies in SPI. In the Proceedings of European Software Engineering Process Group Conference.

Osterweil L (1987) Software processes are software too. Proceedings of the 9th international conference on Software Engineering. IEEE Computer Society Press: 2–13.

Palmer SR & Felsing M (2001) A practical guide to feature-driven development. Pearson Education.

Parnell-Klabo E (2006) Introducing lean principles with agile practices at a fortune 500 company. Agile Conference. IEEE: 232–242.

Pernstål J, Feldt R & Gorschek T (2013) The Lean Gap: A Review of Lean Approaches to Large-Scale Software Systems Development. The Journal of Systems and Software, http://dx.doi.org/10.1016/j.jss.2013.06.035.

Petersen K (2010) Implementing Lean and Agile software development in industry. Blekinge Institute of Technology. Doctoral Dissertation Series No. 2010:04

Pikkarainen M, Haikara J, Salo O, Abrahamsson P & Still J (2008) The impact of agile practices on communication in software development. Empirical Software Engineering 13(3): 303–337.

Poppendieck M (2013) Lean software development: the lean mindset, learning to surf. Keynote at Software Engineering in Practice. International Conference on Software Engineering (ICSE).

Poppendieck M & Cusumano MA (2012) Lean Software Development: A Tutorial. IEEE Software 29(5): 26–32.

Poppendieck M & Poppendieck T (2009) Leading lean software development: Results are not the point. Pearson Education.
Poppendieck M & Poppendieck T (2006) Implementing lean software development: From concept to cash. Addison-Wesley Professional.
Poppendieck M & Poppendieck T (2003) Lean software development: An agile toolkit. Addison-Wesley Professional.
Rakitin S (2001) Manifesto elicits cynicism. IEEE Computer 34(12): 4.
Raman S (1998) Lean software development: Is it feasible? Digital Avionics Systems Conference, 1998. Proceedings, 17th DASC. The AIAA/IEEE/SAE, IEEE 1: C13/1-C13/8 vol. 1.
Reinertsen DG (2009) The principles of product development flow: second generation lean product development. Celeritas Redondo Beach, Canada.
Rosenberg D & Stephens M (2003) Extreme programming refactored: the case against XP. Apress.
Royce WW (1970) Managing the development of large software systems. Proceedings of IEEE WESCON, Los Angeles, CA, USA: 1–9.
Runeson P & Höst M (2009) Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14(2): 131–164.
Sauerwein E, Bailom F, Matzler K & Hinterhuber HH (1996) The Kano model: How to delight your customers. International Working Seminar on Production Economics. 1: 313–327.
Schwaber K & Beedle M (2002) Agile software development with Scrum. Prentice Hall.
Seaman CB (1999) Qualitative methods in empirical studies of software engineering. IEEE Transactions on Software Engineering 25(4): 557–572.
Shah R & Ward PT (2007) Defining and developing measures of lean production. Journal of operations management 25(4): 785–805.
Sharifi H & Zhang Z (1999) A methodology for achieving agility in manufacturing organisations: An introduction. International Journal of Production Economics 62(1): 7–22.
Sherehiy B, Karwowski W & Layer JK (2007) A review of enterprise agility: concepts, frameworks, and attributes. International Journal of Industrial Ergonomics 37(5): 445–460.
Shull F, Singer J & Sjøberg DI (2008) Guide to advanced empirical software engineering. Springer.
Sjøberg D, Johnsen A & Solberg J (2012) Quantifying the effect of using kanban versus scrum: a case study. IEEE Software 29(5): 47–53.
Sommerville I (2010) Software Engineering. 9th Edition. Addison-Wesley.
Stapleton J (2003) DSDM: Business focused development. Addison-Wesley Professional.
Staats BR, Brunner DJ & Upton DM (2011) Lean principles, learning, and knowledge work: Evidence from a software services provider. Journal of Operations Management 29(5): 376–390.
Stone PJ, Dunphy DC & Smith MS (1966) The General Inquirer: A Computer Approach to Content Analysis.

Strode D, Huff S & Tretiakov A (2009) The impact of organizational culture on agile method use. Proceedings of the 42nd Hawaii International Conference on System Sciences.

Sutherland J & Schwaber K (2007) The scrum papers: Nuts, bolts, and origins of an agile process.

Swanson EB & Ramiller NC (1997) The Organizing Vision in Information Systems Innovation. Organization Science 8(5): 458–474.

Tierney J (1993) Eradicating mistakes from your software process through Poka Yoke. Proceedings 6th International Software Quality Week: 300–307.

Tokatli N (2008) Global sourcing: insights from the global clothing industry—the case of Zara, a fast fashion retailer. Journal of Economic Geography 8(1): 21–38.

Tolfo C & Wazlawick RS (2008) The influence of organisational culture on the adoption of extreme programming. Journal of Systems and Software 81(11): 1955–1967.

Tolfo C Wazlawick RS, Ferreira MGG & Forcellini FA (2011) Agile methods and organizational culture: Reflections about cultural levels. Journal of Software Maintenance and Evolution: Research and Practice 23(6): 423–441.

Trimble J & Webster C (2013) From Traditional, to Lean, to Agile Development: Finding the Optimal Software Engineering Cycle. (HICSS), 2013 46th Hawaii International Conference on System Sciences, IEEE: 4826–4833.

Turk D, France R & Rumpe B (2002) Limitations of agile software processes. Third International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP 2002): 43–46.

Van Hoek RI (2000) The thesis of leagility revisited. International Journal of Agile Management Systems 2(3): 196–201.

Venables M (2005) Boeing: going for lean [lean manufacturing]. Manufacturing Engineer 84(4): 26–31.

VersionOne (2011) The sixth annual 'State of Agile Development Survey' (available: http://www.versionone.com/pdf/2011_State_of_Agile_Development_Survey_Results.pdf, accessed 09.09.2013)

Vijayasarathy L & Turk D (2008) Agile software development: A survey of early adopters. Journal of Information Tech Management, 19(2):1–8

Vilkki K (2008) Juggling with the Paradoxes of Agile Transformation or How to survive in a large scale agile transformation. Flexi Newsletter 2(2008): 3–5.

Vilkki K (2010) When agile is not enough. In: Lean Enterprise Software and Systems. Springer: 44–47.

Vilkki K & Erdogmus H (2012) Point/Counterpoint. IEEE Software 29(5): 60–63.

Wang X & Conboy K (2011) Comparing apples with oranges? The perceived differences between agile and lean software development processes. Thirty Second International Conference on Information Systems, Shanghai 2011

Wang X, Conboy K & Cawley O (2012) 'Leagile' software development: An experience report analysis of the application of lean approaches in agile software development. Journal of Systems Software 85(6): 1287–1299.

Wasserman T (2013) Low ceremony processes for short lifecycle projects. Keynote at the 2013 International Conference on Software and System Process, ACM.
West D, Grant T, Gerush M & D'Silva D (2010) Agile development: Mainstream adoption has changed agility. Forrester Research.
Wohlin C, Runeson P, Höst M, Ohlsson MC, Regnell B & Wessln A (2012) Experimentation in software engineering, Springer Publishing Company, Incorporated.
Womack JP, Jones DT & Roos D (1990) The Machine That Changed the World: The Story of Lean Production: How Japan's Secret Weapon in the Global Auto Wars Will Revolutionize Western Industry. New York, Rawson Associates.
Womack JP & Jones DT (1996) Lean thinking: Banish waste and create wealth in your organisation. New York, Rawson Associates.
Yin RK (2009) Case study research: Design and methods. Sage.

# Appendix 1 Literature analysis and industrial inventory data extraction forms

Extraction date: ____/____/____Reviewer: Study identifier: ____

## DATA EXTRACTION FORM: AGILE ADOPTION LITERATURE REVIEW

### IDENTIFICATION

| | | |
|---|---|---|
| Title | | |
| Author (s) and institution | | ☐ *Academic* ☐ *Industry* |
| Source | | |
| Publication Date | | |
| 1. Has the stage 1 been overcome? | ☐ *Yes* ☐ *No* | |
| | If no, please specify the not overcome criteria<br>☐ *Criteria 3 (publication date)* ☐ *Criteria 11 (English)* | |

If question 1 receives a 'No' response do not continue with the data extraction form.

### STAGE 2: STUDY TYPE Please, read only *title* and *abstract* of the study

| | |
|---|---|
| Study Design | ☐ *Research study*<br>☐ *Empirical **<br>☐ *Non-empirical **<br>** Answer these questions in the following stage*<br>☐ *Report*<br>☐ *Non-valid study* |
| Duplicate Study | ☐ *No*<br>☐ *Yes*<br>Identifier duplicated study _____<br>Is this the most complete version? ☐ *Yes* ☐ *No* |
| 2. Has the stage 2 been overcome? | ☐ *Yes* ☐ *No* |
| | If no, please specify the not overcome criteria<br>☐ *Criteria 7 (duplicated)* ☐ *Criteria 8 (not valid)* |

If question 2 receives a 'No' response do not continue with the data extraction form.

### STAGE 3: STUDY FOCUS Please, read only *introduction* and *conclusions*

| | |
|---|---|
| Is the study focused on agile adoption? | ☐ *Yes* ☐ *No* |
| Does the study directly answer one or more of our research questions? | ☐ *RQ1* ☐ *RQ2* ☐ *RQ3* ☐ *RQ4*<br>☐ *RQ5* ☐ *RQ6* ☐ *RQ7* |
| *Is there a rationale for why the study was undertaken?* | ☐ *Yes* ☐ *No* |
| *Is there a clear statement of the study's primary outcome (findings or contributions)?* | ☐ *Yes* ☐ *No* |
| 3. Has the stage 3 been overcome? | ☐ *Yes* ☐ *No* |
| | If no, please specify the not overcome criteria<br>☐ *Criteria 6 (focused on agile adoption)*<br>☐ *Criteria 9 (rationale commence)*<br>☐ *Criteria 10 (clear outcome)* |

If question 3 receives a 'No' response do not continue with the data extraction form.

## STAGE 4: STUDY QUALITY

Please, make an overview of the study and evaluate each item in the quality assessment.

### EMPIRICAL STUDY QUALITY ASSESSMENT TABLE

| | |
|---|---|
| Objective description | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Context description | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Appropriate research design | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Sample | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Data collection | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Data analysis | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Justified findings and conclusions | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Applicability of results | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Minimized threats | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| References | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| 4. Has the stage 4 been overcome? | If point is 5 or more ☐ *Yes* Otherwise ☐ *No* |

### NON-EMPIRICAL STUDY QUALITY ASSESSMENT TABLE

| | |
|---|---|
| Objective description | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Context description | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Appropriate research design | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Observations collection | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Observations analysis | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Justified findings and conclusions | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Applicability of results | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5)<br>☐ *Non-satisfied* (+0) |
| Minimized threats | ☐ *Completely satisfied* (+1)<br>☐ *Partiality satisfied* (+0,5) |

| | | |
|---|---|---|
| | ☐ *Non-satisfied* | (+0) |
| References | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| 4. Has the stage 4 been overcome? | If point is 4.5 or more ☐ *Yes* Otherwise ☐ *No* | |

**REPORT QUALITY ASSESSMENT TABLE**

| | | |
|---|---|---|
| Context description | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| Observations collection | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| Observations analysis | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| Justified findings and conclusions | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| Applicability of results | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| References | ☐ *Completely satisfied*<br>☐ *Partiality satisfied*<br>☐ *Non-satisfied* | (+1)<br>(+0,5)<br>(+0) |
| 4. Has the stage 4 been overcome? | If point is 3 more ☐ *Yes* Otherwise ☐ *No* | |

If question 4 receives a 'No' response do not continue with the data extraction form.

## STAGE 5: PRIMARY STUDY ANALYSIS

Please, analyse the complete study to answer the following questions.

| | | |
|---|---|---|
| Summary of the study | | |
| Context description | Software process (agile methodology) | |
| | The industry in which products are used | |
| | The nature of the software development organisation/Team | |
| | The skills and experience of software staff | |
| | Small projects or large, multi-site, multi-company or distributed projects | |
| | Others, if any | |
| *Research hypothesis* | | |
| *Adoption strategy description* | Motivation for agile adoption | |
| | How the development process is changed? | |
| | How knowledge and projects are managed? | |
| | Other issues, if any | |
| *Findings and conclusions* | Benefits of agile adoption | |
| | Challenges | |
| | Limitations | |
| | Risk factors | |
| | Cost | |
| | Others | |
| *Applicability and relevance* | Research | |
| | Practice | |
| | Future work | |

**Reviewer's remarks:**

# Appendix 2 Sample items of the Agile and Lean Usage Survey


**Agile and Lean Methods Adoption in Finland**

The purpose of this survey by the University of Oulu and Tietotekniikan Liitto ry is to find out the level of Agile and Lean methods adoption in ICT development in Finland, as well as to find out reasons why agile and lean method adoption sometimes fails. **Data from participants will be treated anonymously and are only available to the researchers conducting the survey. The company names cannot be identified from the results of the survey; they are only going to be used for statistical purposes.**

Participants of the survey will have access to an exclusive report with the results of the study. If you would like to take part in the prize draw for an **iPad 2** and receive the summary, you can fill out your email address (please, use your work account) at the end of the survey. Your email address will not be used for any other purpose and it will be kept private.

Please answer to the questionnaire by **(due date)** at the latest. Filling out the questionnaire will take approximately **20 minutes**. We would appreciate very much if you also **distribute the survey to your colleagues**.

**Thank you for your participation!**

**BACKGROUND INFORMATION - 4 %**

---

**1.** Which of the following roles best describes your current position? (Please choose all that apply) *

- ☐ President/VP/CEO/COO/CIO/CTO
- ☐ Project manager
- ☐ Product manager
- ☐ Process manager
- ☐ Product owner
- ☐ Scrum master
- ☐ IT staff
- ☐ Architect

☐ Developer
☐ Quality assurance/Tester
☐ Consultant/Trainer
☐ Operations/Support staff
☐ Sales/Marketing personnel
☐ Other (Please specify):

**9.** Are you currently applying agile methods in your organisational unit? *

◯ Yes

◯ No

**13.** Are you currently applying lean methods in your organisational unit? *

◯ Yes

◯ No

**14.** How long have lean methods been used in your organisational unit? *

◯ Less than 1 year

◯ 1-2 years

◯ 2-5 years

◯ 5-10 years

◯ More than 10 years

AGILE AND LEAN METHODS ADOPTION - 46 %

---

**18.** What were your organisational unit's goals in adopting agile and/or lean methods? (Please choose all that apply.) *

☐ To improve product and service quality
☐ To improve process quality
☐ To increase productivity
☐ To decrease development costs
☐ To reduce development cycle times and time-to-market
☐ To increase the ability to adapt to changes in the business environment
☐ To improve the management of business/product value
☐ To improve our understanding of the whole value stream
☐ To improve customer understanding
☐ To remove waste and excess activities
☐ To improve organisational learning

☐ To improve development flow
☐ To create transparency within the organisation
☐ To improve stakeholders' satisfaction
☐ To improve team communication
☐ To establish team-side project comprehension
☐ To reduce risks
☐ To achieve success others have achieved using lean methods
☐ I don't know
☐ Other (please specify):

**21.** Please select the principles that are commonly applied in your organisational unit and rate their frequency of use.

Frequency 5=systematically; 4=mostly; 3=sometimes; 2=rarely; 1=Never

| | 1 | 2 | 3 | 4 | 5 | Idon't know |
|---|---|---|---|---|---|---|
| Eliminate waste and excess activities (for example, eliminating anything which does not add value to the final product) | ○ | ○ | ○ | ○ | ○ | ○ |
| Minimize inventory or work in progress (for example, minimizing partially worked requirements or untested code) | ○ | ○ | ○ | ○ | ○ | ○ |
| Continuous flow of small batches in the development process | ○ | ○ | ○ | ○ | ○ | ○ |
| Pull from demand (for example, responding directly to needs initiated by customer requirements) | ○ | ○ | ○ | ○ | ○ | ○ |
| Respect and empower people | ○ | ○ | ○ | ○ | ○ | ○ |
| Focus on creating customer value | ○ | ○ | ○ | ○ | ○ | ○ |
| Do it right the first time | ○ | ○ | ○ | ○ | ○ | ○ |
| Focus on optimizing the whole system and not only local optimizations | ○ | ○ | ○ | ○ | ○ | ○ |

| | | | | | | |
|---|---|---|---|---|---|---|
| Create trusted relationships with suppliers | ○ | ○ | ○ | ○ | ○ | ○ |
| Create a culture of continuous improvement | ○ | ○ | ○ | ○ | ○ | ○ |
| Root source analysis is done after problems are discovered | ○ | ○ | ○ | ○ | ○ | ○ |
| Look simultaneously for multiple solutions | ○ | ○ | ○ | ○ | ○ | ○ |
| Create cadence (establish rhythmic cycles for all development activities) | ○ | ○ | ○ | ○ | ○ | ○ |
| Make decisions as late as possible | ○ | ○ | ○ | ○ | ○ | ○ |

**25.** How has the adoption of agile/lean methods affected the following in your organisation? *

Rating 5=significantly improved; 4=improved; 3=no effect; 2=worse; 1=much worse

| | Much worse 1 | 2 | 3 | 4 | Significantly improved 5 | I don't know |
|---|---|---|---|---|---|---|
| Enhanced software quality | ○ | ○ | ○ | ○ | ○ | ○ |
| Enhanced process quality | ○ | ○ | ○ | ○ | ○ | ○ |
| Increased productivity | ○ | ○ | ○ | ○ | ○ | ○ |
| Reduced costs | ○ | ○ | ○ | ○ | ○ | ○ |
| Accelerated time-to-market/cycle time | ○ | ○ | ○ | ○ | ○ | ○ |
| Enhanced value creation | ○ | ○ | ○ | ○ | ○ | ○ |
| Enhanced ability to adapt to changes | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved organisational transparency | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved team communication | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved learning and knowledge | ○ | ○ | ○ | ○ | ○ | ○ |

| | | | | | | |
|---|---|---|---|---|---|---|
| creation | | | | | | |
| Improved alignment between IT and business objectives | ○ | ○ | ○ | ○ | ○ | ○ |
| Reduced risks | ○ | ○ | ○ | ○ | ○ | ○ |
| Reduced waste and excess activities | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved stakeholder satisfaction | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved customer understanding | ○ | ○ | ○ | ○ | ○ | ○ |
| Improved customer collaboration | ○ | ○ | ○ | ○ | ○ | ○ |
| Other (please specify): | ○ | ○ | ○ | ○ | ○ | ○ |

# Original papers

I Rohunen A, Rodríguez P, Kuvaja P, Krzanik L & Markkula J (2010) Approaches to agile adoption in large settings: a comparison of the results from a literature analysis and an industrial inventory. In: Product-Focused Software Process Improvement. Springer: 77–91.

II Rodríguez P, Markkula J, Oivo M & Turula K (2012) Survey on agile and lean usage in Finnish software industry. Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement. ACM: 139–148.

III Rodríguez P, Markkula J, Oivo M & Garbajosa J (2012) Analysing the drivers of the combination of lean and agile in software development companies. In: Product-Focused Software Process Improvement. Springer: 145–159.

IV Lawrence C & Rodríguez P (2012) The interpretation and legitimization of values in Agile's organizing vision. Proceedings of the European Conference on Information Systems (ECIS). Barcelona, Spain, 10–13 June 2012.

V Rodríguez P, Mikkonen K, Kuvaja P, Oivo M & Garbajosa J (2013) Building lean thinking in a telecom software development organisation: strengths and challenges. Proceedings of the 2013 International Conference on Software and System Process. ACM: 98–107.

VI Rodríguez P, Partanen J, Kuvaja P & Oivo M (2014) Combining lean thinking and agile methods for software development. A case study of a Finnish provider of wireless embedded systems. Proceedings of the 47th Hawaii International Conference on Systems Sciences (HICSS 2014). In press.



Original publications are not included in the electronic version of the dissertation.

602. Juntunen, Kaisu (2012) Tieto- ja viestintätekniikan soveltamiseen perustuvat toimintaprosessien uudistukset terveydenhuollossa : sosio-teknis-taloudellinen näkökulma

603. Seppä, Karri (2012) Quantifying regional variation in the survival of cancer patients

604. Kuvaja, Pasi (2012) Software process capability and maturity determination : BOOTSTRAP methodology and its evolution

605. Laanti, Maarit (2012) Agile Methods in large-scale software development organizations : applicability and model for adoption

606. Hokkanen, Juho (2013) Liquid chromatography/mass spectrometry of bioactive secondary metabolites – *in vivo* and *in vitro* studies

607. Kuokkanen, Matti (2013) Development of an eco- and material-efficient pellet production chain—a chemical study

608. Jansson, Eeva (2013) Past and present genetic diversity and structure of the Finnish wolf population

609. Myllykoski, Matti (2013) Structure and function of the myelin enzyme 2′,3′-cyclic nucleotide 3′-phosphodiesterase

610. Lehto, Tuomas (2013) The importance of persuasive systems design in enhancing consumers' perceptions and adoption of health behavior change support systems

611. Hernoux-Villière, Audrey (2013) Catalytic depolymerisation of starch-based industrial waste : use of non-conventional activation methods and novel reaction media

612. Lawrence, Carl (2013) Innovating with information technology in a globalized world : being proactive about culture

613. Ardanov, Pavlo (2013) Priming capacities of endophytic *Methylobacterium* sp. on potato (*Solanum tuberosum* L.)

614. Koskela, Anni (2013) Wolverine habitat selection, diet and conservation genetics

615. Holm, Jana (2013) Catalytic pretreatment and hydrolysis of fibre sludge into reducing sugars

616. Kemi, Ulla (2013) Adaptation to growing season length in the perennial *Arabidopsis lyrata*

617. Aalto, Esa (2013) Genetic analysis of demography and selection in Lyrate rockcress (*Arabidopsis lyrata*) populations

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

UNIVERSITY OF OULU P.O. Box 8000 FI-90014 UNIVERSITY OF OULU FINLAND

# ACTA UNIVERSITATIS OULUENSIS



ISBN 978-952-62-0331-7 (Paperback)
ISBN 978-952-62-0332-4 (PDF)
ISSN 0355-3191 (Print)
ISSN 1796-220X (Online)

UNIVERSITY of OULU
OULUN YLIOPISTO