ABSTRACT

Context: Flexibility is an important capability for a software organization. Without flexibility a software organization risks losing its competitive advantage. To build software organization flexibility every constituent of the organization needs to be taken into account. Otherwise there are unforeseen trade-offs that could have negative impacts on the rest of the organization. Agile and Lean methodologies are known as approaches to build flexibility. However, currently Agile and Lean implementations are studied at project, and not at organization level. There is a need to understand how to build software organization flexibility.

Objective: The aim of this licentiate thesis to understand the challenges that a software organization faces with respect to flexibility. Particularly in understanding the approaches of building software organization flexibility and the associated trade-offs.

Method: A grounded theory study and a tertiary study were performed as part of this licentiate thesis. The grounded theory study was conducted to gain a better understanding of the challenges and processes in building software organization flexibility. The data were collected from an IT Department of a Fortune 500 financial institution. A tertiary study was performed to identify empirically evaluated Agile and Lean practices and their respective impacts. The findings from the tertiary study were synthesized using meta-study.

Results: The findings in this thesis uncovered a number of challenges that a software organization faces with respect to flexibility. They include budget cuts, overhead due to inconsistent development process, and regulatory changes. These challenges then caused uncertainties that impede the organization's operational efficiency, cause delays, and inefficient use of resources. To cope with the uncertainty, a software organization would build its flexibility through modifications of its organization constituents. Processes and trade-offs associated with achieving flexibility were also identified. This thesis also identified 13 Agile and Lean practices and their associated impacts on scope, quality, budget, schedule, etc. A consolidated view of the impacts of Agile and Lean practices and their empirical support is also provided. Furthermore, this licentiate thesis also highlighted the importance of synthesis method in a secondary study. A secondary study with unclear synthesis method could yield different results from a secondary study with a clear synthesis method, e.g., vote counting, meta-analysis, or meta-ethnography.

Conclusion: With the challenges that a software organization faces, building software organization flexibility is becoming more prevalent. To improve software organization flexibility different constituents of the organization need to be considered. Otherwise, the trade-offs associated to achieving flexibility cannot be thoroughly considered. Furthermore, Agile and Lean practices can have positive, negative, or no impacts on quality, budget, schedule, etc. The findings of this thesis can help practitioners identify flexibility needs, as well as improve their awareness of possible negative trade-offs when building software organization flexibility.

Keywords: Flexibility, Software Organizations, Agile, Lean.
Understanding Flexibility of a Software Organization

Indira Nurdiani
Understanding Flexibility of a Software Organization

Indira Nurdiani

Licentiate Dissertation in Software Engineering

Department of Software Engineering
Blekinge Institute of Technology
SWEDEN
# Contents

**Abstract** iii  
**Acknowledgements** v  
**Overview of Publications** vii  
- Papers in this Thesis vii  
- Related Papers not in this Thesis vii  
- Other Papers not in this Thesis vii  

**1 Introduction** 1  
1.1 Overview 1  
1.2 Background 2  
1.3 Research Gaps and Contributions 9  
1.4 Research Questions 10  
1.5 Research Methodology 11  
1.6 Overview of the Chapters 16  
1.7 Overall Contribution of The Thesis 17  
1.8 Conclusions and Future Work 18  

**2 Towards Understanding How To Build Strategic Flexibility of an IT Organization** 21  
2.1 Introduction 22  
2.2 Background and Related Work 23  
2.3 Research Methodology 24  
2.4 Results and Analysis 26  
2.5 Discussion 35  
2.6 Summary and Conclusions 37
<table>
<thead>
<tr>
<th>3</th>
<th>An Analysis of Change Scenarios of an IT Organization for Flexibility Building</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Introduction</td>
<td>40</td>
</tr>
<tr>
<td>3.2</td>
<td>Background and Related Work</td>
<td>41</td>
</tr>
<tr>
<td>3.3</td>
<td>Research Methodology</td>
<td>43</td>
</tr>
<tr>
<td>3.4</td>
<td>Results and Analysis</td>
<td>46</td>
</tr>
<tr>
<td>3.5</td>
<td>Build Flexibility</td>
<td>51</td>
</tr>
<tr>
<td>3.6</td>
<td>Discussion</td>
<td>56</td>
</tr>
<tr>
<td>3.7</td>
<td>Summary and Conclusion</td>
<td>60</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4</th>
<th>The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study</th>
<th>61</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>62</td>
</tr>
<tr>
<td>4.2</td>
<td>Related Work</td>
<td>63</td>
</tr>
<tr>
<td>4.3</td>
<td>Review Method</td>
<td>64</td>
</tr>
<tr>
<td>4.4</td>
<td>Overview of Agile and Lean Secondary Studies</td>
<td>72</td>
</tr>
<tr>
<td>4.5</td>
<td>Results and Analysis</td>
<td>77</td>
</tr>
<tr>
<td>4.6</td>
<td>Discussion</td>
<td>97</td>
</tr>
<tr>
<td>4.7</td>
<td>Conclusions</td>
<td>101</td>
</tr>
</tbody>
</table>

Bibliography 106
Abstract

**Context:** Flexibility is an important capability for a software organization. Without flexibility a software organization risks losing its competitive advantage. To build software organization flexibility every constituent of the organization needs to be taken into account. Otherwise there are unforeseen trade-offs that could have negative impacts on the rest of the organization. Agile and Lean methodologies are known as approaches to build flexibility. However, currently Agile and Lean implementations are studied at project, and not at organization level. There is a need to understand how to build software organization flexibility.

**Objective:** The aim of this licentiate thesis to understand the challenges that a software organization faces with respect to flexibility. Particularly in understanding the approaches of building software organization flexibility and the associated trade-offs.

**Method:** A grounded theory study and a tertiary study were performed as part of this licentiate thesis. The grounded theory study was conducted to gain a better understanding of the challenges and processes in building software organization flexibility. The data were collected from an IT Department of a Fortune 500 financial institution. A tertiary study was performed to identify empirically evaluated Agile and Lean practices and their respective impacts. The findings from the tertiary study were synthesized using meta-study.

**Results:** The findings in this thesis uncovered a number of challenges that a software organization faces with respect to flexibility. They include budget cuts, overhead due to inconsistent development process, and regulatory changes. These challenges then caused uncertainties that impede the organization’s operational efficiency, cause delays, and inefficient use of resources. To cope with the uncertainty, a software organization would build its flexibility through modifications of its organization constituents. Processes and trade-offs associated with achieving flexibility were also identified. This thesis also identified 13 Agile and
Lean practices and their associated impacts on scope, quality, budget, schedule, etc. A consolidated view of the impacts of Agile and Lean practices and their empirical support is also provided. Furthermore, this licentiate thesis also highlighted the importance of synthesis method in a secondary study. A secondary study with unclear synthesis method could yield different results from a secondary study with a clear synthesis method, e.g., vote counting, meta-analysis, or meta-ethnography.

**Conclusion:** With the challenges that a software organization faces, building software organization flexibility is becoming more prevalent. To improve software organization flexibility different constituents of the organization need to be considered. Otherwise, the trade-offs associated to achieving flexibility cannot be thoroughly considered. Furthermore, Agile and Lean practices can have positive, negative, or no impacts on quality, budget, schedule, etc. The findings of this thesis can help practitioners identify flexibility needs, as well as improve their awareness of possible negative trade-offs when building software organization flexibility.

**Keywords:** *Flexibility, Software Organizations, Agile, Lean.*
Acknowledgements

First and foremost I would like to express my gratitude to my supervisors Professor Jürgen Börstler and Dr. Samuel Fricker for their invaluable feedback and advise. Despite their busy schedule, they always make sure that I obtain the support I require.

I would also like to extend my gratitude to Dr. Kai Petersen for his insights, continuous assistance and good advice.

I appreciate all the support that I received from my colleagues in the SERL Sweden group. I would like to thank Deepika and Farnaz for their friendship and support.

I would also like to extend my gratitude to the industry partner that was involved in this work for their support.

Last but definitely not least I would like to thank my family: my parents, who constantly inspire me, my sisters, for their care and support. Ronald for his constant encouragement, good advise, and understanding. Also, to Axel for bringing fun and joy. I would also like to thank my dear friends, Aftri and Yossy for keeping me level-headed.
Overview of Papers

Papers in this Thesis


Related Papers not in this Thesis


Other Papers not in this Thesis

Paper 1: R. Jabangwe and I. Nurdiani, “Evaluating the Effectiveness in Executing Evidence-Based Software Engineering Steps,” chapter to be in-

Chapter 1

Introduction

1.1 Overview

The business environment of a firm is becoming more volatile with the influence of globalization, shorter product life cycles, technological changes, and aggressive competitors. In this highly competitive environment, a firm needs to be flexible to achieve competitive advantage. Without flexibility, a firm is risking its existence in the market. Such highly competitive environment is also applicable for a software organization.

A software organization is also faced with changes from multiple sources, like customers, suppliers, technology, regulators, and competitors (Knoll and Jarvenpaa, 1994). At the same time, a software organization also wants to keep the development cost low, shorten time to market, and produce high quality product. In the United States (US) software firms that develop medical devices must comply to regulations from Food and Drugs Administration (FDA). A medical devices software firm is often faced by regulatory changes. Often a new regulation emerged as a reaction to incidents involving medical devices. Thus, when the regulator announced a new regulation, it might already be too late to initiate a project. The lead time to implement the regulation is very short. Agile and Lean methodologies emerged to provide solutions for software organizations to deliver working software quickly to the customers and be responsive to environmental changes (Cockburn, 2002), through for instance, incremental deliveries. With incremental deliveries, software is developed in short timeboxed iterations and frequent deliveries, where increments of the software versions are
Introduction

continuously delivered to the customers. However, according to FDA’s regulations it is simply illegal to test intermediate versions of the medical devices with potential users (Vogel 2006). As in some cases, testing a medical device can entail exposing human beings to dangerous materials, like radioactive.

Like any other software firm, a software that develops medical devices also want to strive in the competition and also remain compliance to the regulations. When Agile and Lean practices do not suffice, how can a software organization be flexible and cope with the changes? Should we completely dismiss the possibility of adopting Agile and Lean practices to build or improve software organization flexibility?

There are known flexibility approaches like options thinking (Fichman et al. 2005), strategic decision making (Huber and McDaniel 1986), etc. However these approaches are not yet implemented and evaluated in the contexts of a software organization. There is a need to understand the challenges that a software organization faces that leads to need to be flexible. Better understanding of the challenges pertaining to flexibility can lead to a better formulation of a solution that fits the need of the software organization.

The aim of this thesis is to understand challenges that a software organization faces with respect to flexibility and how flexibility is built, and explore the potential of Agile and Lean practices to build software organization flexibility. There are two sub-contributions of this thesis. First is a descriptive view that elaborates the stages in building flexibility for a software organization. Second is understanding the potential of Agile and Lean practices with respect to building software organization flexibility.

The relevant concepts to this thesis are presented in Section 1.2. The research gap are presented in section 1.3. The research questions are presented in section 1.4. The research methodology used as part of this thesis work is presented in Section 1.5. The overview of the chapters in this thesis is presented in section 1.6. The contribution of the thesis is summarized in Section 1.7. Lastly, the conclusions and future work of this thesis are presented in Section 1.8.

1.2 Background

This section describes the concepts used in the licentiate thesis. First, definition of a software organization is described and why it needs flexibility. Next, definitions of flexibility and describe known approaches to building flexibility from various domains are provided. Lastly, Agile and Lean methodologies as known approaches to build flexibility in Software Engineering (SE) is described.
1.2 Background

1.2.1 Software Organization

Organizations can be conceptualized as entities that interact with an environment through inputs, conversions, and outputs (Jones 2012; Ilgen et al. 2005). Figure 1.1 for a schematic overview.

![Schematic for Software Organization](image)

The environment refers to the entities that generate needs, provide inputs, and consume the outputs generated by the organization. The environment includes customers, legislation, suppliers, competitors, and other stakeholders. The inputs refer to the things that the organization obtains to produce value in response to changes in the organization’s environment. The inputs include raw materials, human resources, and information that are obtained from regulators, suppliers, customers, and competitors. The conversions refer to the things the organization requires to transform the inputs into something of value. IT management described five constituents for conversion, they are, workforce, management, processes, organizational structure, and infrastructure (Tapanainen et al. 2008). Workforce refers to human resources and their skills and knowledge. Process refers to concerted activities that take place in the organization, for example a development process. Organizational structure refers to the organization hierarchy and roles and how they relate to each other. Infrastructure refers to technology platforms that are required to conceive, build, release, and
evolve products and services. The outputs refer to the elements that the organization releases to its environment and include products, services, and value that are released to the environment.

In many ways, a software organization shares the same components with a generic organization or an IT organization. Software organizations play an important role in enabling change for their customers and, while doing so, are subject to change themselves (Boar, 1998; Rockart et al., 1996). Managerial, operational, support, and network processes of business units evolve. For example, a change in regulation can generate the need for new business processes that can be handled only if appropriate automation by software is available (Shaw et al., 2006). Similarly, computing environments, development methods, software technologies, and competitors change. For example, the ability of rapidly re-prioritizing requirements with Agile development processes encourage organizations to abandon plan-driven development (Petersen and Wohlin, 2010). These changes in the context of the software organization generate opportunities and pressure to develop, adapt, and re-compose the products and services provided by the software organization. These changes may also imply a need to re-design the organization and adapt processes, skills, and knowledge that are used for development.

1.2.2 Flexibility

Flexibility is a multidisciplinary concept that is studied in different domains. There are different definitions of flexibility depending on the context and domain (Sethi and Sethi, 1990; Golden and Powell, 2000). Across the different definitions of flexibility it can be summarized as the ability of an entity to cope with changes with: (1) effectiveness, timeliness, and satisfaction with a change (Kara et al., 2002), (2) balance of the amount of change and stability (Regev et al., 2006), (3) minimal difficulty, cost, time, effort, and risk of the change (De Toni and Tonchia, 1998; Huber and McDaniel, 1986; Sethi and Sethi, 1990), (4) and the universality of the entity expected to cope with variations of input (Knoll and Jarvenpaa, 1994).

For a manufacturing firm, flexibility is required to cope with internal and external uncertainties (De Toni and Tonchia, 1998; Sethi and Sethi, 1990). Just like a software organization, there are different constituents that make up a manufacturing firm. Flexibility of a manufacturing firm is contributed by the flexibility of the constituents. Sethi and Sethi (1990) describe a schematic of a manufacturing firm constituents, see Figure 1.2.
Flexibility is viewed as an end-to-end capability from material handling to marketing with the support of the organization structure [Sethi and Sethi, 1990]. The relations among these constituents need to be considered in building flexibility of a firm. To build flexibility, the manufacturing literature suggests three main steps. First is identifying which constituents that require flexibility given the uncertainties that the firm faces (Boyle, 2006; De Toni and Tonchia, 1998; Gerwin, 1993; Slack, 1988). Second is implementing the flexibility building decision on the respective constituents (Boyle, 2006; Slack, 1988). Third is performing an audit of the planned and actual achieved flexibility (Boyle, 2006; Gerwin, 1993; Slack, 1988). For instance, changing market demands lead to uncertainty of the types of product to be manufactured. A manufacturing firm needs to move different parts of the manufacturing facilities as well as increase manufacturing capacity. Therefore, flexibility needs to be built into *material handling* and *expansion*. An audit is then performed to see if the increasing flexibility of material handling and expansion address the uncertainties.
In the organization management literature, flexibility is “the degree to which an organization possesses a variety of actual and potential procedures, and the rapidity by which it can implement these procedures” (Leeuw and Volberda, 1996). Building and improving flexibility is a continuous management task. Management needs to have an overview of the organization constituents to build the organization flexibility. For instance, to cope with new demands of product variations, management can decide to provide training to personnel so they develop multiple skills, as well as hire contractors to provide consultancy pertaining to the new product variations.

Building flexibility can be perceived from the organization’s intention, i.e., proactive and reactive (Leeuw and Volberda, 1996; Golden and Powell, 2000). A proactive organization initiate improvements without being triggered by the environment (Leeuw and Volberda, 1996), e.g., maintaining research and development capability, venturing to a new area which the organization is likely can benefit. Meanwhile, a reactive organization initiate improvement as a reaction to the environmental changes (Golden and Powell, 2000), e.g., modification of roles and their definition in the organization (Van der Aalst and Jablonski, 2000; Sharfman and Dean Jr, 2003), or sufficiently diversify the products to cope with new demands (Leeuw and Volberda, 1996).

In SE, flexibility is a known quality attribute of a software product (ISO/IEC, 2010). Looking at the definitions of flexibility in SE, it is associated with the ease and effort of modifying software products or components of the software product (IEEE, 1990; ISO/IEC, 2010; McCall et al., 1977). Currently the focus of flexibility in SE is more emphasized on the software development process. There are known approaches to build flexibility into the software development process, such as Agile and Lean methodologies. More about Agile and Lean methodologies are described in Section 1.2.4.

Building a firm’s flexibility requires a holistic approach, where all the different organization constituents are considered. Making one aspect more flexible might lead to other aspects requiring more control (Leeuw and Volberda, 1996; Sethi and Sethi, 1990). Without a holistic view, the trade-offs of flexibility cannot be thoroughly considered (Ferdows and Meyer, 1990; Harris et al., 2009). For instance to increase product variations, a machine’s capability to perform different types of operations is to be increased. If a machine is capable of doing different operations, the machine will be on demand for producing multiple products. A certain product variety might be delayed because the machine is currently used to develop other types of product. In this case improving machine flexibility can have a negative impact on production flexibility.
1.2 Background

1.2.3 Flexibility versus Agility

We cannot discuss flexibility without its counterpart, namely agility. The term flexibility and agility are often used interchangeably in the literature (Agarwal et al., 2006; Conboy, 2009).

There are many similarities between agility and flexibility (Conboy, 2009). For instance, one of the definition of agility is the ability to rapidly respond to unpredictable change (van Oosterhout et al., 2006). Such definition is also shared by flexibility.

Some suggests that the difference between agility complements flexibility with respect to the following factors:

- Speed. Agility adds the dimension of “speed” in flexibility (Tan, 1998; Kusiak and He, 1997), such as quickly, rapidly, etc. This is true to some extent as definition of flexibility emphasizes the ability to cope with change with “little penalty of time”, that does not seem to emphasize swiftness.

- Iterative approach. Goldman et al. (1995) suggests that agility is about continuously adapting to change. Meanwhile the adverb continuously is not discussed in flexibility.

- Economic aspect. Agility adds the economic aspect to flexibility. Agility is also aimed at gaining profit from the changing situations (Bernardes and Hanna, 2009; Goldman et al., 1995).

- Learning. Agile adds the aspect of knowledge management and learning (Conboy, 2009).

Another distinction between agility and flexibility is what they seek to achieve. Flexibility is a mean to achieve certain goals (Bernardes and Hanna, 2009), e.g., fulfill business needs or comply with regulations. Flexibility is the ability to cope with uncertainties by having options under the current situations or settings. Meanwhile, agility is the ability to change or rapidly reconfigure the organization’s setting.

1.2.4 Agile and Lean Software Development

Agile software development is a group of iterative and evolutionary methods that are aimed at delivering working software quickly to the customers and at the same time improve responsive to market changes (Cockburn, 2002). These methods include Scrum, eXtreme Programming (XP), Crystal, Dynamic System
Development Method (DSDM), Adaptive Software Development (ASD), and Feature Driven Development (FDD) (Leffingwell 2007; Cockburn 2002).

Agile development includes practices that enable flexibility into the software development process such as short iterations, collective code ownership, and refactoring (Abrahamsson et al. 2002; Petersen 2011; Beck 1999). Short iterations allow changes to be integrated as soon as the need for change is discovered from customer feedback (Petersen 2011). Collective code ownership increases knowledge sharing and supports flexibility by making it easier to assign people to tasks (Abrahamsson et al. 2002). Refactoring systematizes the structure of working code to improve its readability and understandability, should changes are required, learning time is reduced (Fowler et al. 1999).

Lean Software Development is derived from lean manufacturing and lean product development. Many suggest that Lean is one of Agile methodologies (Dybå and Dingsoyr 2008; Leffingwell 2007). Poppendieck and Poppendieck (2003) suggest that Lean software development is the expansion of Agile’s theoretical foundation through application of well known lean principles into software development. A comparison of Agile and Lean methods suggest that these methods share similar principles and practices, yet there are aspects that are unique to each method (Petersen 2011). Lean does not propose specific process models like Scrum or XP (Petersen 2011). Lean proposes principles and tools, such as Value Stream Mapping (VSM), pull-systems, etc.

Lean principles such as removing wastes, decide as late as possible, deliver as fast as possible, and see the whole also promote flexibility (Poppendieck and Poppendieck 2003). Removing waste entails avoiding anything that does not contribute in delivering value to the customers. This principle is in line with the definition of flexibility that entails minimizing penalty with respect to time, cost and performance. Decide as late as possible allows decisions to be made based on the most facts and not on speculations, rework is less likely to be done. Deliver as fast as possible allows quick feedback from the customer which in turn will improve the software product. See the whole allows bottlenecks in delivering the software product to be identified and quickly removed.

Agile and Lean methodologies provide software organizations with principles, practices and tools to make software development projects more flexible. Benefits like higher productivity, improved time to market, higher software product quality, and lower development cost are often mentioned (Leffingwell 2007). However, currently Agile and Lean methodologies are implemented at the project level. The applicability of Agile and Lean practices to build flexibility of a software organization is not yet examined.
1.3 Research Gaps and Contributions

The following research gaps have been identified through the background study of this licentiate thesis:

- To build software organization flexibility, different constituents of the organization need to be considered. As building flexibility for one constituent can compromise another constituent. However, Agile and Lean methodologies are primarily concerned with the process constituent to manage projects for a software organization. There is a need to understand how to build software organization flexibility.

- Agile and Lean practices are currently implemented at the project level. Before Agile and Lean practices can be scaled up to build software organization flexibility. The impacts of Agile and Lean practices need to be evaluated. Currently there is no consolidated view of empirically evaluated Agile and Lean practices and their impacts.

The first contribution of this licentiate thesis is a better understanding of building software organization flexibility which was achieved by identifying challenges and trade-offs pertaining to flexibility. The role of decision-makers of a software organization in improving software organization flexibility was identified. Furthermore, through grounded theory we formulated a model that elaborates the different events that take place as a software organization build flexibility. The model offers a unified view of flexibility building for a software organization, from ad-hoc and latent experience to a well-constructed and explicit point of view. The first contribution partially addressed the need to understand how to build software organization flexibility. This licentiate thesis does not provide a detailed guideline or framework as the one seen in manufacturing (see Figure 1.2). However, findings in this thesis provide a better understanding pertaining to the need of flexibility in a software organization, actions that were taken to achieve flexibility, and the associated trade-offs.

The second contribution of this licentiate thesis is a consolidation of Agile and Lean practices and their impacts. Agile and Lean practices that have been empirically evaluated and synthesized in secondary studies were collected. This licentiate thesis identified positive and negative impacts on Agile and Lean practices on different aspects of a project, such as cost, schedule, budget, etc. These aspects are also known as project constraints in Project Management Body of Knowledge (PMBOK) ([Project Management Institute](https://www.pmi.org) [2004]). The tertiary study did not find any discussion of the impacts of Agile and Lean
practices at the organizational level. Furthermore, for each practice and its impact on project constraint, the number of primary studies that reported on it was consolidated.

1.4 Research Questions

The aim of this thesis is to understand the challenges that a software organization faces with respect to flexibility and how flexibility is built, and explore the potential of Agile and Lean practices to build software organization flexibility. The following research questions have been formulated to achieve the thesis aim:

• RQ1. What does software organization flexibility entail?
  – RQ1.1. What is the process of building software organization flexibility?
  – RQ1.2. What are the challenges in building software organization flexibility?

• RQ2. What is known about the impacts of Agile and Lean practices?
  – RQ2.1. What are the studied Agile and Lean practices?
  – RQ2.2. What are the impacts of Agile and Lean practices?
  – RQ2.3. What is the level of empirical support of studied Agile and Lean practices and their impacts?

In understanding software organization flexibility, what flexibility entails for a software organization needs to be examined (RQ1). To aid in answering RQ1 a better understanding flexibility building processes (RQ1.1) and the challenges that a software organization face (RQ1.2) need to be addressed. RQ1 and its sub-questions are addressed through a grounded theory study reported in Chapter 2 and Chapter 3.

When discussing about software organization flexibility, we need to consider Agile and Lean practices. Before adopting Agile and Lean practices to help build software organization flexibility, a better understanding of their impacts is needed (RQ2). First Agile and Lean practices which have been studied need to be identified (RQ2.1). Furthermore, the impacts of Agile and Lean practices need to be examined (RQ2.2). Lastly, to judge whether the impacts of Agile and Lean practices are empirically substantiated the empirical support needs to be evaluated (R2.3). RQ2 and its sub-questions are addressed in a tertiary study reported in Chapter 4.
1.5 Research Methodology

In this section, the methodologies used in the licentiate thesis are presented. The research methods used in this licentiate thesis were grounded theory and tertiary. An overview of the research methods and why they were used adopted in this licentiate thesis is discussed.

1.5.1 Grounded Theory

Grounded theory methods were used as a research method in the empirical study reported in Chapter 2 and Chapter 3. Grounded theory methods consist of “systematic, yet flexible guidelines for collecting and analyzing qualitative data to construct theories from the data themselves” (Charmaz 2014). Grounded theory is an inductive research approach. Instead of deducing a theory and the doing data collection, grounded theory starts with the data to formulate a theory. Grounded theory involves iterative steps between data collection analysis and allows the researcher to continuously interact the data and the emerging analysis.

Grounded theory is a well-established research method that is often used in SE, Information Technology, and Information Systems research (Coleman and Connor 2007; Seaman 2008). Grounded theory is suitable when:

• Little is known about the study area. Yet the need is not to explore but to explain the phenomena (Birks and Mills 2011). Flexibility is a well researched area in manufacturing and management. Flexibility is also a concern in software engineering, as expressed in ISO 25010 (ISO/IEC 2010), and Agile and Lean literature. However, little is known what flexibility building processes entail.

• The phenomena to be studied is inherent, complex, and socially constructed (Birks and Mills 2011; Charmaz 2000). SE is considered a multifaceted area that covers the technical aspects as well as social aspects on software development (Easterbrook et al. 2008). Many aspects of software development is also part of human activities. So is building software organization flexibility.

• The intended outcome of the study is a theory (Birks and Mills 2011). The intention is not to build a theory per se, however a generic view of flexibility building for a software organization was intended to be formulated. Furthermore, the generic view was to be formulated from experiences of people who are involved in the process.
Method Selection

There are several options of research method in flexible research other than grounded theory. One of the flexible research approaches is inductive case study (Eisenhardt, 1989). Inductive case study is similar to grounded theory. Instead of testing a theory or a hypothesis, inductive case study is aimed to formulate a theory or a hypothesis from the cases. Inductive case study also involve iterative process between data collection and analysis. However, unlike a grounded theory study, in inductive case study a priori constructs are to be defined prior to data collection (Eisenhardt, 1989). The aim of this thesis is to capture the process of building software organization flexibility. In this case, a priori constructs might bias and deviate the data collection and data analysis process. Instead of building the theory from the data, a priori construct might force the data to fit the construct.

Sampling Strategy

In grounded theory, the sampling strategy is purposive or theoretical sampling (Robson, 2011). In recruiting the industry partner, we were interested in a software organization that has gone through changes and/or re-organizations.

Case Company

For Chapter 2 and Chapter 3 an empirical study was conducted at an IT department that provides services to one of the business units of a Fortune 500 financial institution. The business unit provides financial service for private and corporate clients in investment banking, assets, and wealth management. The IT Department itself has around 200 developers who take part in developing, hosting, and maintaining the software solutions required by the business unit. As part of their effort to be more flexible, the IT department has undergone several changes in their organization over the past decade.

Recruiting Interviewees

In recruiting interviewees purposive sampling (Charmaz, 2014) was selected. The interviewees were selected given their experiences as initiators or partakers of the organizational changes. Selection of interviewees was also performed to ensure maximum variation sampling of change cases (Miles and Huberman, 1994). More details pertaining to recruiting interviewees can be found in Chapter 2 and Chapter 3.
1.5 Research Methodology

Data Collection

Semi-structured interview was selected as an interview strategy. The aim of semi-structured interviews is to get a holistic understanding of the situation in the organization, what changes had been faced, what kind of challenges were faced, what had been done to address the change, and how they were perceived (Weiss, 2008). The interview questions were continuously modified as the researchers’ understanding of flexibility building evolved. Memos were also made during and after the interviews to document emerging concepts captured during the interviews.

Apart from interviews, project documentations, company policies, standards, and lessons learned were also collected. Documentations were important source of data for the purpose of data triangulation (Robson, 2011).

Analysis

In performing the data analysis, the coding procedures mentioned in Charmaz (2014) were followed. The coding process started with initial coding where line-by-line coding was performed. The next coding steps were axial coding and lastly theoretical coding. The coding steps were done iteratively. More detail on each coding step can be found in Chapter 3, Section 3.3.

Grounded theory prescribed that there should be a feedback loop between data collection and analysis (Charmaz, 2014). Therefore, the coding steps were performed for each round of interviews. Memos made during data collection were also used during analysis. Furthermore, memos were also made during the coding steps, to document emerging codes and categories and aid the iterative coding process.

1.5.2 Tertiary Literature Review

In Chapter 4 a tertiary review was performed. A tertiary review is a systematic review of systematic reviews (Kitchenham and Charters, 2007). The purpose of the tertiary study is twofold: (1) get an overview and trends of published secondary studies in Agile and Lean (2) to identify Agile and Lean practice and their impacts on project constraints, and examine the level of empirical support.

Method Selection

A tertiary review was conducted because there are already many secondary studies in Agile and Lean (Kitchenham and Charters, 2007). A tertiary review
has the same steps as a systematic literature review. The protocol described in [Kitchenham and Charters, 2007] were used as a guideline. The steps include, planning the review, defining a search strategy, performing screening with well defined inclusion/exclusion criteria, performing quality assessment, extracting data, and synthesizing the data. More detail on the review steps can be found in Chapter 4, Section 4.3.

Retrieving Potential Secondary Studies

To retrieve the secondary studies automated search in five publisher and reference databases were performed, they are: IEEE Explore, ISI Web of Knowledge, Compendex, Scopus & Inspec, and ACM. A search string which has been piloted and consulted with an expert was used. The search string contained keywords pertaining to Agile and Lean methods, and secondary study and its synonyms.

Data Extraction

From the included secondary studies, data pertaining to the studied Agile and Lean practices and their impacts on project constraints were collected. Empirical methods used in the primary studies from which the evidence was obtained were also collected.

Synthesis

In synthesizing the data from the secondary study, qualitative meta-study [Patterson et al., 2001] was selected. Meta-study is “a research approach involving analysis of the theory, methods, and findings of qualitative research and the synthesis of these insights into new ways of thinking about phenomena” [Patterson et al., 2001]. Meta-study was selected because it considers the data and the methodological soundness of the studies. In presenting the results of the tertiary study, it is also important to consider the quality of the secondary studies themselves.

1.5.3 Validity Threats

Like any research work, the works included in this licentiate thesis are subjects to validity threats. There are validity threats that are often associated with flexible research [Robson, 2011]. They are:
1.5 Research Methodology

- Credibility. This is an issue both in the grounded theory study (Chapter 2 and Chapter 3), and the tertiary study (Chapter 4). Credibility is compromised when the collected data are insufficient. For the grounded theory study the issue of credibility was mitigated by having multiple data sources, or data triangulation. Documentations were also collected in addition to the interviews. For the tertiary study, credibility issue was mitigated by having broad search terms and being inclusive in the early phase of the secondary studies screening.

- Bias. This is an issue both in the grounded theory study (Chapter 2 and Chapter 3), and tertiary study (Chapter 4). In grounded theory the source of bias could be from the interviewees as well as the researcher. To minimize bias from the interviewees, recruited interviewees must participate by their own will and not by force. Bias from the researcher could be influenced by the preconceived notions from the researchers. Bias was mitigated by having two researchers involved in the data collection and three researchers involved in the coding process. The tertiary study is a subject to reviewer bias. Bias in the tertiary study was mitigated by performing pilot reviews, test-retest (Kitchenham, 2010), and post-hoc validation.

- Reliability. This is an issue both in the grounded theory study (Chapter 2 and Chapter 3), and the tertiary study (Chapter 4). Reliability is compromised when the tools and measures used the study is inconsistent. To minimize reliability issue in grounded theory study, we developed a preliminary list of open-ended question which were later modified during the interview. To maintain audit trail in the coding process, a tool called Atlas.ti was used. In the tertiary study, to improve the reliability, the inclusion/exclusion criteria and data extraction form went through iterations piloting process, involving more than one reviewers, and calculation of Kappa coefficient.

- Generalizability. This is an issue primarily for the grounded theory study (Chapter 2 and Chapter 3). Generalizability is a risk to the applicability of the results outside the case company. We minimized this risk by selecting a variety of change cases that covered the different organization constituents defined by Tapanainen et al. (2008). Furthermore, the selected change cases are typical cases in other software organizations that support financial institutions.

1http://atlasti.com/
More detailed on specific issues pertaining to the different research methods employed as part of this thesis are addressed in individual chapter. Validity threats related to the grounded theory and their respective mitigation strategies are discussed in Chapter 2 and Chapter 3. Validity threats related to the tertiary reviews and their respective mitigation strategies are discussed in Chapter 4.

1.6 Overview of the Chapters

Chapter 2 presents the early results from the grounded theory study. The data collected for this chapter were from nine interviews with managers from the case company. Chapter 2 presents the results from the open and axial coding stages of grounded theory. The early result from the grounded theory study reveals the challenges that a software organization faces pertaining to flexibility. A software organization faces challenges not only to changing market or customers demands, but also changes in regulations, and also budget cuts. Chapter 2 shows the key role that decision-makers have in building flexibility in a software organization. The results from the axial coding showed that flexibility building is initiated due to emerging uncertainty. The decision maker then recognized problems caused by the uncertainties and then initiated a change in the organization. In enacting the change, the decision maker decided which organization constituents to modify (see Section 1.2 for definitions of constituents). Decision-makers explored one or multiple options to address the uncertainties the software organization was confronted with. The selected option was the one that could deliver the most advantage with minimal disadvantage for the software organization organization.

Chapter 3 presents the result from the subsequent iterations of the coding process as well as the theoretical coding. The data used in this chapter were the same interviews used in Chapter 2. Chapter 3 reveals the flexibility for a software organization is not about being able to constantly change the organization structure given the uncertainties. However, flexibility is about being able to change the organization to be able to withstand uncertainties. Furthermore, the data shows that there are three main transitional stages as a software organization build its flexibility, they are pre-change, transitional, and post-change.

In pre-change, uncertainty emerged and led to the need for flexibility. Uncertainty could be changes in legislation to led to the need to be compliant. In transitional stage, flexibility building is taking place where an organization constituent is modified. A constituent modification could entail the decision to build the information system in house so the system could be tailored to achieve
compliance. In post-change, flexibility is achieved with associated benefits and also trade-offs. Achieving flexibility entailed developing a system that allowed the business to quickly enable or disable certain functionalities depending on the new regulation issued by the legislator. However, achieving this flexibility had higher costs than developing the functionalities on demand. Furthermore, flexibility building process could be iterative, if the associated flexibility trade-off introduces new uncertainty, another round of modification could be initiated.

Chapter 4 presents the results from the tertiary study. The secondary studies were obtained through automated search on five databases, they are ACM, IEEE Explore, Inspec, Compendex & Scopus, and ISI Web of Knowledge. From the tertiary study, an overview of published secondary studies in Agile and Lean was obtained. A large number of published secondary studies with large variations of topics was found. The tertiary study also revealed that most secondary studies were aimed at identifying factors pertaining to Agile and Lean. Very little secondary studies were aimed at studying the outcome of the Agile and Lean practices. The tertiary study also revealed that there was little overlap of primary studies among the included secondary studies.

Furthermore, 13 Agile and Lean practices were identified. The impacts of Agile and Lean practices on quality, budget, etc., were also identified and mapped to project constraints as mentioned in PMBOK (Project Management Institute 2004). The impacts (positive, negative, or no impact) of each Agile and Lean practice on project constraints and their empirical support were examined. The impact of TDD on external quality is the most studied practice in secondary studies.

The tertiary study revealed that many of the secondary studies that studied the impacts of Agile and Lean practices had low quality. Many of the secondary studies also did not perform evidence synthesis and merely summarized what was mentioned from the primary studies. Furthermore, the tertiary study also showed that secondary studies with clear synthesis method could yield different outcomes from those with unclear synthesis method.

1.7 Overall Contribution of The Thesis

This licentiate thesis provided a better understanding pertaining to building software organization flexibility. Challenges pertaining to flexibility were also identified. Through a grounded theory study, this thesis revealed that a software organization builds its flexibility through modification of one or more of its constituents. For a software organization, flexibility is not about make changes
to the organization structure quickly every time it faced an uncertainty. A flexible software organization is an organization that is able to restructure its organization constituents in a way that allow the organization to withstand a specific type of uncertainty. Flexibility is built to fulfil a certain flexibility need that emerges from uncertainties that the organization faces. Strategies to build flexibility could include establishing a standardized process model, or over-specifying a software system’s functionalities. The role of the decision-makers in the organization is also crucial to recognize difficulties that are caused by the uncertainty and to initiate the modification to the constituents.

Furthermore, this licentiate thesis also revealed that Agile and Lean practices are still implemented at the project level. The tertiary study that was performed as part of this licentiate did not reveal any evaluation of Agile and Lean practice to build flexibility at organizational level. This thesis provided a consolidated view of Agile and Lean practices on project constraints, as well as the number of empirical studies that supported them. The consolidated view provides an overview of Agile and Lean practices and whether they could have positive, negative, or no impact on quality, schedule, budget, etc. Furthermore, the tertiary study revealed the importance of selecting an appropriate synthesis method in a secondary study. As the selection of synthesis method could influence the results of the secondary study.

The findings from this licentiate thesis uncovered the need for a thorough evaluation before adopting any approach to build software organization flexibility. Therefore the adopted approach can actually address the software organization flexibility needs. The consolidated view can provide practitioners with the insights of which Agile and Lean practice could have positive, negative, or no impact on project constraints. In summary, the findings from this licentiate thesis can aid practitioners to identify flexibility needs, as well as better evaluate which approach is most suitable for their needs and the associated trade-offs.

1.8 Conclusions and Future Work

Flexibility is an important ability for a software organization to strive. Agile and Lean methods have emerged to offer solutions for software organization to react to changes more quickly. However, Agile and Lean methods address flexibility for software projects and not on organization level.

The first contribution of this thesis is a better understanding of building software organization flexibility. The findings from this thesis show that flexibility is built into the organization constituent to address the uncertainties. To improve
1.8 Conclusions and Future Work

software organization flexibility different constituents of the organization need to be considered. Otherwise trade-offs associated to achieving flexibility cannot be thoroughly considered. Through grounded theory study an explanatory model of events that took place as a software organization built flexibility was uncovered. The model does not detail guidelines for decision-makers to build flexibility. However, it elaborates the events that took place as an IT organization built flexibility of its different constituents. The model offers a unified and explicit view of flexibility building for an IT organization, gathered from discrete and latent experiences. The grounded theory study also revealed the strategies used by a software organization to build flexibility.

The second contribution of this thesis is a consolidated view of Agile and Lean practices impacts on project constraints with the relevant empirical support. This was achieved through a tertiary study of Agile and Lean secondary studies. The tertiary study reveals that most secondary studies in Agile do not provide thorough evaluations of the quality of the primary studies and strengths of evidence are often neglected. Furthermore, secondary studies with unclear synthesis approach could yield results that are different from secondary studies that employed qualitative or quantitative synthesis approaches, e.g., meta-ethnography, meta-analysis, or vote counting.

With regard to future work, there is a need to improve the ability of practitioners to better assess the possible trade-off associated with the actions taken to build flexibility. A map can be formulated to help practitioners find associations between their decisions in building flexibility and their trade-offs. Such a mapping allows practitioners to identify flexibility building decision that is the most suitable for their flexibility need with acceptable trade-offs. Practitioners can then use the map to understand the possible consequences of implementing the decision. Furthermore, practitioners can tailor their own approach or approaches to fit their organization and flexibility needs.
Chapter 2

Towards Understanding How To Build Strategic Flexibility of an IT Organization

Indira Nurdiani, Samuel A. Fricker, Jürgen Börstler

Abstract

IT organizations need to react to changes in the business, the domain (e.g., regulatory issues), and the technological development. While some of these changes can be handled by adopting agile practices, others might have large, irreversible effects on the organization as a whole. While flexibility and agility have found their way into software project methodologies, IT organizations struggle with their adaptation at organizational level. This paper presents preliminary results of a grounded-theory study aimed at understanding how experienced managers handle flexibility. The results are a rich empirical source for improving flexibility of an IT organization at the strategic level and also a good starting point for further research towards generalizing agile ideas beyond software projects.
Towards Understanding How To Build Strategic Flexibility of an IT Organization

2.1 Introduction

Information Technology (IT) has become an integral part of an organization’s business strategy \cite{Henderson1993}. An IT organization helps in implementing these strategies by developing, operating, and managing software systems that support the organization’s work. To offer its services to the business units, the IT organization provides relevant expertise, skills, processes, and infrastructure \cite{Knoll1994, Byrd2000}.

An IT organization needs to be able to cope with changes of business demands \cite{Byrd2000}. Such changes can influence the IT organization’s structure, processes, workforce, IT infrastructure, and services. Changes that require such strategic flexibility can originate from customers, suppliers, competitors, governing bodies, and developments in technology \cite{Knoll1994, Feeny1998, Jones2012, Rockart1996}. To handle changes in a turbulent business environment, IT organizations need to develop and improve their strategic flexibility. Without a flexible IT organization, a company might lose its market position and competitive advantage \cite{Byrd2000}.

Today, a holistic understanding of how to achieve strategic flexibility is missing. Advice can be found at the level of product planning decisions \cite{Fricker2012}, development processes \cite{Dingsøyr2012, Petersen2011}, and software architecture \cite{Eriksson2006, Clements2001, Chen2009}. However, frameworks that target the strategic levels of a software organization, such as ITIL and CMMI, merely state that the organization needs to be flexible. They fail at explaining how to assess and improve such flexibility.

When best practices for specific improvement goals have not been established yet, inductive approaches are used to guide process improvements \cite{Briand1995, Pettersson2008}. An inductive approach elicits past experience and exposes it to the organization to learn from it \cite{Garvin2000}. If the results from inductive process improvement are attractive, they should be spread and used for enhancing prescriptive benchmarking frameworks or for creating new ones.

This paper reports on early results of a grounded-theory study to better understand how strategic flexibility is built in an IT organization. To elicit past experience, interviews were conducted with managers of the IT organization about representative changes that they encountered. The results illuminate when managers initiate flexibility-related strategic change and how they act when improving strategic flexibility.
2.2 Background and Related Work

The remainder of this paper is organized as follows. Section 2 presents the background and related work. Section 3 describes the research approach. Section 4 presents the results, which are discussed in Section 5. Section 6 summarizes and concludes the paper.

2.2 Background and Related Work

To handle changes in technology, markets, regulations, and budget, flexibility has become an important concern for IT executives. Such changes may affect any aspect of the IT organization, including infrastructure, applications, data, hardware, knowledge, capabilities, and staff needed to build, manage, and evolve the IT resources. Flexibility is the ability to change easily and effectively (De Toni and Tonchia, 1998; Saleh et al., 2009) and is characterized as follows:

- Up-to-dateness and effectiveness of a change and satisfaction with the results of the change (Nurcan, 2008; ISO/IEC, 2010; Kara et al., 2002),
- Difficulty, cost, time, effort, and risk of implementing a change (De Toni and Tonchia, 1998; Saleh et al., 2009; Sethi and Sethi, 1990; IEEE, 1990; Naab and Stammel, 2012),
- Extent of change and stability (Regev et al., 2006), and
- Universality of the entity expected to be flexible (Knoll and Jarvenpaa, 1994).

Strategic flexibility is the ability of an organization to adapt quickly to changes and thereby be in a position of competitive advantage (Hitt et al., 1998). Such adaptation leads to acceptance of new inputs, modification of the organization’s constituents, and production of new results (Knoll and Jarvenpaa, 1994; Hitt et al., 1998; Benamati and Lederer, 1998). The inputs are the goals, constraints, and resources the organization considers to create outputs.

The constituents refer to the workforce, management and leadership, processes, organizational structures, and infrastructure that are used to transform the inputs into value (Tapasainen et al., 2008). The outputs are the things of value, such as products and services that the organization produces and releases to the environment.

Strategic flexibility has been studied extensively in engineering and the manufacturing of physical goods, where changes are addressed by raw material volumes, routings, production capacities, and product mixes (Saleh et al., 2009). Software, in contrast to manufacturing systems, is characterized as being intangible, knowledge-intensive, and having complex interdependencies (Cusumano, 2004; Fricker, 2012). These differences shift interest away from material routing
Towards Understanding How To Build Strategic Flexibility of an IT Organization

and logistics to development, evolution, and integration of new or changed software. Existing knowledge on strategic flexibility of manufacturing organizations has therefore limited applicability for IT organizations.

The understanding of how an IT organization builds flexibility is limited to tactics and to the process and infrastructure constituents [Benamati and Lederer 1998] [26]. Development is made flexible with agile processes that enable collective code ownership, incremental delivery, and rapid feedback [Dingsøyr et al. 2012; Petersen 2011]. The IT infrastructure is made flexible by planning the reuse of core assets across multiple products and services [Eriksson et al. 2006; Clements and Northrop 2001; Chen et al. 2009].

A holistic understanding of how to build strategic flexibility is required to allow an IT organization to embrace change. All constituents of the IT organization should be considered as well as possible trade-offs between flexibility and other concerns such as cost and control [Sethi and Sethi 1990]. If not addressed holistically, the achieved flexibility might be limited or compromised by unforeseen rework and cost [Benamati and Lederer 1998].

2.3 Research Methodology

The present study aims at developing a generic understanding of the processes decision-makers in an IT organization use when they build strategic flexibility. It used grounded theory, as suggested by Charmaz (2006) to uncover the processes that decision-makers go through as they build IT organization strategic flexibility. Grounded theory is a well-established research method used in software engineering, IT, and IS research (Coleman and Connor 2007).

To obtain relevant information, we formulated the following leading research question: how do decision-makers of an IT organization improve strategic flexibility? The study elicited and analyzed experiences from actual changes that the managers performed for improving their IT organization’s strategic flexibility.

We used a combination of typical case and maximum variation sampling [Miles and Huberman 1994] to identify the cases of strategic change to be studied. First, management representatives of the IT organization identified a large set of recent, relevant change cases. From this list they then selected a small set of representative, but diverse cases. For each case, the responsible manager and other staff, suggested by the manager, were identified and interviewed.

Semi-structured interviews were used as the primary data collection method [Robson 2011]. Each interview lasted approximately 60 minutes. Prior research was used to frame the initial interview questions. Following the grounded theory
guidelines (Charmaz, 2006), we made modifications to these questions during data collection as our understanding of strategic flexibility evolved. The interviews helped us to develop detailed descriptions of the change process, how the decision-makers perceived the changes, and how they reacted and reflected on them. This helped us to develop a holistic understanding of the different phases in implementing a change (Weiss, 2008).

Analysis and coding were conducted iteratively as encouraged by grounded theory methodology (Charmaz, 2006). All interviews were transcribed and coded line by line with a priori codes that allowed us to focus the study of flexibility around changes and their impacts (Saldana, 2012). Documentation obtained from the interviewees and field notes were also included in the coding steps. That analysis allowed us to understand the managers’ perspectives on the changes, identify the important processes that took place along the changes, and understand the impact of the changes on the organization.

During axial coding (Charmaz, 2006) we sorted the coded materials into problem-based initiation of change, the situations before the change where change impact predictions were made, and the results achieved with the change. This axial coding resulted in a structured, rich description of the decision-makers’ views on the phases of a change that aimed at building strategic flexibility. The results of axial coding presented in this paper, give a rich empirical basis for theory-building and the construction of an assessment and improvement framework for strategic flexibility.

2.3.1 Case Organization

We studied the IT organization of the business units of a Fortune Global 500 company in the financial services market. At the time of the study, the IT organization had 200 employees who were responsible for developing, hosting, and managing software solutions. According to independent consultants, the IT organization is representative for other IT organizations in the services industry.

We studied the following four change cases. In Reorganization 1, the organization built project staffing flexibility by shifting from a hierarchical organization to a matrix organization. In Reorganization 2, the organization simplified management by shifting from the matrix organization to a pool organization. In the third case, RUP Introduction, the organization built process and staffing flexibility by adopting the Rational Unified Process as an organization-wide development methodology. In the fourth case, Regulatory Change, the organization adapted its IT infrastructure to flexibly implement changes enforced by legislation.
Towards Understanding How To Build Strategic Flexibility of an IT Organization

Taken together, these four cases had an impact on a broad set of constituents of the IT organization. Reorganization 1 and Reorganization 2 affected the workforce, organization structure, and management. RUP Introduction affected the processes and the workforce. The Regulatory Change Implementation case affected the infrastructure of the organization and the services provided to the company’s business units. As a result, the four analyzed cases give a good base for a generic understanding of how decision-makers improve strategic flexibility of an IT organization.

2.3.2 Threats to Validity

This research is subject to the validity threats common to grounded theory, including credibility, resonance, originality, and usefulness (Charmaz 2006).

Credibility refers to whether sufficiently broad and deep data was used for analysis. This was achieved through interviews with multiple interviewees, as well as access to the organization’s documentation of processes and lessons learned for triangulation. Credibility was also supported by studying representative cases that together covered all important constituents of an IT organization.

Resonance refers to whether the research performed was able to capture the actual experience. We addressed this threat by informing our interviewees of our presence, the purpose of the study, and the kind of information to be elicited. Interviewees also participated on their own will and interest in the study. Company representatives confirmed the correctness of the results presented here.

Originality refers to the novelty from the outcome of the study. The study was undertaken because of the limited understanding of how to build strategic flexibility in an IT organization. It resulted in an illumination of the decision-makers’ perspectives on this management task. Usefulness is a risk to applicability of the results obtained beyond the studied setting. To address usefulness, we selected a varied set of cases from a representative IT organization. The results provide a rich basis to identify models and frameworks for assessing and improving strategic flexibility of an IT organization. The results are also useful for creating theories by looking at the reoccurring patterns.

2.4 Results and Analysis

The data collected during the study showed that decision-makers had a key role in building strategic flexibility. They reacted to problems that implied un-
2.4 Results and Analysis

certainty for the IT organization. Change was initiated because the uncertainty was uncomfortable for the decision-maker. He adapted the organization’s constituents and thereby built uncertainty-matching flexibility. Each adaptation implied a set of trade-offs for the organization, some of which were predicted and some not. Some trade-offs represented so severe problems that yet another change was initiated.

We identified three phases that a decision-maker went through for each change. The first phase was recognize problem and initiate change. Such initiation required that the decision-maker perceived the problem to be causing enough discomfort. The second phase was predict the effects of one or more alternative changes and select one of these options. Here the decision-maker identified the aspects of the organization that had to be changed and evaluated possible options in implementing the change. The decision was based upon an option that yielded the desired flexibility and led to acceptable strengths and weaknesses of the future IT organization. The third phase was to reflect on results. Here the decision-maker reviewed the change outcome and reflected on the flexibility, strengths, and weaknesses achieved with the change. The following subsections describe each of the three phases in detail.

2.4.1 Recognize Problem and Initiate Change

In the first phase, the decision-maker recognized the problem that had to be addressed. He or other initiators identified the problem and saw that it led to uncertainties for the IT organization. In each of the cases, the decision-maker then initiated a change because the uncertainties were of too great discomfort for him.

The core roles in this phase were the initiators who reported the problem and the decision-maker who decided on the problem’s resolution. The initiators were individuals or groups of people who perceived the need for a change. Reorganization 1 and Reorganization 2 were initiated by the Chief Information Officer (CIO). RUP Introduction was initiated by the IT organization’s internal governance board. Regulatory Change Implementation was initiated by the concerned program manager. The decision-makers were also the initiators and had the authority to alter the respective organization constituents.

“[The CIO] decided to merge them [the business-oriented key accounts] into one big pool organization.”

The problems that implied change originated not only in the organization’s environment, but could also relate to issues internal to the IT organization. Re-
Towards Understanding How To Build Strategic Flexibility of an IT Organization

organization 1 and Regulatory Change Implementation originated from problems external to the IT organization. Reorganization 2 and RUP Introduction originated from internal problems.

In **Reorganization 1**, the organization structure was hierarchical (see Figure 2.1, left); structured in departments and sections. Each section had a fixed number of people and was responsible for the development and maintenance of a particular solution.

The strength of the hierarchical structure was the ability of the development teams to produce software solutions quickly and with high quality. The people who worked on the product were experts in their domain.

A problem then emerged related to reduced financing of the IT organization. Budget reductions led to uncertainties about the focus of the development projects.

“Now, [...] sometimes they like to invest on [consumer e-services], a year later they like to invest money on a different topic.”

The uncertain project foci implied that development sections risked losing budgets that were allocated previously. The hierarchical structure of the organization hindered team members of one development section to be reallocated to another one. Preferred staff were running idle because their managers would not let them to be reallocated. This ineffective resource utilization was such a discomfort for the decision-maker that he initiated a reorganization.

“We had a situation where in one [section] we had no money.”

“For instance, [the group] SAP implementation had 180 people. After a year you don’t really need the [rest] 60 people, but these 60 people could not be relocated, and they did not have anything to do.”

![Figure 2.1: Reorganization from hierarchical organization (left) to matrix organization (right)](image-url)
In **Reorganization 2**, the organization was organized in a matrix structure (see Figure 2.2, left) with three main Service Provider Units (SPUs). Each SPU was responsible for the development and maintenance of a product and had pools of people grouped by functional roles. Across the pools there were Key Accounts who acted as product owners and were responsible for conveying business needs to the development team.

The matrix organization’s strengths were the flexibility and mobility of the team members to move across different projects. Also, it provided excellent career perspectives for the staff.

In Reorganization 2, the problem was related to process inconsistency across the SPUs. The inconsistent processes led to uncertainties regarding how to escalate projects issues when two SPUs had to work together. A common issue would be escalated to different bosses. The resulting confusions and delays caused such a discomfort for the decision-maker that he initiated a reorganization.

In **RUP Introduction**, the organization lacked a common development methodology. It had templates, for example for requirement specifications, but no common guidelines for how to use the templates. The lack of a common methodology was problematic because it implied uncertainties in the allocation of staff to projects. Each allocation had to consider whether the staff was compatible with the project. If no compatible staff could be identified, the project got delayed due to the time spent by the new staff for learning how to work in that project.

“We wanted to standardize the way of working so we can support each other, if a new architect coming to your team and [he/she] knows how you work.”
The uncertainty of how projects work and how to effectively train staff caused such discomfort for the decision-maker that he initiated the RUP introduction for standardizing ways of working.

In **Regulatory Change Implementation**, the problem was related to bilateral agreements that the company’s home country planned to sign with other countries. The agreements varied across countries, and the exact terms and rules that were going to be defined were not known.

“No one really knew the [country Y] or [country Z] law. Sometimes it was surprising what ideas that [country Y] government has or what the [country Z] government considers relevant.”

The problem generated uncertainty about the services to be provided by the company. The company needed to comply with each agreement once it would be in place, or it would be considered non-compliant. It was unclear, however, how the company could ramp up each respective service rapidly enough. Each ramp-up would generate significant work. This discomfort encouraged the decision-maker to build the necessary flexibility. This subsection showed that each change of the IT organization was triggered by a problem that caused uncertainty. The uncertainty became such a discomfort for the respective decision-maker that he initiated a change that would address the uncertainty or mitigate its impacts.

### 2.4.2 Predict Effects of Options and Select One

In the second phase, the decision-makers selected a change of one of the IT organization’s constituents for addressing the uncertainty. If multiple options were identified, each such option was evaluated in terms of its positive and negative effects on the organization. In Reorganization 1, Reorganization 2, and RUP Introduction only one option was explored. In Regulatory Change, the decision-maker used multiple options.

In **Reorganization 1**, the decision-maker addressed the project focus uncertainty by changing the organization structure to a matrix (see Figure 1, right). Staff would be grouped according to functional roles and could be assigned to different projects. This reorganization was expected to increase the flexibility of staff allocation because staff would have the possibility to move between development projects without being bound to a particular section or department.

In **Reorganization 2**, the decision-maker decided to address the escalation uncertainty by removing the SPUs and transforming the organization into one
single pool organization (see figure 2, right). The removal of the SPUs was expected to clarify the decision-making authority and thereby avoid confusions. Also staff would have clear roles and responsibilities: one pool of staff would handle the development of new business solutions, one pool would handle solution provision, and one pool would be responsible for testing and support. Each pool reported to the CIO only. The existing strength, the flexible resource utilization, was expected to be retained.

In RUP Introduction, the decision-maker decided to address the uncertainty about the ways of working by adopting RUP as the standard development methodology for the whole IT organization. The adoption of RUP was anticipated to help staff to move between projects. Staff reallocation was simplified because RUP would introduce a common way of working. A common terminology would also ease communication between staffs. Decision-makers selected RUP because some teams were already using RUP, and RUP was perceived to be an industry standard.

"Some of them did RUP already, RUP was kind of the industry standard at that time."

In Regulatory Change Implementation, the decision-maker addressed the uncertainty by exploring two options: to procure an IT solution from a third-party, or to build its own IT solution.

The option to procure was discarded, even though it was estimated to be cheaper than the other options. However, no tailored solution existed in the market and integration of an off-the-shelf product would have required significant modification effort.

The decision-maker selected to build an own IT solution. It was expected to provide the advantage of influencing the industry of how the regulations would be implemented and that it would fit the rest of the organization’s IT systems and the company’s operations.

"We decided that this [buy approach] cannot be the right way because there’s nothing on the market available, so whatever solution they buy they have to change that system [...]. We clearly said, OK, do it on a [build] approach in our infrastructure. It will be not cheaper, but in the end the system will fit in our environment."

In developing the IT solution, there were again alternative options. Maintainability and development speed needed to be traded-off with usability. Maintainability required decoupling the solution from the rest of the IT systems. However such decoupling would sacrifice usability for the clients.
“[The system was] simplified in a way that we could we created a known system just for the [regulation]. So we could test it independent to all of the system.”

“We clearly have chosen the option which has the fewest impact on all of the system. It might not be the best for the clients, because clients will get 2 confirmations out of one [...] transaction.”

Given the time constraints to deliver the solution, maintainability was favored over usability. To further accelerate the critical project, the IT organization’s three most senior business analysts were allocated and received the task to closely follow the whole development project.

“In the normal project,[...] you would not waste the most senior guy doing some functional test on the code level [...] We need the best guy to ensure that [...] the calculation is correct. Otherwise we are searching bugs and we do not know where we can find it and can search on the wrong place and just lose time”.

Also, the decision-maker accepted that the project temporarily deviated from the IT organization’s standard development processes to cope with time constraints. Such deviation was expected to require correction of the project results once the system would be in place. This subsection showed that decision-makers explored one or multiple alternative options for addressing each of the uncertainties the IT organization was confronted with. Each option targeted an uncertainty-matching flexibility and was evaluated in terms of advantages and disadvantages for the IT organization.

2.4.3 Reflect on Results

In the third phase, decision-makers reflected on the results that were achieved with the change. They assessed how the change addressed the uncertainty and the strengths and weaknesses it implied for the organization.

All four changes were concluded successfully and achieved the intended flexibility. However, not all outcomes were acceptable and some generated unpredictable weaknesses. In particular, Reorganization 1 generated new problems that could not be mitigated.

In Reorganization 1, the matrix organization structure (Figure 1, right) delivered the flexible resource utilization that was anticipated by the decision-maker. It provided additional strengths, in particular by improving the career perspectives of the organization’s staff.
However, the matrix organization led to inconsistent escalation processes for projects that spanned multiple SPUs. This weakness was not anticipated. The uncertainty of how issues would be escalated again caused so much discomfort for the decision-maker that he initiated Reorganization 2.

In **Reorganization 2**, the change resulted in a pool organization structure (Figure 2.2, right). Staff was grouped into large pools based on their roles: Solution Provision, Development Projects, Testing and Support, and Managed Services. The Key Accounts from the three SPUs in the previous matrix organization were merged into one large Solution Provision pool. The Development Project pool was responsible for development activities. It contained five sub-pools: project management (PM), business analysis (BA), development (DEV), and integration (INT). The Testing and Support pool was responsible for deploying the solutions that resulted from the development projects into the IT infrastructure.

The pool organization resolved the uncertainty of how project issues had to be escalated. It established clear responsibilities for each of the pools. Each pool was headed by a common manager for resolving project issues, the CIO.

“So if we have some escalations between these two organizations [pools], then you have to go to [CIO].”

Another strength of the pool organization was resource utilization flexibility that was further improved. The organization of staff into role-specific pools made allocations to projects easy and allowed respecting changing business needs.

“We are sourcing the project depending the business needs and budget they have.”

“The way things are working is that the project lead […] would be the first one assigned to the project. […] solution provision […] would request a project lead and the project lead would staff the remaining team and plan the whole project.”

The pool organization also had weaknesses, which concerned staff motivation, knowledge management, and lead-time for staffing. Motivation suffered because employees did not feel to belong to a group. Colleagues changed when project teams would be recomposed. In-depth product knowledge got lost when people moved from one project to another. Since the roles required for a project were spread over multiple pools, it took more time than before to find the right people to source a project.
“Normally the lead time to find resources is 4 weeks.”

Even though the weaknesses did not lead to a reorganization, they had to be mitigated. For example, the loss of product knowledge was mitigated by active sharing of lessons learned.

“We have some specialists groups, they have very strong interactivity, so they learn from each other and then the group will communicate this lessons learned to the whole organization.”

As a result of **RUP Introduction**, all development projects followed the same process. The introduction of RUP increased flexibility of resource allocation and improved the mobility of employees that switched between projects. These advantages were anticipated.

“We wanted to standardize the way of working so we can support each other, if a new architect coming to your team and [he/she] knows how you work.”

The introduction of RUP also delivered another strength: a common terminology. It eased the communication among team members, particularly when a team member joined a new project team.

RUP also introduced a need for documentation, a weakness from the perspective of the staff. Project milestones obliged projects to deliver documentation. Employees felt they were being watched and were reluctant to adopt the change.

“They just want to work as they have always worked, for them it is just an overhead. When we introduced the milestones it was just additional work for them they did not see the benefit.”

This weakness only occurred at the beginning of RUP adoption. The temporary nature of the weakness did not warrant additional strategic changes.

**Regulatory Change Implementation** resulted in the deployment of a new software solution into the IT infrastructure of the organization. To address the uncertain contents and timing of the agreements, the system allowed switching tailored services on and off. This was the intended and anticipated purpose of the software solution.

“Unfortunately, [country X] jumped off few days before going live, but nevertheless the software is ready.”
2.5 Discussion

“We prepared the software in a way we could switch on or off a country.”

The construction of a completely new system yielded the other anticipated strengths. It allowed the decision-maker to influence how the agreements would be implemented. Also it led to a low solution integration effort and good maintainability.

“[The system was] simplified in a way that we could create a known system just for the [agreement]. So we could test it independent to all other systems.”

This solution also had drawbacks. The decision to build the system from scratch required a large investment. The decision to prioritize system maintainability reduced usability for the clients.

In this case, the decision-maker was aware of the strengths and weaknesses prior to the delivery of the software solution. However, the weaknesses were not severe enough to initiate mitigation actions or further strategic change.

This subsection showed that each change delivered the intended flexibility, other strengths, and both expected and unexpected weaknesses. Some of these weaknesses could be mitigated, while others initiated another strategic flexibility improvement project.

2.5 Discussion

The cases suggest that when building strategic flexibility, one is concerned with two levels of change: strategic and operational changes. The strategic changes pertained to the organization, the development process, and the IT infrastructure. The operational changes concerned the re-allocation of staff to projects and switching on and off services. Each strategic change was performed by the decision-maker for building flexibility at an operational level. The operational flexibility was then utilized for addressing the uncertainty caused by the problem the IT organization was confronted with.

The two levels of change can be seen as a dichotomy of designed for flexibility and designed with flexibility [Hitt et al., 1998; Schonenberg et al., 2008]. Strategic change is used for (re-)designing the flexibility of the IT organization. Operational change is thereby enabled to rapidly and easily answer the outcomes of the uncertainty the change has been designed for.
Hitt et al. (1998) suggested that for an organization to become flexible, it needs to engage in strategic actions such as to build dynamic core competences. Hitt et al. (1998) also recommended specific actions to be undertaken. Our analysis results suggest that the actions are the pragmatic result of the decision-maker’s knowledge of the change evaluation of the change options in terms of strengths and weaknesses for the organization. Furthermore, although the studied organization addressed the kinds of conversions suggested by Hitt et al. (1998), it did well by not following the recommendations. Instead of building a horizontal organization, a pool organization was built in which horizontal projects could be easily ramped up and down. This suggests that there is a broader set of options for achieving flexibility than was anticipated earlier. This also suggests that trade-offs of strengths and weaknesses of a change need to be evaluated in the situation, in which an organizational change is performed.

Earlier research had proposed that organizations adapt to change by adjusting to new inputs, by modifying its processes, and by producing outcomes (Knoll and Jarvenpaa 1994). We could confirm this inputs-conversion-outputs view of flexibility. Reorganization 1 and Reorganization 2 addressed variation of inputs, pertaining to budget instability. RUP Introduction is an example of process modification as part of achieving flexibility. Regulatory Change Implementation is an example of producing outcome variation, as the services of the IT solution could be modified depending on the services required for each agreement.

Our data also confirms the view that IT organizations adapt their constituents for building flexibility. Each of the cases involved a change of one of the constituents suggested by Tapanainen et al. (2008). Strategic flexibility involved much more than operational tactics such as those suggested by Bena- mati and Lederer (1998). While practices like IT compliance, staff motivation, customized education, and staying up-to-date provide good tactical advice for flexible IT organizations, our cases provided evidence that strategic change is at least as important to build a flexible IT organization. The present paper presents the cases in a way that gives rich insights into how decision-makers regard strategic flexibility and how the change process for building such flexibility unfolds. Future research is needed to identify a model of how strategic flexibility is built for an IT organization and to develop a framework for assessing and improving such flexibility.
2.6 Summary and Conclusions

This research has studied how decision-makers build strategic flexibility of an IT organization. Earlier research on flexibility in the IT context focused on the development process, on the software infrastructure, or on flexibility tactics only. Our industry partner, though, wanted an approach that allows improving flexibility while considering the whole IT organization and the trade-offs flexibility implies.

We used grounded theory as a research method for understanding how decision-makers interact with the IT organization when building strategic flexibility. We studied four diverse cases of flexibility improvement that our industry partner considered representative. Data was collected by interviewing decision-makers about the actual cases they were involved in. The results presented here are the results of the open and axial coding procedures suggested by grounded theory methodology.

Up to axial coding step, the research resulted in a rich description of the four cases. Each description explains how the decision-maker interacted with the IT organization for evolving the IT organization’s strategic flexibility. The results indicate that strategic flexibility of an IT organization relates to two levels of change: strategic and operational. It is the result of pragmatic decision-making based on predicted strengths and weaknesses of the strategic change.

Future research is needed to describe a fitting model of how strategic flexibility is built for an IT organization and to develop a framework for assessing and improving such flexibility.
Towards Understanding How To Build Strategic Flexibility of an IT Organization
Chapter 3

An Analysis of Change Scenarios of an IT Organization for Flexibility Building

Indira Nurdiani, Samuel A. Fricker, Jürgen Börstler

Abstract

Flexibility is important for software organizations to cope with changes demanded in the business environment. So far, flexibility has been extensively studied from a software product and software development process point of view. However, there is little work on how to build flexibility at the level of the whole software organization. Thus, there is no clear understanding of how to effectively improve the ability of an organization to respond to changes in a timely fashion and with little effort. This paper presents the results of a grounded theory study on how flexibility is built and improved in an IT organization and provides a holistic and explanatory view of how this is achieved. Implications for research and practices are also provided.
3.1 Introduction

A software organization is an entity that develops and delivers software to an external mass market, a company, or supports business operations within the same company. An IT organization is a form of software organization that operates, develops, and manages human resources, processes, and infrastructure (both hardware and software) that supports and is of importance for the business (Byrd and Turner 2000; Knoll and Jarvenpaa 1994).

Software organizations in general, and IT organizations in particular, need to be flexible and cope with the changes demanded in the business environment. Changes not only originate from the business (customers of an IT organization), they can also originate from suppliers, competitors, governing bodies, and also technological development (Byrd and Turner 2000; Knoll and Jarvenpaa 1994). In a turbulent business environment, IT organizations need to develop and improve strategies to be flexible. Without flexibility, an IT organization might lose its ability to support the business’ market position and competitive advantage (Byrd and Turner 2000).

Despite the importance of flexibility, it is not well elaborated how organizational flexibility is built. There are known prescriptive approaches in software engineering, such as Agile methodologies (Beck 2000; Schwaber and Beedle 2001), software product line engineering (Clements and Northrop 2001), and software product management (Fricker and Schumacher 2012). Agile methodology emerged as an approach that offers flexibility by welcoming changes in customer needs (Beck 2000). Software product line engineering also offers flexibility through variability management (Chen et al. 2009). The area of product planning decision also offers approaches to achieve software product flexibility (Fricker and Schumacher 2012). These approaches offer prescriptive suggestions on how to include flexibility in the software product, but not at an organizational level.

In this paper, we elaborated events that took place as an IT organization built and improved its flexibility. These events are then presented as a model that was developed through grounded theory. The data were collected through interviews with managers to identify scenarios where flexibility was built and improved in the organization. The scenarios that took place in the case company were related to change of organization structure, implementation of a new software development method, and implementation of a new legislative regulation. The elaboration of these events can be used as a stepping stone to further improve flexibility in a software organization.
The remainder of this paper is organized as follows. Section 3.2 presents the background and related work. Section 3.3 describes the research methodology. Section 3.4 presents the results and analysis. Section 3.6 presents the discussions of the result. Lastly, the paper is summarized and concluded in Section 3.7.

3.2 Background and Related Work

Organizations can be conceptualized as entities that interact with an environment through inputs, conversions, and outputs (Jones, 2012; Ilgen et al., 2005), as shown in Figure 3.1. The environment refers to the entities that provide inputs and consume the outputs generated by the organization. The environment includes customers, legislation, suppliers, competitors, and other stakeholders. Inputs refer to the things that the organization obtains from the environment to produce value such as raw materials, human resources, and information. Conversions refer to the constituents that the organization requires to transform the inputs into something of value. For an IT organization to be flexible its constituents need to be built for flexibility. These constituents are workforce, management, processes, organizational structure, and infrastructure (Tapanainen et al., 2008). Workforce refers to human resources and their skills and knowledge. Process refers to concerted activities that take place in the organization, for example a development process. Organizational structure refers to the organization hierarchy and roles and how they relate to each other. Infrastructure refers to technology platforms that are required to conceive, build, release, and evolve products and services. Outputs refer to the elements that the organization releases to its environment and include products, services, and value that are released to the environment.

Figure 3.1: Organization Model Adapted from Jones (2012) and Tapanainen et al. (2008)
Flexibility is a multidisciplinary concept that relates to the ability of an entity to change easily and effectively (De Toni and Tonchia 1998). There are different definitions of flexibility depending on the context and domain (Sethi and Sethi 1990). Across the different definitions of flexibility it can be summarized as the ability of an entity to change with: (1) effectiveness, timeliness, and satisfaction with a change (Kara et al. 2002), (2) balance of the amount of change and stability (Regev et al. 2006), (3) minimal difficulty, cost, time, effort, and risk of the change (De Toni and Tonchia 1998; Huber and McDaniel 1986; Sethi and Sethi 1990), (4) and the universality of the entity expected to cope with variations of input (Knoll and Jarvenpaa 1994).

Manufacturing recognizes that flexibility needs to be built from different aspects of the manufacturing processes, such as, material handling, production line, to marketing process, as well as management and organizational structure (Sethi and Sethi 1990). Flexibility is required to cope with internal and external uncertainties (De Toni and Tonchia 1998; Sethi and Sethi 1990) and achieve competitive advantage (Boyle 2006; Sethi and Sethi 1990). To build flexibility, manufacturing literature suggests the importance of identifying the required flexibility derived from the uncertainties, implement the flexibility building decision, and perform audit of the planned and actual achieved flexibility (Boyle 2006; De Toni and Tonchia 1998; Gerwin 1993). However, the intangible, changeable, complex nature of software and the unpredictability of development projects differs from manufacturing of physical goods (Brooks 1987). Thus, approaches in manufacturing may not be easily transferable to IT organizations.

Literature in management has two perspectives of organization flexibility. The first perspective views organization flexibility as how quickly an organization can change its structure given the changes in the environment (Huber and McDaniel 1986). This can be achieved by modifying people’s roles and responsibilities (Van der Aalst and Jablonski 2000), or having a manager with openness to change (Sharfman and Dean Jr 2003). The second perspective views organization flexibility as how the organization structure can remain stable given changes in the environment (De Haan et al. 2011). This can be achieved by employing highly skilled workers (Sethi and Sethi 1990), and adopt a flat organization structure to allow quick problem resolution (Huber and McDaniel 1986). Management literature has provided suggestions to achieve organizational flexibility, but there is no descriptive guidance on how to build flexibility.

The need for holistic view of flexibility is also expressed in IS literature. Byrd et al. (2010) proposed a framework of IS flexibility with four dimensions, people, IT, data, and process. Each dimension is associated with various factors that support its flexibility which in turn will support the organization flexibility.
Allen and Boynton (1991) discuss two main approaches to achieve IS architecture flexibility, high-road, and low-road. Each approach has its own pros and cons. Thus organizations need to tailor them to their own contexts. Current solutions are still lacking descriptive approach that allows organizations to build flexibility that suits their needs and contexts.

Software engineering has offered approaches on how to build flexibility on the level of the software product. Software product line engineering (SPLE) is intended as an approach to deal with mass-customization of software products (Clements and Northrop 2001). This is achieved through variability management where commonalities and variabilities of a set of software products’ artefacts (e.g., requirements, codes, test cases, etc..) are identified (Chen et al. 2009). Agile methodologies enable product flexibility through a process that welcomes frequent user requirement changes (Dingsøyr et al. 2012). Agile methodologies include practices that enable flexibility such as short iterations, collective code ownership, and refactoring (Abrahamsson et al. 2002). Short iterations allow changes to be integrated as soon as the need for change is discovered from customer feedback (Petersen 2011). Collective code ownership increases knowledge sharing and supports flexibility by making it easier to assign people to tasks (Abrahamsson et al. 2002). Refactoring systematizes the actual change of a software system (Fowler et al. 1999).

Today, we lack an understanding of how to build flexible software organizations. Software engineering literature addresses product flexibility but does not do so on an organizational level. A holistic organizational coverage is missing. Manufacturing and IS literature confirms this need for a holistic view (Huber and McDaniel 1986; Sethi and Sethi 1990). Making one aspect more flexible might lead to other aspects requiring more control (Leeuw and Volberda 1996; Sethi and Sethi 1990). Without a holistic view, the trade-offs of flexibility cannot be thoroughly considered (Ferdows and Meyer 1990; Harris et al. 2009).

### 3.3 Research Methodology

We studied an IT Department that serviced one of the business units of a Fortune Global 500 company operating in the financial sector. The company provides services like investment banking, asset and wealth management for private, corporate and institutional clientele worldwide. The IT Department had around 200 developers who were responsible for developing, hosting, and maintaining the software solutions required by the business unit.
An Analysis of Change Scenarios of an IT Organization for Flexibility Building

The IT Department had undergone changes in the past decade as part of their efforts in improving flexibility. With the assistance of the IT Department management representatives, we identified change scenarios that were (a) sufficiently representative and (b) covered sufficient kind of change variation that they encountered \cite{Miles and Huberman, 1994}. In Reorganization 1, the organization shifted from hierarchical to matrix organization. In Reorganization 2, the organization shifted from matrix to pool organization. Both Reorganization 1 and Reorganization 2 were intended to improve resource allocation flexibility. In RUP (Rational Unified Process) Introduction, the organization adopted RUP as the standard development methodology to improve resources flexibility to support different development projects. In Regulatory Change Implementation, the organization modified its IT system(s) due to regulatory change that the business had to comply.

Together, these four scenarios covered all five constituents of an IT organization which were mentioned by \cite{Tapanainen et al., 2008}. Reorganization 1 and Reorganization 2 influenced workforce, organization structure and management. RUP Introduction influenced the development process, and workforce’s knowledge of RUP. Regulatory Change Implementation influenced the infrastructure of the organization, as the system becomes part of their infrastructure.

3.3.1 Data Collection and Analysis

In this research, we aim to reveal a holistic view on how flexibility is built or improved in an IT organization to cope with changes. To achieve our research aim, we adopted grounded theory as suggested by \cite{Charmaz, 2014} to uncover the events that take place as an IT organization improves its organizational flexibility. The following research question was formulated to guide our study:

*RQ1. How does an IT organization build or improve flexibility?*

Grounded theory is a well-established research method used in software engineering, IT, and IS research \cite{Coleman and Connor, 2007}. Grounded theory allows us to develop concepts and theories of building flexibility based on the data, evidence and not through pre-conceived notions \cite{Charmaz, 2014}. Grounded theory allows us to remain open and critical to what flexibility building really is.

We performed data collection and data analysis iteratively. Data collection was done through semi-structured interviews and documentation reviews for data triangulation \cite{Robson, 2011}. The aim of semi-structured interviews is to get a holistic understanding of the situation in the organization, what changes
3.3 Research Methodology

had been faced, what kind of challenges were faced, what had been done to address the change, and how they were perceived. A preliminary list of open-ended questions was developed. These questions were continuously modified as we progressed with the interviews and concepts were emerging, as suggested in grounded theory (Charmaz 2014). Documentations were collected to gather standards, policies, and lessons learned that are relevant to get an overview of the flexibility improvement efforts that have taken place in the organization.

We conducted interviews with nine managers in the case organization. Each interview lasted approximately 60 minutes, and they were recorded and transcribed. Prior to performing the interviews with the managers, we conducted interviews with three of the management representatives of the IT organization to get an overview of the organization and why it needed to improve its flexibility. They also provided an overview of recent changes that the organization had to face. From these preliminary interviews, we saw how the different changes influenced different constituents of the organization, as explained by Tapanainen et al. (2008). The representatives then provided a list of managers that we could interview pertaining to these changes, whilst ensuring that we covered wide variation of scenarios based on the impact of changes to different organization constituents.

We followed the coding steps in grounded theory as prescribed in (Charmaz 2014). In initial coding, we adopted line by line coding. It was selected to allow us to see processes and distance us from the interviewees’ perspective. We went through the interview transcriptions line by line and assign one code or more as they emerge from each line. From the open coding, we identified recurring themes across interviewees and scenarios.

Then we performed axial coding to categorize the change scenarios based on the recurring themes such as, triggering uncertainties, impact of the uncertainties to the organization, actions that were taken to address uncertainties, and their respective outcomes with respect to achieved flexibility and relevant trade-offs. The concepts that emerged from the axial coding can be seen in Table 3.1.

In theoretical coding, we analyzed the relationships between the categories. We observed that the categories that emerged from the axial coding were a series of events that took place as an IT organization improved its flexibility. The outcome of the theoretical coding is shown in Figure 3.4.
3.3.2 Threats to Validity

This research is subject to validity threats common to grounded theory, such as credibility, resonance, originality, and usefulness (Charmaz, 2014).

**Credibility** is compromised when the research performed did not have data with sufficient breadth and depth. Thus, the arguments are not strongly supported by the data. This was mitigated through having multiple interviewees and reviews of the organization’s documentations. Credibility was also supported by having change scenarios that covered all five of the organization constituents.

**Resonance** refers to whether the research performed was able to capture the actual experience. We addressed this threat by informing our interviewees of our presence, the purpose of our study, and also the kind of information we would like to elicit from them. Interviewees also participated on their own will and interest in the study.

**Originality** is a risk that the outcome of the study did not offer new insights of flexibility building. The study that we conducted was built on top of existing work on flexibility from different domains. In this study, we uncovered how the existing knowledge of flexibility building is applicable for an IT organization. Furthermore, in this study we uncovered a series of events in building flexibility.

**Usefulness** is a risk to the applicability of the results beyond the studied setting. To improve the usefulness, we selected scenarios that covered the different constituents that were identified by Tapanainen et al. (2008), which should be present in other IT organizations. We believe that reorganizations, adoption of a new development method, and implementation of a new regulation are typical scenarios in other IT organizations supporting a financial institution.

3.4 Results and Analysis

We uncovered key concepts in flexibility building that were present in all four scenarios (Reorganization 1, Reorganization 2, RUP Introduction, and Regulatory Change Implementation). The scenarios, flexibility building concepts, and interviewees’ statements (marked S1 – S43) are summarized in Table 3.1. Flexibility building needed a key person or a group of people who initiated the process. We referred as *initiator*. *Situation* referred to the condition of the organization prior to the modification of the constituent. Given some internal or external changes, the organization faced variations or unclear inputs (budget, information, or regulation) referred as *uncertainty*. *Need for flexibility* was the
problems or inefficiencies that emerged in the organization due to the uncertainty. *Flexibility building* referred to the modifications made to the specific organization constituent. *Achieved flexibility* referred to the situation after the organization constituent modification and how flexibility was improved. *Trade-off* referred to the identified disadvantages of the selected flexibility building option. The following subsections describe how these concepts contribute to flexibility building of an IT organization.

### 3.4.1 Uncertainty and Need for Flexibility

The initiation of flexibility building was triggered by emergence of uncertainty that the organization could not cope. Such uncertainty led to the need for flexibility.

In **Reorganization 1**, the organization was in hierarchical structure (S2), Figure 3.2 left. A problem emerged related to the financial institution’s budget reduction that led to uncertainty in the focus of the development projects (S3). The need for flexibility emerged as the hierarchical structure of the organization hindered team members of one development section to be reallocated to another one (S4). Preferred staffs were running idle (S5).

In **Reorganization 2** the organization was in matrix structure (S6), Figure 3.3 left. A problem of inconsistent processes across development units or Service Provider Units (SPUs) emerged due to unpredicted outcome from Reorganization 1. The inconsistency led to uncertainty in the process of project issues escalation (S9). The differences in escalation process resulted in confusion and delays.

In **RUP Introduction**, initially the organization did not have a development methodology (S20). The problem was related to inconsistent development methodology that led to uncertainty on how a project would be performed (S21). This uncertainty limited the possibility to allocate staff into different projects (S22,S23), as they might require time to learn how the other project worked.
Table 3.1: Key Concepts and Relevant Interviewees’ Statement

<table>
<thead>
<tr>
<th>Reorganization 1</th>
<th>Reorganization 2</th>
<th>RUP Introduction</th>
<th>Regulatory Change Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initiator</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1: “[The CIO] was a change manager”</td>
<td>S10: “[The CIO] decided to merge them in one big pool organization.”</td>
<td>S19: “It was from internal governance.”</td>
<td>S28: “It was my [program manager] decision what can be descope.”</td>
</tr>
<tr>
<td>S2: “We had some groups, and each group was responsible for a product and the people.”</td>
<td>S6</td>
<td>S20: “We had [..] templates, but not really a methodology.”</td>
<td>S29: “There is nothing [regulated service system] on the market available.”</td>
</tr>
<tr>
<td><strong>Situation</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S3: “Years ago we were really stable in the budgeting [..]. Now, [..] sometimes they like to invest on [application A], a year later they like to invest [..] on a different topic.”</td>
<td>S9</td>
<td>S21: “There was no common way of working, [..] how they work was different from project to project</td>
<td>S30: “The bilateral agreement between [country A] and [country X], [country Y], and [country Z].”</td>
</tr>
<tr>
<td>S21: “There was no common way of working, [..] how they work was different from project to project</td>
<td>S31: “This is a completely new agreement, [..] nobody really had any experience.”</td>
<td>S31: “This is a completely new agreement, [..] nobody really had any experience.”</td>
<td>S32: “Unfortunately, [Country X] jumped off few days before going live.”</td>
</tr>
</tbody>
</table>

*Continued on next page*
<table>
<thead>
<tr>
<th>Need for flexibility</th>
<th>Flexibility building</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4: “If […] an employee […] would like to move [his] manager can stop it because they did not like to have […] moves to the other department.” S5: “[The group] SAP implementation had 180 people. After a year you do not really need the [rest] 60 people, but these 60 people could not be relocated, and they did not have anything to do.”</td>
<td>S6: “We had a matrix with […] four pools: project management, business analysis, development, and integration. Across we had the key accounts, and we had the head of this unit called Service Provider Unit.” S11: “We are organized in the delivery unit in five pools: […] project leads, business analysts, and we have [two] development pools.” S12: “So if we have some escalations between these two organizations [pools], then you have to go to [the CIO].”</td>
</tr>
<tr>
<td>S9</td>
<td>S10: “…” S11: “…”</td>
</tr>
<tr>
<td>S22: “There was a problem that each project work in their own ways.” S23: “we wanna standardized the way of working so we can support each other.”</td>
<td>S24: “[…] all projects are running RUP.”</td>
</tr>
<tr>
<td>S33: “[These] regulatory things you have to deliver, there is no choice.” S34: “[…] to become a compliant financial institution of 2012.”</td>
<td>S35: “The software is ready […]” S36: “We […] did it on a make [build] approach in our infrastructure.” S37: “We […] had chosen the option which has the fewest impact on all of the system.”</td>
</tr>
<tr>
<td>Reorganization 1</td>
<td>Reorganization 2</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------</td>
</tr>
<tr>
<td>S7: “People had the possibility to move around, to grow, to have a good future perspective.”</td>
<td>S13: “The pool setting offers flexibility to cope with demand from the business. You can focus people on a solution whenever required.”</td>
</tr>
<tr>
<td>S8: “We [the staff] had more power to switch [...] to other solutions.”</td>
<td>S14: “Whatever the business decides, you can allocate people to that particular solution.”</td>
</tr>
<tr>
<td>S15: “Here [pool organization], the responsibilities are clear. You have one organization responsible for product, one organization responsible for deliveries [...]”</td>
<td>S16: “People miss having their home and may be unmotivated.”</td>
</tr>
<tr>
<td>S25: “Now we all have the same terminology [...]”</td>
<td>S16: “People miss having their home and may be unmotivated.”</td>
</tr>
<tr>
<td>S9: “He [a manager] could decide himself on an escalation view. The problem was all these three SPUs [Service Provider Delivery Units] had small differences in the process.”</td>
<td>S17: “Knowledge management is much more complicated [...] Something gets lost when [people] go away from a solution domain.”</td>
</tr>
<tr>
<td>S10: “The pool setting offers flexibility to cope with demand from the business. You can focus people on a solution whenever required.”</td>
<td>S18: “With the pool I [project manager] need time to find people.”</td>
</tr>
</tbody>
</table>
In Regulatory Change Implementation, new bilateral agreements between financial institution’s home country were planned with other countries (S30). There were no IT systems that supported such bilateral agreements (S29). The contents of these agreements varied across countries (S30), and nobody knew the exact rules that were going to be agreed (S31). Furthermore, uncertainty escalated as there was a possibility that a country could withdraw from the agreement (S32). This uncertainty led to a difficulty in determining what services need to be provided for every bilateral agreement, and if they need to be provided at all. This led to the possibility of performing unnecessary work.

From the four scenarios, we learned that an uncertainty could originate from outside the organization, like changes in the business environment, as shown in Reorganization 1 and Regulatory Change Implementation scenarios. An uncertainty could also originate from within the organization and not necessarily caused by changes in the environment, as shown in Reorganization 2 and RUP Introduction. The uncertainty limited the software development organization’s ability to operate efficiently or lead to additional or unnecessary work. These limitations then led to the need for flexibility.

3.5 Build Flexibility

Once the need for flexibility was identified, an IT organization built flexibility through modifying one or more of its constituents. The role of a decision-maker was crucial for initiating the required modifications.

In Reorganization 1, there was a need for more flexible resource allocation to high priority projects (S4, S5). This rigidity of resource allocation led the Chief Information Officer (CIO) to initiate a reorganization (S1). The option was to shift from hierarchical organization to a matrix organization structure, as shown in Figure 3.2, where staffs were grouped according to their roles (S6) and not by product development. The new organization structure should accommodate team members’ mobility to be easily allocated into different projects.

In Reorganization 2, there was a need to overcome the inconsistency of escalation process across SPU (Q9). The CIO, again, opted to initiate a reorganization (S10). The inconsistency needed to be addressed by having a more centralized escalation process. The option was to shift from matrix to pool organization structure, as shown in Figure 3.3. In the pool organization, the SPU were merged into different pools, and all pools only reported
to the CIO (S11). The centralized reporting should eliminate variations in issue escalation process.

In **RUP Introduction**, there was a need for flexible resource allocation. It was hindered due to variation of development process across projects. The internal governance group decided that a common way of working was required to improve flexibility (S19). The option to address the inflexible resource allocation was by adopting RUP as a standard organization development methodology (S24). A common way of working and terminology should ease communication between staff and transitions between projects.

In **Regulatory Change Implementation**, there was a need for flexibility in IT service provision to support the new bilateral agreements (S30,S32), as well as to remain compliance with the regulation (S31). For this scenario, we uncovered different options on how to build flexibility of the IT service.

First option was whether to **build** or **buy** the regulated service system. The program manager (S28) of the implementation program decided on **build** approach (S36). Prior to executing the decision, the program manager had foreseen the pros and cons of building the IT system in-house. The **build** approach would require less integration effort (S39), and but it would be more expensive (S42).

There was also an option to build the system with high maintainability or high usability. Prior to executing the decision, the project manager had also seen the pros and cons of each option. To have better maintainability, the regulated service system would be decoupled from the rest of the IT system (S37). However this option would sacrifice usability for the clients (S41). It was decided to go with the option that prioritized maintainability over usability.

There were different decision-makers involved in deciding how the constituents were to be modified to build flexibility. They could be from upper level management or middle level management. Uncertainty that involved the whole organization, as in Reorganization 1, Reorganization 2 was handled at executive level, the CIO. The RUP Introduction was initiated by the internal governance responsible for development processes. Regulatory Change Implementation was handled by the program manager of the development program. Furthermore, the need for flexibility influenced which constituents to be modified and how they were modified. Unlike the other scenarios, in Regulatory Change Implementation, the decision-maker was able to foresee the outcome of the different options in improving flexibility.
3.5 Build Flexibility

3.5.1 Achieved Flexibility and Trade-Offs

After the constituents were modified, the organization achieved the flexibility that they required. However, there were trade-offs associated with the achieved flexibility.

In Reorganization 1, the matrix organization structure delivered flexibility in resource allocation. The team members were able to move different solution development or projects (S7, S8). The freedom for team members to move across projects also allowed them to improve their career perspectives in the organization (S7).

However, this new flexibility of resource allocation had a trade-off at the management level. The separation of the organization into different SPUs led to inconsistent issue escalation processes across the SPUs (S9). The variations of issue escalation process led to delays and confusions, and led the CIO to initiate Reorganization 2 (S10). This suggests that the decision-maker could not anticipate the trade-off of resource allocation flexibility from the reorganization to matrix organization.

In Reorganization 2, the pool organization solved the problem of project issue escalation, as all pool heads only reported project issues to the CIO (S12). The pool organization also strengthened the flexibility of resource allocation. As team members are no longer permanently designated to a particular product, they could be assigned to projects or initiatives which the business really needed (S13, S14). The pool organization also established clearer responsibility for the team members and managers (S15).

The pool structure allowed team members to be allocated into different projects which eliminated a sense of belonging to a group (S16). There was loss of product knowledge as people were not designated to a particular product anymore and moved between projects (S17). More time was required to find the right people to source projects (S18) because people were spread out in different pools. Decision-makers initiated a mitigation approach to mitigate loss of knowledge problem through sharing of lessons learned. As expressed by one of our interviewees: “[We have some specialists groups, [...] then the group will communicate this lessons learned to the whole organization.]”

RUP Introduction improved communication and flexibility of the team members to be assigned in different development projects. However, the introduction of RUP practices like project milestones and mandatory documentations, made team members felt like they were being watched (S27). This trade-off only occurred in the beginning of RUP adoption. Thus no mitigation actions were initiated.
The result of the **Regulatory Change Implementation** was the deployment of a regulated service system into the infrastructure of the organization (S35). The decision to build the system in-house (S36) minimized the integration effort (S39), and allowed the project team to steer the development process (S40). Furthermore, the decision allowed the design of the infrastructure to be flexible in handling uncertainties of the bilateral agreement (S36). However, the decision to build the system was more expensive (S42) and required a large investment (S43). The decision to prioritize maintainability resulted in a regulated service system which was decoupled from the other IT systems (S37). However the decision compromised the system’s usability (S41).

The selected option to modify a constituent allowed the organization to achieve the flexibility required to address the uncertainty. However, each option to improve flexibility could have other negative influence on other constituents. In the case of Reorganization 1, Reorganization 2, and RUP Introduction, it was difficult to properly foresee the outcome of the flexibility building option. However, in the case of Regulatory Change Implementation the trade-offs were consciously made by the decision maker.
3.5 Build Flexibility

3.5.2 Flexibility Building Model: Revisit RQ1

Using interview data from nine IT managers and grounded theory our research explored how an IT organization improved or built flexibility through strategic change of one of the IT organization constituents. Our research uncovered three states that the organization went through as part of the strategic change: pre-change, transitional state, and post-change. Pre-change was the state prior to the strategic change. Transitional state was the state when the constituent modification was taking place. Post-change was the state after the strategic change was in place. If another strategic change was required, the post-change state would be the pre-change state.

Through these transitional states in the organization, our data show that there were different events that took place in each state, emerging uncertainty, need for flexibility, build flexibility, and achieved flexibility and trade-offs. Across four distinct change scenarios, we observed that the uncertainty emerged in the pre-change state of the organization. The uncertainty triggered the need for flexibility. During transitional state, flexibility was built by modifying one or more of the organization constituents to address the uncertainty. In the post-change state, after the modification was implemented, flexibility was achieved yet with associated trade-offs. Furthermore, as demonstrated in Reorganization 2, flexibility building process could be cyclical, if the associated flexibility trade-off introduces new uncertainty, another round of flexibility building could be initiated.

The result suggests a model that elaborates the different events that took place in the transitional states as shown in Figure 3.4. The model does not detail the decision making points that managers need to take to build flexibility. However, it identifies and describes the events that took place as an IT organization built flexibility of its different constituents. The model offers a unified view of flexibility building for an IT organization, from ad hoc and latent experience to a well-constructed and explicit point of view.

Furthermore, we discovered that a flexible organization was not an organization that could quickly make changes to its structure every time it faced an uncertainty. Yet, a flexible organization was an organization that was able to restructure its organization constituents in such a way that they allowed the organization to withstand a specific type of uncertainty.
3.6 Discussion

This research has discovered the key events that took place as an IT organization built or improved its flexibility. Our data show that building organizational flexibility was done through modifying one or more of the IT organization’s constituents to address the uncertainty that exposed the organization’s need for flexibility. Through the constituents modification, the organization was able to achieve flexibility and this aided efficient response to changes and uncertainty, and with little additional effort. Our study has provided a holistic explanatory view of the processes in improving organizational flexibility.

Manufacturing literature suggests that building flexibility entails identifying uncertainty, implementing the appropriate decision, and monitoring the achieved flexibility (Boyle, 2006; Gerwin, 1993). Similar view is also shared in organizational change management literature (Orlikowski, 1993). The results of our analysis show similar pattern with existing literature in flexibility and change management. However, our study suggests that a flexible organization is not the one that is capable of constantly change the structure of its organization constituents on the face of uncertainties. Organizational change is one of the tools to build organizational flexibility. Building organization flexibility is about purposefully restructure of the organization constituents so the organization is able to cope with uncertainties. Further research could be allocated on looking at known approaches in organizational change management to complement the current model that was developed in this study.

In software reusability management, reuse can be used from two perspectives “developed for reuse” and “developed with reuse” (Börstler, 1992). Developed for reuse is when software artefacts like documentations and codes are predefined and purposefully developed for later reuse usually by implementing well known design principles like encapsulation, coupling, and cohesion. Developed...
with reuse deals with the use of the previously predefined artefacts. The data in our study shows that to achieve flexibility organization constituents could also be “developed for flexibility” and “developed with flexibility”. In Reorganization 1, Reorganization 2, and RUP Introduction, the organization structure and development process were developed for flexible resource allocation to different projects, whilst each initiated project was developed with flexibility of the organization structure and development process. In Regulatory Change Implementation the infrastructure was developed for flexibility of providing financial services for different bilateral agreements, whilst each supported bilateral agreement was developed with flexibility of the infrastructure. This shows that software engineering has mechanisms for improving flexibility that are applicable at the organizational level, but currently are used at product level. Future research can be directed towards identifying mechanisms and design principles that could be transferred from software artefacts to organizational level to build organizational flexibility.

Real options thinking is often suggested for managers to cope with uncertainty (Fichman et al., 2005). Uncertainty can be addressed by choosing an option. One of these options is defer where decisions are delayed for a certain period of time. Our study shows that modifications of organization constituents allowed managers to exercise the defer option. In Reorganization 1 and Reorganization 2, the decision to change the organization structure, allowed them to defer the decisions which team members to be allocated to a certain development effort until a project was funded. In the Regulatory Change Implementation scenario, the decision to design the infrastructure to cope with variation of bilateral agreements, allowed them to defer decisions on which services to activate or deactivate. Future research on real options thinking can be pursued to get insights on how to develop criteria to help managers to select the best option in improving organizational flexibility.

Previous studies suggest that flexibility is a trade-off with values, such as control (Harris et al., 2009), performance (Ferdows and Meyer, 1990), and cost (Van Biesbroeck, 2007). Our study shows that flexibility is not necessarily an immediate trade-off with other values. Scenarios clearly show that flexibility trade-offs were the indirect effects of the selected option for the strategic change. For example, in Regulatory Change implementation, the option to build the system in-house was more costly, nevertheless it was preferred over purchasing a system that required large integration effort. In Reorganization 2, ease of team member allocation to projects was preferred over quick development project completion. Our study shows that different options could lead to different trade-offs, however these trade-offs were acceptable given the achieved
flexibility. Further research needs to be done to help managers predict and assess the trade-offs of the different options in improving organizational flexibility.

One of the practices in Agile methodologies is iterative development (Aframian et al. 2002). Previous studies have shown that iterative development allows a software organization to reduce rework, improve estimations, and reduce lead time (Petersen and Wohlin 2010). Our study also shows that flexibility improvement is done iteratively. This was clearly demonstrated in Reorganization 1 and Reorganization 2. In Reorganization 1, the organization structure shifted from hierarchical to matrix organization structure to deal with rigid resource allocation. Despite the achieved flexibility of resource allocation, the new organization structure introduced unanticipated inconsistencies into the issue escalation process. The inconsistency initiated a new cycle of flexibility improvement through Reorganization 2. Our study shows that the iterative nature of achieving organizational flexibility is congruent with the iterations in Agile methodologies. Furthermore, our study shows that reflecting the outcome of a strategic change was an important process for a software organization in achieving flexibility. Future research could be directed to seek other key ideas from Agile methodologies for improving organizational flexibility.

Current approaches in software engineering, like Agile methodologies (Beck 2000) and SPLE suggest (Clements and Northrop 2001) that flexibility is built through the products. Our study shows that organizational flexibility could be built into different organization constituents, not only built through the products. Reorganization 1, Reorganization 2, and RUP Introduction show that modifications in organization structure, management, and development process could have influence on the flexibility of project resource allocation which in turn improved the IT organization’s flexibility to initiate projects that suited the business needs. Regulation Change Implementation shows that the infrastructure design improved the IT organization’s flexibility in coping with what the business requires in terms of providing services for different bilateral agreements. This shows that there are wider range of possibilities how an IT organization can be improve its flexibility than previously assumed and further confirmed the need for a holistic approach to flexibility (De Michelis et al. 1998) not only through products.

3.6.1 Implications for Practice and Research

Our research suggests a model that provides a holistic view of how an IT organization improves flexibility. This model was uncovered through analyzing the experiences of IT managers in the case organization and has not yet been
operationalized. However, the findings of our study yield implications both for researchers and practitioners.

The key events that our study uncovered can be used by practitioners as a foundation for process improvement. For example, flexibility could be added as a process area to existing frameworks such as CMMI. Such a process area would include the identification of uncertainty, the prioritization of which kind of flexibility should be built, the selection of the appropriate flexibility improvement approach, and the analysis of the obtained results. Also, such a process should specify how experiences from flexibility improvement would be collected and shared to support future decisions.

Flexibility improvement processes would need to be supported by suitable tools that enable process institutionalization. For example, the uncertainties that need to be addressed by the organization could be captured in a backlog. Backlogs are an established and commonly used tool in Scrum to enable visibility and prioritization of features that a product needs to deliver (Schwaber and Beedle 2001). A backlog of uncertainties would enable sharing of flexibility improvement needs, prioritization, and progress tracking, thus allow the organization to manage flexibility proactively and systematically.

Also, knowledge and experience sharing may improve the organization’s ability to build flexibility. Knowledge sharing is an essential part of organizational learning that has led to performance improvement in many companies (Garvin 2000). Here, knowledge sharing should enable the identification of flexibility-building options and improve the understanding of the options’ impacts on flexibility and trade-offs. Such knowledge could be expressed as stories or patterns (Hagge and Lappe 2005). Their use would reduce the risk of unanticipated negative surprises and minimize the need to re-iterate.

Our study provides a stepping stone for future research in understanding how organizational flexibility is built. To support organizations in improving flexibility, a systematic mapping of flexibility-building options with their associated benefits and trade-offs is needed. Experience transfer and evaluation from domains other than software engineering, like management and manufacturing, can be valuable inputs to build such a mapping. Management literature provides suggestions to build flexibility, for example to adopt a flat organization structure (Huber and McDaniel 1986) and having managers that are open to change (Sharfman and Dean Jr 2003). Manufacturing literature describes mechanisms and measures to build flexibility through suitable material handling and production process design (Sethi and Sethi 1990). Concerns that are unrelated to software need to be replaced, however. Nevertheless, understanding the practices of other established domains can prevent “reinventing the wheels”.
An Analysis of Change Scenarios of an IT Organization for Flexibility Building

To formulate a model that maps flexibility-building options and their associated benefits and trade-offs, we can also pursue experience transfer from existing product-level approaches in software engineering to organizational level. Our study shows that organization constituents could be “developed for flexibility” and “developed with flexibility”. Also, our study shows that flexibility improvement was done iteratively. These are known concepts from software reusability management (Böorstler, 1992) and Agile methodologies (Abrahamsson et al., 2002) with an abundance of empirical studies. Using existing empirical studies from such established approaches can allow researchers to build a map of flexibility-building options and their associated advantages and disadvantages.

3.7 Summary and Conclusion

We adopted grounded theory as a research approach for understanding how an IT organization improved flexibility. We studied four scenarios of flexibility improvement that our industry partner considered representative for an IT organization that had evolved over the past decade. Data was collected by interviewing the respective decision-maker about the strategic flexibility-related change he was involved in. The scenarios suggest that decision-makers have a central role in managing IT organization flexibility. They are the key authoritative figures that allow which changes to be made in the pursue of flexibility. Furthermore, the selected course of actions that the decision-makers take can yield to different flexibility trade-offs.

The research resulted in a model that describes events that take place as an IT organization improves its flexibility. The model suggests that flexibility is built to answer uncertainty in the organization that can cause delays, unnecessary work, and inefficient use of resources. Furthermore, flexibility can be built into different constituents of the organization, and not just through the product. Future research should be directing towards improving the understanding of how to select the course of action that is best suited to address uncertainty, and better predict flexibility trade-offs.
Chapter 4

The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Indira Nurdiani, Jürgen Börstler, Samuel A. Fricker

Abstract

The interest in Agile and Lean software development is growing. The growing interest is reflected by the increasing number of secondary studies performed to identify different factors, like benefits and limitations, of Agile and Lean. The aim of this tertiary study is to consolidate empirically evaluated Agile and Lean practices and their respective impacts. We performed an automated search on five databases. We identified 13 secondary studies that provide discussion on the impacts of Agile and Lean practices. We then adapted meta-study to synthesize the findings. From 13 secondary studies with identified 13 Agile and Lean practices and their impacts on quality, schedule, budget, risk, scope, and communication. The impact of Test Driven Development (TDD) and quality is the most studied. Most of the secondary studies we examined did not provide clear synthesis of the evidence. Furthermore, we found that secondary studies
with unclear synthesis method could yield different results as those with clear synthesis method. There is a need for more attention to rigour and traceability of review process for future secondary studies in Agile and Lean.

4.1 Introduction

The software industry is highly dynamic and competitive. To strive in the competition software organizations need to cope with frequent changes in its environment, i.e., customer needs, regulation, technology development) (Byrd and Turner, 2000). At the same time software organizations need to deliver the product in shorter lead time, better quality, and lower budget. Flexible approaches such as Agile and Lean software development have emerged to provide solution to this situation (Dyba and Dingsøyr, 2008; Leffingwell, 2007).

Agile software development is a group of software development methodologies, e.g., Extreme Programming (XP), Scrum, Crystal, etc., that focus on delivering working software product in small iterations, adaptive towards requirement changes, and collaborate closely with the customers (Abrahamsson et al., 2002). Lean software development (Lean) is a set of principles and tools derived from Lean manufacturing that focus on removing waste, delaying decisions as late as possible, and building quality into the product as early as possible (Poppendieck and Poppendieck, 2003).

The growth of interest in Agile and Lean is reflected by the large number of research papers published between 2001 and 2010 (Dingsøyr et al., 2012). However, the strength of evidence of empirical studies of Agile methods is still very low to support any benefit of implementing Agile software development (Dyba and Dingsøyr, 2008). Almost a decade after the Systematic Literature Review (SLR) that was conducted by Dyba and Dingsøyr (2008), the benefits and limitation of implementing Agile and Lean practices remain in question.

With the introduction of Evidence Based Software Engineering (Kitchenham et al., 2004), the number of secondary studies in Agile and Lean has increased over the years. Many of them reported on the benefits and limitations of Agile and Lean methods and practices. With the increasing number of secondary studies in Agile and Lean, it becomes important to get an overview of what has been done. Furthermore, it is important to examine if the strengths of evidence have improved. Therefore, researchers in Agile and Lean can build on what have been conducted in the secondary studies, or to seek new venues of research. Meanwhile, practitioners can have a better overview of which Agile and Lean
practices have been empirically evaluated and understand their benefits and limitations.

The purpose of this tertiary study is to consolidate empirically evaluated Agile and Lean practices and their respective impacts. First, we provided an overview of what has been studied in Agile and Lean secondary studies. Second, we examined Agile and Lean practices with their respective impacts and provided the extent of empirical support as reported in the secondary studies.

The remainder of the paper is structured as follows: Section 4.2 presents related work. Section 4.3 presents the review steps. Section 4.4 provides the overview of Agile and Lean secondary studies. Section 4.5 presents the detailed results and analysis. Section 4.6 presents the discussions of the results. Section 4.7 presents the summary and conclusions of the review.

4.2 Related Work

With the introduction of Evidence-Based Software Engineering (EBSE) by [Kitchenham et al. (2004)], a more systematic approach of performing secondary studies has become more prevalent. The aim of an SLR is “to identify, evaluate and summarise the findings of all relevant individual studies, thereby making the available evidence more accessible to decision-makers” (Centre for Reviews and Dissemination, 2009). The result of combining multiple relevant individual studies is expected to provide more reliable and precise estimate of the effectiveness of an intervention.

When there are many secondary studies in the area such as in software engineering, it is possible to perform a tertiary study [Kitchenham and Charters (2007)]. Kitchenham et al. (2009) performed a tertiary study aimed at assessing impacts of SLRs in software engineering. A number of issues pertaining to time spent in performing the review and difficulty to assess primary studies quality were identified. Kitchenham et al. (2010) then published another tertiary study to provide a catalogue of SLRs to researchers and practitioners. Kitchenham et al. (2010) also found that the number of SLRs are increasing and the quality is improving.

MacDonell et al. (2010) performed a tertiary study to evaluate reliability of of two SLRs with the same topic of comparing effort prediction models for inter- and intra-company. The tertiary study shows that the two SLRs have different review design and execution, yet the outcomes were rather consistent. The tertiary study shows that systematic reviews are reasonably robust and reliable research method.
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Wohlin et al. (2013) performed a tertiary study to evaluate reliability of mapping studies. They evaluated two mapping studies with the topic of testing of product families. Despite the same topic and classification schemes the two mapping studies had different outcomes. From the tertiary study Wohlin et al. (2013) hypothesized that the findings from the mapping studies are valid, however not yet producing reliable results. A contradictory result from the tertiary study by MacDonell et al. (2010). Secondary studies reliability could be attributed by to the scope of the study and the expertise of the reviewers. Therefore we need to be aware that the results of a secondary study may not be reproducible (Wohlin et al., 2013).

A tertiary study in Global Software Development (GSD) pertaining to risk and risk mitigation strategies was conducted by Verner et al. (2014). Their study indicates that most SLRs in GSD are actually literature summaries or mapping studies for researchers. Most SLRs did not provide evidence for the risk that they identified. Also most SLRs did not provide any advice for practitioners.

Hanssen et al. (2011) performed a tertiary study in the applicability of Agile in Global Software Engineering (GSD). The review shows that there is a growing trend of implementing Agile in GSD. Certain Agile practices, such as continuous integration, pair programming, retrospective, and daily stand up are possible to be done in GSD contexts. Hanssen et al. (2011) indicate that most Agile GSD implementations are successful based on their analysis of two secondary studies. However, it was not made explicit what was the outcome of successful implementations.

Current research in Agile software development is still inconclusive with respect to the impacts of Agile and Lean practices compared to using traditional methods (Dybå and Dingsøyr, 2008). Although there exist one tertiary study pertaining to Agile (Hanssen et al., 2011), the tertiary study only indicated that there are successful implementations, the actual benefits and their supporting evidence are not yet made available. The impacts of Agile and Lean practices, and their supporting evidence, are not unified. The lack of unified view of the impacts of Agile and Lean practices and their supporting evidence would raise question of the applicability and trade-off of implementing Agile and Lean practices.

4.3 Review Method

The aim of this tertiary study is to provide a consolidated view of evidence pertaining to the impacts of Agile and Lean practices. This tertiary first examines...
the overlap of primary studies included in the secondary studies. Then Agile and Lean practices were collected together with their impacts. The number of empirical studies that support the practice and impact association was also examined. The consolidated view will provide researchers and practitioners more information whether the impacts of Agile and Lean practices are substantiated with strong empirical support or not. Therefore benefits and limitations of Agile and Lean practice can be better understood.

The following research questions were developed to help us achieve our aim:

**RQ1:** What is the population of primary studies that the secondary studies draw upon?

- **RQ1.1:** What is the overlap of the populations of primary studies?
- **RQ1.2:** What are reasons for overlap and differences?

**RQ2:** What is the scope of the secondary studies in evaluating Agile and Lean?

- **RQ2.1:** What are the Agile and Lean practices that were studied?
- **RQ2.2:** What are the impacts of Agile and Lean practices that were evaluated?

**RQ3:** What is known about the impacts of Agile and Lean practices?

- **RQ3.1:** What is the level of empirical evidence of the evaluations presented in the secondary studies?
- **RQ3.2:** What is the level of agreement between the secondary studies?

### 4.3.1 Search Process

In conducting this tertiary review we adopted the systematic review guideline by Kitchenham and Charters (2007). We started the search by formulating the search string that was used to perform automated search.

In our search we formulated search strings that allowed us to retrieve secondary studies in Agile development. The search string was composed of three substrings: Agile methods, Lean methods, and secondary study. Table 4.1 shows the substrings and the keywords.

We performed a pilot search on Inspec and Compendex to check if adding or removing a keyword would yield to more or less relevant papers. We also consulted an expert in Agile and Lean methods, external to the review team, to ensure we have included the necessary keywords in Substring 1 and 2 as well as adopting the keywords used in Dyba and Dingsøyr (2008) for Substring 1 and

---

1The expert was one of 18 most active researchers who published in Agile software development as reported in Chuang et al. (2014)
Table 4.1: Keywords in Search String

<table>
<thead>
<tr>
<th>Substring</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>Substring 1</td>
<td>Agile OR scrum OR crystal OR XP OR “extreme programming” OR DSDM OR “dynamic systems development method” OR FDD OR “feature driven development”</td>
</tr>
<tr>
<td>Substring 2</td>
<td>lean* OR kanban OR kaizen OR “value stream mapping” OR VSM OR “continuous improvement” OR “cross* functional” OR “concurrent engineering” OR “integrated product development”</td>
</tr>
<tr>
<td>Substring 3</td>
<td>“review of studies” OR “structured review” OR “systematic review” OR “literature review” OR “literature analysis” OR “in-depth survey” OR “literature survey” OR “meta-analysis” OR “Past studies” OR “systematic map*” OR “secondary study”</td>
</tr>
</tbody>
</table>

Pernstal et al. (2013) for Substring 2. For Substring 3 we referred to Kitchenham et al. (2010) to ensure we included all the necessary keywords and synonyms to secondary study. We performed search on abstract, title, and keyword with the following search string: (Substring 1 OR Substring 2) AND Substring 3.

The search string was then implemented in five electronic databases: (1) Compendex & Scopus, Inspec, IEEE Explore, ACM Digital library, and ISI Web of knowledge. The search was done in December 2014 to identify peer-reviewed secondary studies published since 1990. The search time span was limited between 1990 and 2014 to minimize irrelevant papers from manufacturing, where Agile and Lean concepts were derived from. Also, Agile and Lean software development started as an early 21st century phenomena with the introduction of Agile Manifesto (Beck et al., 2001).

Table 4.2: Database Search

<table>
<thead>
<tr>
<th>Database</th>
<th>Years Searched</th>
<th>Total Hits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compendex &amp; Scopus</td>
<td>1990 – 2014</td>
<td>355</td>
</tr>
<tr>
<td>Inspec</td>
<td>1990 – 2014</td>
<td>414</td>
</tr>
<tr>
<td>IEEE Explore</td>
<td>1990 – 2014</td>
<td>130</td>
</tr>
<tr>
<td>ACM</td>
<td>1990 – 2014</td>
<td>20</td>
</tr>
<tr>
<td>ISI Web of Knowledge</td>
<td>1990 – 2014</td>
<td>191</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>1110</strong></td>
</tr>
</tbody>
</table>
4.3.2 Study Selection

To perform study selected we formulated inclusion/exclusion criteria that we adapted from [Kitchenham et al. (2009)]. The purpose of the inclusion criteria was to ensure that we included all peer-reviewed secondary studies in Agile and Lean. Articles were included if they were:

- **IC1.** Systematic reviews, systematic mappings, meta-analysis or literature reviews in Agile or Lean (Type 1) OR secondary studies in Agile and Lean that were part of empirical studies (Type 2).
- **IC2.** Secondary studies (either Type 1 or Type 2) with defined literature review process.
- **IC3.** Secondary studies that were peer reviewed.
- **IC4.** Secondary studies that have a list of included primary studies.

The purpose of the exclusion criteria was to ensure that we only included unique secondary studies with traceable review process. Articles were excluded if they were:

- **EC1.** Type 2 papers with SLR results that were not distinguishable from the empirical study results.
- **EC2.** Posters and short papers less than four pages.
- **EC3.** Redundant reports of the same secondary study. The same author(s) often published the results of their secondary studies in multiple publication venues, with different titles.

From the automated search the papers were screened. We started by screening through the titles and abstracts, followed by full texts of the papers. The study selection process is represented in Figure 4.1.

The first step was title and abstract screening. In title and abstract screening, we checked for duplicates, as well as non-article papers, e.g., editorials, prefaces, or article summaries. After removing duplicates and non-article papers, we were left with 729 papers. We only excluded papers that were clearly evident from the title not related to software engineering, not secondary studies, or secondary studies not in Agile and Lean. In this stage we were being more inclusive and included all secondary studies in Agile and Lean, and we were left with 122 papers. At this stage only the first author was involved.

The second step was full text screening. We screened the papers against the inclusion/exclusion criteria. At this stage, we also checked the papers for redundancies. We found SLRs published by the same authors to be published in multiple publications venues. In such cases we included the most updated or complete results. For instance we excluded (Jalali and Wohlin 2010) from our review and included (Jalali and Wohlin 2012) instead. We found the journal
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Figure 4.1: Study Selection Process

version [Jalali and Wohlin, 2012] to be an extension of work from the mapping study [Jalali and Wohlin, 2010] and covers more aspects than the mapping study itself. We also found more redundant publications based on the same SLR, they are [Dyba et al., 2007], [Dybå and Dingsøyr, 2008], [Dyba and Dingsoyr, 2009], and [Dybå and Dingsøyr, 2008]. We only included the journal version [Dybå and Dingsøyr, 2008], because we found it to be the most comprehensive and suits the aim of our tertiary study. At this stage all the first author reviewed all papers, meanwhile second and third authors reviewed two thirds of the papers. To ensure that two thirds of the papers were reviewed by at least two reviewers, the second and third author reviewed different sets of papers. We also calculated the inter-rater agreements among the reviewers. First and second author had a kappa coefficient 0.802. First and third author had a kappa coefficient of 0.323. From the first phase of full text screening, there were 41 papers left. We analyzed 41 secondary studies to provide overview of Agile and Lean secondary studies, as reported in Section 4.4.
4.3 Review Method

4.3.3 Data Extraction

From 41 secondary studies that passed inclusion/exclusion criteria, only 13 secondary studies provided discussions on the impacts of Agile and Lean practices. From the selected secondary studies, we extracted the following information:

- Meta-data: title and publication year
- Review method. The adopted literature review method and the reference to the methods as mentioned in the secondary study.
- Aim of study. We collected the stated aim of the secondary study.
- Topic of the review. We collected the main focus of the study, whether it is Agile or Lean in general, or specific Agile or Lean practices, or a combination of Agile or Lean with other topic in SE.
- Practices and their associated impacts. We collected the impacts associated to a particular Agile and Lean practice. To ensure uniformity of interpreting Agile and Lean practices, we referred to the list of practices discussed in [Petersen 2011].
- Empirical support. We collected the number of primary studies that were cited with respect to impacts and practice association. W also collected the research methodologies that were used in the primary studies
- List of primary studies. We collected the primary studies that were included from each secondary study.

The first author performed the data extraction on 13 secondary studies. A post-hoc validation was performed by the second and third authors to validate the inclusion of secondary studies after data extraction, as well as the extracted data. Again, the second and third authors reviewed different sets of papers. In cases where the first author included a secondary study which was excluded by the second or third author, a discussion was made to resolve the disagreements. During the discussions, it was decided to include a secondary study which was included by at least one of the reviewers. To validate the extracted data, the second and third authors also performed independent data extraction. The data extracted by the first author were then checked and compared with the data extracted by the second and third authors. In case where the second or third author extracted different information from the first author, discussions were made on the differences. Then another round of data extraction was performed on particular secondary studies where data differences were found.
4.3.4 Quality Assessment

Quality assessment is critical because the effectiveness of an intervention might be masked by secondary studies that were not properly conducted (Centre for Reviews and Dissemination 2009). For the quality assessment we adopted appraisal criteria from CRD’s guide for reviews (Centre for Reviews and Dissemination 2009) and adjusted them to the context of this study. The quality assessment criteria is summarized in Table 4.3.

Table 4.3: Quality Assessment Criteria adopted from (Centre for Reviews and Dissemination 2009)

<table>
<thead>
<tr>
<th>ID</th>
<th>Quality Assessment Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Was the review question clearly defined in terms of population, interventions, comparators, outcomes and study designs (PICOS)?</td>
</tr>
<tr>
<td>C2</td>
<td>Was the search strategy adequate and appropriate?</td>
</tr>
<tr>
<td>C3</td>
<td>Are the review’s inclusion and exclusion criteria described and appropriate?</td>
</tr>
<tr>
<td>C4</td>
<td>Were appropriate criteria used to assess the quality of the primary studies?</td>
</tr>
<tr>
<td>C5</td>
<td>Were preventive steps taken to minimize bias and errors?</td>
</tr>
<tr>
<td>C6</td>
<td>Were adequate details presented for each of the primary studies?</td>
</tr>
<tr>
<td>C7</td>
<td>Were the evidences synthesized and aggregated?</td>
</tr>
<tr>
<td>C7a</td>
<td>Were the evidences actually synthesized and aggregated, or merely summarized?</td>
</tr>
<tr>
<td>C7b</td>
<td>Were the strengths of evidence of individual studies taken into account in the synthesis?</td>
</tr>
<tr>
<td>C8</td>
<td>Do the authors’ conclusions accurately reflect the evidence that was reviewed?</td>
</tr>
</tbody>
</table>

Each quality criteria was scored with the following procedure: Y=1, P=0.5, and N=0. Details of the quality criteria scoring is provided in Appendix A. Quality assessment was performed in parallel with data extraction. We performed quality assessment on full text papers that were included for data extraction.

In performing quality assessment, the first author performed the quality assessment on 13 secondary studies. The second and third author would then perform a post-hoc validation of the quality assessment. After the post-hoc validation, we calculated the Kappa coefficient to see how much the reviewers agree on the quality scores of the secondary studies. The first and second author had a Kappa coefficient of 0.478. The first and third author had a Kappa coefficient of 0.586. Both Kappa coefficients suggested moderate agreement (Landis and Koch 1977).
4.3 Review Method

4.3.5 Synthesis

In this tertiary we adapted the synthesis steps mentioned in qualitative meta-study. Meta-study is “a research approach involving analysis of the theory, methods, and findings of qualitative research and the synthesis of these insights into new ways of thinking about phenomena” (Paterson et al., 2001). The aim of a meta-study is to transform the aggregation of study results into a coherent knowledge with purpose of generating new theory and informing practice. This tertiary study was not aimed to generate theory, but we intended to generate a consolidated view to inform research and practice.

There are four steps in meta-study: (1) meta-data analysis, (2) meta-method, (3) meta-theory, and (4) meta-analysis (Paterson et al., 2001). In meta-data analysis the data from each secondary study were collected and compared. Data were later tabulated into tables as we presented in Section 4.5. Next is meta-method, in this step methodological soundness of the secondary studies were examined. In our case, we examined the review and synthesis method through the quality assessment process, see Table 4.3 for quality criteria. We did not perform the third step, which was meta-theory, because we did not intend to produce a new theory pertaining to Agile and Lean. The last step was meta-synthesis where the results from the previous steps were integrated. In this step, we examined the reported impacts of Agile and Lean practices on each project constraint together with the reported empirical support. Furthermore we examined if the high quality secondary studies showed the same results as those of low quality. We also examined if the selected synthesis method also influenced the results of the secondary studies.

4.3.6 Validity Threats

Our tertiary study is also a subject to validity threats common to literature review studies.

Missing relevant secondary studies: The limitation of this tertiary study is possibility of missing important secondary studies. The mitigate this issue we used a broad search string, instead of using the individual practices as keywords, we used the Agile and Lean methods in our search. We also omitted “software engineering” or “software development” in our search string to enable broader hits. Our inclusion/exclusion criteria were also defined to be more inclusive, by ensuring that we include secondary studies in Agile with a defined review process. Since the Kappa coefficient between first and third author was low, first author did another round of full text screening on the papers that passed
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

title and abstract screening. From this process, two previously excluded papers were added for full text analysis, from 39 papers to 41 papers.

Missing relevant data from the secondary studies: The first author was the primary reviewer of this tertiary study and thus there was a potential to introduce bias. This issue was mitigated by test-retest and post-hoc validation by second and third author for the full text screening, data extraction, and quality assessment. The first author performed a test-retest for the data extraction as suggested in [Kitchenham, 2010]. In doing the test-retest, the first author performed two rounds of data extraction. The first author then check if she had missed some information the first time. Missing information would then be complemented. A post-hoc validation was also performed by the second and third author to check for consistency in the full text screening, data extraction, and quality assessment.

Publication Bias: Studies with positive results are more likely to be published than those with negative results. This issue did not seem to be a problem as we found secondary studies that reported on negative outcomes. There is a risk that the collected secondary studies are skewed towards particular practices. We mitigated this issue by using the Agile and Lean methods, instead of practices, in our search keywords.

4.4 Overview of Agile and Lean Secondary Studies

The results of this tertiary study is divided into two parts. The first part, which we present in this section, is the overview of secondary studies which has been conducted in Agile and Lean methods. The second part, which we present in Section 4.5 is the evaluation of Agile and Lean practices and their impacts on project constraints, as well as their empirical support.

4.4.1 Trends in Agile and Lean Secondary Studies

The first part of our analysis is to examine the state-of-art in Agile and Lean secondary studies. The list of included secondary studies is summarized in Table 4.4. The mapping is adopted from Diaz et al. (2011).

We found 36 Type 1 papers and 5 Type 2 papers that passed our inclusion/exclusion criteria. We looked at the trend of secondary studies in Agile over the years. In total we found 29 SLR, 4 Systematic Mapping (SM), 1 meta-
Table 4.4: List of Secondary Studies

<table>
<thead>
<tr>
<th>ID</th>
<th>Citation</th>
<th>ID</th>
<th>Citation</th>
<th>ID</th>
<th>Citation</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>Ahmad et al., 2013</td>
<td>S15</td>
<td>Al-Baik and Miller, 2014</td>
<td>S29</td>
<td>Kacheva et al., 2009</td>
</tr>
<tr>
<td>S2</td>
<td>Causevic et al., 2011</td>
<td>S16</td>
<td>Changas et al., 2013</td>
<td>S30</td>
<td>Salah et al., 2014</td>
</tr>
<tr>
<td>S3</td>
<td>Da Silva et al., 2011</td>
<td>S17</td>
<td>Silva da Silva et al., 2011</td>
<td>S31</td>
<td>Salvador et al., 2014</td>
</tr>
<tr>
<td>S4</td>
<td>Dyba and Dingsoyr, 2008</td>
<td>S18</td>
<td>Diaz et al., 2011</td>
<td>S32</td>
<td>Senagath and Srinivasan, 2013</td>
</tr>
<tr>
<td>S5</td>
<td>Gandomani et al., 2012</td>
<td>S19</td>
<td>Diebold and Dahlem, 2014</td>
<td>S33</td>
<td>Silva et al., 2015</td>
</tr>
<tr>
<td>S6</td>
<td>Magdaleno et al., 2012</td>
<td>S20</td>
<td>Hansson et al., 2014</td>
<td>S34</td>
<td>Sietio et al., 2011</td>
</tr>
<tr>
<td>S7</td>
<td>Munir et al., 2011</td>
<td>S21</td>
<td>Hegin and Bremoum, 2014</td>
<td>S35</td>
<td>Stavru et al., 2013</td>
</tr>
<tr>
<td>S8</td>
<td>Mesay et al., 2011</td>
<td>S22</td>
<td>Hossein et al., 2014</td>
<td>S36</td>
<td>Osman et al., 2014</td>
</tr>
<tr>
<td>S9</td>
<td>Perozai et al., 2015</td>
<td>S23</td>
<td>Jalali and Wohlin, 2012</td>
<td>S37</td>
<td>Yanzer Cahra et al., 2014</td>
</tr>
<tr>
<td>S10</td>
<td>Stebsos and Shaneb, 2010</td>
<td>S24</td>
<td>Jurca et al., 2014</td>
<td>S38</td>
<td>Roininen et al., 2010</td>
</tr>
<tr>
<td>S11</td>
<td>Rahique and Mist, 2014</td>
<td>S25</td>
<td>Kupiainen et al., 2014</td>
<td>S39</td>
<td>Stavru et al., 2013</td>
</tr>
<tr>
<td>S12</td>
<td>Shen et al., 2012</td>
<td>S26</td>
<td>Mitchell and Seaman, 2009</td>
<td>S40</td>
<td>Stavru et al., 2013</td>
</tr>
<tr>
<td>S13</td>
<td>Khan et al., 2011</td>
<td>S27</td>
<td>Nguyen-Cong and Tran-Cao, 2013</td>
<td>S41</td>
<td>Kasoju et al., 2013</td>
</tr>
<tr>
<td>S14</td>
<td>Abrantes and Travassos, 2011</td>
<td>S28</td>
<td>Oliveira Albuquerque et al., 2012</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

analysis, 1 quasi-SLR, 1 literature review, and 5 Type 2 papers. Figure 4.2 shows the trend of published secondary studies.

Figure 4.2: Number of Secondary Studies performed according to Year

The number of secondary studies gradually increased between 2008 and 2012. There was a slight decrease in 2013, but then it increased significantly in 2014,
with 13 SLRs. In the figure, we can see that there was only one SLR in 2015. We included this paper because it was available online in 2014. However, it was included for 2015 journal.

Furthermore, since the introduction of Agile manifesto in 2001 (Beck et al., 2001), the area grew rapidly. Dybå and Dingsøyr (2008) reported a secondary study which was conducted in 2005 that included 33 empirical studies. We did not find any secondary studies that fulfil our inclusion criteria prior to 2008. The earliest secondary study we found was by Abrahamsson et al., (2002). However, we excluded the paper because we did not find a description of the review process. The increasing trend of secondary study in 2008 onward was attributed to the introduction of EBSE by Kitchenham et al., (2004) and also the area was still growing in the early 2000 and the number of primary studies were too little to perform a meaningful secondary study.

We found different methods being employed in performing secondary studies. We also found that different guidelines were used by the authors. Table 4.5 summarizes the methods and the referred guidelines. Some papers were mentioned twice in different methods and guidelines because authors mentioned more than one methods and/or guidelines.

<table>
<thead>
<tr>
<th>Secondary Study Method</th>
<th>Guideline to the Method</th>
<th>Adopted in</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLR</td>
<td>Kitchenham and Charters</td>
<td>S1, S2, S4, S5, S7, S10, S11, S12, S13, S15, S16, S17, S18, S20, S21, S22, S23, S25, S26, S27, S28, S31, S32, S33, S34, S36, S37, S38, S40, S41</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>Biolchini</td>
<td>S17</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Pai et al</td>
<td>S14</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Unclear</td>
<td>S30, S39</td>
<td>2</td>
</tr>
<tr>
<td>Semi-SLR</td>
<td>Kitchenham and Charters</td>
<td>S8</td>
<td>1</td>
</tr>
<tr>
<td>Quasi-SLR</td>
<td>Kitchenham and Charters</td>
<td>S6</td>
<td>1</td>
</tr>
<tr>
<td>SM</td>
<td>Kitchenham and Charters</td>
<td>9, 19</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Petersen</td>
<td>S3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Unclear</td>
<td>S24</td>
<td>1</td>
</tr>
<tr>
<td>LR</td>
<td>Unclear</td>
<td>S35</td>
<td>1</td>
</tr>
</tbody>
</table>
As we can see that not all authors adopted SLR guidelines in SE such as Kitchenham and Charters (2007) and Biolchini et al. (2005). Some authors adopted guidelines from medicine such as Pai et al. (2004). We also found modified or simplified version of Kitchenham and Charters (2007) review steps as mentioned as quasi-, and semi-SLR. A number secondary studies did not mention which guideline was adopted in conducting the secondary study.

The large number of secondary studies that adopted Kitchenham and Charters (2007) SLR guideline suggest that researchers in Agile and Lean were adopting a more systematic approach of reviewing the literature. This indicate the area was growing in terms of number of primary studies, and the need of adopting EBSE to examine Agile and Lean implementation.

Topics in Agile Secondary Study

We found a large variety of topics in Agile and Lean secondary studies. The combinations of Agile and Lean with other areas in SE, such as, Model Driven Development (MDD), Global Software Development (GSD), etc., dominated the topic of the secondary studies.

General discussion on Agile and Lean was also a popular the secondary studies topic. Agile and GSD was second most popular topic. There were also topics pertaining to implementation of Agile and Lean in specific domains, such as, embedded systems, scientific software, and regulated domain. There were also secondary studies that focus on specific Agile and Lean practices, such as TDD and frequent deliveries. The large variation of secondary topics in Agile and Lean again further reinforced how the area had grown. It shows the range of application of Agile and Lean in different aspects of software development, like, product line, globalization, user-experience, etc. It also shows the wide range of domains which Agile and Lean are being implemented. Table 4.6 summarizes the topics covered in Agile and Lean secondary studies.

Aims of Secondary Studies

When examining the aims stated in the secondary studies, we identified four types of secondary studies, they were:

- Classify Primary Studies. Secondary studies in this category reviewed the primary studies and classified them into different categories. All SM studies fit into this categories
- Find association. Secondary studies in this category reviewed the primary studies and examined the association between certain interventions and
Table 4.6: Topics of Secondary Studies

<table>
<thead>
<tr>
<th>Topic</th>
<th>References</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combining Agile/Lean with other topics in SE, e.g., User Centered Design (UCD), Software Product Line Engineering (SPLE), Global Software Engineering (GSE), Metrics, Model Driven Development (MDD), Cloud</td>
<td>S17, S24, S30, S31, S3, S13, S18, S25, S27, S36, S22, S23, S20, S21, S40, S39</td>
<td>16</td>
</tr>
<tr>
<td>Agile/Lean General</td>
<td>S4, S14, S19, S32, S35, S38</td>
<td>6</td>
</tr>
<tr>
<td>Application of Agile/Lean in specific domains e.g., Embedded Systems, Scientific Software, Large Scale Software Systems, Automotive</td>
<td>S9, S12, S28, S34, S41</td>
<td>5</td>
</tr>
<tr>
<td>Test Driven Development (TDD)</td>
<td>S2, S7, S11</td>
<td>3</td>
</tr>
<tr>
<td>Comparing Agile with other development process models e.g., Open Source Software (OSS), Plan driven)</td>
<td>S5, S6, S26</td>
<td>3</td>
</tr>
<tr>
<td>Lean - Kanban</td>
<td>S1, S15</td>
<td>2</td>
</tr>
<tr>
<td>Agile and Management (Project Management, Knowledge Management)</td>
<td>S16, S37</td>
<td>2</td>
</tr>
<tr>
<td>Agile and Value Creation and Quality</td>
<td>S10, S29</td>
<td>2</td>
</tr>
<tr>
<td>Agile and CMMI</td>
<td>S33</td>
<td>1</td>
</tr>
<tr>
<td>Frequent Releases</td>
<td>S8</td>
<td>1</td>
</tr>
</tbody>
</table>

certain outcomes. In this case the implementation of Agile or Lean practices and their impacts on different aspects, such as quality, lead time, etc.

- Identify Factors. Secondary studies in this category reviewed the primary studies to identify various factors, like challenges, benefits, limitations, or certain practices.
- Survey the literature. Secondary studies in this category reviewed the primary studies to identify trends and extent of Agile and Lean research.

In categorizing the papers, we looked at the stated aim and how the results were presented. Table 4.7 summarizes the types of secondary studies. The stated aim from each secondary studies is compiled in Appendix B.

Most secondary studies were aimed at identifying various factors pertaining to Agile and Lean methods. We found a number of secondary studies aimed at surveying the literature and classifying the literature. There were also a few secondary studies that were aimed at examining associations between Agile and
Lean methods or practices with certain factors. The secondary studies under “find association” were primarily pertaining to TDD.

There were more studies aimed at to get an overview of Agile and Lean implementations. There were very little secondary studies that aimed at evaluating the methods or practices and their outcomes on software development, which what an SLR was intended to be (Centre for Reviews and Dissemination 2009).

4.5 Results and Analysis

This section presents the detailed analysis of the secondary studies. In this section we presented the analysis of the population of the secondary studies. We also presented the list the impacts of Agile and Lean practices and their empirical support. In the detailed analysis we only included secondary studies S1-S13.

4.5.1 Quality Assessment

We examined at the quality scores of each secondary study. The scores for secondary studies were between two and eight. The maximum score that each secondary study could obtain was nine. The quality scores of each secondary study is summarized in Table 4.8.

More than half (54%) of the secondary studies scored below five. Most of the low-scoring studies were missing criteria for preventative steps to minimize bias (C5) and evidence synthesis (C7a and C7b). Issues pertaining to minimizing bias, providing sufficient information on the included primary studies, synthesizing evidence, and evaluating strength of evidence were often missed or only partially addressed in most the secondary studies.

<table>
<thead>
<tr>
<th>Category</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classify primary studies</td>
<td>S3, S9, S19, S21, S24</td>
</tr>
<tr>
<td>Find Association</td>
<td>S2, S7, S10, S11, S26</td>
</tr>
<tr>
<td>Identify Factors</td>
<td>S4, S5, S6, S8, S12, S13, S14, S15, S16, S17, S18, S20, S22, S23, S28, S29, S30, S32, S33, S35, S37, S38, S39</td>
</tr>
<tr>
<td>Survey the literature</td>
<td>S1, S25, S27, S31, S34, S36, S40, S41</td>
</tr>
</tbody>
</table>
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Table 4.8: Quality Assessment

<table>
<thead>
<tr>
<th>ID</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>C6</th>
<th>C7a</th>
<th>C7b</th>
<th>C8</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RQ Search</td>
<td>Incl./Excl.</td>
<td>QA Bias</td>
<td>Detail</td>
<td>Synthesis</td>
<td>SoE</td>
<td>Concl.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>S2</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>S3</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>4.5</td>
</tr>
<tr>
<td>S4</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>S5</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>S6</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>4.5</td>
</tr>
<tr>
<td>S7</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>S8</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>S9</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>S10</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>5.5</td>
</tr>
<tr>
<td>S11</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>S12</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>S13</td>
<td>0.5</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>2.5</td>
</tr>
</tbody>
</table>

| Sum | 6.5 | 8 | 10 | 5 | 3 | 7 | 6 | 3 | 10 |

Most secondary studies had clear research questions. However most of them were not formulated with respect to population, intervention, comparator, outcome and study design (PICOS). Most secondary studies only refer to another source pertaining to their quality criteria and did not present the results of the quality assessment. With respect to minimizing bias most secondary studies did not report on Kappa coefficient, nor reported piloting the review. Lastly, strengths of evidence were often not considered. We did not exclude secondary studies based on quality scores.

4.5.2 Primary Studies

We examined the primary studies from the 13 secondary studies. We found very little overlap among the secondary studies. We found 50 papers that are shared by at least two papers and four papers at most. Figure 4.3 shows the overlap among the secondary studies. The number in the brackets represents the total number of primary studies from each secondary study.

As we can see from Figure 4.3, there were four clusters of papers. In Cluster 1 (S1, S8, S9) contained papers focusing on Lean methods (S1, S9) and frequent deliveries (S8). Meanwhile Cluster 2 contained papers pertaining to combination of Agile and SPLE. Papers in Cluster 3 had variety of topics, they were, general Agile and Lean, Agile-OSS, and Agile-Embedded systems. Lastly, Cluster 4 was
a cluster of paper focusing on TDD with the exception of S10, that a broader focus on Agile in general.

Both in Cluster 1 and Cluster 2, although the papers in each cluster shared the same topic, they had very little overlap. This could be attributed by the slight different of focus. For instance in Cluster 1, S1 was focused on Kanban, meanwhile S9 had a broader focus on Lean in general. Meanwhile in Cluster 2, S3 had a broader focus on SPLE and S13 had more narrow focus on reference architecture.

In Cluster 3, there were three papers, S12, S6, and S10 that overlapped with S4. Although this cluster had variety of topics the overlap with S4 was because S4 was a general secondary study in Agile and Lean. Meanwhile the overlap between S5 and S6 was because these papers share the topic of Agile and OSS. We could also observe there were five overlapping primary studies between Cluster 3 and Cluster 4.
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

In Cluster 4, we could see that there are many overlaps among the four papers. However, the three papers focusing on TDD (S2, S7, S11), had small number of overlapping primary studies. S2 and S7 had 21 overlapping papers as these secondary studies focused on empirical studies in TDD. Given the similarity of focus we expected more overlap among S2 and S7. However, if we looked further into S2 and S7, S7 only focused on the impact of TDD on quality and productivity, meanwhile S10 had broader focus on the impact of TDD. We did not expect large overlap with S11 because it only focused on experiments. Given the large number of empirical studies conducted in TDD, this showed that TDD as one of Agile practices had been well researched. The lack of overlap suggested that researchers in TDD had separate agendas in examining the impact of TDD.

Both in Cluster 3 and 4 we found papers that was shared by three to four papers. In Cluster 3, S4, S6, S12 shared one paper, meanwhile in Cluster 4, S2, S7, S10, S11 shared six papers. These papers could be considered as key papers in Agile and Lean methods. The list of the papers is presented in Table 4.9.

Table 4.9: List of Most Cited Papers from the Secondary Studies

<table>
<thead>
<tr>
<th>Papers</th>
<th>Cited In</th>
</tr>
</thead>
<tbody>
<tr>
<td>D. Karlström, P. Runeson, Combining agile methods with stage-gate project management, IEEE Software 22(3) (2005) 43 – 49.</td>
<td>S4, S6, S12</td>
</tr>
</tbody>
</table>
4.5 Results and Analysis

4.5.3 Scope of Agile and Lean Secondary Studies

To examine the scope of studied Agile and Lean practices, we used a pre-existing list of Agile and Lean practices from Petersen (2011), so we could see which practices had been examined and which had not. The list from Petersen (2011) was selected as it covers both Agile and Lean practices. In mapping the identified practices from the secondary studies to the correct row, we looked for the description of the practice, whenever provided, as described in the secondary studies against the definitions in Petersen (2011). We also used the definitions presented in Williams (2010) as a cross reference.

For the impacts of Agile and Lean, we collected them bottom-up from the included secondary studies. Each impact was then mapped to the project constraints as defined in Project Management Body of Knowledge (PMBOK) which includes, but not limited to: scope, quality, schedule, budget, resources, and risks (Project Management Institute, 2004). The mapping of studies to practices and project constraints is summarized in Table 4.10.

4.5.4 Agile and Lean Practices

From the list of practices mentioned in Petersen (2011), we found 13 out of 26 practices that were reported in the secondary studies. TDD was the most studied Agile practice with 10 secondary studies, three of them were specifically focused on TDD. Pair programming, and Planning game were each mentioned in two secondary studies. Each of the following practices was mentioned in one secondary study, they are: on-site customer, adaptive planning, stand up meeting, incremental deliveries, short iterations, value stream mapping (VSM), and refactoring, and Kanban pull system.

The secondary studies often did not provide any definition of the practices that were reported. At times, the secondary studies used the same terms as in Petersen (2011), they were, on-site customer, refactoring, pair-programming, the planning game, and VSM.

Secondary studies mapped under TDD and test automation were the secondary studies that discuss the practice of creating test cases prior to coding. TDD as a term was also used rather consistently throughout the secondary studies. There were also other terms, such as, test-first programming (S2, S4) and test-first development (S11).

Rapid releases (S8) was mapped under incremental deliveries, as it entailed releasing new or updated versions of software to the customer frequently (Petersen, 2011). Rapid releases did not fit into the definition of short iteration...
Table 4.10: Practices and Project Constraints Mapping

<table>
<thead>
<tr>
<th>Practice</th>
<th>Scope</th>
<th>Quality</th>
<th>Schedule</th>
<th>Budget</th>
<th>Resources</th>
<th>Risks</th>
<th>Communication</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Onsite customer</td>
<td>S4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Metaphores and user stories</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>refactoring</td>
<td>S10</td>
<td>S10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Coding standards</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Team code-ownership</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Low dependency architecture</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>TDD &amp; test automation</td>
<td>S12</td>
<td>S2, S3, S4, S5, S6, S7, S10, S11, S12, S13</td>
<td>S2, S7, S10, S11, S12, S10</td>
<td>S2, S10</td>
<td>S2, S10</td>
<td>S10</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>Pair programming</td>
<td>S4, S10</td>
<td>S4, S10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Continuous integrations</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Inspection</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Configuration Management</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Incremental deliveries</td>
<td>S8</td>
<td>S8</td>
<td>S8</td>
<td>S8</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Internal/external release</td>
<td>S9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Short iteration</td>
<td>S4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Adaptive planning with highest priority user stories (Adaptive planning)</td>
<td>S1</td>
<td>S1</td>
<td>S1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Time-boxing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Planning game</td>
<td>S4, S10</td>
<td>S4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Co-located development</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Whole team (cross-functional team)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>40h week</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>stand up meeting</td>
<td>S4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Team chooses own task</td>
<td>S1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>VSM</td>
<td>S9</td>
<td>S9</td>
<td>S9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Inventory management</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Chief Engineer</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Kanban pull systems</td>
<td>S1</td>
<td>S1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td><strong>Sum</strong></td>
<td>1</td>
<td>13</td>
<td>8</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

as it entails performing development activities in short period of time, where a feature was delivered to obtain feedback from the customers. The definitions provided for short iterations both in Petersen (2011) and Williams (2010) do not entail releasing a version. However short iterations fit the practice of delivering in-progress-software to obtain continuous feedback mentioned in S1.
The secondary study mapped under *separating between internal and external releases* (S9) discussed the practice of dividing development task into major and minor enhancement, called concurrent development system (CDS).

S1 mentioned two Kanban practices, they were visualization and prioritization of tasks. S1 did not provide a description of visualization, we assumed it was a visual mean for tracking and prioritizing tasks, as in a Kanban board. Meanwhile, S1 defined prioritization as a practice to aid which work item needs to be pulled next to optimize values. Although S1 separated the two practices apart, they both fits under the definition for *adaptive planning with highest priority user stories*.

The secondary study mapped under *stand-up meeting* discussed a Scrum practice that pertains to daily meeting that involves different stakeholders (S4).

The secondary study mapped under *team chooses own task* discussed a Kanban practice where team members were allowed to decide their own task management and prioritization, without upper level management approval (S1).

The secondary study mapped under *Kanban pull systems* discussed a Kanban practice that limited work in progress by pulling the next work item according to the team’s capacity (S1). This description fits the definition of *Kanban pull systems* where team members are not to be overloaded with work (Petersen, 2011).

### 4.5.5 Impacts of Agile and Lean Practices

The impacts of Agile and Lean practices mentioned in the secondary studies were mapped into different project constraints, see Table 4.11. From Table 4.10 we can see that quality was the most studied impact of Agile practices with 13 secondary studies, followed by schedule with eight secondary studies. Resource and communication each was reported in three secondary studies. Budget was reported in two secondary studies. Meanwhile scope, and risks each was reported in one secondary study.

In general, mapping the secondary studies to the relevant impacts were quite straightforward. However, the lack of definition from the secondary studies sometimes made the mapping rather difficult. In such events, we referred to available guidelines and standards, like PMBOK (Project Management Institute, 2004) and ISO 25010 (ISO/IEC, 2010). We mapped fulfilling requirements (S12) under scope because activities in scope management includes managing requirements (Project Management Institute, 2004). We mapped productivity (S7, S10, S11) under schedule, because productivity was often measured with respect to time. Meanwhile, effort (S9) and estimating work size (S4, S10) were
mapped under resources, as determining effort and estimating work level is part of human resource management knowledge area in the PMBOK. As for quality aspects, we found different variables and measures for quality. In mapping quality impacts, we also used [ISO/IEC (2010)] as a reference to map the secondary studies.

There were different variables and measures reported in the secondary studies with respect to each project constraint. Table 4.11 summarizes the mapping of different variables and measures and project constraints.

We found that number of defects and defect density were commonly used to measure external quality. Meanwhile complexity was commonly used to measure internal quality. However, we found different variations of measures to measure time and productivity. Many secondary studies did not provide information pertaining to the measures that they collected.

Table 4.11: Mapping of Variables and Measures to Project Constraints

<table>
<thead>
<tr>
<th>Project Constraints</th>
<th>Variables</th>
<th>Measures</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope Requirements</td>
<td>number of defects/bugs/trouble reports, defects per KLOC/defect density, number/percentage of acceptance/black-box/external tests passed, number/percentage of unit tests passed, quality mark given by client</td>
<td>S11</td>
<td></td>
</tr>
<tr>
<td>Quality</td>
<td>Defect count, defect density, mean time assertion per method, total assertions (passed/failed), number of test cases passed</td>
<td>S7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>number of passed acceptance tests or the total number of defects or number of defects/KLOC (defect density), number of defects found before release or defects reported by customers</td>
<td>S10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Number of defects</td>
<td>S3, S8, S9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Robustness</td>
<td></td>
<td>S7</td>
</tr>
<tr>
<td>Customer satisfaction</td>
<td>Not provided</td>
<td></td>
<td>S1, S4</td>
</tr>
<tr>
<td>Internal Quality</td>
<td>Condition coverage, Branch coverage, Nested block depth, Cyclomatic complexity, Number of parameters, Coupling between objects, Information flow, Weighted class per method, Lack of cohesion metrics, Mutation score indicator</td>
<td>S7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td></td>
<td>S2</td>
</tr>
<tr>
<td></td>
<td>Code size, Cyclomatic complexity, Code reuse, Coupling and cohesion</td>
<td>S10</td>
<td></td>
</tr>
</tbody>
</table>

Continued on next page
### 4.5 Results and Analysis

Table 4.11 – *Continued from previous page*

<table>
<thead>
<tr>
<th>Project Constraints</th>
<th>Variables</th>
<th>Measures</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code quality</td>
<td>Not provided</td>
<td>S2, S4, S5, S6, S12</td>
<td></td>
</tr>
<tr>
<td>Reusability software components</td>
<td>Not provided</td>
<td>S13</td>
<td></td>
</tr>
<tr>
<td>Schedule</td>
<td>Size</td>
<td>Not provided</td>
<td>S2, S8, S9</td>
</tr>
<tr>
<td></td>
<td>Time</td>
<td>Release cycles, Time between two subsequent releases</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mean-time per assertion, Mean time, Total/overall time, Average hours worked, Time taken in person hours, Time taken on testing as a percentage of total time, Time spent on coding as a percentage of total time, Total person hours</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Development time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Lead time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Waiting time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Development time, feedback time</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Testing time</td>
<td></td>
</tr>
<tr>
<td>Productivity</td>
<td>Output/unit effort, Total time to complete a story card, New lines of code/development effort, LOC/person hour, LOC/development effort, User stories/person hour</td>
<td>S7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Development time/person hours spent/task time, total LOC divided by total effort, or the number of LOC per hour, total noncommented LOC, number of delivered stories per unit effort (or implemented user stories per hour), delivered noncommented LOC per unit development effort (or effort per ideal programming hour), and hours per feature/development effort per LOC</td>
<td>S11</td>
<td></td>
</tr>
<tr>
<td>Budget</td>
<td>Cost</td>
<td>Not provided</td>
<td>S2</td>
</tr>
<tr>
<td></td>
<td>Development cost</td>
<td>S10</td>
<td></td>
</tr>
<tr>
<td>Resource</td>
<td>Estimation</td>
<td>Work size</td>
<td>S4, S10</td>
</tr>
<tr>
<td></td>
<td>Effort</td>
<td>Amount of Rework</td>
<td>S9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Testing effort, development effort</td>
<td>S10</td>
</tr>
<tr>
<td>Risk</td>
<td>Control Risk</td>
<td>Not provided</td>
<td>S8</td>
</tr>
<tr>
<td>Communication</td>
<td>Customer collaboration</td>
<td>Not provided</td>
<td>S4</td>
</tr>
</tbody>
</table>
Table 4.11 – Continued from previous page

<table>
<thead>
<tr>
<th>Project Constraints</th>
<th>Variables</th>
<th>Measures</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Team work</td>
<td>Not provided</td>
<td></td>
<td>S1, S4, S10</td>
</tr>
</tbody>
</table>

4.5.6 **Empirical Support for the Impacts of Agile and Lean Practices**

We found that a practice could have positive, negative, or no impact on different project constraints. The ‘+’ sign indicated that the respective Agile and Lean practices had positive impact on project constraints. Meanwhile ‘0’ indicated that the Agile and Lean practice had no impact. Lastly, ‘-’ indicated that the respective Agile and Lean practice had negative impact on project constraint.

Furthermore, we also indicated the synthesis methods used the secondary studies. If the synthesis method is not described, we indicated as “Unclear”. For systematic mapping studies, we indicated the synthesis method not applicable (N/A), because it is unlikely for systematic mapping study to have in-depth synthesis like narrative synthesis or meta-analysis ([Kitchenham and Charters](#))

We also found that secondary studies without a clear synthesis method tend to have low quality scores, i.e., below four. Meanwhile, secondary studies with clear synthesis method had quality score greater or equal to five.

Unless the type of empirical methods of the included primary studies was mentioned in the secondary studies, we indicated unclear empirical study as “study”. Otherwise we indicated the empirical methods, for example experiment, case study, or survey. If a secondary study only provided statements pertaining to the practices and their impacts without clear trace of the cited primary studies, we indicated as “Unclear”. We included secondary studies with unclear study type or unclear citation as long as brief statistics pertaining to empirical research methods of the primary studies was mentioned. Also, if an impact cell was left blank, it indicated that the respective secondary study did not find any primary study that reported the influence of a practice on a particular impact (+,0,-). If a secondary study did not report particular impact, we indicated as “Not reported”.

**Scope**

We only found one variable and one reference pertaining to scope. The secondary study suggested that TDD helped the developers to better understand
4.5 Results and Analysis

user requirements. The statement was supported by three studies. However, it was not clear what type of studies supported the statement. The secondary study itself reported on that it collect different types of empirical studies, but the primary studies referred regarding TDD and requirements were not clear.

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD</td>
<td>Requirements</td>
<td>S12</td>
<td>Unclear</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1 case study, 2 studies</td>
</tr>
</tbody>
</table>

**Quality**

We found a number of Agile and Lean practices that had been reported to have impacts on different aspects of quality. Table 4.12 summarizes the impacts on different Agile and Lean practices on quality. Refactoring was reported to have a positive impact on quality. However, it was not clear which type of quality was studied. Furthermore, the type of research method used in the primary study and the synthesis method were not clear.

The association between TDD and quality aspects was the most studied practice-impact pair. It was mentioned in 10 secondary studies, and more closely examined in three secondary studies (S2, S7 and S11). As we can observe from Table 4.12, there was a significant number of empirical studies that supported the positive impact of TDD on external quality, code quality, and complexity.

In S7 and S11, the impact of TDD was compared to another methods, like waterfall, test last development, or iterative test last development. However, this was not apparent in other secondary studies. Therefore it was hard to tell what was this direction of impact meant.

A meta-analysis (S11) study suggested that TDD had positive impacts on external quality when examining at all the experiments, regardless of the experiments’ subjects. However, when looking at experiments specifically with students subject, TDD showed no impact on external quality. Meanwhile, TDD showed negative impact on external quality in experiments that compared TDD and to iterative-test-last.

A vote counting study (S7) also suggested that TDD had overall positive impact on external quality. Given the rigour and relevance of the primary studies, the positive impact was supported by a number of empirical studies with high rigour and high relevance. Meanwhile, high rigour and low relevance
studies suggested that TDD had no impact on external quality. Meanwhile TDD showed no impact on robustness. However, there was only one experiment with low rigour and low relevance that supported it.

Secondary studies with unclear or not applicable synthesis method also suggested that TDD had positive impact on external quality. However, negative and impact primary studies were not reported.

The impact of TDD on internal quality, however was not as straightforward. There were different variables or measures used in different secondary studies.

The vote counting studies (S2,S7) suggested that TDD had positive impact on complexity. The impact of TDD on code size was inconclusive, because the two vote counting studies found contradicting primary studies. One of the vote counting studies (S7) suggested that TDD had positive impact on internal quality in general. Meanwhile S2 reported that TDD had positive impact on code quality.

A meta-ethnography study also suggested that TDD had positive impact on code quality. Meanwhile studies with unclear synthesis method also suggested that TDD had positive impact on code/internal quality and reusability, both at code and software component levels.

We found that regardless of the synthesis method, the secondary studies suggested that TDD showed positive impact on external quality. We also found that secondary studies with low and high quality were in agreement that TDD had positive impact on external quality. The impact of TDD on internal quality varied, depending on which specific measure of internal quality was examined. The high quality secondary studies, with clear synthesis methods, reported that TDD could have different impacts on internal quality. However, the low quality secondary studies, with unclear synthesis methods, were in agreement that TDD had positive impact on internal quality.

With respect to pair programming, a meta-ethnography study showed that pair programming had positive impact on quality. However, it was not specified which aspect of the quality was measured. Meanwhile a secondary study with unclear synthesis method showed that pair programming had positive impact on internal quality.

For the rest of the identified practices, i.e., incremental deliveries, short iterations, adaptive planning, VSM, and Kanban pull systems, the secondary studies suggested that these practices had positive impact on different quality attributes. We found that the secondary studies suggested the same outcome, regardless of the synthesis method or the quality of the secondary studies. However, negative and no impact primary studies were not reported.
Table 4.12: Impact of Agile and Lean Practices on Quality

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/ Measures</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Refactoring Quality</td>
<td>S10</td>
<td>Unclear</td>
<td>2 studies</td>
<td>Not reported</td>
</tr>
<tr>
<td>TDD</td>
<td>External quality</td>
<td>S11</td>
<td>Meta-analysis</td>
<td>25 experiments (18 experiments comparing to waterfall)</td>
</tr>
<tr>
<td></td>
<td>External quality</td>
<td>S7</td>
<td>Vote counting</td>
<td>6 case studies, 1 survey&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>External quality</td>
<td>S3</td>
<td>N/A</td>
<td>6 studies</td>
</tr>
<tr>
<td></td>
<td>External quality</td>
<td>S8, S9, S10</td>
<td>Unclear</td>
<td>7 experiment, 8 case studies, 2 studies</td>
</tr>
<tr>
<td></td>
<td>Robustness</td>
<td>S7</td>
<td>Vote counting</td>
<td>1 mixed case study + experiment, 3 case studies, 1 experiment</td>
</tr>
<tr>
<td></td>
<td>Complexity</td>
<td>S2</td>
<td>Vote counting</td>
<td>11 case studies, 2 experiments</td>
</tr>
<tr>
<td>Complexity</td>
<td>S7</td>
<td>Vote counting</td>
<td>2 case studies&lt;sup&gt;a&lt;/sup&gt;</td>
<td></td>
</tr>
<tr>
<td>Internal Quality</td>
<td>S7</td>
<td>Vote counting</td>
<td>1 experiment&lt;sup&gt;c&lt;/sup&gt;</td>
<td>10 experiments&lt;sup&gt;e&lt;/sup&gt;</td>
</tr>
<tr>
<td>Code quality</td>
<td>S2</td>
<td>Vote counting</td>
<td>11 case studies, 2 experiments</td>
<td>2 experiment</td>
</tr>
<tr>
<td>Code quality</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
<td>Not reported</td>
</tr>
</tbody>
</table>

<sup>a</sup> 4 case studies<sup>b</sup> 3 experiments<sup>c</sup> 1 survey<sup>d</sup> 6 experiments<sup>c</sup> 1 experiment<sup>c</sup>
Table 4.12 – Continued from previous page

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measures</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Code/Internal quality</td>
<td>S5, S6, S10, S12</td>
<td>Unclear</td>
<td>2 experiments, 3 case study, 1 simulation study, 4 studies</td>
<td>1 experiment, 1 case study (experiment + case study)</td>
</tr>
<tr>
<td>Code reusability</td>
<td>S10</td>
<td>Unclear</td>
<td>1 experiment</td>
<td>Not reported</td>
</tr>
<tr>
<td>Software components reusability</td>
<td>S13</td>
<td>Unclear</td>
<td>2 studies</td>
<td>Not reported</td>
</tr>
<tr>
<td>Code cohesion</td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
<td>1 mixed study (experiment + case study)</td>
</tr>
<tr>
<td>Size</td>
<td>S2</td>
<td>Vote counting</td>
<td>1 mixed case study + experiment</td>
<td>1 experiment</td>
</tr>
<tr>
<td>Size</td>
<td>S7</td>
<td>Vote counting</td>
<td>1 experiment</td>
<td>2 experiments</td>
</tr>
<tr>
<td>Pair Programming</td>
<td>Quality</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
</tr>
<tr>
<td>Internal Quality</td>
<td>S10</td>
<td>Unclear</td>
<td>14 experiments, 1 case study, 1 survey, 1 mixed study (survey + experiment)</td>
<td>1 experiment</td>
</tr>
<tr>
<td>Incremental External Quality Deliveries</td>
<td>S8</td>
<td>Unclear</td>
<td>2 studies</td>
<td>Not reported</td>
</tr>
<tr>
<td>Short Iterations</td>
<td>Customer satisfaction</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
</tr>
</tbody>
</table>

Continued on next page
Table 4.12 – Continued from previous page

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measures</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Adaptive planning</td>
<td>Customer satisfaction</td>
<td>S1</td>
<td>Unclear</td>
<td>5 studies</td>
</tr>
<tr>
<td>VSM</td>
<td>External Quality</td>
<td>S9</td>
<td>N/A</td>
<td>1 study</td>
</tr>
<tr>
<td>Kanban pull systems</td>
<td>Work quality</td>
<td>S1</td>
<td>Unclear</td>
<td>Unclear study</td>
</tr>
</tbody>
</table>

a Studies with high rigour and high relevance

b Studies with low rigour and high relevance
c Studies with high rigour and low relevance
d Studies with low rigour and low relevance
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Schedule

We also found a number of Agile and Lean practices that had been reported to have impacts on schedule. Table 4.13 summarizes the impacts on different Agile and Lean practices on schedule.

**Refactoring** was reported to have positive impact on productivity. However, it was only reported in one secondary study with unclear research method. Furthermore, the type of research method used in the primary study was not clear.

Again, **TDD** was the most studied practice regarding its impact on schedule. The impact of TDD on schedule was inconclusive. The empirical support seemed to be equally distributed for positive, negative, and no impact. Furthermore, there were variations of variables or measures used with respect to schedule.

A meta-analysis study showed that the impact of TDD on productivity was inconclusive. Experiments with students subjects and experiments that compared TDD to waterfall suggested that TDD had positive impact on productivity. However, in experiments conducted with practitioners TDD showed negative impact on productivity. Negative impact of TDD on productivity was observed in experiments that compared TDD to iterative test last development.

There were two vote-counting studies that studied different variables of schedule, such as, development time, feedback time, time, and productivity. TDD showed positive impact on feedback time. However, the impact on development time was inconclusive with a tendency towards negative. Meanwhile, primary studies with low rigour or low relevance suggested that TDD had positive impact on time. The impact of TDD on productivity was not observable on experiments with high rigour and low relevance. However, two case studies with high rigour and high relevance suggested that TDD had negative impact on productivity.

Secondary studies with unclear synthesis method studied time, testing time, and productivity. TDD showed negative impact on time, however it was supported by two case studies. Meanwhile, TDD showed positive impact on testing time, however there was only one study that supported it. Meanwhile with respect to productivity, the impact of TDD was inconclusive.

The impact of TDD on schedule across different synthesis methods and quality scores varied. Given there were different variables of schedule. The meta-analysis study and one of the vote counting studies were in agreement with respect to the negative impact of TDD on productivity. Meanwhile a secondary with unclear synthesis method suggested that the impact of TDD on productivity was inconclusive.
### Table 4.13: Impact of Agile and Lean Practices on Schedule

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Refactoring</td>
<td>Productivity</td>
<td>S10</td>
<td>Unclear</td>
<td>3 case studies, 1 mixed study</td>
</tr>
<tr>
<td>TDD</td>
<td>Development Time</td>
<td>S2</td>
<td>Vote counting</td>
<td>3 experiments, 2 case studies</td>
</tr>
<tr>
<td></td>
<td>Feedback Time</td>
<td>S2</td>
<td>Vote counting</td>
<td>3 case studies, 1 survey</td>
</tr>
<tr>
<td>Testing Time</td>
<td>S12</td>
<td>Unclear</td>
<td>1 study</td>
<td>Not reported</td>
</tr>
<tr>
<td>Time</td>
<td>S7</td>
<td>Vote counting</td>
<td>4 experiments&lt;sup&gt;c&lt;/sup&gt;</td>
<td>2 experiments&lt;sup&gt;c&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Productivity</td>
<td>S7</td>
<td>Vote counting</td>
<td>1 case study&lt;sup&gt;a&lt;/sup&gt;</td>
<td>2 case studies&lt;sup&gt;a&lt;/sup&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 experiment&lt;sup&gt;c&lt;/sup&gt;</td>
<td>10 experiments&lt;sup&gt;c&lt;/sup&gt;</td>
</tr>
<tr>
<td>Productivity</td>
<td>S11</td>
<td>Meta-analysis</td>
<td>18 academic experiments, 7 experiments comparing to iterative test last</td>
<td>8 industry experiments, 18 experiments comparing to waterfall</td>
</tr>
<tr>
<td>Productivity</td>
<td>S10</td>
<td>Unclear</td>
<td>2 experiments</td>
<td>1 case study</td>
</tr>
<tr>
<td>Pair programming</td>
<td>Time</td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
</tr>
</tbody>
</table>

Continued on next page
<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/ Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Schedule</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Time</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Incremental deliveries</td>
<td>Productivity</td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
</tr>
<tr>
<td>Release cycle</td>
<td>S8</td>
<td>Unclear</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Internal/external release</td>
<td>Release cycle</td>
<td>S9</td>
<td>Unclear</td>
<td>1 study</td>
</tr>
<tr>
<td>Adaptive planning</td>
<td>Efficient Work</td>
<td>S1</td>
<td>Unclear</td>
<td>2 studies</td>
</tr>
<tr>
<td>Team chooses own task</td>
<td>Time</td>
<td>S1</td>
<td>Unclear</td>
<td>Unclear</td>
</tr>
<tr>
<td>VSM</td>
<td>Waiting time</td>
<td>S9</td>
<td>N/A</td>
<td>1 study</td>
</tr>
<tr>
<td>Kanban pull systems</td>
<td>Lead time</td>
<td>S1</td>
<td>Unclear</td>
<td>Unclear</td>
</tr>
</tbody>
</table>

a Studies with high rigour and high relevance
b Studies with low rigour and high relevance
c Studies with high rigour and low relevance
d Studies with low rigour and low relevance
4.5 Results and Analysis

With respect to **pair programming**, a meta-ethnography study (S4) suggested that TDD had negative impact on time. Furthermore, a secondary study with unclear synthesis (S10) suggested that TDD had negative impact both on time and productivity.

For the rest of the practices, i.e., **incremental deliveries, internal/external release, adaptive planning, team chooses own task, VSM, and Kanban pull systems** were all reported in secondary studies with unclear or not applicable synthesis methods. The secondary studies showed that all these practices had positive impact on different variables of schedule. However, negative and no impact primary studies were not reported.

**Budget**

TDD was the only reported practice that had impact on budget. We can see from Table 4.14 there were very few primary studies that reported on the association between TDD and budget. A vote counting and a secondary with unclear synthesis method both suggested that TDD had positive impact on cost.

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>Budget</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD</td>
<td>Cost</td>
<td>S2</td>
<td>Vote counting</td>
<td>+</td>
</tr>
<tr>
<td></td>
<td>DevelopmentCost</td>
<td>S10</td>
<td>Unclear</td>
<td>0</td>
</tr>
</tbody>
</table>

**Resource**

There were also very few practices reported with respect to resources. Table 4.15 summarizes the impact of TDD and Planning game on resources.

With respect to **TDD**, it was interesting to see how the contradictory results. A secondary study with unclear synthesis method suggested that TDD had negative impact on testing effort, but had positive impact on development effort. This could be attributed by the implementation of TDD itself. Test cases and automated unit test were developed prior to coding. This added more work on testing side, but less work on development side.

Meanwhile, for **planning game**, a secondary study with unclear synthesis method and a meta-ethnography study both suggested that planning game had positive impact on work size estimation.
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

Table 4.15: Impact of Agile and Lean Practices on Resources

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>+</th>
<th>0</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDD</td>
<td>Testing effort</td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
<td>Not reported</td>
<td>1 experiment</td>
</tr>
<tr>
<td></td>
<td>Development effort</td>
<td>S10</td>
<td>Unclear</td>
<td>1 experiment</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Pair Programming</td>
<td>Effort</td>
<td>S10</td>
<td>Unclear</td>
<td>Not reported</td>
<td>Not reported</td>
<td>1 experiment</td>
</tr>
<tr>
<td>Planning game</td>
<td>Work size estimation</td>
<td>S10</td>
<td>Unclear</td>
<td>1 study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td></td>
<td>Work size estimation</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
</tbody>
</table>

Risk

Risk is another project constraint that had very little attention. We found only one secondary study with unclear synthesis method that reported the result from one case study. As we can see from Table 4.16, incremental deliveries had positive impact in managing development risk. Meanwhile negative and no impact were not reported.

Table 4.16: Impact of Agile and Lean Practices on Risk

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>+</th>
<th>0</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Incremental</td>
<td>Development risk</td>
<td>S8</td>
<td>Unclear</td>
<td>1 case study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
</tbody>
</table>

Communication

With the focus on interaction over processes and customer collaboration, communication was definitely a key point in Agile methods. There were a number of practices that were reported to have impacts on communication, as summarized in Table 4.17.

A meta-ethnography study reported that on site customer and stand up meeting had positive impact on customer collaboration. Meanwhile planning game had positive impact on team work. Incremental deliveries was also reported to have positive impact on customer collaboration by a secondary study
4.6 Discussion

with unclear synthesis method. Two secondary studies with unclear synthesis method suggested that pair programming and adaptive planning had positive impact on team work.

Table 4.17: Impact of Agile and Lean Practices on Communication

<table>
<thead>
<tr>
<th>Practice</th>
<th>Variable/Measure</th>
<th>Ref.</th>
<th>Synthesis</th>
<th>+</th>
<th>0</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>On site customer</td>
<td>Customer collaboration</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Stand up meeting</td>
<td>Customer collaboration</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Incremental Deliveries</td>
<td>Customer Collaboration</td>
<td>S8</td>
<td>Unclear</td>
<td>1 study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Pair programming</td>
<td>Team work</td>
<td>S10</td>
<td>Unclear</td>
<td>2 experiments</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Adaptive planning</td>
<td>Team work</td>
<td>S1</td>
<td>Unclear</td>
<td>5 studies</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
<tr>
<td>Planning game</td>
<td>Team work</td>
<td>S4</td>
<td>Meta-ethnography</td>
<td>1 case study</td>
<td>Not reported</td>
<td>Not reported</td>
</tr>
</tbody>
</table>

4.6 Discussion

Our tertiary study revealed large number of secondary studies found, 122 based on title and abstract screening, many of them do not include clear review processes. As mentioned in MacDonell et al. (2010) that it is impossible to trust the outcome of a review unless the process is transparent and well documented. Therefore, we only included 41 secondary studies after full text review. The earliest Agile and Lean SLR was published in 2008, and the number of secondary studies gradually increased the following years with highest peak in 2014. There are different topics being covered in the secondary studies, mainly combining Agile and Lean with other subject in SE, like GSD, MDD, etc. Our study shows that Agile and Lean are becoming more widespread in terms of application in software development.

We also found that most secondary studies are aimed at identifying factors in Agile and Lean, such as, challenges or benefits. This result is similar to the tertiary study by Verner et al. (2014), where they found that most SLRs in GSD produced summary and mappings of the literature. Very few secondary studies are aimed at investigating the impacts of Agile and Lean practices. Our study shows that Agile and Lean is still maturing and expanding, as most secondary studies are exploratory.
From 41 secondary studies only 13 provided discussions on the impact of Agile and Lean practices. From 13 secondary studies we performed more detailed analysis to study their population of primary studies, as well as the studied Agile and Lean practices. From 13 secondary studies that we included for detailed analysis, we found very little overlaps of primary studies. There are only 50 primary studies that are shared by two to four secondary studies. The largest overlap of 21 papers is shared by S2 and S7, which is about 50% of the population of each of the secondary study. This finding is similar to the result reported in Wohlin et al. (2013). However, unlike previous tertiary studies that examined overlap of primary studies (MacDonell et al., 2010; Wohlin et al., 2013), the secondary studies that we included do not share the same research questions. Despite commonalities of topics, the finer details of the secondary studies are slightly different. For instance S2 and S7 are both vote-counting studies examining TDD. However S2 and S7 have different research questions.

The overall small overlap of primary studies among the secondary studies shows that research of Agile and Lean is very broad and continuously expanding. There are so many primary studies performed with so many different topics. However, at the same time also suggests the lack of unified focus among Agile and Lean researchers. As suggested by Ciolkowski and Münch (2005) researchers are often performing research for their goals, e.g., individual publications or theses.

We found a number of Agile and Lean practices discussed in the secondary studies. Out of 26 practices mentioned in Petersen (2011), there are 13 practices that was mentioned in the secondary studies. Four out of 13 secondary studies are dedicated to specific practices, i.e., TDD (S2, S7, S11) an incremental deliveries (S8). The most studied practice is TDD, with 10 secondary studies. Pair-programming and planning game are far second most studied practice with 2 secondary studies each. Our study shows there are many Agile and Lean practices are still under-studied. Our findings do not imply that the rest of Agile and Lean practices are not evaluated at all. Our finding suggests the lack of secondary studies of other Agile and Lean practices is because the number of primary studies are not significant enough to perform a secondary study.

We also found an interesting observation pertaining to the number of overlapping primary studies. We found three secondary studies, i.e., S2, S7, and S11, focusing on TDD, however we found 10 secondary studies that reported on the impact of TDD. However, we did not find any overlap between S2, S7, and S11 and the remaining seven secondary studies that reported on the impact of TDD. This shows that TDD there are even more primary studies on TDD and initially expected. However, the lack of overlap among the secondary studies
could indicate the TDD researchers are not building on each others work. It could also indicate that some of the studies on TDD lack rigour and of low quality, as many of them were not captured by the high quality secondary studies such as S7 and S11.

Our tertiary study also identified number of impacts of Agile and Lean practices, such as cost, time, quality, etc. The most studied impacts are quality, schedule, and communication. Meanwhile, scope and risk are studied in one secondary study each.

The impact of TDD on quality is the most studied association, followed by the impact of TDD on schedule. The secondary studies that studied the impact of TDD on quality suggest that TDD showed positive impact on external quality. However, with respect to internal quality, different secondary studies suggest different impact, depending on the measure of internal quality. For code quality and internal quality in general, the secondary studies suggest that TDD showed positive impact. However, with respect to size, the impact of TDD is inconclusive. The impact of TDD on schedule is inconclusive depending on the measures of schedule. The secondary studies suggest positive impact of TDD on feedback time. However, TDD showed tendency of negative impact on productivity. One of the secondary study indicate that TDD showed positive impact on development effort, but TDD showed negative impact on testing effort. Our tertiary study shows that implementing TDD can have various impacts on different project constraints. This shows that there are trade-offs that are associated with implementing a particular Agile practice.

As suggested by [Kitchenham and Charters] (2007), a tertiary review is dependent on high quality secondary studies. Our tertiary study also shows that most of the secondary studies that evaluate the impact of Agile and Lean practices are of low quality. We purposefully included such secondary studies because we did not want to dismiss any attempt to collect evidence pertaining to Agile and Lean practices. Most of the secondary studies did not clearly discuss or show a clear synthesis method. For the most part, evidence was not synthesized. The secondary studies merely repeated or summarized what was mentioned in the primary studies. Most secondary studies also did not provide enough information pertaining to the included primary studies. We often could not trace the research methods used in the primary studies. Furthermore, most of the secondary study also did not provide evaluation on the strengths of evidence. Our study shows that secondary studies in Agile and Lean are often summaries of existing primary studies, it has not yet reached the quality that they could be used to aid decision making as it has been practiced in medicine [Centre for Reviews and Dissemination] (2009).
We also examined how the synthesis method used in the secondary studies could influence the outcome. When examining at the secondary studies the impact of TDD on external quality, secondary studies with unclear synthesis method yield the same outcome as those that used vote-counting or meta-analysis. However, the same cannot be said when examining secondary studies that examined the impact of TDD on productivity. Vote counting and meta analysis studies suggested that TDD showed negative impact on productivity, however, secondary studies with unclear synthesis method was inconclusive. Our tertiary study indicates that the selection of synthesis method could yield different outcome from a secondary study. The same can be said pertaining to the quality of the review, secondary studies with rigorous review process can yield different outcome to less rigorous secondary studies. This shows that the result of secondary studies may not be reproducible, as previously examined by Wohlin et al. (2013).

Our tertiary study identified 13 Agile and Practices and their impacts on project constraints. Although we collected empirical support, nothing definitive can be said pertaining to the impacts of Agile and Lean practices. First of all, the data from the primary studies are heterogeneous. For instance with respect to adherence to TDD, often primary studies did not report how much they conformed to the description of TDD (S11). Second, often the primary studies performed to evaluate impact of Agile practices were based on short term observation (S7, S11). Third, number of empirical support does not reflect effect size. As suggested in S11, when comparing TDD to traditional method, the improvement of external quality is marginal.

### 4.6.1 Implications for Research and Practice

This tertiary study yields a number of implications both for research and practice.

The first implication for research, our tertiary study shows the need for a clear focus in Agile and Lean research. The breadth of topics in Agile and Lean secondary studies shows that Agile and Lean implementation is getting more pervasive in software development. Given that the success of Agile and Lean implementation is primarily based on stories and anecdotes Cohen et al. (2004), more effort should be put into in synthesizing evidence on the impacts of individual Agile and Lean practice, as well as a combination of them.

Our study also shows that secondary studies in Agile and Lean lacks meaningful information pertaining to the primary studies. Researchers who are in the process of or considering performing a secondary study, particularly an SLR,
need to include as much information as possible pertaining to the included secondary study. Furthermore, strength of evidence should be included as part of synthesis. When examining impacts of Agile and Lean practice, it is also important to state clearly what measures are used and what they entail. As our study suggests that secondary studies often do not provide detailed information what exactly was being measured.

Our tertiary study also reveals gaps in Agile and Lean research which opens up a venue for further research. There are still a number of Agile and Lean practices that have not been explored. Even the ones that have been identified in this tertiary study, with the exception of TDD, still lacks in depth evaluation.

For practitioners, our study compiled a list of Agile and Lean practices and their impacts (positive, negative, or no impact) on project constraints. Although we did not provide suggestive propositions, practitioners can use our findings to help assessing suitability of adopting an Agile and Lean practice. For instance, evidence seems to point that TDD has positive impact on external quality. However TDD seems to compromise productivity. If a practitioner wants to adopt TDD, they could expect improvement in external quality however, it might decrease the productivity.

4.7 Conclusions

We identified 122 secondary studies in Agile and Lean. However, many of them did not have clear review process. Only 41 secondary studies passed the inclusion/exclusion criteria. Most secondary studies are still exploratory, very few of them are aimed at examining the outcomes of the intervention, in this case Agile and Lean practices.

Out of 41 secondary studies, only 13 are relevant to our tertiary studies. After examining the primary studies included by the secondary studies, we found very little overlap. We also identified 13 Agile and Lean practices and their impacts on project constraints being studied in the secondary studies. TDD is the most studied practice, followed by pair-programming and planning game. Impacts of TDD and quality is the most studied associations of Agile and Lean practice and project constraints. Meanwhile the impacts of other practices are not as thoroughly studied as TDD.

When it comes to empirical support of the impacts of Agile and Lean practices, for the most part it was difficult to aggregate the results from the secondary studies. Most secondary studies did not provide synthesis and sufficient information pertaining to the primary studies. However, when it comes to the
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study

positive impact of TDD on quality, we found significant number of empirical support, obtained from high quality secondary studies.

This tertiary study reveals there is a lack of focus in Agile and Lean research. Many are doing research for their own purpose, as shown by the small overlaps among the primary studies. The evidences are often not synthesized nor examined in the secondary studies. There is a clear need for researchers to build upon existing and each others’ work. Also, the initiation of future secondary studies, should be more focused on investigating the impacts of Agile and Lean practices with attention to the rigour and traceability of the review process.

Appendix A: Quality Criteria

Table 4.18: Quality Assessment Score

<table>
<thead>
<tr>
<th>Criteria*</th>
<th>Score**</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>Was the review question clearly defined in terms of population, interventions, comparators, outcomes and study designs (PICOS)? Y: PICOS are used and RQs are clearly defined. P: PICOS are not used but RQs are clearly defined. N: RQs are not clearly defined.</td>
</tr>
<tr>
<td>C2</td>
<td>Was the search strategy adequate and appropriate?*** Y: The authors have either (searched 4 or more digital libraries and included additional search strategies) or (identified and referenced all journals addressing the topic of interest). P: The authors have searched 3 or 4 digital libraries with no extra search strategies or they searched a defined but restricted set of journals and conference proceedings. N: The authors have searched up to 2 digital libraries or an extremely restricted set of journals.</td>
</tr>
<tr>
<td>C3</td>
<td>Are the review’s inclusion and exclusion criteria described and appropriate? Y: The inclusion/exclusion criteria are explicitly defined in the study. P: The inclusion/exclusion criteria are implicitly defined. N: The inclusion/exclusion criteria are not defined and cannot be readily inferred.</td>
</tr>
<tr>
<td>C4</td>
<td>Were appropriate criteria used to assess the quality of the primary studies?*** Y: The authors have explicitly defined quality criteria, quality scoring procedures, and extracted them from each primary study. Information about the quality scores for the primary studies is provided (e.g., in a table or a distribution). P: The research questions involve study quality issues that are addressed by the study. The quality criteria are listed. The quality scoring procedure is described. The scores for individual primary studies are not documented. N: There is no explicit quality assessment of individual primary studies or the quality assessment has been described insufficiently.</td>
</tr>
<tr>
<td>C5</td>
<td>Were preventive steps taken to minimize bias and errors? Y: Measures for inter-rater agreement are provided, e.g., Kappa-coefficient(s), or a test-retest (in case of a single reviewer) P: The authors mention preventive actions, e.g., piloting of the review steps.</td>
</tr>
</tbody>
</table>

Continued on next page
4.7 Conclusions

Table 4.18 – Continued from previous page

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Score**</th>
</tr>
</thead>
<tbody>
<tr>
<td>C6</td>
<td>Were adequate details presented for each of the primary studies?***</td>
</tr>
<tr>
<td>Y: Information is presented about each paper, so that the data summaries can clearly be traced to relevant papers. E.g., we can trace the research methods, information about study context, or collected measures to individual primary studies.</td>
<td></td>
</tr>
<tr>
<td>P: Only summary information is presented about individual papers. E.g., papers are grouped into categories, but it is not possible to link individual studies to each category.</td>
<td></td>
</tr>
<tr>
<td>N: Information/results cannot be traced to individual primary studies.</td>
<td></td>
</tr>
</tbody>
</table>

C7a | Were the evidences actually synthesized and aggregated, or merely summarized? |
| Y: The synthesis pools the studies in a meaningful and appropriate way. Differences between studies are addressed. |
| P: A partial synthesis/aggregation of evidence is provided. |
| N: There is no real synthesis. The evidences from the individual studies are basically only repeated/summarized. |

C7b | Were the strengths of evidence of individual studies taken into account in the synthesis? |
| Y: Yes, to some extent. |
| N: No. |

C8 | Do the authors’ conclusions accurately reflect the evidence that was reviewed? |
| Y: The conclusions are clearly supported by the provided data/results. |
| N: The conclusions are not well supported by the provided data/results. |

* Criteria C1–C8 are adopted from the Centre for Reviews and Dissemination (2009).
** Y(es)=1; P(artially)=0.5; N(0)=0.
*** Details adopted from Kitchenham et al. (2010).

Appendix B: Aims of Secondary Studies

Table 4.19: Aims of Secondary Studies

<table>
<thead>
<tr>
<th>Type</th>
<th>Ref</th>
<th>Aim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classify Primary Studies</td>
<td>S3</td>
<td>Collect evidence about how Agile Software Product Line (SPL) research is structured, synthesize current evidence on the integration of both approaches and identify further challenges for the integration of Agile methods and SPL.</td>
</tr>
<tr>
<td></td>
<td>S9</td>
<td>Identify and classify state of the art in large-scale software development influenced by Lean Product Development (LPD) approaches.</td>
</tr>
<tr>
<td></td>
<td>S19</td>
<td>Analyze agile practices in order to explore their industrial usage with respect to their distribution over different domains and processes from the perspective of software engineers.</td>
</tr>
<tr>
<td></td>
<td>S21</td>
<td>Investigate the state of the art on how standardized model like Agile can work with Model Driven Development (MDD)</td>
</tr>
<tr>
<td></td>
<td>S24</td>
<td>Identify relevant research and understand what the field of Agile/User Experience (UX) looks like at present.</td>
</tr>
</tbody>
</table>

Continued on next page
## Table 4.19 – Continued from previous page

<table>
<thead>
<tr>
<th>Category</th>
<th>Ref</th>
<th>Aim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find association</td>
<td>S2</td>
<td>Investigate potential factors that are limiting the industrial adoption of Test Driven Development (TDD).</td>
</tr>
<tr>
<td></td>
<td>S7</td>
<td>Investigates impact of TDD on different variables whilst taking two study quality dimension into account, namely rigor and relevance.</td>
</tr>
<tr>
<td></td>
<td>S10</td>
<td>Evaluate according to the ISO/IEC 12207 and ISO/IEC 9126 standards, synthesize, and present, the empirical findings on quality in agile methods.</td>
</tr>
<tr>
<td></td>
<td>S11</td>
<td>Investigate the impact of TDD on external code quality and productivity.</td>
</tr>
<tr>
<td></td>
<td>S26</td>
<td>Gather and synthesize empirical evidence to provide convincing and illuminating support for software project managers who need to make informed choices about software development approaches for their projects, with respect to cost, duration and quality</td>
</tr>
<tr>
<td>Identify Factors</td>
<td>S4</td>
<td>Collect evidence on benefits and limitations of agile software development, understand the strength of evidence of the findings, practical and research implications</td>
</tr>
<tr>
<td></td>
<td>S5</td>
<td>Assess the relationship between Agile Software Development (ASD) and Open Source Software Development (OSSD).</td>
</tr>
<tr>
<td></td>
<td>S6</td>
<td>Characterize reconciliation among the plan-driven, agile, and free/open source software models of software development.</td>
</tr>
<tr>
<td></td>
<td>S8</td>
<td>Identify the origin, prevalence, benefits, enablers, and problems of rapid releases.</td>
</tr>
<tr>
<td></td>
<td>S12</td>
<td>Evaluate, synthesize, and present the existing findings that will give the latest state of research on applying agile software development method to embedded software development.</td>
</tr>
<tr>
<td></td>
<td>S13</td>
<td>Present a detailed view about uses of reference architectures in agile methodologies</td>
</tr>
<tr>
<td></td>
<td>S14</td>
<td>Identify the software practices usually used into the context of agile approaches for the development of software.</td>
</tr>
<tr>
<td></td>
<td>S15</td>
<td>Provide insight into the Kanban approach and its elements (concepts, principles, practices, techniques, and tools) that have been empirically reported by scholars and practitioners.</td>
</tr>
<tr>
<td></td>
<td>S16</td>
<td>Identify the characteristics of agile project management in organizations using agile methods and maturity models; regarding support approaches employed; from the viewpoint of researchers; in academic and industrial context.</td>
</tr>
<tr>
<td></td>
<td>S17</td>
<td>Provide empirical support for a proposal of a methodology for integration of User Centered Design (UCD) and Agile, identifying most common practices and artefacts used.</td>
</tr>
<tr>
<td></td>
<td>S18</td>
<td>Identify what barriers have been dealt with, and what challenges have to be addressed in the near future to apply Agile Product Line Engineering (APLE) to the software industry.</td>
</tr>
<tr>
<td></td>
<td>S20</td>
<td>Find out to what experiences there are of Model-driven Agile Development (MAD), from an empirical context.</td>
</tr>
<tr>
<td></td>
<td>S22</td>
<td>Identify, synthesize, and present the findings reported about using Scrum practices in Global Software Development (GSD) to date.</td>
</tr>
</tbody>
</table>

*Continued on next page*
### 4.7 Conclusions

<table>
<thead>
<tr>
<th>Category</th>
<th>Ref</th>
<th>Aim</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S23</td>
<td>Systematically reviewing and summarizing the existing research literature and investigating which Agile practices have been used effectively in Global Software Engineering (GSE) contexts.</td>
</tr>
<tr>
<td></td>
<td>S28</td>
<td>Present a detailed view of how agile methods have been used in the development of embedded systems, and to describe their benefits, challenges, and limitations.</td>
</tr>
<tr>
<td></td>
<td>S29</td>
<td>Identify how Agile create business values</td>
</tr>
<tr>
<td></td>
<td>S30</td>
<td>Identify various challenging factors that restrict Agile and User Centred Design Integration (AUCDI) and explore the proposed practices to deal with them.</td>
</tr>
<tr>
<td></td>
<td>S31</td>
<td>Gain a comprehensive understanding of the various factors that impact the sustained usage of agile methods.</td>
</tr>
<tr>
<td></td>
<td>S33</td>
<td>Evaluate, synthesize, and present results on the use of the Capability Maturity Model Integration (CMMI) in combination with agile software development.</td>
</tr>
<tr>
<td></td>
<td>S35</td>
<td>Examine industrial surveys published in 2011 and 2012, determine the extent to which we could trust their reported high rates of agile method usage and provide recommendations on how quality of research could be improved in the future.</td>
</tr>
<tr>
<td></td>
<td>S37</td>
<td>Provide an overview of studies within knowledge management in agile projects, what kind of concepts have been explored, what the main findings were and what were the research method</td>
</tr>
<tr>
<td></td>
<td>S38</td>
<td>Amass current knowledge about agile adoption and to identify essential future research issues for empirical studies, especially on agile in the large settings.</td>
</tr>
<tr>
<td></td>
<td>S39</td>
<td>Identify technical and organizational challenges of Service Oriented Architecture (SOA) and cloud in Agile context</td>
</tr>
<tr>
<td>Survey the literature</td>
<td>S1</td>
<td>Investigate the status of Kanban in software development, in terms of its presence in existing literature.</td>
</tr>
<tr>
<td></td>
<td>S25</td>
<td>Review the literature of actual use of software metrics in the context of agile software development.</td>
</tr>
<tr>
<td></td>
<td>S27</td>
<td>Review the current research literature on effort estimation in agile, iterative and incremental software projects (AIISPs) and evidences about common trends and gaps</td>
</tr>
<tr>
<td></td>
<td>S31</td>
<td>Provide a synthesis of relevant studies in this issue in order to identify the scope of current research and the shortcomings that need to be addressed in the future.</td>
</tr>
<tr>
<td></td>
<td>S34</td>
<td>Investigate the extent to which agile practices have been used in scientific software projects. Second, we aim to investigate the impact on testing and requirements activities in projects with agile practices.</td>
</tr>
<tr>
<td></td>
<td>S36</td>
<td>Provide a detailed overview of the state of the art in the area of effort estimation in ASD.</td>
</tr>
<tr>
<td></td>
<td>S40</td>
<td>Evaluates the potential of agile methods and techniques to address the challenges of Model-Driven Modernization.</td>
</tr>
<tr>
<td></td>
<td>S41</td>
<td>Identify testing related problems in the context of the automotive software domain and solutions that have been proposed and applied in an industrial context.</td>
</tr>
</tbody>
</table>
The Impacts of Agile and Lean Practices on Project Constraints: A Tertiary Study
Bibliography


Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J.,


ABSTRACT
Context: Flexibility is an important capability for a software organization. Without flexibility a software organization risks losing its competitive advantage. To build software organization flexibility every constituent of the organization needs to be taken into account. Otherwise there are unforeseen trade-offs that could have negative impacts on the rest of the organization. Agile and Lean methodologies are known as approaches to build flexibility. However, currently Agile and Lean implementations are studied at project, and not at organization level. There is a need to understand how to build software organization flexibility.

Objective: The aim of this licentiate thesis to understand the challenges that a software organization faces with respect to flexibility. Particularly in understanding the approaches of building software organization flexibility and the associated trade-offs.

Method: A grounded theory study and a tertiary study were performed as part of this licentiate thesis. The grounded theory study was conducted to gain a better understanding of the challenges and processes in building software organization flexibility. The data were collected from an IT Department of a Fortune 500 financial institution. A tertiary study was performed to identify empirically evaluated Agile and Lean practices and their respective impacts. The findings from the tertiary study were synthesized using meta-study.

Results: The findings in this thesis uncovered a number of challenges that a software organization faces with respect to flexibility. They include budget cuts, overhead due to inconsistent development process, and regulatory changes. These challenges then caused uncertainties that impede the organization’s operational efficiency, cause delays, and inefficient use of resources. To cope with the uncertainty, a software organization would build its flexibility through modifications of its organization constituents. Processes and trade-offs associated with achieving flexibility were also identified. This thesis also identified 13 Agile and Lean practices and their associated impacts on scope, quality, budget, schedule, etc. A consolidated view of the impacts of Agile and Lean practices and their empirical support is also provided. Furthermore, this licentiate thesis also highlighted the importance of synthesis method in a secondary study. A secondary study with unclear synthesis method could yield different results from a secondary study with a clear synthesis method, e.g., vote counting, meta-analysis, or meta-ethnography.

Conclusion: With the challenges that a software organization faces, building software organization flexibility is becoming more prevalent. To improve software organization flexibility different constituents of the organization need to be considered. Otherwise, the trade-offs associated to achieving flexibility cannot be thoroughly considered. Furthermore, Agile and Lean practices can have positive, negative, or no impacts on quality, budget, schedule, etc. The findings of this thesis can help practitioners identify flexibility needs, as well as improve their awareness of possible negative trade-offs when building software organization flexibility.

Keywords: Flexibility, Software Organizations, Agile, Lean.