PONTIFICAL CATHOLIC UNIVERSITY OF RIO GRANDE DO SUL
FACULTY OF INFORMATICS
COMPUTER SCIENCE GRADUATE PROGRAM

CHARACTERIZING THE PRESENCE OF AGILITY IN LARGE-SCALE AGILE SOFTWARE DEVELOPMENT

GREICE DE CARLI ROMAN

Thesis submitted to the Pontifical Catholic University of Rio Grande do Sul in partial fullfillment of the requirements for the degree of Master in Computer Science.

Advisor: Prof. Dr. Sabrina dos Santos Marczak
Co-Advisor: Prof. Dr. Josiane Kroll

Porto Alegre
2016

Ficha Catalográfica

R758c Roman, Greice De Carli

Characterizing the Presence of Agility in Large-Scale Agile Software Development / Greice De Carli Roman . – 2016.
144 f.
Dissertação (Mestrado) – Programa de Pós-Graduação em Ciência da Computação, PUCRS.

Orientadora: Profa. Dra. Sabrina dos Santos Marczak.
Co-orientadora: Profa. Dra. Josiane Kroll.

1. Large-Scale Agile Development. 2. Agile Software Development. 3. Presence of Agility. 4. Empirical Study. I. Marczak, Sabrina dos Santos. II. Kroll, Josiane. III. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica da PUCRS com os dados fornecidos pelo(a) autor(a).

Pontifícia Universidade Católica do Rio Grande do Sul
FACULDADE DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

TERMO DE APRESENTAÇÃO DE DISSERTAÇÃO DE MESTRADO

Dissertação intitulada "Characterizing the Presence of Agility in Large-scale Agile Software Development" apresentada por Greice De Carli Roman como parte dos requisitos para obtenção do grau de Mestre em Ciência da Computação, aprovada em 15 de dezembro de 2016 pela Comissão Examinadora:

Dra. Sabrina dos Santos Marczak — PPGCC/PUCRS
Orientadora

Dra. Josiane Kroll — PNPD/PPGCC
Coorientadora

Dr. Avelino Francisco Zorzo — PPGCC/PUCRS

Dra. Rafaela Mantovani Fontana — UFPR

Homologada em...../...../....., conforme Ata No. ..... pela Comissão Coordenadora.

Prof. Dr. Luiz Gustavo Leão Fernandes
Coordenador.

Campus Central
Av. Ipiranga, 6681 – P. 32 – sala 507 – CEP: 90619-900
Fone: (51) 3320-3611 – Fax (51) 3320–3621
E-mail: ppgcc@pucrs.br
www.pucrs.br/facin/pos

“The important thing is not to stop questioning. Curiosity has its own reason for existing.”
(Albert Einstein)

ACKNOWLEDGMENTS

A Deus por estar sempre ao meu lado, me guiando para conquistar os meus objetivos.

Aos meus pais, Leonor e José Roman, pelo amor, dedicação, esforço e suporte dados para a realização deste sonho.

À Rita, Vítor e Luiza, por serem meus anjos, por estarem sempre comigo em todos os momentos, por me apoiarem e entenderem as incontáveis vezes em que tive que me dedicar aos estudos.

A todos meus Professores do PPGCC, em especial à Profa. Sabrina, Post-doc Josiane, Profa. Alessandra, Prof. Rafael e Prof. Avelino. Vocês ajudaram nos momentos em que mais precisei de apoio e conselhos. Todo meu conhecimento sobre pesquisa vem do aprendizado que obtive com vocês.

À Rafa pelas dicas, ideias, atenção sobre o meu estudo.

Aos meus queridos colegas: Carol, Bernardo, Elisa, Mateus, Cris, Lauri, Flavio, Will, Gabriel, Luis, Thiago, Marina, Júlia, Leti. vocês são demais, levo-os no coração.

A todos os participantes que contribuíram para a construção deste estudo.

CHARACTERIZING THE PRESENCE OF AGILITY IN LARGE-SCALE AGILE SOFTWARE DEVELOPMENT

RESUMO

Em fevereiro de 2001, o Manifesto Ágil foi proposto tendo como princípio equipes pequenas e co-localizadas. No entanto, ao longo destes 16 anos, a agilidade também foi posta em prática em outros contextos, como por exemplo: equipes distribuídas e sistemas complexos, utilizando-se o termo "Desenvolvimento Ágil em Larga Escala". Não há uma definição clara e compreensiva de como a agilidade está presente neste contexto. Assim, nosso trabalho preenche essa lacuna com o objetivo de caracterizar a agilidade no Desenvolvimento Ágil em Larga Escala. Neste trabalho, realizou-se um estudo organizado em duas fases. Na Fase 1, denominada Base Teórica, realizamos um estudo do estado-da-arte da área. Na Fase 2, denominado Estudo Empírico, nós realizamos duas investigações: um estudo de campo em uma empresa ágil em larga escala, para identificar o desenvolvimento durante o processo de transformação da empresa para esta nova abordagem e, um grupo focal, para identificar como as equipes ágeis em larga escala que vêm utilizando os métodos ágeis o quanto se percebem em termos de aspectos de maturidade ágil. Estes resultados contribuem para os pesquisadores e profissionais entenderem melhor como a agilidade é definida e percebida nestes grandes ambientes. O conhecimento é útil para aqueles que querem entender como o desenvolvimento ágil se adapta a tais ambientes e para pesquisadores com o objetivo de se aprofundar sobre o tema.

Palavras-Chave: Desenvolvimento Ágil em larga-escala, Desenvolvimento de Software Ágil, Agilidade, Estudo Empírico.

CHARACTERIZING THE PRESENCE OF AGILITY IN LARGE-SCALE AGILE SOFTWARE DEVELOPMENT

ABSTRACT

The Agile Manifesto was proposed in February 2001 having in mind small and collocated teams. However, agile has also been put in practice in other settings (e.g. large teams, distributed teams, complex systems) under the term 'Large-Scale Agile Development' (LSAD). There is no clear definition for and understanding of how agility is present in this setting. Thus, our work fills in this gap aiming to characterize agility in LSAD. We conducted a study organized in two phases. In Phase 1, named Theoretical Base, we conducted the state-of-the-art of the area. In Phase 2, named Empirical Study, we conducted two investigations: a field study in a large-scale agile company to identify how agility was developed during the transformation process of the company to this new approach, and a focus group to identify how large-scale agile teams that have been using agile for a certain while perceive themselves in terms of maturity in agile aspects. Findings contribute to researchers and professionals better understand how agility is defined and perceived in large settings. This knowledge is useful for those who want to enter the agile journey in such similar environments and for researchers aiming to further explore the topic.

Keywords: Large-Scale Agile Development, Agile Software Development, Presence of Agility, Empirical Study.

LIST OF FIGURES

LIST OF TABLES

CONTENTS

1. INTRODUCTION

Agile development was proposed as an alternative to traditional software development aiming to help teams to better respond to unpredictability through incremental, iterative work that values individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan [13]. However, the Agile Manifesto [13] and the subsequent methods derived from its principles were designed to support small, co-located teams, allowing people to collaborate and coordinate work face to face, facilitating communication. The software industry has changed since them, and nowadays it is common to have agile large projects developed by large teams distributed over the globe.

To confirm this trend, VersionOne's Annual State of Agile Survey (2015) [77] [1] reported that nearly two-thirds of the respondents in their survey said they worked for software organizations with more than hundred people, and 31% stated that they worked for software organizations with more than thousand people. The number of large enterprises that are embracing agile continues to increase each year. For instance, more than 24% of the respondents worked for organizations with over twenty thousand employees, compared to 21% in 2014. These results are confirmed in literature reporting cases of agile adoption from large organizations such as BMC (2006) [46], Amazon (2009) [9] [21], IBM (2010) [5], Ericsson (2013) [68], HP (2013) [117], among others.

Despite the benefits acquired with the agile adoption, including higher satisfaction, feeling of effectiveness, increased quality and transparency, increased autonomy and happiness, and earlier detection of defects [66], better team communication and customer collaboration [93], and self-organizing teams [15][59], the transformation is arduous and requires a lot of effort and the overcoming of challenges and barriers. For instance, Alzoubi, Gill, and Al-Ani (2016) [3] found six critical challenges categories when a distributed company adopted agile, namely: distance differences, team configuration, customer communication, project characteristics, organizational factors, and human factors.

There are also cases in which a large company with globally-distributed teams has failed in its scaled agile transformation (e.g., [88]); open questions about the main challenges related to requirements in large-scale agile software development [97], and recurring communication problems when teams were only geographically-distributed [3].

Literature reports practices to support and mitigate the main challenges and concerns teams face during an agile transformation in large settings. For instance, Larman and Vodde (2008 - 2010) [70] [71] and Schiel (2009) [100] wrote books based on their experiences with industry about how to conduct agile adoption on a large scale. They provide step by step guidance for leading an enterprise into the transformation. There are also

[1] VersionOne is a well-known worldwide consulting company in agile development.

frameworks on how to consider agile at the enterprise level. For instance, the Scaled Agile Framework [74], SAFe in short, integrates agile, the Lean philosophy, and product development flow thinking with a new operating model that successfully coordinates work at all levels: team, program, and portfolio. The Large-scale Scrum (LESS) framework [72] aims to offer the most direct, concise, actionable guide to reaping the full benefits of agile in distributed, global enterprises. The Disciplined Agile Delivery (DAD) framework [6], which focuses on light-weight guidance to help organizations streamline their IT processes in a context-sensitive manner, showing how the various activities such as solution delivery, operations, enterprise architecture, portfolio management, and many others work altogether. All frameworks above have been defined by practitioners or experienced consulting professionals.

Although the availability frameworks above, being an 'agilist' involves more than merely adopting practices or frameworks. Teams and organizations need to acquire agile mindsets, master new behaviors and change a culture to follow better the twelve principles proposed in the Agile Manifesto [13]. These characteristics are abstract and can be difficult to be incorporated by a team. For instance, the study of Eklund, Olsson, and Strøm (2014) [37] identify as an open issue the translation of agile ideas (manifesto and principles) in its large-scale agile team in a context of mass-production. On the other hand, Lagerberg et al.(2013) [68] found positive impacts with the usage of agile principles on two large-scale projects at Ericsson. The findings of Lagerberg et al. are related to improved knowledge sharing, increased project visibility and productivity, and coordination effectiveness. These two studies confront an open question mentioned by Reifer, Maurer and Erdogmus (2003) [96]: "How scaling agile without sacrificing the underlying principles of the Agile Manifesto?"

Despite the popularity of the topic, the LSAD field is incipient and there is no clear characterization of definitions, settings and current challenges. Many studies report on the term LSAD identifying a large company in an agile transformation [61]. Other studies have reported that large scale agile is a term defined for large projects [38], or for large and distributed teams [87] [55]. Other studies go further and include collocated teams using agile inside large companies [40] [78].

Those dispersed information may mean a great effort for researchers and practitioners to identify the applicability of the available studies and to report and search studies of an empirical nature. Also, we realized that the presence of agility in such domain needs to be comprehensive and have a consolidated body of knowledge to be a resourceful source of use by both industry and academia professionals.

1.1 Research Question and Objectives

The main goal of this research is to characterize agility in LSAD, which includes identifying the settings in which it takes place, strategies taken, and the challenges faced by a company and teams to be agile in this scenario.

In order to achieve our goal, the following objectives were defined:

- **(Obj1).** To identify the state-of-art of LSAD in literature. (Study 1).
- **(Obj2).** To empirically identify how a large-scale agile company goes through the transformation of becoming agile. (Study 2).
- **(Obj3).** To empirically identify whether agility is present in large-scale agile teams and the challenges faced by such teams to be agile in such setting (Study 3).

1.2 Contributions

The contributions are related to each of the three studies conducted in this research, as presented next.

From the Study 1, the systematic mapping review in Objective 1, we identified the state-of-art of LSAD research including definitions, settings involved, and challenges in this context. Our contribution with the systematic mapping review showed challenges faced in LSAD, and the importance of scaling agile practices to support LSAD. Furthermore, it was possible to characterize LSAD.

From the Study 2, the field study in a large-globally company in Objective 2, we identified lessons learned about how achieved agility in such scenario. We also identified the reasons and strategies taken by this company to adopted agile in their processes. Furthermore, this study has shown significant challenges and gaps from the literature to be investigated in future works.

From the Study 3, the focus group study were conducted from two different companies in Objective 3, revealing that agility persists at different levels within the agile maturity. However, we cannot generalize the results of the presence of agility in LSAD. Thus, new studies can be conducted to build new theories on the question presented.

Overall, from the software engineering perspective, this work contributed to the understanding of definitions, the presence of agility, project's settings, practices, and challenges faced in LSAD. From the software industry perspective, this work contributes to providing new information about LSAD characteristics, the strategies taken by a company and teams to adopt LSAD in their work processes, and how to agility agility in large settings.

1.3 Structure of this Dissertation Thesis

The remainder of the Master Thesis is organized as outlined next. Chapter 2 presents the theoretical background of the main topics discussed this Master Thesis. Chapter 3 introduces the research methodology followed in this research. Chapter 4 presents results from a systematic mapping review on LSAD. Chapter 5 and Chapter 6 present results from our empirical study phase which is comprised of a field study in LSAD (Chapter 5) and a focus group study (Chapter 6). Chapter 7 concludes the work by revisiting the posed research objectives, the main contributions of this Master Thesis, limitations of this research, and future studies.

2. THEORETICAL FOUNDATION

This chapter presents the theoretical background related to this work. Section 2.1 reviews the agile software development concept to build the foundation required for this work. This section is organized as follows: Section 2.1.1 introduces the Scrum method, Section 2.1.2 describes the Extreme Programming method, Section 2.1.3 presents the Kanban method, and Section 2.1.4 introduces the philosophy behind agility. Section 2.2 briefly describes the LSAD concept. Section 2.3 describes the related work of this Master Thesis.

2.1 Agile Software Development

Agile software development arose in the mid-1990s [34] with the purpose to attend the need to define a new way to produce software in a scenario driven by changes, in which traditional methods such as the Waterfall were not satisfiable any longer. Agile software development is based on releases and faster delivery to the customer, as defined by Sommerville (2011) [110].

Agile methods such as Scrum [101] and Extreme Programming (XP) [12] are based on the Agile Manifesto [13], proposed based on empirical knowledge of 17 renowned developers and software business consultants such as Martin Fowler and Kent Beck. The Manifesto was based on four values, as follows:

- Individuals and interactions over processes and tools;
- Working software over comprehensive documentation;
- Customer collaboration over contract negotiation;
- Responding to change over following a plan.

According to the Manifesto, while there is value in the items on the right, the items on the left are valued more. To better understand the value of agile software development, after the Manifesto, twelve fundamental principles were added [13], as follows:

- The highest priority is to satisfy the customer through early and continuous delivery of valuable software;
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage;
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale;

- Business people and developers, must work together daily throughout the project;
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done;
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation;
- Working software is the primary measure of progress;
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely;
- Continuous attention to technical excellence and good design enhances agility;
- Simplicity–the art of maximizing the amount of work not done–is essential;
- The best architectures, requirements, and designs emerge from self-organizing teams;
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Agile software development has individuals values and principles going beyond of a vision of following a software development process. In essence, agile methods promote values and practices to achieve the term 'agility', which refers to the culture and behavior of those developing software [23].

The values are based, mostly, on the interaction between individuals to share the knowledge to produce software with quality in a short time (releases), which in turn requires customer collaboration and frequent communication among the team members. This regular interaction between individuals does not require to have a formal documentation as prescribed in traditional development [22].

2.1.1 Scrum

Scrum provides practices for developing software represented by self-organization, management of empirical processes, and knowledge creation. Schwaber and Sutherland (2013) [115] define Scrum as a framework where people can address complex adaptive problems productively and creatively delivering products of the highest possible value.

The Scrum team consists of a Product Owner, the Development Team, and a Scrum Master. Each responsibility of a Scrum Team Member according to Schwaber and Beedle (2001) [102] and Schwaber and Sutherland (2013) [115] is:

- Product Owner: The person responsible for the project, by having to manage, control, and make the product backlog visible. The product owner makes the final decision of the tasks related to the product backlog and turns issues in the backlog into features to be developed.
- Scrum Master: The person that helps everyone in the Scrum Team to understand and embrace the values, principles, and practices of Scrum. This role acts as a coach, leading the process and helping the Scrum Team and the entire organization to develop its high performance.
- Development Team: Development team is structured and empowered by the organization to organize and manage their work. Such as a team is responsible for turning the product backlog into increments of potentially releasable functionality during the period of a work release.

Scrum events and artifacts are specifically designed to enable critical transparency and inspection as described by Schwaber and Sutherland (2013) [115] and Rubin (2013) [99]:

- Sprint: In Scrum, the work is performed in iterations or cycles of approximately one month of time-boxed, named sprint. The work done in each sprint should create a product of tangible value to the customer. Once the Sprint begins, its duration is fixed and cannot be reduced or increased. A new sprint starts immediately after the last sprint has finished. As a rule, no change in scope is allowed during a sprint.
- Sprint Retrospective: The sprint retrospective is an opportunity for the Scrum Team to inspect itself and create a plan for improvements to be enacted during the next sprint.
- Sprint Planning: The work to be performed in the sprint is planned at the sprint planning. This plan is created by the collaborative work of the entire scrum team. Sprint planning is time-boxed to a maximum of eight hours for a one-month sprint.
- Daily Scrum: Is a 15-minute time-boxed event for the development team to synchronize activities and create a plan for the next 24 hours.
- Sprint Review: A sprint review is performed at the end of the sprint to inspect the increment and adapt the product backlog if needed. During the sprint review, the scrum team and stakeholders collaborate about what was done in the sprint.
- Product Backlog: The product backlog is an ordered list of everything that might be needed in the product and is the single source of requirements for any changes to be made to the product. The product owner is responsible for the product backlog, including its content, availability, and ordering.

- Sprint Backlog: The sprint backlog is the set of product backlog items selected for the sprint, plus a plan for delivering the product Increment and realizing the sprint goal.
- Increment: The increment is the sum of all the product backlog items completed during a sprint and the value of the increases of all previous Sprints.

2.1.2 Extreme Programming

XP is an agile and flexible way to manage changes in the software development scenario [12]. The method was developed for small to medium-sized teams (10 people in the same environment). The team is prepared to take the principles and empirical practices to extreme levels [12]. In the XP method, requirements are identified through "user stories" and implemented as a "series of tasks." Programmers work in pairs and develop tests even before the starts to developed code. There is a small gap between system releases, and the integration is performed after the new code insertion, which necessarily must have all completed tests [110].

Beck (2000) [12] stares that the practices are simple and fundamental solutions to follow the XP method. The following list presents a quick summary of each major practices in XP as explained by Beck and Fowler (2001) [14] and Beck (2000) [12]:

- The Planning Game: The customer decides the scope and timing of releases based on estimates provided by programmers.
- Small releases: The system is put into production before solving the whole problem. New releases are often made.
- Stand-up Meeting: The development team meets every day to evaluate the work that was performed in the day before and prioritize what will be implemented for the next day.
- Metaphor: Each software project in XP is guided by a single comprehensive metaphor. The metaphor helps everyone involved in the project to understand the essential elements and their relationships.
- Simple Design: A simple design where there is no duplicated logic, tests are successful, and that all important intentions for the development team and customer are expressed clearly in the project.
- Tests: Programmers write unit tests which are collected and must run correctly. The customer writes functional tests for the stories in an iteration.

- Refactoring: Is the action to change a code due to duplicate code or poor cohesion in writing implementation without affecting the functionality it implements.
- Collective Code Ownership: Every developer improves code anywhere in the system at any time if they see the opportunity.
- Pair Programming: In XP method, developers implement the system in pairs, i.e., there are always two developers to produce the same code.
- Continuous Integration. The system is integrated and tested after a few hours, at most one day of development.
- On-site customer: A customer sits with the team full-time.
- 40-hour weeks: No one can work a overtime for two consecutive weeks.
- Open workspace: The team works in a large room with small cubicles.
- Coding standards: XP teams follow a standard of coding. Thus the whole system code looks like it was written by a single - very competent - individual.

2.1.3 Kanban

The Kanban method is an implementation of Lean development principles. Its focus involves the flow and in the context of what will be developed, offering a less prescriptive approach compared to other agile methods [8]. Kanban is not only a process used for software development but the word 'Kanban' is also used to describe the system that has been used for decades by Toyota to control and balance the production line visually. The concept of Kanban is relatively new to the software industry. However, it has been used for over 50 years in the Lean production system at Toyota [7]. The first virtual Kanban system for software engineering was implemented in Microsoft in 2006, but it was David J. Anderson and Don Reinerstsen who expanded the knowledge of the use of Kanban to view and optimize the workflow in software development, as showed in the Figure 2.1. The Figure illustrates a well-known resourced used by agile practitioners, the Kanban Board [7].

Kanban is a framework for improvement, highlighting that work process should be customized for each team in each project [8]. The Kanban method limits the work in progress of a team to define the capacity and to demand balance about the performance of the work that will be delivered. By applying this idea one can achieve a sustainable development path, which reaches a balance between work and personal life of each member belonging to the team. Kanban also provides visibility to the problems that affect the performance and workflow of a team making it easier to eliminate these problems that compromise project performance [7].

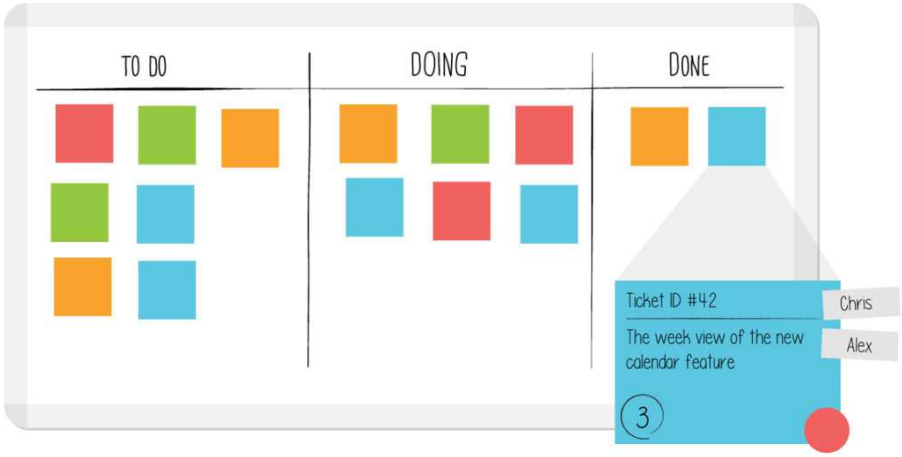


Figure 2.1: A physical Kanban board with a basic, three-step workflow. Source: LeanKit (2015) [73]

2.1.4 Characterization of Agility

Although there are many publications about agile software development (both state-of-practice and state-of-art e.g., [24, 39, 35, 32]) since the creation of the Agile Manifesto in 2001 [13], the existing literature suffers from a lack of clarity about to what comprises "agility" [104]. For instance, Conboy (2009) [25] underlines this gap occurs given that each study (in systems information) adopts a unique interpretation of what agility means. On the other hand, Lyytinen and Rose (2006) [76] consider that agility is related to culture and project environment.

Linked to culture and project, Cockburn (2001) [23] and Highsmith (2002) [54] reinforce the definition of agility as *"the ability to rapidly and flexibly create and respond to change in the business and technical domains"*. Erickson et al. (2005) [39] associate such statement with a quick change in user requirements and accelerated project deadlines.

While there are several considerations on the agility meaning in agile software development studies, we sought (1) to characterize its definition and (2) to identify its presence during work processes. We identified in literature (from our study in Objective 1) three empirical studies that disscuss the concept of agility in traditional [1]agile software development. The first one is a comparison of agility in agile organizations and agile teams [34], the second is a conception of maturity in agile teams [42], and the third is agility characteristics and agile practices for software processes [2].

- Doyle et al. (2014) [34]

The first study concerns the Comparative Agility (CA) [34]. It is a survey-based assessment tool, used by companies to compare their agility implementations to others. This

[1]We refer to the term traditional agile development to non-large scale agile settings.

survey approach assesses agility across eight dimensions (1) teamwork; (2) requirements; (3) planning; (4) technical practices; (5) quality; (6) culture; (7) knowledge creating; and (8) outcomes. Each statement of the survey is an agile practice for which the respondent indicates the truth of the statement about their team or company. For instance, *upfront planning is helpful without being excessive*. Team members leave planning meetings knowing what needs to be done and have confidence they can meet their commitments. Figure 2.2 presents an overview of the CA survey based assessment tool.

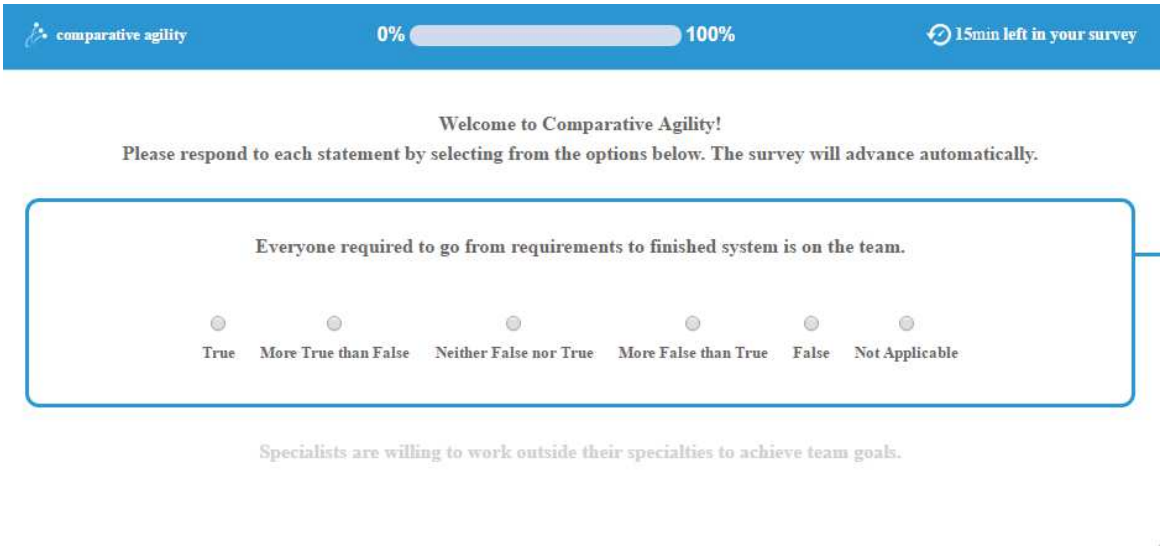


Figure 2.2: Comparative Agility Survey. Source: Doyle et al (2014) [34]

- Fontana et al. (2015) [42]

This study characterizes maturity in agile software development. Through the progressive outcomes, agile teams follow agility. This studies also describes ambidexterity as a key ability to maturity, which ambidexterity does not prescribe practices nor a process. It rather describes outcomes that agile teams pursue to improve their working processes. The complex adaptive systems theory was used to guide the framework definition of the empirical study conducted to evaluate the framework. The findings show that agility means fostering subjective capabilities, such as collaboration, communication, commitment, care, sharing, and self-organization. Figure 2.3 shows a Progressive Outcomes Framework that we used as "agility compass" [43]. Each line of this framework represents a category of outcomes. To facilitate our understanding, we translated the "outcomes" of each category as "characteristics".

The practices learning category presents the agility when the team decides to change the way of they work. In agile trial characteristic, the team tries to adopt agile practices, but, usually, they learn on the fly, without successful implementation of the practices. In the agile learning characteristic, the teams implement an agile method by the book. After, in the sense-making of the work processes characteristic the process included taking

Figure 2.3: Progressive Outcomes Framework. Source: Fontana et al. (2015) [42]

the method learned and tailored it to particular needs. By the end, comprehension of situation characteristic includes using tools to track the process, to have the team to report work status, to understand stories sizes, and using simple metrics.

The team conduct category describes the team behavior in the usage of agile. They start with a responsive characteristic, with practices that demand a leadership position of command and control. After, the team evolves to a confident team characteristic having the team members, starting to express their opinions about the decisions. Later, the assertive team characteristic is when the members are active voices in the project and promote the process improvement initiatives. In the end, a sparkling team characteristic is when the team is still assertive but also characterized by technical excellence, high performance, and motivation to continuous learning.

The deliveries pace category describes how outcomes for deliveries evolve. Teams start investing in iterations to control the coding process. The code is not delivered. It is kept for further testing and integration (expected frequently finished coding characteristic). In the next characteristic expected frequent deliverable the team implement processes that make this code ready for delivery, but would not be delivered yet due to many reasons. Then, to

actual deliveries at the end of the iterations – usually late deliveries. The team then, in the next characteristics, starts working on practices to have a defined delivery, that is, a delivery that is performed on time.

Features disclosure category describes outcomes teams pursue when describing the features the software will comprise. The requirements gathering characteristic represents a process of eliciting requirements similar to traditional software process. In the requirements discovery characteristic, the team improves the quality of the requirements to make sure they meet customer expectations. After, in the requirements quality characteristic, the team uses different techniques in requirements definition.

In the software product category, the team implements the practices to improve the software itself. First, in the awareness of failures characteristic the team perform agile practices to have a good resulting source code. The next characteristic is to focus on high-level delivered software, i.e., ensuring a good delivery. The efficient coding category, when practices such as integration, testing, and deploy automation are implemented.

The customer relationship category comprises when the team implements practices to improve its relationship with the customer. The first characteristic represents the team gaining awareness of the customer. Then, the second characteristic is when the customer gets acquainted with the team's capabilities and the agile work processes. The confident customer characteristic is when the deliveries are going to happen and what is going to be delivered. The partner customer is when the team helps define requirements and solutions for the customer's business problems.

The organizational support category describes the characteristics related to the company's position to agile transformation. When a company is in agile motion, there are isolated, and small, bottom-up initiatives start for agile adoption. After, in agile commitment, the top management starts supporting the agile adoption. Next, in agile priority the changes that change its structures, roles, and processes to enable the agile transformation. Finally, in the agile business characteristic the company is recognized for being agile.

We choose to use Fontana et al.'s (2015) [42] work as a reference, given their framework helps agile teams to identify agility stages. Also, different from the previous study (Doyle et al. [34]) the progressive outcomes framework does not compare the agility of different companies, but rather emphasizes the maturity characterization of the agile team. More details on how the framework was used in our study is presented in Chapter 3 research methodology and in Chapter 6 focus group study with large-scale agile teams.

- Abrantes and Travassos (2013) [2]

In this work, Abrantes and Travasssos (2013) [2] assessed the pertinence and relevance of agility characteristics and agile practices for software processes. Through a systematic literature review and a survey method, they identified 16 agility characteristics and 15 agile practices were considered relevant to agility in software processes, as follows:

- Characteristics of agility: Being collaborative, being cooperative, being incremental, adaptability, self-organization, being iterative, constant testing, emergency, feedback incorporation, leanness, modularity, people orientation, reflection and introspection, small teams, time-boxing, and transparency.
- Agile practices: Coding standards, collective code ownership, continuous integration, metaphor, on-site customer, planning game, product backlog, project visibility, refactoring, simple design, short releases, daily meetings, and sustainable pace.

2.2 Large-Scale Agile Development

LSAD is a generic term that includes large companies. From the perspective of size, literature reports the number of developers in a team, the number of teams, project size, systems and line codes, and also project duration [29]. For example, LSAD might be having more than two teams [33], a project size with more than 150 developers [117], part of a product consisting of 10-12 million lines of code, or 30 subsystems partly developed in Sweden and China [79].

Furthermore, some characteristics should be taken into consideration about the geographical distribution of teams, agile teams working on a large company with entrenched traditional culture, agile teams working on a system complexity requiring a viable architectural strategy, top management support.

Based on findings from our systematic mapping review (see Section 4), we defined LSAD to denote software development organizations with 50 or more people [29] and teams composed of more than one team [33]. People in the team do not need to be developers, but must belong to the same software development organization developing a standard product or project, and thus have a need to collaborate. The company can be in a traditional structure, but with agile initiatives. Chapter 4 details all definitions reported from literature, and also other characteristics that underlie the concept of LSAD.

2.3 Related Work

There are only a few empirical studies in literature discussing LSAD [29]. These studies are mainly focused on experiences reporting agile transformation, strategies, and challenges to adopt LSAD. Next, we present the related work to this Master Thesis, both to point out the contributions of previous research and to place our contributions in the proper context. First, we will present a paper from Dikert et al. (2016) [29] that identify the lack of description of has been followed in reported empirical studies.

- Dikert et al. (2016) [29]

The main contribution of this study was to identify the lack of description of what research method has been followed in reported empirical studies of LSAD. From the 55 studies analyzed in this study, only 10 of them reported this information. Also, for being the first systematic literature review on the topic, it included studies on LSAD those that remain more than 50 people working for the same company. Furthermore, challenges and success factors were analyzed and described. The study reported 29 success factors organized into eleven categories and 35 challenges grouped into nine categories (see Table 2.1). The challenges categories that received the most citations are difficulties to implement agile, integrating non-development functions, change resistance and, requirements engineering challenges. The success factor categories that received the most citations are: choosing and customizing the agile approach, management support, mind- set and alignment, and training and coaching.

Table 2.1: Success factors and challenges in LSAD transformations

ID	Success Factors	Challenges
1	Management support	Change resistance
2	Commitment to change	Lack of investment
3	Leadership	Difficulties to implement agile
4	Choosing and customizing the agile	Coordination challenges in multi-team
5	Piloting	Different approaches emerge in a multi-team
6	Training and Coaching	Hierarchical management
7	Engaging people	Requirements engineering challenges
8	Communication and transparency	Quality assurance challenges
9	Mindset and Alignment	Integrating non-development functions
10	Team autonomy	
11	Requirements management	

- Lagerberg et al. (2013) [68]

The purpose of this multiple-case study research was to contribute to empirical evidence on the impact of using agile principles and practices in LSAD projects at Ericsson. These findings were organized into seven areas: software documentation, knowledge sharing of teams, project visibility, pressure and stress, coordination, and productivity. Thus, the identified agile adoption effects in the studied projects were: (a) correlate with a more balanced use of internal software documentation, (b) contribute to knowledge sharing, (c) correlate with increased project visibility and coordination effectiveness, (d) reduce the need for other types of coordination mechanisms, and (e) possibly increased productivity. No correlation with an increase in pressure and stress was found.

- Paasivaara (2016) [90]

This study presented a case study on scaling Scrum in a large globally distributed software development project at Nokia. In this study the authors investigated *"How did the case*

project scale Scrum over multiple teams and sites?", growing from two collocated Scrum teams to 20 teams located in four countries with a total of 170 persons taking over agile approach at once. Moreover, they reported the challenges faced during this 2,5 year of scaling agile practices. They gathered data from 19 semi-structured interviews of project personnel from two sites. Interviewees comprised different roles including managers, architects, product owners, developers, and testers.

The project was highly successful from the business point of view, as an agile-enabled fast response to customer requirements. However, the project faced significant challenges in scaling Scrum despite attempts at adopting the LESS framework. This naturally leads to the question of whether the problems were due to inherent problems in the framework itself, or due to it being poorly implemented. The organization experimented with different ways of implementing scaling practices like implementing common sprint planning meetings, Scrum-of-Scrums meetings, common demos and common retrospectives, as well as scaling the product owner role.

- Dingsøyr and Moe (2014) [31]

This study is a revised agenda for LSAD emerged from workshop discussions on the topic of LSAD aiming to understand and establish a set of definitions and aspects for LSAD. Some definitions and characteristics described for LSAD in the workshop are listed below:

- Definition (1): Agile software development in over 50 developers, 1 or 1/2 million lines of code, or more than three sites / time zones;
- Definition (2): Agile software development in over 50 persons, over five teams, developing together the same product / project using agile method;
- Definition (3): Agile being applied to more than one team,
- Aspect (1) - Architecture: figuring out how work is coordinated. Architectural work supports the implementation of high-priority business features without risking excessive redesign later or requiring strong coordination between teams [83].
- Aspect (B) - Inter-team coordination: creating effective knowledge networks is essential due to the knowledge-intensive nature of software development [53];
- Aspect (C) - Portfolio management: providing continuous feedback from the portfolio to project levels enables the teams and project members to take decisions that are consistent with the goals of the large-scale agile portfolio [67];
- Aspect (D) - Scaling: describing the context for agility and scale is essential for understanding how to improve it LSAD is when a company would be in an agile thinking on all levels of the company [65].

- Fitzgerald et al. (2013) [41]

This paper presents a case study of how agile methods can be scaled to regulated environments (non-agile environments). The results of this study show that scaling agile development at a leading supplier of regulatory compliance management solutions for document and quality management, submissions management, and regulatory approval in the life sciences sector has worked very well. For example, compliance is more immediate and evident in real-time-continuous. Also, the concept of living traceability has been coined to reflect the end-to-end traceability that has been facilitated by the toolset that has been implemented to support agile development.

Thus, the assumption of incompatibility between agile methods and regulated environments is more accidental than essential. Also, the V-life cycle model frequently adopted in controlled environments appears compelling in that there is a clear sense of traceability between the levels of resting and the levels of analysis, design and coding activities. However, as pointed out by one of our practitioner interviewees: “Agile is a lot of small Vs” and the levels traceability can be clearly accomplished in the agile mode of development. According to the authors, these findings may provide right motivation to other organizations that wish to explore how they can benefit from adopting agile methods.

- Heikkilä et al. (2010) [53]

This case study was conducted in five release planning events and four retrospectives in two projects at a large F-Secure software company. First, the authors described how a release planning method was employed in these projects, identifying the benefits which the projects gained from the method, and analyze challenges in the cases and improvements made to the method during the case projects. The aimed of the method was release planning events where the whole project organization gathered to plan the next release. They identified ten benefits which included improved communication, transparency, dependency management and decision making. Also, nine challenges were revealed in the study, which included the lack of preparation and prioritization of requirements, unrealistic schedules, inadequate architectural planning, and the lack of an agile mindset.

The biggest improvements to the method were the introduction of frequent status checks and a big visible planning status board. The release planning process ameliorated many challenging characteristics of the release planning problem, but its efficiency was negatively affected by the performing organization that was in transition from a plan-driven to an agile development mindset. Even in this case the benefits clearly outweighed the challenges, and the method enabled the early identification of the issues in the project.

2.4 Chapter Summary

In this chapter, we presented concepts covered in this research in which included: Agile software development and LSAD. For agile software development, we identified that agility concept changes for each setting and depending on the local culture. However, for this Master Thesis, we have identified three relevant studies [42][34][2] covering the different definitions found on agility. We chose the Fontana et al. (2015) [42] study because it represents characteristics of agility in traditional agile teams. We aim to identify the presence of agility in large settings instead.

We identified that LSAD area is a generic term that includes large companies and can refer to different aspects of software development: the number of members in a team, the number of teams, project size, systems and line codes, and also project duration. This work adopts the definition of more than one software team working on software development organizations with 50 or more people or more than one software development team working on the same project, product or system. We chose this definition to empirically understand how a company and teams follow agility in LSAD.

3. RESEARCH METHODOLOGY

This chapter discusses the aspects related to the research methodology followed in this work. Section 3.1 provides a brief definition of each of the research methods adopted in this research. Section 3.2 describes the adopted research design.

3.1 Methodological Background

This research is exploratory. Yin (2013) [123] defines exploratory research as a problem which has not been studied previously on the topic under study. Moreover, it helps the researcher to identify new research questions and problems to be investigated.

Exploratory research enables the adoption of different research methods to perform a study. In this work as detailed in Section 3.2, we decided to use as primary research methods the folowing: systematic mapping review, field study, and focus group study. We summarize each research method next.

3.1.1 Systematic Mapping Review

Systematic mapping review is a comprehensive review of primary studies on a particular topic, which aims to identify what evidence is available on this subject. A mapping study provides a systematic and objective procedure for determining the nature and extent of the empirical study data that is available to answer a particular research question [19].

The analysis of results focuses on frequencies of publications for categories within a scheme. Thereby, the coverage of the research field can be determined. Different facets of the plan can also be combined to answer more specific research questions [94]. The process of a systematic mapping review is similar to a systematic literature review [56], but there are differences of the goal of each method. Table 3.1 depicts these differences [17] [56].

Table 3.1: Comparison of systematic mapping review and systematic literature review

Systematic Mapping Review	**Systematic Literature Review**
High level analysis	In-depth analysis
Categorization / classification by evidence	Analysis focused on comparison / description / critical
Performed when the topic is too broad (mapping the area)	Performed when the topic is more specific
It helps to identify topics of interest to conduct a systematic review	Supported by systematic mapping review

We performed a systematic mapping review. This systematic mapping review helped us to identify definitions for LSAD, the state-of-art of research in LSAD, settings involved, and challenges for its implementation. Furthermore, the empirical evidence from a systematic mapping review provides the information about research findings and terms and definitions used in the current study area.

3.1.2 Field Study

Field study in software engineering is a research method based on a recognition that software engineering is fundamentally a human activity [108]. To explore real practitioners as they solve real problem field studies are required.

To conduct a field study is necessary to use data collection techniques such as interviews, surveys, brainstorming or notes. The most data collection technique used in field studies are interviews (the technique used in this research), to understand general information on the process, personal knowledge, and others [108].

There are three types of categories for interviews [103]: structured, unstructured, and semi-structured. In structured interviews, the researcher has specific goals in which it has particular questions and closed scope. In unstructured interviews, on the other hand, the goal is to collect as much information as possible. This is used when one wants to join a broad data set on a topic. The semi-structured interviews, in turn, are comprise of closed and open questions, allowing the researcher to capture their specific topics of interest and also those that emerge from the conversation [118].

The type of interview defines the control that a researcher intends to have. In our study, we followed semi-structured interviews with open-ended questions. In Chapter 5 we present our research approach detailed.

3.1.3 Focus Group Study

The focus group is a carefully planned discussion, designed to obtain personal perceptions of group members in a defined area of research interest [60]. Morgan [81] describes it as a technique to collect data through group interaction on a specific topic. The interaction is the heart of the focus group [58]. This means that the researcher not only considers the direct answers to interview question but also the discussions among the participants [28].

The participants are typically in the number of 3 to 12 persons, and the discussion is guided and facilitated by a moderator-researcher, which follows a predefined ques-

tioning structure so that the discussion stays focused [60]. The group environment allows participants to build on the ideas of other participants, which increases the richness of the information gained [69]. Two or more focus groups increasing the chances of success [63].

In this study, we conducted a focus group method to identify the presence of agility in large-scale agile teams. We chose this because this would provide us with the richness of interactions among the participants in contrast to the individual opinion of a person. Through the promoted discussions and reach consensus among the teams themselves, we collected the perceived opinion of the team on the presence of agility in large-scale agile teams.

3.2 Research Design

The research design was organized in two major phases as shown in Figure 3.1: (1) Theoretical Base and (2) Empirical Work. Figure 3.1 summarizes the selected methods. Each phase is described in detail next.

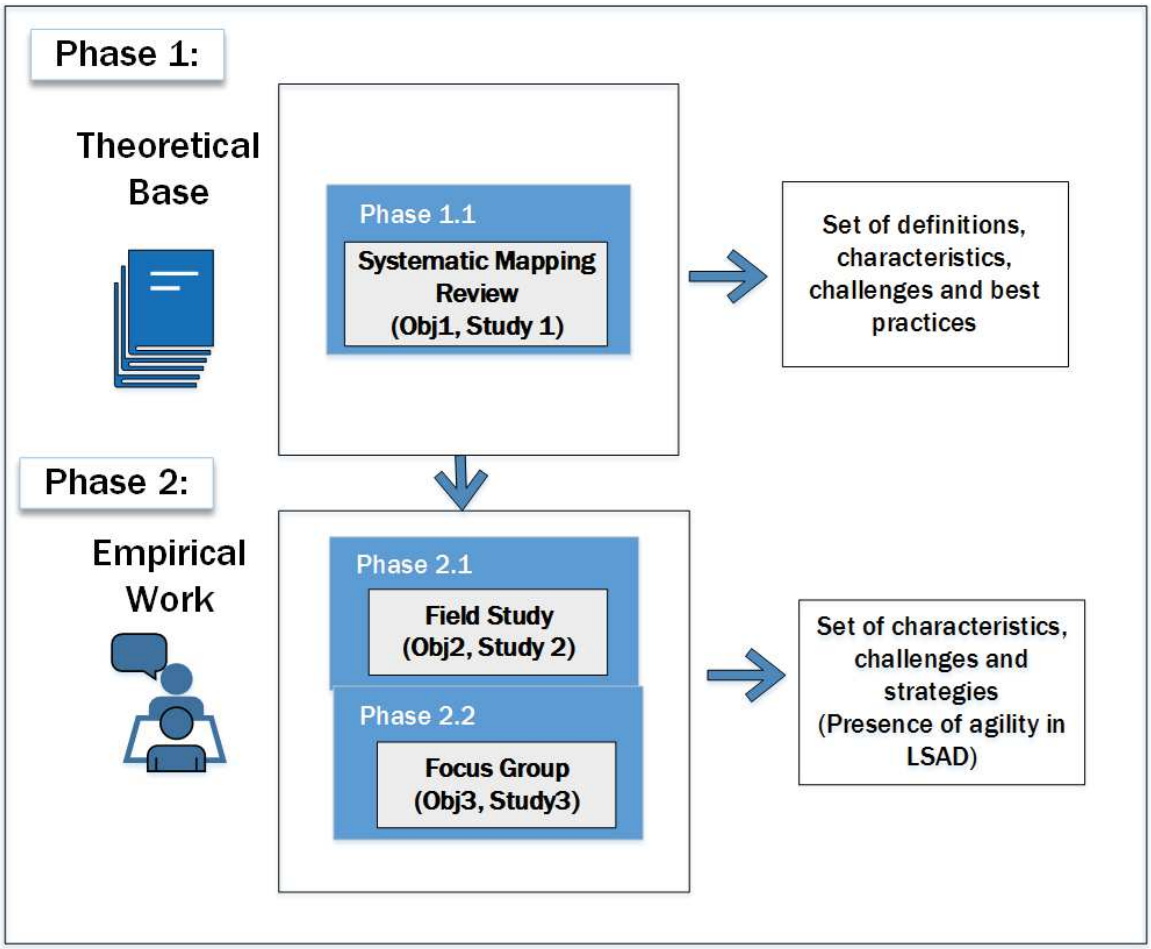


Figure 3.1: Research Design

3.2.1 Phase 1: Theoretical Base

On the Theoretical Base phase, we focused on getting familiar with the concepts of agile software development and large-scale agile development (LSAD) by skimming through literature. This informal literature review provides us enough knowledge to conduct the Systematic Mapping Review to identify the state-of-the-art on LSAD, which was the first study (Study 1) out of the three studies conducted in this research (Objective 1, Phase 1.1). More specifically, the Systematic Mapping Review aimed to identify (1) what are the definitions for LSAD, (2) in what settings LSAD takes place (agile practices adopted, project sizes, team sizes, enterprise backround, etc), and (3) what are the challenges faced by agile teams in such settings. The knowledge developed with this review served to guide the design of the studies in Phase 2. Details on how the Systematic Mapping Review was conducted is presented in Chapter 4.

3.2.2 Phase 2: Empirical Work

We conducted two empirical studies. The first (Study 2) was a field study that aimed to provide insights on the agile transformation of a large-globally distributed company (Objective 2, Phase 2.1). The investigated company falls into the definition of LSAD given its large teams, the software development organization with 50 or more people, and teams composed of more than one team. We had the chance to follow the company from the moment that senior management made the decision to adopt agile to the time the teams had to change their working processes and adjust to a new organizational culture. This exploratory study followed the recommendations defined by Singer, Sim and Lethbridge (2008) [108] for field studies. A set of 18 senior managers were interviewed over a period of six months (Dec'14 and Jun'15). Preliminary insights from this agile transformation initiative in large settings were published at the 2015 Brazilian Workshop on Agile Methods (WBMA) [98]. We present in Chapter 5 details of this Field Study.

Next, we conducted a focus group (Study 3) with two large-scale agile teams. These teams fall into the LSAD definition given their number of people and number of teams. Our main goal with this study was to identify whether agility is perceived as present by large-scale agile teams (Objective 3, Phase 2.2). By applying a questionnaire to collect the individual opinion of the team members and later discussing their feedback as a group to reach consensus in a focus group session we managed to conclude which aspects of agility are present in each of the two investigated teams as per the Progressive Outcomes Framework [42]. To perform this study, we rigorously followed the recommendations on how to conduct a focus group study proposed by Kontio (2008) [60] and Krueger (1994)[63]. Before

selecting the two groups to be investigated, we validated and piloted the data collection instruments following Kitchenham (2007) [57] guidelines. The results of the focus group study is presented in Chapter 6.

4. SYSTEMATIC MAPPING REVIEW ON LSAD

This chapter presents the results of the Study 1, the systematic mapping study to identify the state-of-the-art on LSAD (Phase1), which included to identifying definitions, in which LSAD takes place, and challenges faced by LSAD teams. This chapter is organized as follows. Section 4.1 describes the research protocol. Section 4.2 presents the results of the systematic mapping review. Section 4.3 discusses the results and threats to their validity. Section 4.4 summarizes the chapter.

4.1 Research Protocol

This study was undertaken as an systematic mapping review following the guidelines proposed by [56][19][94]. An systematic mapping is known as an exploratory study. It is a comprehensive review of primary studies on a specific subject, i.e., aims to identify what evidence is available on a certain topic. This Section addresses the systematic mapping protocol composed of the research questions posed to guide the mapping review, search strategies used to search for the aimed papers, the papers selection criteria, and the data extraction strategy.

4.1.1 Research Question

The LSAD term has first been used in academia in 2001 by Sutherland (2001) [114]. Despite the growing literature on the topic, the lack of subsequent empirical studies [29] makes an incipient and unconsolidated area. Our work aims to answer the posed research question:

- What is the state-of-art in empirical studies on LSAD?

We aimed to understand existing research directions within LSAD and in particular to identify adopted methods in the literature, identify related definitions, determine which settings are reported in such studies, and investigate challenges. To better state own intended goal with this review we broke down our primary research questions into four secondary research questions as presented in Table 4.1.

Table 4.1: Research questions

Research Questions	Motivation
RQ1. Which research methods are adopted into LSAD studies?	To identify which are the most frequently adopted methods used in the reported studies.
RQ2. How LSAD is defined in literature?	To identify the homogeneity of the definitions related to LSAD.
RQ3. What settings LSAD takes place?	To identify what kind of project, enterprise background, and agile practices are used in LSAD.
RQ4. What challenges are faced in LSAD?	To identify which challenges faced in LSAD.

4.1.2 Data sources and search strategy

To ensure the efficiency of the search, we used boolean search expressions merged per keywords. Although the standard approach is to characterize the search expressions based on population, intervention, control, and outcome (PICO) components [17], recent guidelines on systematic review for software engineering suggest the omission on the PICO components [56]. Based on these recommendations, we kept the population and intervention components and overlooked comparison and outcome.

In order to perform the automatic search of the selected digital libraries we used the keywords (synonyms) "Scaled Agile", "Scaling Agile", "Large-Scale Agile", "Large Agile" and "Agile at Scale" presented in the papers used as a control of this systematic mapping [33, 89, 65].

Thus, we established the search expression following:

(A) Scaled

(B) Scaling

(C) Large Scale

(D) Large

(E) Agile

All these search terms for LSAD were combined by using the boolean "OR" and "AND" operator which entails that paper only had to include any one of the terms to be retrieved and the expression "Agile," therefore, the query:

((A OR B OR C OR D) AND E)

After setting the boolean string, the next step was to choose the databases to run it. Thus, we chose well-known digital databases arose the following: IEEEXplore, ACM Digital Library, Springer Link, Science Direct, and Wiley InterScience. These are often used in software engineering studies.

After the search engines and the main research question defined, we selected some criteria aligned to the main research question and the secondary research questions

to help in the data selection and extraction process. We established the following criteria for the inclusion of studies: The paper must be available online, written in English, published in workshops, or in conferences, or in journals. Also, papers must have targeted the following topics: scale agile, large agile, large-scale agile or scaled agile, or even large scale projects, large projects using agile, and also be empirical studies.

The exclusion criteria consisted of removing duplicate papers or papers that are not of scientific nature. In all steps, the author and the supervisors of this Master Thesis read each paper independently. For conflicting evaluations, researchers further discussed the papers to reach a consensus. In cases where there was no consensus or there was doubt, the study was included to avoid premature exclusion.

4.1.3 Study selection

We used the string presented in Section 4.1.2 to retrieve the candidate papers from the digital library in January 2016. We retrieved 906 papers in our search. After applying the selection criteria, we selected 94 papers for data extraction and analysis. Next, we present the detailed results of executing the steps shown in Section 4.2. Figure 4.1 shows the phases of our selection process and detailed in the following list:

- (Step 1) **Query entered in the selected digital libraries**. We queried digital libraries, and the references of the retrieved studies were stored in the Start tool [1] to be further analyzed. As a result, we gathered 906 candidate papers: 216 from IEEE, 257 from ACM, 133 from Science Direct, 82 from Wiley InterScience, and 218 from Springer Link.
- (Step 2) **Reviewed title, keywords, abstract according to criteria.** For this step titles, abstracts, and keywords were read to verify which studies met the inclusion and exclusion criteria. Here the titles, abstracts, and keywords were independently analyzed by two researchers. During this step, the researchers identified and dismissed 95 duplicates gathered from different libraries. After that, by applying the inclusion and exclusion criteria, 144 studies were returned.
- (Step 3) **Introduction and conclusion read.** The initial and closing sections of the studies were evaluated regarding their objectives and results. This analysis enabled the researchers to verify further if the papers answered the research question and met all inclusion and any exclusion criteria. When the reading of the opening and closing sections was inconclusive, the entire paper was read to decide on its inclusion or exclusion. All 144 papers had their introductions and conclusions analyzed by the researchers. Among them, 50 were dismissed. We did not find the full text of four papers.

[1] A known tool that supports systematic reviews.

One of the paper was a previous version of another more complete study. We found 42 papers comprised in research agendas, books, editorials, and tutorials panels.Three papers were not written in the English language, and four were not available.

- (Step 4) **Full paper read**. Finally, after selecting a paper by its introduction and conclusion, the entire paper was read to decide on its inclusion or exclusion in the review.

We also summarize the total papers found by each source and approach in Table 4.2. This table also includes the search efficacy, which refers to the studies found and selected from the search string a step taken. For example, out of the 257 papers found in the ACM Digital Library, 22 papers were selected, resulting in an efficiency of 8.56%. Regarding the reasons for excluding papers, we account for them in Table 4.3.

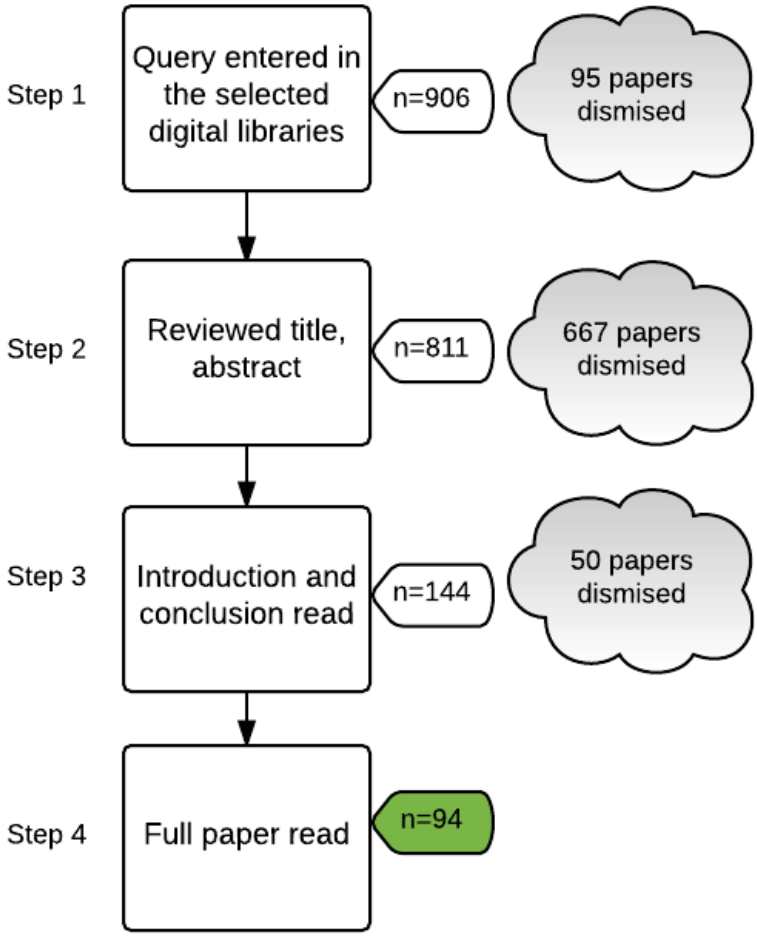


Figure 4.1: Systematic mapping review phases followed in this study

Table 4.2: Papers found and included per source.

Source	Found	Relevant Papers	Search Efficacy (%)
ACM Digital Library	257	22	8,56
Springer-Link	218	26	11,93
IEEExplore	216	67	31,02
Science Direct	133	7	5,26
Wiley InterScience	82	9	10,98
Total	906	131	14,02
Duplicated	58	37	
Sub-Total	848	94	11,08

Table 4.3: Papers excluded per exclusion criteria

Exclusion criteria	Excluded	Total of Studies
Total (without exclusion)	-	906
Related to Agile but not to Large Scale Agile or Large Scale Projects using Agile	667	239
Duplicated Papers	95	144
Prefaces, Editorials, Books, Discussions, Comments, Positions, Summaries of tutorials, panels.	42	102
Paper/Article not Available	4	98
Not in English language	3	95
Previous version of a more complete paper	1	94

4.1.4 Data extraction strategy

The data extraction strategy employed was based on providing the set of possible answers for each research secondary questions that had been defined. This approach ensure the application of the same extraction data criteria to all selected papers and it facilitate their classification. The possible answers to each research secondary question are explained in detailed on the following list.

- In RQ1. Adopted research methods: We extracted adopted research methods of each paper. They can be classified in one or more of the following answers: (A) Case study, (B) Survey, (C) Literature review, (D) Grounded Theory, (E) Others Action Research, Focus Group, etc.

- In RQ2. Definition for LSAD: A paper can be classified in one of the following answers: (A) yes, a definition paper, (B) yes, but not a definition paper, (C) no (excluded paper). This question describes the definition of large-scale agile software development of each paper. When the paper is a definition paper the field is filled as: “to read the full paper.”

- In RQ3. LSAD settings takes place in LSAD: We extracted (3.1) company information, (3.2) project information, and (3.3) agile software development characteristics. In the company information category (3.1) the fields extracted were: (A) Company’s domain

(B) company's location, and (C) company's name. In the project information category (3.2) we extracted in papers the fields: (A) Project type, (B) distributed or collocated, (C) number of sites (if applicable), (C) number of teams, (D) number of people, and (F) project's domain. (3.3) In the agile software development information category (3.3) we extracted the (A) agile methodologies, (B) scaling agile frameworks, (C) agile practices and (D) scaling agile practices.

- In RQ4. Challenges faced in LSAD: We extracted the from the paper where references to challenges were found and later coded them excerpt.

4.2 Results

In this section, we present the findings of the systematic mapping review (Study1) based on 94 papers selected upon our search. The results are structured based on the research questions stated in Section 4.1.1 (All 94 papers selected in this systematic mapping are listed in Appendix A).

4.2.1 Overview of the studies

We analyzed all papers checking the publications per year. Figure 4.2 describes such distribution. The topic first appeared in 2001, and the last publication was in 2015 (current year of search). The largest number of publications appeared in the last three years (40 papers out of 90). Also, it is important to highlight that this research was performed in December 2015. Thus, papers published after December'15 were not indexed in this study.

Out of the 94 papers selected in this study, ten were published in journals, and 84 in conference proceedings. The primary studies are spread in different conferences. Agile Conference has the highest number of studies published (26 out of 94). Following XP Conference with 20 papers out of 94 published.

4.2.2 Empirical research methods and data collection methods adopted by the identified studies (RQ1)

Based on the Shull, Singer and and Sjøberg (2008) guidelines [107], we identify the research methods and data collections used in the selected papers. Figures 4.3 and 4.4 present the empirical methods and collect data from each paper.

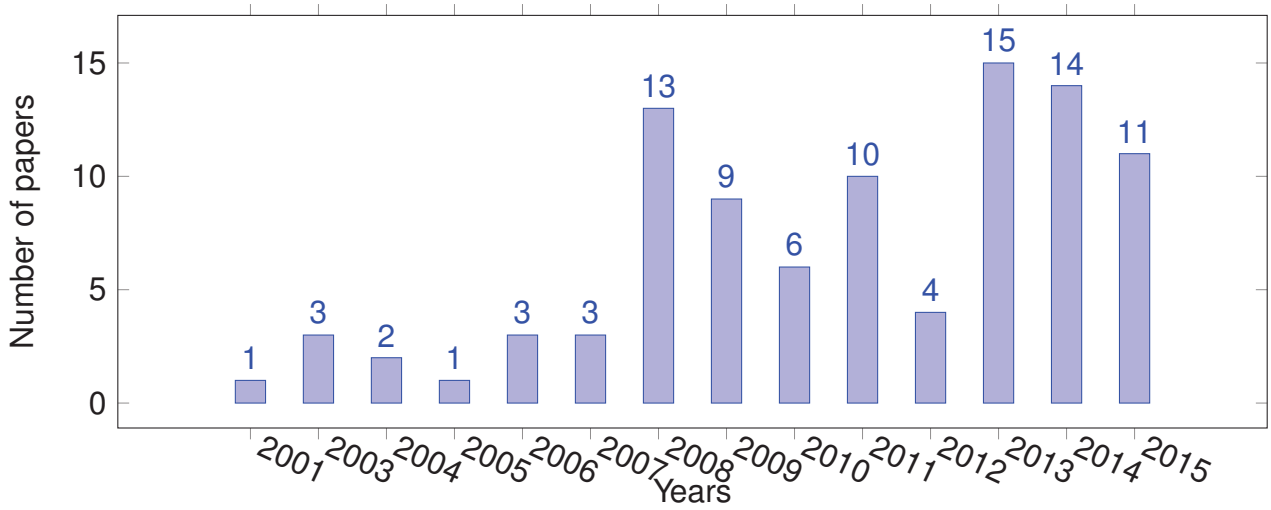


Figure 4.2: Temporal view of the publications (queries from January 2016).

The majority of the papers use the case study as a research method (47.87% - 47 out of 94 results). Survey method appeared in the second position with five papers. Action research had two papers; Grounded theory had two papers. The focus group is a method to construct new findings and to investigate a research question with a panel of people [107], resulting in 2 papers. Controlled experiment (1 paper) were the least used methods employed.

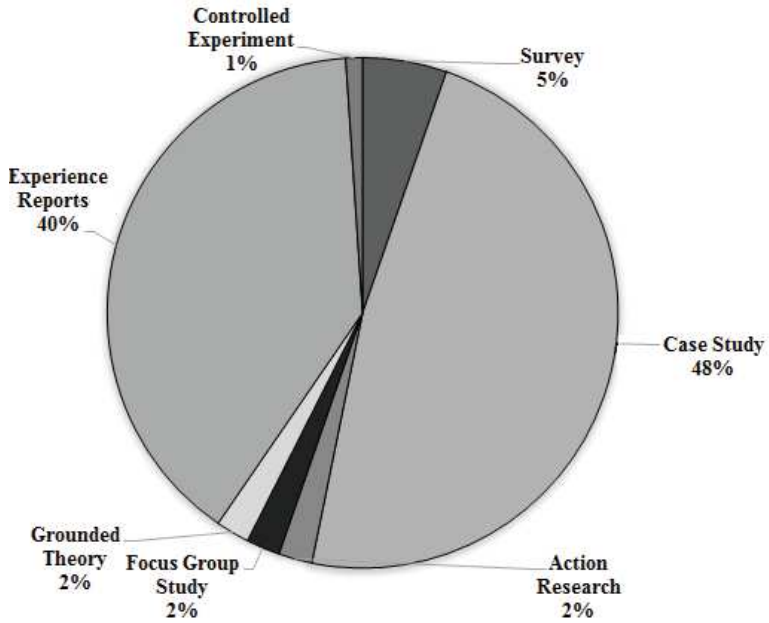


Figure 4.3: Research methods adopt in LSAD studies

Third-nine (40%) of 94 papers do not describe their research method, or it is not clear. About this, we recognized that most of these papers are experience reports [65] [4]. This category includes a significant amount of studies of experts authors and practitioners on the subject regardless of the potential bias issues, as mentioned in other systematic review on the topic in quality assessment section [29].

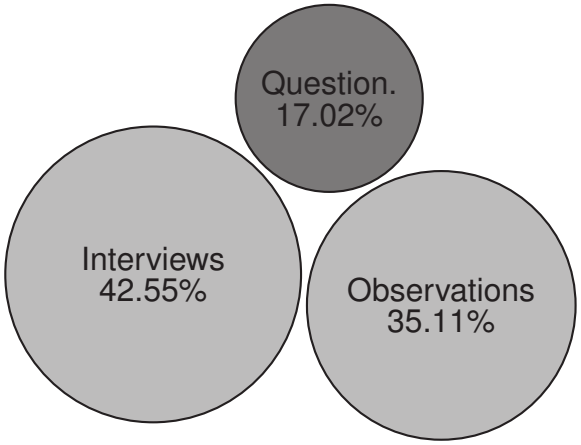


Figure 4.4: Data collection methods

For instruments to collect data, interviews had the largest number of papers (40 out of 94), following by observations with 33 papers and questionnaires with 16 out of 94 papers. All support Case Study and Survey Method.

4.2.3 Definition of LSAD in literature (RQ2)

In this research question we classify the same set of papers in LSAD definitions under different perspectives, as Tables 4.4, 4.5, 4.6. Next, we detailed these three classifications.

Based on information of each paper about LSAD definitions, we categorized the information in four categories in Table 4.4. Each paper can be allocated in more than one category. The findings were consolidated in (1) distributed large-scale settings using agile methods with 51 papers (54,26% percent of 94); (2) large-complex projects (complex integrations, dependencies among different areas, overheads) using agile methods with 35 papers (37,23% percent of results), (3) multiple teams who develop the same project we found in 11 papers (11,7% of results), and (4) collocated teams who develop the same project in 3 papers.

Afterwards, we identified three papers in the systematic mapping [33][4][65] that helped us to categorize the definition on the topic in other aspects. These papers describe characteristics of LSAD settings.

The first definition for LSAD was proposed by Dingsøyr, Fægri, and Itkonen (2014) [33]. The authors suggested common characteristics focusing on three evidences in LSAD. The first evidence is that *neither cost, code size, the number of requirements, nonfunctional requirements and the size of software regarding code or requirement, technologies and project contexts are not a sufficient criterion to define LSAD.* However when more people

Table 4.4: Definition of LSAD per study

Definition	Ref.	Freq.
Distributed large-scale settings using agile	[p1] [p4] [p5] [p6] [p8] [p10] [p11] [p14] [p18] [p21] [p23] [p25] [p27] [p28] [p30] [p31] [p32] [p33] [p34] [p37] [p39] [p44] [p45] [p48] [p49] [p51] [p53] [p54] [p55] [p56] [p57] [p58] [p59] [p61] [p62] [p64] [p65] [p66] [p67] [p68] [p69] [p70] [p78] [p83] [p85] [p86] [p88] [p91] [p92] [p93] [p94]	51
Large-complex projects (complex integrations, dependencies among different areas, overheads) using agile	[p4] [p5] [p6] [p9] [p11] [p18] [p19] [p15] [p22] [p24] [p43] [p13] [p17] [p23] [p27] [p30] [p33] [p35] [p40] [p41] [p46] [p50] [p52] [p59] [p60][p63] [p74] [p75] [p79] [p82] [p83] [p84] [p90] [p92] [p93] [p96]	36
Multiple teams who develop the same product using agile	[p3] [p13] [p20] [p26] [p29] [p35] [p36] [p38] [p52] [p78] [p85]	11
Collocated teams who develop the same product using agile	[p20] [p24] [p55]	3

were involved in a project, the work needed to be divided among several teams requires a new level of coordination.

These three evidences help to find the first definition on the topic and also to map the papers. Table 4.5 shows a mapping between papers extracted from our systematic mapping. We analyse the number of each team described in those papers to described in this systematic mapping.

Table 4.5: Taxonomy categorization of LSAD. Source: Dingsøyr, Fægri, and Itkonen (2014) [33]

Definition	Description	Ref	Freq.
Small scale (one team)	Team coordination can be done using agile practices such as daily meetings, planning, review and retrospective meetings	[p30]	1
Large-scale (between two to nine teams)	Teams coordination can be achieved in a new forum such as a Scrum of Scrums forum	[p8] [p30] [p81]	3
Very large-scale (ten or more teams)	Several forums are needed for teams coordination, such as multiple Scrum of Scrums.	[p30][p38] [p55] [p3] [p57] [p68] [p65] [p70] [p49] [p61] [p56] [p37] [p64] [p32]	14

The only study found with small scale definition was the proposed by the author of this categorization. No other studies describes about one single team representing a small scale agile. A reason for this finding is described in the Power's (2014) study [95]: if the team is relatively decoupled the concept of 'large-scale agile' likely does not apply. One paper was found in category large-scale between two to nine teams. In this paper one project that its

management decided to start agile in a software team with 15 people, organized in four sub-teams [106]. For very large-scale we found thirteen papers with more than ten teams. For instance, Heikkila et al. (2010) [53], explored the release planning by observing events and conducting questionnaires to over ten development teams, who spent several days in a shared space doing release planning together.

After, we got Ambler (2008) [4], and Laanti (2014) [65] papers which identified the characteristics needed for a company to be considered a large-scale agile: The team size, the geographical distribution of the team, the entrenched culture, the system complexity and legacy, the regulatory compliance, the organizational distribution, the governance and the enterprise focus. Thus, Table 4.6 shows a consolidated set of characteristics for LSAD mapped in the papers found from the systematic mapping review. The list below describes the seven characteristics and the frequency of related papers.

Table 4.6: Characteristics of LSAD environments. Source: Ambler (2008) [4]

Characteristics	Description	REF	Freq.
Enterprise focus	Addressing enterprise issues, including enterprise architecture, portfolio management, and reuse within an agile environment.	[p4] [p5] [p6] [p10] [p11] [p14] [p17] [p18] [p19] [p22] [p25] [p26] [p29] [p30] [p32] [p33] [p35] [p36] [p39] [p40] [p44] [p45] [p46] [p47] [p48] [p52] [p53] [p55] [p56] [p58] [p59] [p62] [p63] [p66] [p64] [p67] [p71] [p72] [p74] [p75] [p77] [p80] [p81] [p82] [p85] [p87] [p92] [p94]	48
Complexity	Agile teams working on a system complexity requiring a viable architectural strategy	[p1] [p4] [p5] [p6] [p7] [p9] [p13] [p15] [p17] [p18] [p19] [p20] [p23] [p24] [p27] [p30] [p33] [p35] [p41] [p43] [p44][p46] [p49] [p50] [p52] [p59] [p60] [p63] [p71] [p74] [p75] [p77] [p81] [p83] [p88] [p89] [p90] [p93]	38
Team size	Large teams (10+) using agile	[p3] [p4] [p8] [p13] [p15] [p19] [p20] [p25] [p27] [p28] [p30] [p32] [p35] [p36] [p37] [p38] [p39] [p42] [p49] [p55] [p56] [p57] [p58] [p60] [p61] [p64] [p65] [p66] [p68] [p70] [p72] [p75] [p76] [p79] [p88] [p90] [p93]	37
Geographical distribution	Distributed teams (including stakeholders) using agile	[p1] [p3] [p4] [p5] [p14] [p15] [p21] [p23] [p25] [p27] [p30] [p32] [p33] [p37] [p39] [p49] [p56] [p57] [p58] [p61] [p64] [p65] [p68] [p69] [p88] [p70] [p71] [p78] [p91] [p93]	30
Organizational distribution	Agile team working for different divisions, or from different companies (such as contractors, partners, or consultants), then management complexity rises.	[p1] [p3] [p8] [p9] [p13] [p24] [p25] [p27] [p28] [p31] [p34] [p39] [p41] [p49] [p58] [p65] [p69] [p70] [p74] [p76] [p86] [p89] [p94]	23
Culture	Agile teams working on an large company with entrenched traditional culture.	[p5] [p6] [p13] [p14] [p23] [p27] [p35] [p37] [p45] [p50] [p52] [p57] [p67] [p68] [p80] [p84]	16
Governance	Top management support, multiples IT projects using agile	[p5] [p10] [p15] [p42] [p44]	5

- **Enterprise Focus**: Is related to large companies using agile thinking (practices, principles, frameworks) to solve problems in different others disciplines such as: architecture [83], design, marketing [41], portfolio management [116], program management [65]. For enterprise focus we found 50 related papers.

- **Complexity:** Large systems which involve subcontracts with other businesses (sometimes competitors) for building or integrating substantial parts of the system [59]. Also, they require architectural coordination for teams developing in the same application or structure [83]. In system complexity characteristic we found 39 related papers.

- **Team Size:** In Scrum Guide [115] having more than nine members in one team of development requires a lot of coordination, being a discouraged practice. Given such a statement we categorized software development teams with more than ten members (not including Product Owner (PO) and Scrum Masters if they are not working on a Sprint Backlog [115]). For instance, Tabib's (2013) [117] project at HP involved over 150 developers, Quality Assurance (QA) and PO distributed in Israel, Ukraine, Czech, Vietnam, China and the US. Lagerberg et al. (2013) [68] on the other hand, researched the impact of using agile principles and practices in a Project A and Project B that are large commercial projects with 420 and 120 members respectively. These projects also have a large number of teams (15 and 14 respectively). Then we mapped papers containing a background with more than ten team members, totalizing an amount of 38 papers.

- **Geographical distribution:** This characteristic comprises having distributed teams. Some members of a team, including stakeholders, may be in different locations. Paasivaara et al. (2014) [92] identified how distributed product development company at Ericsson use Value Workshops to align the different sites and teams when adopting agile and lean software development. Currently, the development company is distributed to five sites located in three countries. Four of the sites are in Europe, and one is in Asia. Abdullah and Abdelsatir (2013) [1] presented how XP practices could enhance the development and implementation of a large-scale and geographically distributed system. In this case, we mapped each papers containing distributed teams totalizing an amount of 30 papers.

- **Organizational distribution:** Characterized by having a large team of people working on different divisions, or from various companies (such as contractors, partners, or consultants), thus management complexity rises. For instance, agile software development in a multi-site off-shore environment [51]. For organization distribution, we found 23 papers.

- **Culture:** In a large company, sometimes agile teams need to work with hybrid methods. This includes working on a matrix structured hindering the empowerment of teams

[80]. Or tailoring agile methods focus on planning and controlling aspects of software development. An adapted agile development process is ideal for long-term projects and the development of large systems [75]. For culture characteristic we found 16 related papers.

- **Governance:** This characteristic is related to top-management containing one or more IT projects governance in a place. Government decisions to change policies about taxation or social security systems often imply large scale software implementations. Then the product owner provides a technical governance framework to project teams working on a program [11]. For governance, we found five papers.

4.2.4 Settings where LSAD takes place (RQ3)

To identify those where LSAD settings takes place we analyze the extraction of data into three different categories based on our data extraction (see in Section 4.1.4). Thus, we organize in the next paragraphs the results of project information, companies information, and agile software development extracted from each paper in this systematic review. Not all papers are mapped in all categories listed, due to the lack of information.

Project Information

In the project information category, we collected data from the projects' characteristics, the type of the projects, the number of teams inside the project, the number of people in the same team, and the projects' domain.

In the projects' characteristics, we collected data from studies that report information about their projects (see Table 4.7). More than half of all papers found (52.13 %) describe their cases through the projects. Moreover, we found more information characterizing the type of project, as a product with twelve studies, and system with four studies. We identified that 47.87% no reported no information related to the projects researched.

Related to the type of projects we collected data between distributed and collocated projects. We discover 38 distributed projects, and three papers studying collocated projects (see in Table 4.8). Even with a small number of collocated projects compared to the total number of studies, this result can prevent generalizations about using LSAD term in distributed environments. In the distributed projects we found the number of sites between two to six sites. The most frequent studies that report among 4 to 2 sites. There are also no studies which described if the project is distributed or collocated, as is the case of [P9], [P17], [P55], and others.

Table 4.7: Number of projects and characteristics found in the identified studies

Inf.	Ref.	Freq.	Charact.	Ref.	Freq.
Project	[p1] [p3] [p4] [p5] [p8] [p13] [p14] [p15] [p17] [p20] [p21] [p22] [p23] [p24] [p25] [p27] [p30][p31] [p32] [p33] [p35] [p37] [p38] [p39] [p42] [p49] [p53] [p54] [p55][p56] [p57] [p58] [p61] [p64] [p65] [p68] [p69] [p70] [p71] [p72] [p74] [p78] [p81] [p84] [p85] [p88] [p89] [p90] [p91] [p93]	51	**Product**	[p3] [p5] [p9] [p21] [p23] [p31] [p54] [p69] [p72] [p78] [p84] [p93]	12
			System	[p4] [p13] [p42] [p89]	4

Table 4.8: Types of projects in the identified studies

Type	Ref.	Freq.	Number of Sites	Ref.	Freq.
Distributed	[p1] [p3] [p4] [p5] [p14] [p15] [p21] [p22] [p23] [p25] [p27] [p30] [p32] [p33] [p37] [p39] [p49] [p53] [p56] [p57] [p58] [p61] [p64] [p65] [p68] [p69] [p70] [p71] [p78] [p85] [p88] [p91] [p93]	38	2	[p30] [p37] [p61] [p69] [p71]	5
			3	[p32] [p39] [p56] [p57]	4
			4	[p4] [p58] [p64] [p68] [p70] [p71]	6
			5	[p22] [p23] [p53	3
			6	[p85]	1
Collocated	[p20] [p24] [p55]	3	-	-	-

Regarding the number of teams inside a project, we found studies describing projects between two teams to thirty teams (Table 4.9). In this table, we have not presented projects that contained a single team. On the number of people in a team, the Table 4.10 shows numbers between 10 to 1200 people working on the same team. On the projects' domain, we found eight different areas. Figure 4.5 shows that AirCraft and Telecom are the projects with more studies.

Table 4.9: Number of teams per study

Number of teams	Ref.
2	[p8] [p81]
10	[p32] [p38]
14	[p49]
15	[p49] [p55] [p85] [p90]
20	[p37] [p64] [p68] [p70]
25	[p57] [p68] [p70] [p56]
30	[p3] [p61]

Table 4.10: Number of people in the same team per study

Number of People	Ref.
10	[p8]
15	[p90]
22	[p13]
30	[p93]
Over 50	[p72]
58	[p39]
60	[p58]
76	[p27]
78	[p35]
80	[p30]
170	[p20]
Over 150	[p88]
190	[p69]
300	[p25]
1200	[p55]

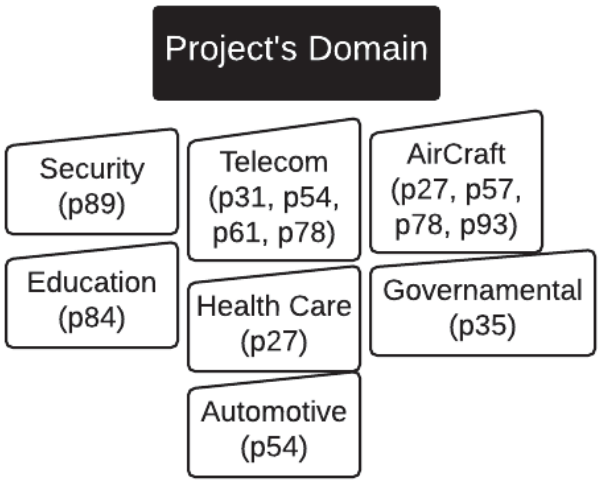


Figure 4.5: Projects' domain per study

Company Information

In the company information category, we extracted company's domain, the company's location, and the company's name. Related to company's domain, we found companies as IT (32 studies). Telecom domain (16 studies), for instance, Ericsson, Nokia, Qwest and Siemens companies. Also, Automative Industry at Volvo Cars company appeared in the third position. Air Force, Coaching, Retail and others appear in the lastly, as seen in Table 4.11. According to the location of each company we found companies in different locations covering four mainlands. Figure 4.6 shows the geographical map with the countries where the companies have their offices.

Table 4.11: Companies' information

Domain	Ref.	Freq.	Name	Ref	Freq.
Air Force	[p90]	1	none	none	none
Army	[p17]	1	none	none	none
Automotive Industry	[p19] [p53] [p54]	3	Volvo Cars	[p19]	1
Aviation Industry	[p27]	1	none	none	none
Bank	[p11] [p92]	2	ABC Bank	[p11]	1
Coaching	[p21]	1	Landmark	[p21]	1
IT	[p3] [p5] [p8] [p10] [p13] [p15][p19][p13] [p14] [p22] [p24] [p25] [p28] [p33] [p38] [p39] [p40] [p51] [p52] [p53] [p55][p56] [p60] [p62] [p69] [p73] [p80] [p81] [p83][p86] [p88] [p94]	32	BMC	[p25]	1
			Borland	[p51]	1
			Channel Advisor	[p40]	1
			Cisco	[p86]	1
			FinApp	[p13]	1
			F-secure	[p39]	1
			HP	[p88]	1
			IBM	[p10] [p15]	2
			Microsoft	[p55]	1
			Salesforce	[p3] [p22] [p24]	3
			UOL	[p60]	1
Mechatronic Driven:	[p7]	1	none	none	none
Oil Gas	[p30]	1	none	none	
Retail	[p28]	1	Gap	[p28]	1
Telecom	[p34] [p36] [p37] [p45] [p48] [p49] [p53] [p54] [p56] [p57] [p58] [p61] [p62] [p66] [p65] [p67][p72] [p85]		Ericsson	[p34] [p36] [p49] [p58] [p65] [p72]	6
		16	Nokia	[p45] [p48]	2
			Qwest	[p85]	1
			Siemens	[p56] [p57]	2
TV Station	[p46] [p76]	2	BBC	[p76]	1

Agile Software Development

For this category we selected data from:

(A) agile methods and agile practices,

(B) scaling agile frameworks, and

(C) scaling agile practices.

Related to (A) agile methods and practices were found Scrum, Kanban and Lean methods. Table 4.12 the most common method used is Scrum with 62 studies, following XP in the second position with 53 studies. Lean with 11 studies and lastly Kanban out of 4 studies. Also, Agile Practices were extracted from each study. All these practices in one way or another have been changed according to the setting. For instance, in the study about Qwest Communications (2003) [111] the practice pair-programming was adapted for their Company's setting. This phrase was written to identify how they used this practice day-to-day: *"(They started) pair-programming across distance (using Net Meeting)", and it was an*

Figure 4.6: Countries' company per study

evolutionary innovation. "Developers loved it too, even pair programming (after the initial resistance of course)"..."Testers worked in partnership with customers and did pair testing." [p78]. For this reason, we do not mapped how each practice was adopted.

About (B) scaling agile frameworks, we found SAFe (2 papers p46 and p47), Disciplined Agile Delivery (DAD) (1 paper, p10) to support large-scale agile. And about scaling agile practices, we extracted in each paper the most successful practices created to LSAD in which large environments needed to scale agile practices to adopt agile software development. The following list detailed each scaling agile practice.

- **Technical Area Responsible (TAR)** [p58]: TARs are central in the knowledge network and act as the boundary spanners among teams and sites [79]. The goal of this role is to support teams to solve problems, to ensure knowledge sharing within the project and safeguard the quality. In this role they usually have a senior developer working half-time or full-time. This practice was used in a large-scale globally distributed agile project at Ericsson.
- **Communities of Practice (COP)** [p18, p65, p66, p90, 94]: The aim of a COP is to develop member capabilities. It is formed by members who selected themselves, and they interact on a continuing basis to build and exchange knowledge. At a project at IBM, COP reveals important elements in agile transformation process in an LSAD setting. They also have three levels of COPS: the first level of community involves a core group of business and technical leaders who represent their units as change agents. The second level of community is a single broad community that provides a social network for building awareness of the latest evolutions, resources, and events

Table 4.12: Agile methods and agile practices adopted per study

Method	Ref.	Freq.	Agile Practices
Kanban	[p10] [p37] [p46] [p65]	4	Kanban Board
Lean	[p2] [p9] [p10] [p24] [p46] [p47] [p65] [p79] [p82] [p86] [p92]	11	-
Scrum	[p1] [p3] [p5] [p6] [p8] [p10] [p11] [p14] [p15] [p17] [p19] [p20] [p22] [p23] [p24] [p26] [p28] [p29] [p30] [p32] [p35] [p36] [p37] [p38] [p39] [p43] [p40] [p41] [p44] [p45] [p48] [p49] [p51] [p52] [p53] [p54] [p55] [p56] [p57] [p58] [p64] [p65],[p66] [p68] [p69] [p70] [p72] [p74] [p75] [p76] [p78] [p80] [p81] [p82] [p83] [p84] [p88] [p90] [p91] [p92] [p93] [p94]	62	Planning Poker (Planning Sprints), Restrospective and Review, Product Owner, Scrum Masters, Metrics (burn-up, burn-down), Standup Meetings, Product Backlog
XP	[p1] [p3] [p5] [p7] [p10] [p11] [p13] [p17] [p18] [p20] [p21] [p22] [p23] [p24] [p26] [p29] [p30] [p31] [p32] [p36] [p37] [p38] [p39] [40] [p41] [p42] [p43] [p45] [p49] [p50] [p52] [p55] [p56] [p57] [p60] [p63] [p64] [p68] [p69] [p71] [p72] [p74] [p75] [76] [p84] [p85] [p86] [p88] [p89] [p90] [p91] [p93] [p94]	53	Pair Programming, Unit Testing, Test Driven Development, Planning Game, Release Planning, Continuos Integration, Colletive Code Onwership, Simple Design, Refactoring, Clean Code, Sustainable Pace, Daily Meetings

in software engineering practices. The last community focuses on specific software engineering practices. Each Practice Community is responsible for helping teams to adopt and use a particular software engineering practice. Also, to help agile, lean transformation at Ericsson [86, 89] they used COP. They had over 20 groups of COP (some, for example: Feature COP, Coaching COP, Developers COP, End-to-end COP), each one gathering weekly, bi-weekly or on a need basis to talk about their topics.

- **Agile Architects** [p1, p18, p33, p40, p57, p59, p68, p75]: Architecture provides a way to partition work around large chunks of software development, guiding the company into teams. Moreover, the role and support for the architecture depend not only on the degree of the size but as well on the degree of complexity [36]. On large-scale software development endeavors, agility is enabled by architecture, and vice-versa [p57]. Agile allows developers to focus early on key architectural decisions, spread these decisions over time, and validate architectural solutions early as the architecture of the system, the structure of the development company, and the production infrastructure. This role (sometimes also called architecture owner or chief architects) is required to work in close cooperation with all different feature teams. Allowing the chief: (a) to understand

the needs of the teams; (b) to ensure the teams understand the architecture, and (c) to help to improve the architecture.

- **Area Product Owner (APO)** [p37, p39, p58, p64, p68, p70]: Larman and Vodde (2008) [70] suggest APOs for scaling the PO role, a practice adopted by globally a distributed software and hardware project company. [87]. Each APO was responsible for features in one specific product area, and the idea was that the APO would work with 2-3 teams developing those features. The role of APO was divided between two people: a system architect and a solution architect. The system architect was a technical person, whose duties included performing feasibility studies for new suggested features, taking care of the architecture planning, splitting features into user stories and frequently communicating with the teams. The solution architect was a product management representative, who could either have a business or technical background [p68, p70].

- **Agile Feature Teams** [p3, p7, p8, p9, p22, p24, p29, p30, p36, p37, p52, p58, p64, p65, p70, p71, p77]: Teams working vertically across multiple components of the overall product in order to realize a specified feature end-to-end [18]. An agile feature team can be organized in more complex projects as cross-functional or cross-component production accountants sharing their knowledge to the developers. Feature teams work on a single feature at a time grouped into technical product areas [p37]. Also, there are virtual feature teams who performed their feature-related tasks for their component and then passed the work on [p65]. Agile feature teams would be able to develop a new feature very fast without extra handovers, from end-to-end [p71].

- **Scrum-of-scrums** [p3, p93, p68, p27, p38, p46, p48, p52, p54, p56, p63, p68, p76, p88, p93]: A technique to handle inter-team coordination in large-scale Scrum [p68]. The scrum-of-scrums is used to tactically manage and coordinate the progress of iterations through the various scrum teams [p6]. At Salesforce [10] scrums-of-scrums were introduced to a group of teams to discuss their work, focusing mainly on the area of overlap and integration.In other LSAD case was created multiple feature-oriented teams, using scrum-of-scrums to help in agile transformation at large companies.

- **Customer Specific Teams** [p61]: Customer-specific teams are designated teams who work exclusively with one selected, and highly prioritized customer, to quickly respond to their particular needs and requests after product deployment [85]. The use of this role was advantageous for improving customer responsiveness, customer satisfaction and feature quality during software evolution for a large-scale embedded systems.

- **Agile Portfolio Management** [p46, p87]: Is to deal with transparency of resources and work items, improving trust, decision-making, and resource allocation; close collaboration based on routinized interaction and artifacts enabling frequent feedback loops

across domains, commitment to strategically managed portfolios and removing unrest in resource allocation and building capabilities in teams [67]. Agile portfolio management was a practice taken at Finnish Broadcasting Company Yle in the area of Internet development. There are existing practices inside of Agile Portfolio Management, they are: prioritization of epics (to avoid a long queue of development items that will get outdated); portfolio backlog (clear visibility and communication of implementation needs through epics); epic owner(s) (responsible for making all the decisions regarding the contents of that epic); enterprise architect(s) (to lead to better decisions as all opinions are heard and viewpoints considered); program portfolio management (strategy planners and directors that make portfolio decisions and prioritize the Portfolio Backlog jointly); Strategic Themes (express the intent to which direction the enterprise would like to develop its portfolio) and portfolio metrics (Portfolio metrics measure the enterprise's performance at the highest level) [67].

- **Agile Requirements Management Hierarchy** [p25, p38, p76, p83, p86, p88]: This practice is based on a five-level hierarchy following Leffingwell's book [74]. First, the *strategic themes* denote strategic focus areas for the company's business. Within these, *epics* form high-level functional goals for the product(s). Epics can be split into *features*, which in turn can be further split into *user stories*. Finally, user stories are refined into development *tasks*, which denote what needs to be done technically to implement a user story. Companies [53, 52, 50] using this strategic during their planning.

4.2.5 Challenges faced in LSAD (RQ4)

The last, but not least, research question was set to identify challenges in LSAD studies. We extracted challenges using some guidelines of coding data [26]. This field was defined in three main categories as to the Nerur's study [82]: management and organizational issues, people-related issues, and process issues. The result of this RQ is shown in Figure 4.7. In the following paragraphs, we detail the challenges found in these 94 papers.

Management and organizational issues. This category is related to organizational culture and management style in software development issues [82]. In this category we found three challenges, in 11 studies. The first problem is related to has a **C1.Organizational structured not adequate** to adopt agile [p7, p57, p65, p84]. This challenge is a well-known fighting of agile versus matrix structure (traditional structured). Shatil, Hazzan, and Dubinsky (2010) [106] related their matrix-structured company that contains teams of people from several sections of the business. These teams were created to the purpose of a specific project, often existing only for the duration of the project, and are led by a project manager and this is perfectly run in the traditional process. Other challenges that

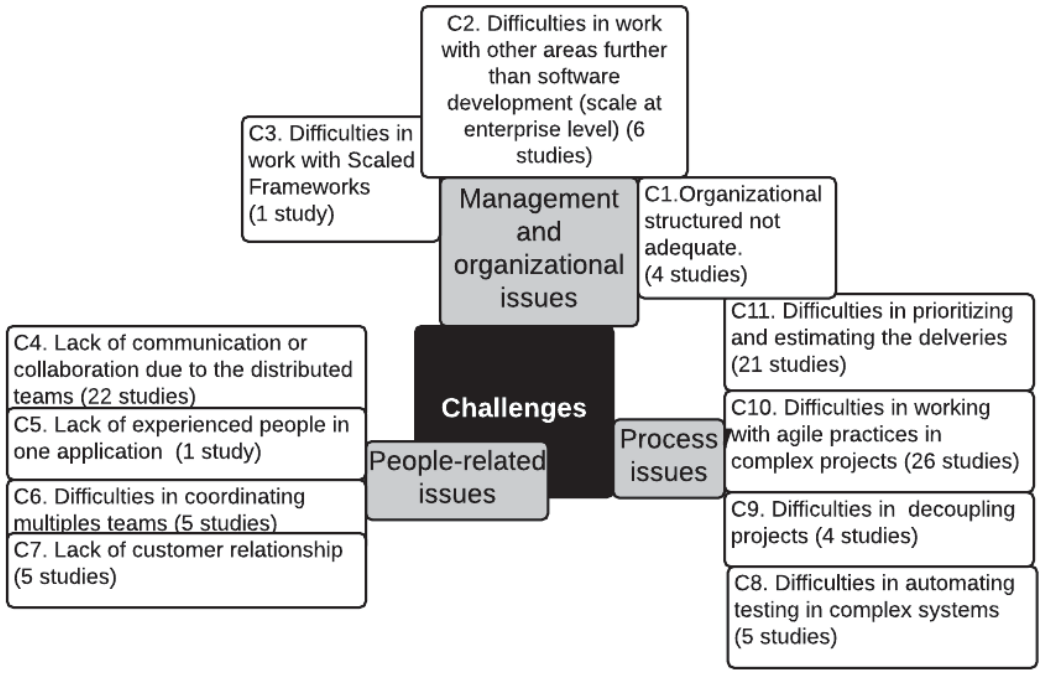


Figure 4.7: Challenges faced in LSAD

the matrix organizational structure faces when implementing the agile approach in a system project, address the team structure and definition, hindering the team to be self-organized [80]. For Berger and Eklund (2015) [16] the organizational structures that aligned leaders with a small set of teams (hierarchical and matrix structures with multiple sub-team leaders) tended to influence more managing versus leading as their role was more narrowly focused on these teams and their members. Paasivaara et al. (2014) [86], on the other hand, identify that "old structures" are a change resistance with the agile mindset at Anything-as-a-Service (XaaS) platform in Ericsson.

The challenge **C2. Difficulties in work with other areas further than software development (scale agile at enterprise level)** [p21, p22, p29, p55, p84, p88] is related to work with Agile Software Development including other disciplines. Shatil et. al. (2010) [106] found in their study that, in addition to the software component, a system project encompasses other disciplines as well, such as hardware, algorithms, and mechanics impacted the agile approaches. Greening et al. (2010)[48] in the same way identify the biggest challenge of all is scaling Scrum to the operation of an entire company. Enterprise Scrum is encroaching into this area, in part because it promotes the use of Scrum in other large, struggling creative departments, such as marketing.

The last challenge in this category is **C3. Difficulties in work with Scaled Frameworks** [p64]. This curious challenge is related to Scaling Agile Frameworks and its implications. One study was identified relating to maturity in Scaled Agile Software Development that described it implication through the usage of SAFe Framework in industry. The cur-

rent wholesale adoption approach of the SAFe is considered risky and complex. Moreover, driven by practitioners' efforts in the industry, there are very few studies about SAFe adoption reported in the academic literature [112].

People-related issues. In this category, we highlight challenges faced for the critical success in agile software development: individuals. People-related issues have four major challenges, as described next.

C4. Lack of communication or collaboration due to the distributed teams [p4, p7, p8, p19, p25, p27, p28, p34, p39, p41, p44, p53, p57, p59, p64, p69, p70, p73, p83, p89, p91, p96]. In large companies, it is common to use distributed teams to work in the same application, system or project. For Tabib (2013) [117] the major reason is the distribution of teams in different locations and time zones. Distributed teams may suffer many limitations as depend on confs calls, online programs to collaborate and communicate. This implies in a hard transition, as Smits' [109] study. The most difficult transition is getting everything else that must feed and support the product development teams in place. Hallikainen (2011) [51] on the other hand, mentioned about physical limitations may add an extra challenges large scale company with a long history in the industry. The ideal case is to have the whole company in the same building and at the same corridors. It can easily lead to that the physical limitations are used as an excuse not to communicate and collaborate physically. Failure with agile practices as in the planning practice where all developers collocated should suffer consequences [112].

C5. Lack of experienced people in one application [p84].This challenge is related to the individual expertise in knowledge of an application. In complex scenarios, it is difficult by the factor beyond the expertise of their work this individual needs to understand the expertise domain of their work. Furthermore, they should have the ability to think on different levels of abstraction, both on the detailed level and to a higher standard of abstraction, and to move between these abstraction levels wisely, when needed. This ability is especially important in integration tasks, when an explicit and systematic analysis of complex problems is required to locate the problematic component of the system and then activate the right people to coordinate a fine-tuned analysis and find a solution as Shatil et al. highlight this issue [106].

C6. Difficulties in coordinating multiple teams [p4, p19, p37, p68, p73] In LSAD we need to pay attention to multiple teams. But multiple teams require good coordination to people can work in an independent way, regardless the project size and not affecting the collaboration and team communication. Scrum of Scrum is a common practice to deal with coordination. By the way, Paasivaara (2012) cited shows that [91] Scrum-of-Scrum meetings involving representatives from all teams are a real challenge. The audience was too wide to keep everybody interested, and the participants did not know what to report that might be valuable to other teams, often ending up not reporting anything. Also, for Eklund et al. (2014) [37] the challenge of scaling the number of involved teams is described as a

challenge in two-folds. First, scaling the number of involved teams. Second, scaling up the necessary system engineering activities in the iterations/sprints prescribed by different agile methodologies. Power (2014) [95], on the other hand, mentioned coordination cost relates to the cost incurred in coordinating the people and systems that perform the work. It can be measured in time and money.

C7. Lack of customer relationship [p7, p19, p27, p31 p44, p62].This challenge is related when a customer is not available and in cases that customer is not even in the same time zone. Ktata and Ghislain (2009) [64] identify in their study the misunderstanding stakeholders' needed. Scalability issues increase the level of difficulty since Agile methods lie on face-to-face communication to transfer knowledge. Also, this is critical for early feedback. If the customer is not available, someone who used the similar system on customer site may also review the system and give early feedback. These timely feedback forces the programmers to do customers priorities, not their priorities [49]. For Olsson and Bosch (2015) [84], customer availability is hard when you are in an LSAD. Typically, and as reported in the interviews, the clients who "scream the loudest" get recognized while other customers get forgotten.

In the **Process issues** category we describe challenges related to process in software development and their complexity in LSAD environments. Here we found four more challenges and presented in detail in the next paragraphs.

C8. Difficulties in automating testing in complex systems [p25, p29, p55, p85 p91]. The lack of test automation due to large and legacy systems requires a significant investment. As Goodman (2008) [47] revealed at GAP Inc. that TDD on a large existing code base requires a significant investment. Also, another challenge is to identify dangerous areas early enough to ensure the quality of the application in production. Tabib (2013) [117] found a challenge at one HP application cause since they have moved from 18-24 month-releases in previous projects to 2 weeks release sin the current one. Not only that testing has to be done earlier in the life-cycle rather than last step, it had to be also much focused on the risk areas of the application.

C9. Difficulties in decoupling projects [p7, p19, p56, p57] This challenge is highlighted by Moore (2009) [80]. One of the primary challenges for the large-scale development project is decomposing the project into smaller projects that can be executed in parallel. To apply Agile Software Development in such environments is required to organize teams with individuals from across the program to minimum cross-team dependencies and coordination, but this is a challenging and hard thing to do [80]. However, related challenge is concerned cross-functional team expertise with component inter-dependencies. Usually, companies realize that many elements in a large-scale system are technically robust and interdependent, and require years of experience to be fully understood by developers [37].

C10. Difficulties in working with agile practices in complex projects [p4, p7, p19, p23, p24, p27, p28, p29, p31, p40, p44, p45, p48, p50, p54, p55, p58, p62, p63, p67,

p72, p73, p82, p84, p85, p88, p89, p91, p92, p95]. This challenge is related to coordination and construction of the requirements or user stories. Is hard to write user stories with a minimum level of detail due the project or software complexity and interlocks. Goodman (2008) [47] identify in HP project that writing good stories is difficult. Getting to the right level of detail and breaking up large stories into small enough stories has proven to be challenging. Also, we found in literature problems with backlog inconsistent due to a lot of unused features, as Olsson and Bosch (2015) [84] identify that due to limited mechanisms to monitor feature usage. Their five companies are convinced that a large number of the features they develop are never used and that investments are put on functionality that is not proven valuable to customers challenge.

C11. Difficulties in prioritizing and estimating the deliveries [p7, p8, p19, p23, p28, p29, p31, p33, p36, p37, p38, p39, p44, p50, p62, p73, p78, p89, p91, p93, p96]. In each stage of agile planning, prioritization depends on both effort and value estimation. In LSAD environments is difficult to do such thing due to the complexity. In Greening's (2010) [48] study (Citrix Online Industry) their team has faced challenges with an immature Product Owner as identity in first planning period for Enterprise Scrum, a server team added a 600 ESP project to our Enterprise Backlog and requested prioritization. But the description was indecipherable to everyone outside the team: it was a refactoring project to reduce hardware requirements. In an early Product Board meeting, members dropped it off the Quarter Backlog. The server group realized it had to articulate the project's value better.

Release Planning helps to estimate better, filled their iterations with stories based on the teams. In Heikkila et al.'s (2010) [53] study they showed conclusions of one Event: *Many details of the new architecture were unknown and the complexity of the whole system was high, which resulted in effort estimates that were too optimistic.*

4.3 Discussion

This section we discuss the main findings of this systematic mapping. Also, we highlight the threats to validity and the majors implications for research and practice.

4.3.1 Main Findings

The goal of this systematic mapping study was to explore the current state-of-art in LSAD studies. The main findings are described next.

Although the area has publications since 2001, more than 40% of the studies (38 papers) do not present their research methods. In this systematic mapping we titled them by reported experiences. This nomenclature is commonly used in other systematic reviews to

studies written by practitioners who are knowledgeable on the subject. This demonstrates a significant number of cases reported by authors with a strong background on the LSAD as, for instance, Laanti (2014) [65] paper. On the other hand, we evidenced that studies on the LSAD area need more rigor and description of its research methods [29].

The LSAD definition is incipient, there are many different variables to compare. Thus this research question (Definition of LSAD in literature (RQ2)) is our most significant contribution to this work. There are several studies that consider distributed teams (54,26%), others complex projects (38,30%), multiple teams (10,34%) and, others even collocated teams (3,19%) using agile software development. Such issue is not a particular problem of this study. It is an issue in an area of software engineering as a whole, Wohlin (2014) [122] present three areas where improvements are needed to become more successful in synthesizing empirical evidence in software engineering: terminology, paper content and reviewing.

Although more than a half of the studies describe their projects, there is a fair amount of studies that do not characterize their projects (43 studies on 94 selected). Most studies indicate their projects as distributed geographically (35,1 %), but there are also studies that consider projects with multiple collocated teams (3 results). This may mean collocated teams also participate in globally distributed organizations, as the case of Microsoft Inc. (2007) [78]. Another characteristic from LSAD is a team with more than ten people, differently of Scrum Guide propose [115] from 5 to 9 people in the same software development team. About the companies' information category we reported the existence of others areas besides IT domain interested in agile methods. These studies that contain complex areas such as Airlines, Automotive, and Banks among others, show the spread of use of LSAD in other domains.

Considering agile methods Scrum and XP are the most used methods, followed by Lean and Kanban, proving a similar statistical data of Version one magazine [77] on LSAD usage, approaching the practice of theory. Scaling practices are workarounds to follow agility in LSAD scenario. This factor is strongly evident when studies reveal and propose new practices to achieve agility, as described in Section 4.2.4.These scaling agile practices foster the coordination and collaboration of multiple teams showing the inclusion of agile development to larger settings.

On the challenges, we identify 11 challenges grouped into three broad areas. The challenge category with the most mentions is related to *Process* with 55 studies of four challenges. The major challenge in this category is difficulties in working with agile practices in complex projects (mentioned by 32,98% of the cases), followed by difficulties in prioritizing and estimating the work (mentioned by 22,34% of the cases). These results show that LSAD seems to be harder to implement in complex and larger places. Open issues as automation testing in highly complex systems and how architecture influence on the size of the working done needed to be studied.

The second one category *people related issues* with 33 studies mentioned, we found the greatest challenge related to teams and customers not available to communicate and collaborate (referred to by 25,53% of the studies). This refers when a global team wants to adopt agile development without thinking about communication. This issue broke a principle of the Agile Manifesto when individuals and interactions need to be over than processes and tools. Senior people in one application is another challenge in LSAD. This issue can be solved with enough architecture to decouple the dependencies of tasks. Related to coordination of multiple teams problem we found in literature some good practices to adapt as scrum-of-scrums, agile architects, feature teams and so on. However, each case is different and depending on the context can be even more challenging, as in the study of Ericsson Company [89] several people distributing in a large scrum-of-scrums is hard to control and coordinate.

In the category *management and organizational* we found 12 studies. The LSAD challenges faced are related to traditional organizational structures and troubles with other areas that do not work with agile development. These two issues are open questions in literature. About scaling frameworks is a curious finding from literature cause instead to support LSAD, this framework denotes fragile and risky.

4.3.2 Threats to Validity

A systematic mapping aims to discover and extract knowledge of a particular topic guided by research questions. Although the process is systemic, several limitations can be identified in these studies.

Several precautions were taken to ensure the construct validity of the study. First, the author and the supervisors of this Master Thesis designed the protocol of this study, presented in the Section 4.1. We aligned the goals of the study, the digital libraries and the terminologies to be adopted. This step has undergone a rigid control analysis were chosen and databases that include the conference proceedings of the area as known how: Conference Agile (IEEExplore) and XP Conference (SpringerLink). But a threat to validity would be databases that were not chosen, leaving behind a few items that could be relevant to the study.

To ensure the accuracy of the expected results, we use pilot studies as an attempt to find these studies by applying the search strings in the databases. However, because the study answer on knowledge of state-of-art a wide area, studies with different terminologies in the string may not have been included.

The definition of the scope of our systematic mapping is our major limitation. LSAD containing different areas of knowledge. The definition made our research question extensive and not limited to empirical research.

4.4 Chapter Summary

Since 2001, the term LSAD has been proposed both in the academia and industry to identify large settings using agile. Curiosity is that even though this topic be investigated by several studies, literature is very confusing and with several gaps that were earlier presented in this work. In order to better understand the term and identify possible patterns to be a standard alignment studies, this systematic mapping aimed at investigating the state-of-art LSAD, featuring in four aspects: which adopted methods are used, what is the definition of the topic LSAD, who are involved and which challenges were faced in LSAD studies.

We selected 94 papers for our investigation. Thus, we believe that this systematic mapping has much to contribute to both the academic scope as for the industry. The results are useful for identifying the knowledge to the area so that new publications can be guided more appropriately.

Also, this systematic mapping contributed to identify characteristics and definitions for LSAD used in the next empirical studies. Next Chapter presents a field study in a large-scale agile company to identify how agility was developed during the transformation process of the company to this new approach.

5. FIELD STUDY AT A LARGE-SCALE AGILE COMPANY

This chapter presents the results of the Study 2, a field study to empirically identify how a large-scale agile company goes through the transformation to become agile. This chapter is organized as follows. Section 5.1 describes the research method. Section 5.2 presents the results of the field study. Section 5.3 discusses the results and threats to their validity. Section 5.4 summarizes the chapter.

5.1 Field Study Design and Settings

To empirically attend the Objective 3, first, we conducted a field study in one large-scale company. The semi-structured interviews were conducted in person on-site at the American and the Brazilian IT (Information Technology) offices. The interviews with the 10 American-based representatives took place in the fourth month after the agile transformation process kick-off (Dec'14) and was conducted by one of the researchers while the eight remaining ones with the Brazilian-based representatives took place during the ninth and tenth months (May and Jun'15) and were conducted by two other researchers.

The interview was focused on 4 major themes as follows: *i) participant's background*, we asked the participant to introduce herself and talk about her job description and responsibilities within the company; *ii) reasons for the transformation*, we asked her to elaborate on the reasons the company is going through such transformation that she was aware of; *iii) strategies taken*, we invited the participant to introduce us to the activities she was engaged on or has been communicated about related to the transformation; and *iv) challenges*, we requested the participant to express her challenges about the transformation process as a whole considering the organization background. The interview script is presented in Appendix B.

Participants were pointed out by a senior manager, the focal point of the research project that this study is part of, and then invited to participate on a volunteer basis. All 19 participants accepted our invitation; however, a US-based member called out in sick during our visit to the United-States of America (USA) office and had the interview canceled. Thus, we interviewed 18 participants.

All participants are managers at ORG [1] and are either a member of the CIO Committee board, a director or a development manager. They are working at ORG for at least four years and at most for 16 years. All participants are currently a member of the IT department, but 5 of the US-based representatives had worked in the business office either as a salesman, a manufacturing manager, or a business analyst. Also, out of our pool of

[1] A Globally-Large Scale Software Development Company

respondents, 8 of them are male (1 in the USA, 7 in Brazil) and ten are female (9 in the USA, 1 in Brazil). Table 5.1 summarizes the participants' job position.

Table 5.1: Participants' profile

ID	Job Title
P1	Manager on Process Improvement, Member of the CIO Board
P2	Quality Assurance Analyst (Process-based quality)
P3	Director on Process Improvement, Head of the CIO Board Committee and of the World-wide Agile Transformation Initiative
P4	Manager on Quality Management (Product-based quality), Member of the CIO Board
P5	Business Analyst Team Leader
P6	Business Analyst, Leader of the World-wide Business Analyst Community of Practice
P7	Business Analyst Manager
P8	Portfolio Planning Manager, Member of the CIO Board
P9	Portfolio Planning Manager, Head of the Roadmap Planning Department, Member of the CIO Board
P10	Director on Finance Application Development area
P11	Manager on Services Application Development area
P12	Director on Infrastructure area
P13	Manager on Finance Application Development area
P14	Manager on Services Application Development area
P15	Manager on Financial Services Application Development area
P16	Director on Infrastructure area
P17	Director on Services Application Development area
P18	Manager on Financial Services Application Development area

All interviews were voice recorded and later transcribed by each one of the interviewees. The shortest interview lasted 20 minutes and the longest 90 minutes, and they took in average 53 minutes. Our subsequent analysis was guided by grounded theory procedures [26]. We coded the interviews identifying factors for each of our topics of interest–reasons for the transformation, strategies taken, and challenges, until we have exhausted the data and reached a final set of merged factors. Coding was done by each of the interviewees and later reviewed by the author and the supervisors of this Master Thesis that is part of our research project. Next, codes were emerge from the interviews transcriptions into a single document (see Appendix C - Network of Results). In the next paragraphs, we present company background and the results coding.

5.1.1 Company Background

ORG is a large IT multinational company with offices located in 5 continents. The IT department develops software products to support the organizational processes. Demands to develop or to update these products come from the business departments, mainly located in the headquarters' office in the USA but with business, representatives spread out over 30 countries. IT development teams are distributed among the headquarters' office and in Brazil, India, and Malaysia. There also-also IT employees in China, Japan, Ireland, and Russia.

The IT department, at the beginning of the transformation initiative, used to follow a matrix structure based on business areas (e.g., sales) and IT functions (e.g., developers). Development assignments were mainly organized in projects that vary from the development of new products to the maintenance of legacy systems, and project teams would follow the waterfall model. Some Scrum practices were scarcely adopted in a project-based fashion to support project management. Software development processes would vary from formal (following CMMI Level 3 practices) to informal (defined by the project members upon their needs).

A well-known practice at ORG is still in place: an annual project roadmap is defined in December based on the requests made by business representatives and recorded by business analysts. Business analyst managers in conjunction with project managers prioritize the requests and define a set of projects to be developed throughout the year. Priorities are defined based on business impact and development costs and approved by a committee board composed of senior business and IT personnel who directly report to the CIO board.

Up to the beginning of the agile transformation, distributed software teams were formed to develop the elected projects. Members were assigned to projects based on their skills and domain knowledge, despite their physical location. Therefore, a project often has its roles distributed over several places. By mid-February, each team would receive a business request document. The software team would start working to translate the business into software requirements led by the software requirements analysis. These would consult with business analysts to clarify business requirements and, when necessary; business representatives would be invited to join the discussion. Project managers would monitor the project progress based on a set of organizational performance measures that would be reported to senior management on a regular basis. Results from these measurements were used to determine whether a project failed, attended, or exceeded its performance goals. Although ORG has gone through a major reorganization, this roadmap assignment process has not been reconsidered yet as mentioned in Section 5.2.3.

5.2 Results

Next, we present the first steps of ORG's transformation to become agile and its major challenges on agile transformation. We first describe the findings of the reasons why the company decided to move towards this new trend (in Section 5.2.1), next what was done so far to promote the change (in Section 5.2.2). And at the end, we describe on challenges related to this agile transformation (In Section 5.2.3).

5.2.1 Reasons for the transformation (Motivation)

ORG has started its transformation to agile about a year ago. It all started when a manager with experience in a large e-commerce company was hired to lead the development of ORG's Online Store. One of his major changes as a manager was to introduce agile to the Online Store development team. ORG's Online Store is an independent application and has a dedicated team to take care of it. For over four years the team improved its processes and the online store itself based on the expertise developed using agile to guide their development. A recent reorganization has put this manager in charge of the IT department, and as the new CIO his first worldwide announcement was that the company would go "agile."

One of the main reasons behind this decision was **to deliver faster to the customer** given that *"the faster, the better"*, as argued by the Manager on Process Improvement [P1].

One of the Managers on Services Applications [P11], the Manager of Finance Applications [P13], and yet one of the Managers on Financial Services Application Development [P18] also reported, respectively:

> *"Agile aims to deliver faster to the customer. Instead of spending ten months specifying requirements, then five more months coding them and five more testing, the customer will be able to quickly see added value to the application in a shorter time." [P11]*
>
> *"It would be great if we could deliver faster since we have this daily pressure to reduce costs, as any other IT company also has." [P13]*
>
> *"(...) requirements that take four months to be developed and two more to be deployed, we want to change such scenario and deliver more quickly and often to the customer." [P18]*

The current long time time-box between the customer requesting a software and receiving it is also an important motivation for this transformation, said one of the Financial Services Managers [P15]. He also said:

> *"Agile means frequent deliveries, causing the impact of the customer's decision to be handled in smaller slots of time and allowing changes during the process."*

The same reason was also mentioned by one of the Directors on Infrastructure area:

> *(...) there is a gap in the customer expectations since IT takes too long to deliver what is requested." [P16]*

Agility as **an attempt to be more innovative** is another important reason according to one of the Directors on Infrastructure area. He mentioned: *"We seek agility as a 'driving force' to innovation. We need to (...)"* [P12]

The transformation to agile also aims **to reduce the communication gaps between business and IT** as mentioned by one of the current Directors on Infrastructure area:

> *"[Agile] expects closer interactions between business and IT, having the customer more involved. This approximation is welcomed, and we do need it, but it can also be risky if the customer does not get engaged." [P12]*

The Business Analyst Team Leader added based on his large experience working on the business department:

> *"We can not forget that a lot that takes place in an organization goes through informal channels. Thus, this agile model will likely be good to make informal things formal." [P5]*

In addition, participants also reported that another important reason for adopting agile company-wide is **to be more flexible to changes**. For instance, *"The requirements are defined early in the year and sometimes they are obsolete when development starts."* as reported by the Director of Services Application Development area [P17]. Therefore, it is important that the development teams have a closer interaction with business to frequently discuss and re-prioritize what requests are more important at a given time, as mentioned by one of the Managers on Services Application Development area:

> *"Flexibility means we can change requirements with a cheaper cost and more easily." [P14]*

5.2.2 Strategies Taken

Once the decision has been made, the CIO and his committee board discussed a set of actions to be taken to promote the transformation and engage management in developing them. At the mid of the fourth month, a worldwide announcement was made to the entire IT department marking the kick-off of the agile transformation. At this time, teams were informed that by the end of a 12-months period all projects had to be "acting agile". We report next all actions taken during the four months of discussions at the executive and strategic levels and the seven months of changes at the operational level.

The company prioritized **reorganizing the former organizational structure**, which was a matrix structure. As reported by one of the Managers on Services Application Development area, *"The new structure is defined now by business areas that have their internal functions."*. He supplemented:

> *"An interesting change we made was that the executive leadership was completely realigned. We have now a mix of new and 'ancient' people of the business department that are within a portfolio area to ensure the new ideas will be welcomed but that we will also not lose important implicit knowledge." [P11]*

"Leadership is slowly realizing how positive this change will be. It is a matter of internalizing it and later sharing with their team members", added the Director of Process Improvement, the head of the worldwide,' transformation initiative [P3].

One of the main advantages of such organization change and moving more towards a model in which teams are responsible for what was assigned to them is *"To have clear accountability for decisions and deliveries. In our matrix form, we had too many people in control and no one controlling anything."*, said the US-based Director on Finance Application Development area [P10].

After the major discussions about the organization structure, the focus turned to **training teams on agile practices**. Team members start than to learn how to adopt such practices in the context of ORG. Several presentations and debate sessions are organized by managers, development leaders, and architects to promote discussions on how agile within an enterprise with complex applications and that is globally widespread can take place. For instance, one of the Managers on Services Application Development area reported:

> *"Local presentation sessions to all members of a certain office were organized to take place worldwide in the same week in which teams were already going through the transition were motivated to report their experience to others as a way to encourage the adoption of agile and to share good practices." [P14]*

At the same time, some members were encouraged to carry out the Scrum Master's certification and use their knowledge to help other project members to align their actions with the transformation initiative, as mentioned by one of the Managers on Financial Services Application Development area: *"Some members were already trained as Scrum Master, but on Monday four more people of my team will be going to a training to learn new things. They are very motivated about it. They enjoy learning new things."* [P18]. Also, managers are also learning along with their team member as mentioned by one of the Directors on Infrastructure area:

> *"We are receiving several trainings, both at the technical and at the managerial level on agile practices and we have been learning from each other as we put them in practice." [P12]*

Communities of Practices (COPs) was a key mechanism behind the success of the large-scale agile implementation in Ericsson, helping teams to mitigate some pressing issues of the transformation [89], and also a source of motivation to ORG's employees too, argued the Business Analyst who leads world-wide the Business Analyst Community of Practice [P7]. The Manager of Quality Management added:

> *"As important as training people is sharing what we are learning with one another, and here is a good way to do it. I hope they do not cut this practice off." [P4]*

Another action taken was **to identify and prioritize pilot projects** to serve as testbeds for assessing the side effects of the reorganization and overall changes promoted during the initial months of the transformation as commented by one of the Directors on Infrastructure area:

> *"We are prioritizing some projects to be pilot projects based on our understanding that they are more prone to be adherent to the agile philosophy. We have a priority list, but we still keep some legacy projects running on 'traditional' [Waterfall] processes." [P12]*

Also, these pilot projects have been closely **supported by coaches hired to assist the company**. *"Local coaches were hired to support each of the IT offices. We looked for experienced professionals who have faced similar issues than ours in other large corporations."*, said the Director of Process Improvement [P3].

The benefits of having experienced coaches working with the teams are recognized as follows: *"They can more easily and quickly to recognize ways to enable the transformation."*. One of the Managers on Finance Application Development area supplements by saying:

"The coaches are already working with some pilot projects supporting refining the organization structure when projects involve other projects and teams. You know, we have all these dependencies (...)" [P13]

Another reported advantage of having coaches supporting the transition process is to ensure the agile practices are correctly used as exemplified by one of the Managers on Financial Services Application Development area: *"The coaches participate in all the ceremonies and help us understand if we are doing it right. For example, they helped us to revisit our team structures and set up the feature teams in a way that makes much more sense now."* [P15] It was also argued by the Director of the same area, located in the USA:

"We are finally learning how to do things. We do understand what it means to do a stand-up meeting. We are not playing anymore." [P10]

The teams are also aware of the role and extent of responsibility of the coaches as reported by one of the Managers on Services Application Development area:

The coaches are responsible for helping the organization to make the transformation happen, but they are not responsible for the transformation itself, this is a responsibility of each portfolio area. They provide us with the tools and helps us with their expertise. We have to make it work. [P11]

Another major action taken was **to refine the team structures** based on the new organization structure. The teams began to take shape as 'feature teams' as described by one of the Managers on Financial Services Application Development area:

"Before this agile transformation, I usually had a team with resources allocated to it, and the team members worked on the project, from day 1 to the delivery. When another project was up, then maybe I would have another resource allocated to work with me to deliver it. Now, I have an organized structure responsible for a major feature, and this structure receives demands from several different portfolios and systems programs, and we try to prioritize the demands according to the customer needs, by sprints, not as it used to be." [P15]

Still about the refinement of the team structures, one of the Directors on Infrastructure area [P16] and the Director of Services Application Development area mentioned [P17], respectively:

"We have teams still working on a project-based form but trying to create new structures like feature teams. It is not an easy transition, for example, earlier today I discussed with a colleague about the testing area. The performance testers

> *are still separated from development. This is something we have to change for us to be completely agile. We are not sure how to do it, but I think we are going on the right track." [P16]*

> *"In the past, a project manager would receive reports from all members allocated to projects of a certain portfolio. He would manage how these people were working. Now, with the agile transformation, these managers are responsible for the portfolio management as a whole, they need to know how the features the portfolio is responsible for are progressing." [P17]*

The US-based Director on Finance Applications believes that: *"It is great to have a multidisciplinary team, or this feature thing–I can not recall the name–where everyone needs to know all skills and to be trained in all aspects. This will make people more focused and committed. I like that!"* [P10]

To achieve that, the company is also *"refining the job descriptions and revisiting skills and competencies they expect each role to have"*, as commented the Business Analyst Team Leader [P5]. The Business Analyst also reinforced:

> *"It is important that we reconsider the competencies each role requires in this new model. For instance, we had six levels of business analysts, now we have business analysts, and business software analysts merged in one single role, we are still revisiting their set of expected competencies and skills." [P6]*

Further, some teams identified the need to **mix agile practices from different methods** to compose what they need to support their work as highlighted by one of the Managers on Financial Services Application Development area:

> *"Some teams are adopting only Kanban, some only Scrum. But several teams are using 'Scrumban,' as we named here. This is when a team only uses the concept of Sprint from Scrum but adopts the Kanban way to work." [P18]*

5.2.3 Challenges

Despite the initial strategies taken by the company, there are still open questions in the opinion of the managers. They are aware that in such a transformation process one can expect that as decisions are taken and implemented new issues will come up. Certain actions will be taken and will result in positive outcomes; others will have to be reconsidered. They agree that this is part of any maturation process. However, there are key points they consider critical, mainly due to their large-complex globally distributed configuration, and that are still to be discussed as presented next.

The biggest concern reported by all participants is the transformation occurring in a company with **a complex ecosystem of software applications.** This is of concern given the large number of interconnected applications that attend multiple business areas (e.g., sales, manufacturing, finances, HR, etc.) and legacy products maintained by several organizational departments and by highly distributed teams. Such concern was expressed by the US-based Business Analyst as follows: *"Interlocks–as interdependencies are mostly named at ORG–will be a big headache."* [P6]

One of the Managers on Services Application also mentioned: *"We have over 2 thousand applications forming a well-connected web, an incredibly complex data stream that is globalized and yet serving regional needs in several cases where our client–several company's departments–have no globalized processes."* [P11]

Another concern about the high amount of interdependent applications is how the 'feature teams' are being set up. Some managers understand that a team at ORG will never be completely independent of other teams given the dependency among the applications. Thus, releases have to be coordinated. One of the Managers on Services Application Development area reported:

> *"All applications are very well interconnected, then now we have a concept of release. In theory, it would be ideal to have a release every three months, but in practice, this can vary because I depend on others to get my feature done and some projects have still not been migrated to this release idea." [P14]*

One of the Directors on Infrastructure area also mentioned about this same issue: *"If I need to automate a process, the process will probably hit several systems to complete the task. Thus, the automation needs to have all applications with a 'ready status' to be able to be delivered."* [P16]

The **limited spread of the transformation to parts of the IT department** is the major challenge of several managers. For instance, training and coaching have been only assigned to support development teams. Infrastructure and services teams are having to provide services to development in an 'agile fashion' without having been included in the transformation initiative as reported by one of the Directors on Infrastructure area: *"It is necessary to look at the software life-cycle as a whole, including the question of provisioning infrastructure. The training offered is very good for the development team, but infrastructure has no clear guidelines defined yet so we feel kind of lost"* [P12]. He added:

> *"The technological complexity of the environment, I think, is another important factor. We are talking thousands of database, thousands of applications [silence]. Then, when I have to talk about a transformation that will affect, for example, 'refreshing a hardware structure' to all applications of a portfolio, then I have to ensure that I will be able to handle such major change, and this has not been discussed yet but I am already facing such an issue." [P12]*

The other Director on Infrastructure area is concerned about the available infrastructure itself:

> *"How can we support constant deliveries if we are not sure which are the infrastructure needs for that and we do not have enough persons to work or even servers to support the applications?" [P16]*

On the other hand, the Director of Finance Application Development worries that *"the quality of the service provided by production support people can be jeopardized since their way of work will not be in sync with development. These guys still have to be fast to provide solutions to live issues, but they might not know how to interact with development teams anymore since they are not aligned with what the coaches are doing."* [P10] The Business Analyst has a similar worry:

> *"We need to ensure consistency in some level, so we stay functional and make people's life easier." [P6]*

To make the customer adopt agile in their processes is another concern related to the transformation boundaries within ORG. For instance, the Director on Finances Application shared:

> *"What we need is a proper customer involvement. This is a 'big sticking part of all this.' We need a customer who can use requirements in an effective way. They have to be committed to doing it, and we also need to learn to be disciplined about it." [P10]*

One of the Portfolio Planning Managers added:

> *"We urgently need a centralized solution for business personnel to make their requests and prioritize them." [P8]*

The Financial Services Application Manager commented:

> *"Customer representatives are aware that the company-wide is going agile, but are not directly involved in the actions taking place so far, so I guess that they have not yet realized that they will have to be more active overall, to respond faster to our requests and to more quickly consider what are their priorities, to be able to handle new deliveries at a faster pace, among so many other things." [P15]*

The Business Analyst Team Leader argued:

"What customers have to realize is that we need a day-to-day proximity with them throughout the development cycle but when coding the contacts will likely slow down. We do understand that interacting with IT people takes away from their daily job duties, but in the end, we are providing them with solutions that will help them, in the end, to better do their work. So they need to find a balance." [P5]

"We can work with PO [product owners] proxies, if necessary to make it work. We are okay with that" concluded the Process Improvement Manager [P1].

The Manager of Quality Management supplemented:

"We believe that by involving more the customer we can increase the overall customer satisfaction with IT services. This is what my team is looking for in this transformation: to have better results in our periodic customer satisfaction surveys." [P4]

The Director of Services Application Development area highlighted:

"We need the customer closer to the IT department, and fast." [P17]

The **annual roadmap** to decide on a budget for the departments is also a concern. The Manager of Finance Application Development area is afraid ORG might "crash" in the coming year:

"(...) We are still learning how to distribute our projects on an annual basis now (...) In fact, what we need is to learn how to prioritize requests as the year goes by so the customers will always get what they need faster. It does not matter what model we are following; we need rules to decide on what adds more value, so prioritization will be made easier and clearer." [P13]

The US-based Finance Director argued: *"Agile fits well when there is a lot of unknowns' but it cannot be good to fix contracted models like ours."* [P10]. His office partner mentioned: *" Our budget is fixed, we will likely never change that. Effective prioritization is the key."* [P10]

In addition, one of the Directors on Infrastructure area reported:

"The company has an annual roadmap, so we plan by the resources and budget we receive; I still have no clear vision of how it will look like really with agile." [P12]

One of the Portfolio Planning Managers, responsible for the annual roadmap planning confessed:

> *"We do not know we have to change the way we do our budget forecast, but we still have not found a way and the clock is ticking. It is dependent on the business budget funds, so it is not just changing the process, it is more complicated than that." [P9]*

The Manager of Process Improvement added: *"We need to work based on priorities."* [P1] Her colleague supplemented: *"(...) We just are not sure how to move from a cost-based model to a priority-based one. I think the first step is to have a demand supply staff, like a roadmap change management board, to ease things down next year. Then we buy ourselves some time."* [P8]

The **global distribution of the teams** is another factor that concerns most of the managers. ORG started in the USA, later created an office in Brazil, next in India and Malaysia. There are also groups of IT professionals allocated along with business offices in Ireland and Russia, and other places. Over the last 13 years, the teams went from co-located to distributed between two countries (e.g., USA-Brazil, USA-India) and finally to distributed over three continents (e.g., USA-Brazil-Ireland-Malaysia). Differently from a large number of agile companies in Europe that are distributed up to 5 countries within the same time zone or, at most, 1 or 2 hours apart, at ORG teams have the challenge to have to coordinate with remote teams members that are often 8-14h distant from one another. This concern can be perceived in the following excerpts by one of the Directors on Infrastructure area, the Manager of Finance Application Development area, the Manager of Services Application Development area, respectively:

> *"We work with distributed teams. Everything we do is distributed. My team is distributed over four continents. It is very challenging to have a synchronous meeting, even when we make an effort to compromise our working hours. Not even to mention that for more that we try, it will never be the same than working side-by-side." [P12]*

> *"There are just too many people over too many countries around the world to be flexible and agile in our processes effectively." [P13]*

> *"We have five thousand people distributed among Americas, Europe, and Asia, and this will not change because it is how our business survive. We need to be where our end-client is (...)" [P14]*

Most of the applications at ORG born when the company was still small and with a single office. Despite, they have been maintained by teams of senior professionals that are well aware of how the applications work. As people retire, new members are moved from junior to senior positions and being assigned to be in charge of keeping the applications working. With this company-wide transformation, **old technologies and legacy systems** become a serious concern as reported by one of the Portfolio Planning Managers:

"We are still learning about whether agile is fit for all projects we have. Some might never be able to go agile like the legacy systems. We are still not sure yet." [P9]

The Manager on Services Application said:

"Our applications were not designed thinking of agile methods. They are 10-15 years-old with legacy code that is tough to have a feature team responsible for it, for example." [P14]

The Manager on Finance Applications said:

"In a complex environment with different systems communicating with each other and integrated with old technologies, it is insane to try to move legacy systems to agile." [P13]

Other concern is about **countries' laws and fiscal year's budget** as mentioned by the Manager on Financial Services Application:

"The shares must be tendered, and we are still managed by a quarter to quarter within the fiscal year. Management is tied up to the fiscal year, which ensures pre-delivery and planning visibility different from what happens in agile, I think. The cost is likely discussed for the short-term deliveries, without considering a closed scope." [P15]

The Manager of Finance Application Development reported:

"I have some demands that vary greatly according to the [Brazilian] government and the law under ORG is hosted here [in Brazil]. For instance, if the law changes, we need to change to be compliant with it. Also, the [Brazilian] government is always changing rates. Thus we have to adjust the systems in a very frequent fashion." [P13]

Agile evangelists is also a concern because they often believe only in agile in itself, not in a process that can be effective through a transition and that represents a culture change. Transformation is a slow process, so many people do not believe that teams can be useful in cases where the company is still running some projects on 'traditional' processes. This issue is mentioned by the Services Application Development and Financial Services Application Managers, respectively:

"Evangelists do not believe that change can occur and be effective. They just believe in what is described in the agile manifesto. They end up damaging the transformation process. We know it will be slow (...)" [P14]

"(...) a problem that I see today is that agile is a religion, there are many people who strongly believe that it can even cure cancer [laughs]." [P15]

Lack of formal documentation for requirements is another issue cited by a few participants as showed by the excerpt below:

"So, if you do not have any documentation, it has happened several times to us, to have projects that go back and forth, and then our customers say no–customer here is always internal departments. They say 'it was not what I wanted,' then we say 'but it is what you documented' so we have how to defend our position. We always had this fear to miss formalities despite all the interactivity that Agile offers." [P11]

The adaptation and the redesign of tools to support work throughout the organization is a less concerning issue but something the company knows that will have to be considered sometime sooner than later. For instance, the Process Improvement Manager mentioned:

"We will need tools to support virtual stand up meetings, visualization of data exchanged among people to facilitate comprehension of what is going on given that most team members do not work with co-located colleagues and have large time zone differences, and so many others that I could spend the entire hour listing here." [P1]

5.3 Discussion

As results, ORG has two particular configurations in relation of other large companies that have already been through this transformation are: (i) it has an ecosystem of applications that are dependent on each other and (ii) it has large teams globally distributed around the globe with no or little overlapping work hours. Both characteristics make ORG's situation unique, thus likely requiring specific measures to leverage success.

This opportunity brought several findings not known in the literature, adding a complex environment in a traditional structure geared towards to large-scale agile development involving all continents. Thus helped us to understand the subject in a detailed empirically manner.

As other LSAD companies [120], ORG has different motivators to aim to 'become' agile. We could realize that the most important reason why ORG decided to go Agile is similar to those reported in the literature (e.g., [21]): to deliver faster. This is one of the main characteristics of agility and proven to be still a common issue in the industry.

The reorganization of the former organizational structure reported by some of the participants is a natural reaction when a large-scale company is moving to a new paradigm (e.g., from Waterfall to Agile). When the company was using the Waterfall model, it was fit to have a matrix structure. However, when the migration to LSAD started it was easy to realize that the organizational structure needed to change to support teams working based on features using evolutionary and iterative development.

Training team members on agile practices is an incisive step to the large-scale agile transformation that includes qualifying people on the 'basics' of the new mindset (e.g., self-training, trial, and error, changing priorities) [45]. Our findings also indicated that agile training was one of the first strategies to be taken at ORG, despite a large amount already familiar with agile practices. The training offered not only the discussion of agile concepts but also how to put them into perspective at ORG based on the company context and background. This was cited as one of the main benefits of having the training sessions approaching the success factor found in LSAD companies [29].

Pilot projects are common strategies to test whether something is working and it was also used by ORG. The pilot projects in LSAD environments helped create confidence in agile way of working and the general acceptance of agile in the teams [29]. Also, this was considered one of the successful factors to support the transformation in LSAD companies such as in Gap, as reported in [47].

Coaches hired to assist the company is among other of the successful factors cited by Gandomani (2014) [45]. To provide training on agile helped people to become more positively inclined towards the new way of working, and improve the chances to be agility in LSAD [44].

Customizing the agile methods and practices was often seen as a necessary step in the LSAD implementation [29]. Applying agile at scale teams need to change agile practices to fit the large and distributed environment [111]. This allow teams to become innovative and perform well. Thus also took place at ORG and was also pointed out as a strategy to LSAD implementation.

On the challenges reported here by the ORG managers, some of them are similar, and others are new to literature as discussed in the coming paragraphs.

The concern related to the complex ecosystem of software applications that ORG has present in its daily activities in still an open question in the literature for LSAD. We found challenges and limitations reported related to the usage of agile practices in a distributed environment (e.g., [119]) and on a project with dependent projects (e.g., [44]), but none referred to the same complexity as in ORG (having both aspects altogether on the same project, for instance).

The limited spread of the LSAD discussion to the IT department only is a new concern. For being a large company with several financially independent departments, the

implementation has been so far just been discussed among IT personnel. Business people are aware of the transformation, but they still are not involved in it in practice. Dingsøyr and Moe (2013) [30] proposed a research agenda for LSAD in which they cite that customer collaboration is still an open issue in this setting. ORG is, facing this issue and little is know in literature to help them overcome this challenge.

As per the annual roadmap concern, we found a study from Borland (2009 [121] that showed an annual roadmap was one obstacle, having the company moved to a catalyst solution. The new strategy has been established based on a common understanding of the involved parts, and the agility was achieved across their product delivery value chain. ORG knows this is one of their key topics of concern and that changing it involves more than improving software engineering processes and task allocation.

The global distribution of the teams is 'big' challenge to the LSAD at ORG. Managers have already realized that there are some practices that do require close coordination and fast decisions have to be made for short releases be a feasible reality at the company, and that long distances with the lack of overlapping hours make them discussions almost impossible to happen. Korhonen (2013) [62] was identified that distance does make it hard for agile teams to work but that the distance itself does not imply or affect the quality of working practices per se, contradicting the perception of ORG managers.

ORG has no clear direction about the large-scale transformation of old technologies projects and legacy systems. Literature reports successful cases when implementing agile in such scenarios. For example, Shah and Nies (2008) [105] reported practices to move from large legacy applications to agility, as follows: To have and effective agile attitude with training, to establish a plan and work incrementally, to have a dedicated role to inspect and adapt the process, to break monolithic projects into smaller projects, and also to try initiate something already accepted rather than something new.

Lack of formal document for requirements also was found in a large-scale traditional projects reported by Cristal, Wildt, and Prikladnicki (2008) [27]. An open issue in literature, on the other hand, local laws and fiscal year's budget, as well as agile evangelists, are still open challenges not yet mentioned in literature.

To became a large-scale agile company is not a simple process to go through. However there are reports in the literature describing successful factors for this transformation (e.g., [29]), there are still open questions when the transformation took place in a large-complex globally distributed scenario as reported in this paper and discussed above.

5.3.1 The presence of agility

In the discussion three categories describe the presence of agility in Fontana et al.'s (2015) [42] study (Progressive Outcomes Framework). In Figure 5.1 we designed the practices learning category as being in agile trial and in agile learning characteristics. The second category identified in our study was customer Relationship. In this category only customer awareness of team level was identified by our field study. In the organizational support category we found agile commitment level because the overall initiative of the company's adoption of agile methods. We did not identify other results on the categories as team conduct, requirements, software and deliveries. To respond all characteristics of the Progressive Outcomes Framework, we performed another study with LSAD teams, a focus group study presented in chapter 6.

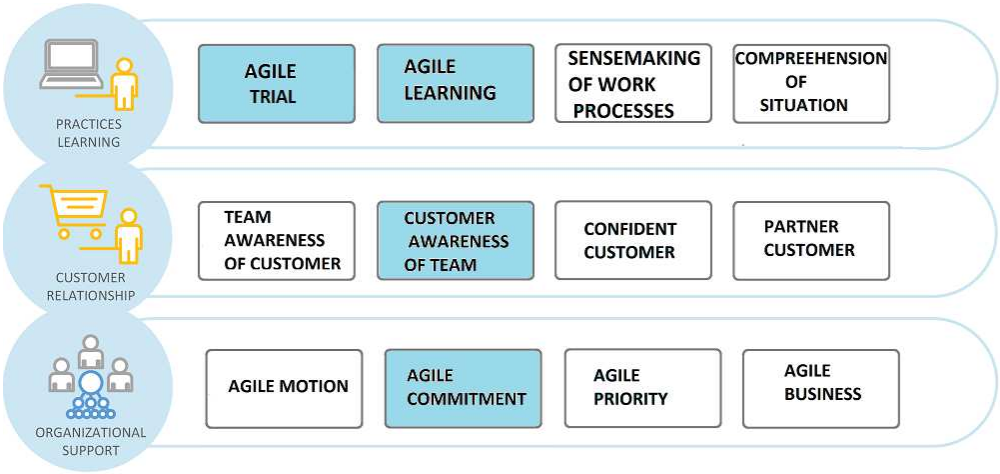


Figure 5.1: Representation of the presence of agility in a large-scale agile company.

5.3.2 Threats to Validity

This study includes some threats to validity. Firstly, we interview managers, directors, and analysts. In this study, we do not consider the opinion of softtware development teams because we would like in the first phase to identify only a macro view on the perspective of the company as a whole, and we just interviewed people directly involved in the agile transformation. About the opinion of software development teams, we performed another study using a focus group method, see in Chapter 6.

There other limitation about the number of respondents in this study (18 respondents). We do not known if this number is enough. But we conducted interviews with all people directly related to the agile transformation of the company. We also have sought a theoretical basis for a better interpretation of the results. We used methods of collection and

analysis known to mitigate the bias of the researcher, following Singer, Sim and Lethbridge (2008) [108] recommendations.

The last threat to validity is the subjectivity of data classification. Since three persons performed the qualitative analysis and the supervisor looked the review of the codes created. Thus, Grounded Theory was used to reduce this threat. We follow Strauss and Corbin (1990) [113] guidelines of cyclical process of collection-analysis-reflection.

5.4 Chapter Summary

This chapter presented a field study about agility in the initial steps of ORG' company-wide transformation to agile. Our aim was to identify through interviews what were the main reasons that motivated ORG to enter this journey; the strategies performed so far to implement the LSAD and to name the majors challenges foreseen by 18 seniors management to succeed in such endeavor. This results contributed to understanding to what extent agility in LSAD.

Despite the challenges and the early stage of the LSAD process itself, we could observe that ORG is following the same direction path of most successful change initiatives as reported in the literature. However, we do know that the challenges revealed by our study will have to be discussed and handled by ORG when the time comes.

The results of the field study led us to conduct a further study with the teams working on large-scale agile companies to identify the presence of agility in teams. In this study we identified the presence of agility using the Progressives Outcomes Framework.

6. FOCUS GROUP STUDY WITH LARGE-SCALE AGILE TEAMS

This chapter presents the results of the Study 3, a focus group study to empirically identify whether agility is present in large-scale agile teams and the challenges faced by such teams to be agile in such settings. This chapter is organized as follows. Section 6.1 describes the research method. Sections 6.2, 6.3 present the research executions, backgrounds, results, and discussions from each company studied. Section 6.4 discusses the threats to validity of this research. Finally, Section 6.5 summarizes the chapter.

6.1 Focus Group Planning

We performed a focus group study to answer the follow question: *how agility is present in LSAD?* The study involved an homogeneous group of participants from large-scale organizations with 50 or more people, or more than two teams [33] (defined in the systematic mapping review Chapter 4), and from different domains. The name of the companies was not disclosed due to the agreement made with the author of this Master Thesis and companies. This study was performed following recommendations defined by Singer, Sim and Lethbridge (2008) [108], Krueger (1994) [63], and Morgan (1996) [81].

To achieve our goal we defined the following Research Questions (RQs):

- RQ1: Is agility present in LSAD teams?
- RQ1.1: What are the challenges faced by LSAD teams to achieve agility?

The focus group research process consisted of the following phases (Figure 6.1): (1) **Focus group sessions planning** - To define the purpose and research questions of the study, to elaborate questionnaire based on research questions, to organize the materials to be used during the session, to estimate the duration of each session, and to select participants and to schedule sessions. (2) **To conduct focus group sessions** planned, to take notes, records, and after each meeting to transcribe all data collected. (3) **Consolidation of results** - To examine and to code data transcribed in the second step using Grounded Theory procedures, and finally, report these results. Next paragraphs details the process.

Based on the research questions already presented, we have identified the need to develop a questionnaire to facilitate the understanding of each participant about the purpose of the study. As aforementioned, we used the Framework of Fontana et al. (2015) [42] to develop the questions. We had Author's authorized to use the Framework in our study.

We estimated one hour and a half duration to perform each focus group session. We performed in two phases the session, Phase 1 - Individual perception and Phase 2 - Discussion and collective perception.

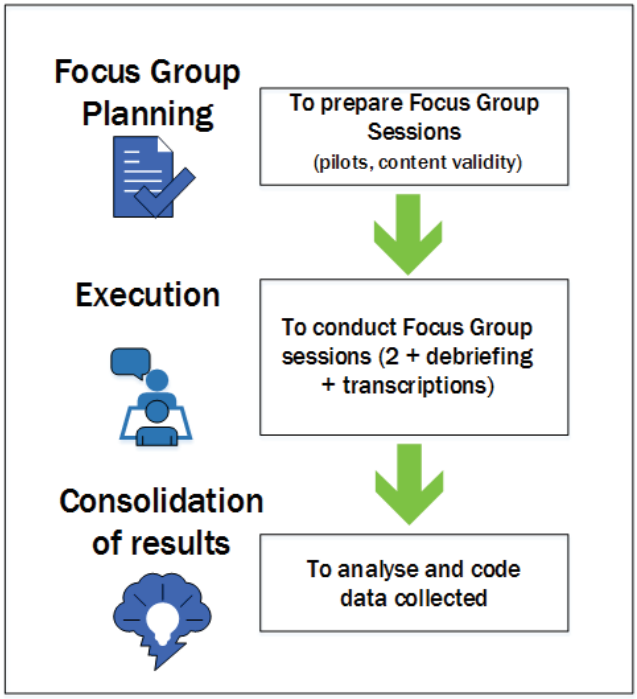


Figure 6.1: Phases of the focus group research method

- Phase 1 - Individual perception: In this phase, the moderator presented study goal and distributed a questionnaire to the participants. The questionnaire has seven questions related to the Progressive Outcomes Framework aforementioned. This phase was estimated to take 30 minutes duration.

- Phase 2 - Discussion and collective perception: In this phase, the team discussed the findings related to answers given in the questionnaire. This phase was estimated to take one hour.

Six pilots studies were conducted to evaluate the study protocol feasibility, time, and adverse events. First, we interviewed five persons who working on LSAD teams. And second, we conducted a pilot focus group study with seven graduate students. In this pilot focus group, we created an LSAD hypothetical usage scenario to simulate an LSAD team. All professionals and students expressed their opinion on the study. As a result the following notes were taken to improve the focus group study:

- We identified choice changes of the participants by comparing the results from phase 1 and 2. For instance, in the features disclosure (i.e. fourth category of the framework), all characteristics from that level were identified by all participants. However, the participants chose not to identify any traits in phase 2. We understood it as an outcome of the study. The thinking individually is different from thinking together by the team. It helps to characterize such study is the team's interaction and perception.

- To collect as much information as possible, we define 58 questions in the questionnaire. However, the participants found the length of questionnaire inadequate to be

answered in a short period.Thus, we decided to change questionnaire length keeping the focus only in the main characteristics of the each category with a multiple choice.

Appendix D presents the questions reviewed and applied during the focus group sessions. These questions are in Portuguese language because participants were not fluent in English. Also, we organized the materials used to conduct the study: post-it notes, sheets of cards, pens, questionnaires, study goal printed for each participant, two audio recorders and blank pages for notes.

We selected candidates of LSAD teams from different companies. Five companies are located at Porto Alegre and the metropolitan area and, two companies from Curitiba. Table 6.1 details the candidate's participants for our study and the status of conversations with each Company. At the end, we performed the study with two companies from Porto Alegre.

We executed the study in the way we planned. The session moderator conducted the study accompanied by two researchers. While the moderator guarantees the execution of the study, others researchers controlled the time, took notes and delivered the materials to the participants. The Figure 6.2 shows how we represent the results during a focus group session.

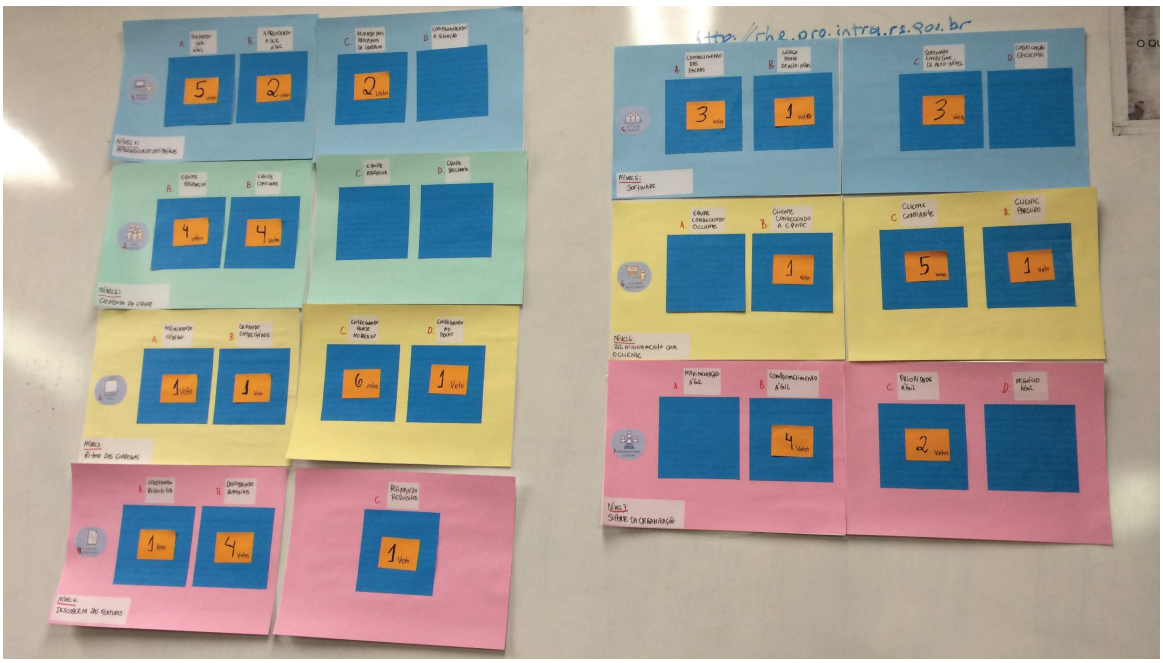


Figure 6.2: Sample of results showed in the focus group sessions

Once the transcriptions were done, we started the data analyzis process inspired by the grounded theory procedures of Corbin and Strauss (2014) [26].We used MAXQDA [20] to support the creation of codes. First, in open coding stage, we separated the data into categories and subcategories. Subsequently, we performed the axial coding to obtain the understanding of what the data meant. In the end, we obtained the selective coding, described in Section 6.2, and 6.3, which presents the results of this study.

Table 6.1: Candidate participants of the focus group study

Comp.	Description	Status
A	An American multinational information technology company providing hardware components, software and related services to consumers.	Invitation sent. No confirmation.
B	An American multinational computer technology company that develops, sells, repairs, and supports computers and related products and services.	Invitation sent. No team has confirmed to participation in the study.
C	Rio Grande do Sul's public department.	Study accomplished with one team.
D	A Telecom Company having the largest development team in Latin America, providing manufacturers of telecommunications	Study accomplished with one team.
E	A Company who develop projects of software development and outsourcing processes in retail, finance, health-care, government and others.	Invitation sent. No team has confirmed to participation in the study.
F	The biggest government-owned corporation of IT services of Brazil.	Initial decision done to Senior Manager who confirmed interest. No (research) budget to perform the study
G	The largest Telecom company in Brazil.	Initial decision done to Senior Manager who confirmed interest. No (research) budget to perform the study

6.2 Focus Group at Company A

We conducted our first focus group study in August 2016. This study was perfomed in a governmental justice department in the Rio Grande do Sul, Brazil. All software area of the company has about 100 people between permanent and outsourced employees.

The project of this company aimed to implement an electronic process to the Special Courts and Treasury at the Rio Grande do Sul. This project began in 2013 with a small development team, but the team gradually grew. There is no forecast date for the end of this project. About 40 people are working on and the teams involved are system development, architecture, support (internal system team), DBAs, and infrastructure. Also, there are peo-

ple of other outsourced companies in this project. All teams (architecture, support, DBAS, and infrastructure) adopt agile methods.

We collected data from one LSAD team of this project. This LSAD team has 18 people and is separated into two smaller teams with nine people each. There is development, testing, PO, Scrum Master and support roles out. Of the 18 people, 12 are outsourced. Due to the teams' availability, we conducted the focus group with six members. The team adopts Scrum and Kanban in their work processes, and the current sprint's goal is expanding the scope of the project to allow a greater number of locations and users to use the electronic process. Developers have different levels of agile experience. Three developers had less than three years of working experience, and other three have more than three years' of work experience. Table 6.2 presents the participants' details.

Table 6.2: Participants' profile - Company A

Participants	**Job Title**	**Agile Methods experience**	**Working experience with the team**
P1	Developer	4 years	2 years
P2	Developer	5 years	3 years and 4 months
P3	Developer	5 years	2 years and 4 months
P4	Developer	4 years and 10 months	3 years and 10 months
P5	Developer	2 years	1 years and 2 months
P6	Developer	3 years and 10 months	3 years and 7 months

6.2.1 Results from the Questionnaire

In Phase 1 the author of this Master Thesis (moderator) presented the study goal and distributed a questionnaire to the participants. The results from Phase 1 are shown in Figure 6.3. Each question in the questionnaire represented the seven categories of the framework. In each category, the characteristics were described so that the participant could choose which characteristics belong to their large-scale agile team according to each level. To identify the responses of each participant, Figure 6.3 also describes the participant ID: P1, P2, and so on.

In the **practices learning category**, the majority of participants believes to be part of an agile trial characteristic with five choices a total of 83.33% (choices from P1, P2, P3, P4, and P5). Following agile learning and sensemaking of work process characteristics, we found two choices each. Comprehension of situation characteristic had one choice from P6.

In the **team conduct category**, we found a draw on the responses from participants. The responsive team had four choices (from P1, P2, P3, and P4), and also 4 (choices from P1, P2, P5, and P6) choices for confident team characteristic. Assertive team and sparking team characteristics were not indicated by the participants.

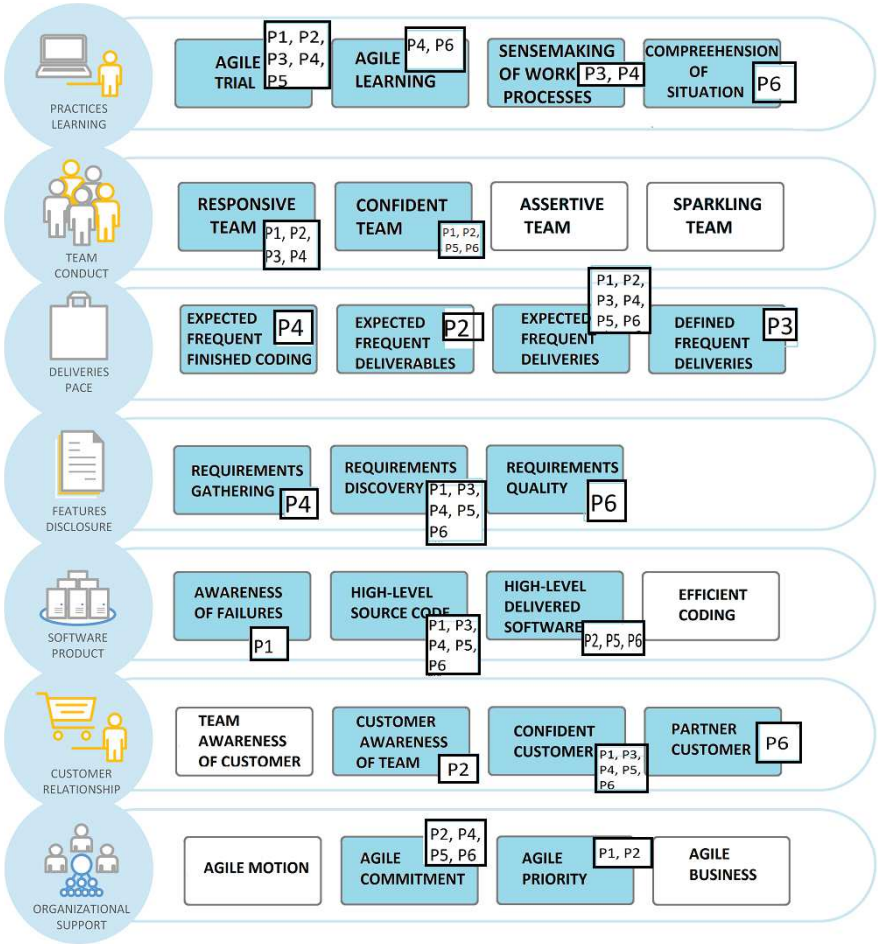


Figure 6.3: Findings from the questionnaire - Company A

In the **deliveries pace category** all participants chosen expected frequent deliveries as a characteristic that represents their frequent deliveries to the customer. The other characteristics: expected finished coding, expected frequent deliverables and defined frequent deliveries had one choice each (choices from P4, P2, and P3 respectively).

In the **features disclosure category,** the requirements discovery characteristic answered by most of the participants with a total of 5 choices from P1, P3, P4, P5, and P6. Others characteristics as requirements gathering and requirements quality had one choice each (choices from P4, and P6 respectively).

In the **software product category** we found high-level of source code as a characteristic more answered by the participant with five choices (choices from P1, P3, P4, P5, and P6). Following high-level delivered software with three choices from P2, P5, and P6. Awareness of failure had one choice from P1 and efficient coding with no response.

In the **customer relationship category** the majority of participants chose confident customer as a characteristic that represents their customer with five responses (choices from P1, P3, P4, P5, and P6). Customer awareness of team and partner customer characteristics had one choice each (choices from P2, and P6). Team awareness of customer had no answer.

The **organizational support category** had four choices for agile commitment characteristic (choices from P2, P4, P5, and P6) following agile priority characteristic with two answers (choices from P1, and P2 respectively). Agile motion and agile business characteristics had no answers.

6.2.2 Results from Focus Group Session

After the end of the mapping from questionnaire responses to the framework, we started the second phase. In Phase 2, results from the Phase 1 were discussed with the whole team to get a deeper conclusion about the results of the first phase and the reasons why each characteristic chosen belongs to the team. We also extracted challenges faced in all categories. The discussion and collective perception meant that the results were confirmed by all participants about their characteristics in the Progressive Outcomes Framework.

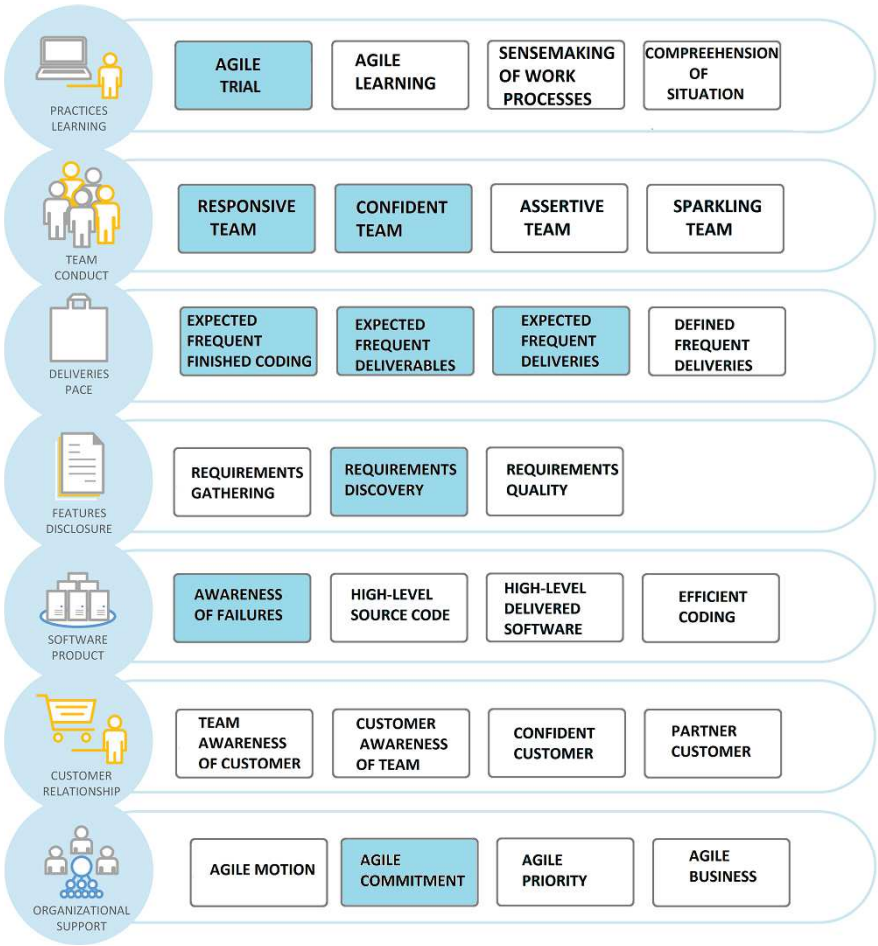


Figure 6.4: Findings from the focus group - Company A

The contributions from each participant are shown in Table 6.3. The statement estimates how many contributions each speaker made and what percentage of all statements

that calculates to [20]. And the label words means the number of words spoken by the participants. We identified that Participant 6 (P6) had zero statement because he left the study before the end to solve a work problem. Participant 5 (P5) had the major statements about the group of 34,37% percent of contributions.

Table 6.3: Contributions from each participant in the focus group - Company A

Participants	Statements	% Statements	Words	% Words
P1	32	12,45	2426	8,24
P2	43	16,73	5175	17,58
P3	39	15,18	3319	11,27
P4	25	9,73	2892	9,82
P5	55	21,40	10119	34,37
P6	0	0,00	0	0,00

When the discussion started, we noted that some characteristics were discarded, other were included, and also other had no changes. The following paragraphs detail the group choices of the participants on the characteristics (outcomes) represented by the team. Codes [O(number)], [E(number)], and [C(number)] represents the outcomes, evidences and challenges, respectively, extracted in each category. This codes were merged from the interviews transcriptions. (See in Appendix E - Network of Results).

In the **practices learning category**, all participants agreed that **agile trial characteristics** [O1] belongs to the team. Participants discussed the adoption of some agile practices tailored to the large-scale development context. Others practices are not implemented nor encouraged. **They are all hands learning about how to follow agility [E1].** As mentioned by [P2]:

> *"(...) We don't have all Scrum Method applied by the book in our development process. What I think is that (...) We only adopted management part of Scrum. We use kanban board, daily meetings, and others. However, we haven't used quality code, refactoring, and pair programming. We are still learning how to implement agile, how to deliver value on time." [P2]*

Participants P4 and P5, also discussed that they had a coach, but he left the project due to the contract agreement finished. Now they do not have a coach to help them to implement agile practices. Thus, the biggest challenge in this category is the **lack of training in agile practices** [C1]. This challenge characterizes a fragile team, which is not confident to do their work in an agile environment:

> *"In the past, we had a coach, we had a mentor of agile methods. But he left the project for some time and, since then, we did not have agile training in our project. (...) We do agile on the fly. However, sometimes nothing works!" [P4]*

Also, the developer P5 added:

> *“So also we, as a team, had this lack of training. The unavailability of a public company that depends of others outsourced companies. This is so difficult. (...) So, we have been adapted the techniques, the practices, but actually, me and my work colleagues do not follow all practices by the book.” [P5]*

Other characteristics were not covered by the participants because they believe they do not use any agile method ’by the book’. They vaguely understand the agility value in each practice, and they do not understand the situation of their work processes with simplicity using metrics.

In the **team conduct category**, the **responsive team** [O2] and **confident team characteristics** [O3] were fully agreed by all participants. They admit being a responsive and confident team because **they need leadership to perform the activities demands** [E2]. They have a little autonomy over their work processes and require approval for changes about the business scenario, even in small changes as reorganize the product backlog. About communication, they are free to communicate with each other, and they use a tool called *Communicator* to talk to other distributed team. **They know each other and are free to communicate** [E3], even in distributed teams.

Thus, Product Owners (POs) are the leadership inside of the team. The POs hold meetings with customers to define the project scope and the product backlog in a high-level understanding to start the sprints. During the sprints, in the planning poker, POs meet the entire team (POs, developers, and testers) to define which activities have to be done. These activities are described as user stories and estimated. In this process, the participants have partial autonomy to make decisions about how activities will be developed, and by whom, also they choose the technologies used for the work development. Only POs define which are the top priorities of work to be done. Thus, no one developers have autonomy to change it or even to change the time of accomplishing a user story, as developer P2 said:

> *”(...) Our Product Owner is our informal client. He evolves workshops in the clients and they define the project scope and they pass for us to develop (...) we do not define what it will be developed. (...) the autonomy we have is to change some sprint backlog, we need not implement exactly what comes, we suggest the best way to accomplish tasks” [P2]*

This hierarchical structure within the entire team bothers the developers [C2]. Because some activities need to be modified and the developers do not have this autonomy. Thus, deliveries not understood by development are exposed to errors not being performed in the estimated time, as developer P2 claims:

"The senior management does not ban our work process (...) we have a management within the team, the POs. (...) They came ready at most ended up scoring the difficulty we would have. Regardless, if the task is easy, we did. If the task is impossible, we had to do the same but, we can deliver late. we cannot complain about this complex tasks, I see this as a challenge." [P2]

Participants did not choose an assertive team, and sparkling team characteristics during the focus group session because they do not feel responsible for the project, and they did not create their work processes focusing on technical excellence and self-autonomy.

In the **deliveries pace category** participants identified three characteristics (same characteristics found in the questionnaire): **expected frequent finished coding** [O4], **expected frequent deliverable** [O5], and **expected frequent deliveries** [O6]. In their work process **the deliveries are partials** [E4] (finished code and integrated in into the repository). Also, their **technological environment support automated build and continuous integration** [E5] using Jenkins tool (an open source automation serve). As aforementioned, the deliveries are planned and done, but usually late. This occurs due to **the extra demands appear** [E6] and the team has to address them. Thus, the **sprint size sometimes varies** [E7] and change of scope happens, as mentioned by P3, and P5:

"It is quite rare to deliver on time" "(...) Very often we deliver on the last minute that we can. We always anticipate that some tasks will be delivered after the production or "golive". We have bugs in our development process. These bugs can be corrected in the next Sprint, or during the release."(...) "Here, we call that as a partial agile delivery." [P3]

"Yes, In several situations we delivered late. (...) In 45 minutes from time to deliver – they (PO and customer) – diced to change the project scope, and – we (developers) – need to fix it and deliver on time. I only remember one or two tasks that have been delivered on time. (...) For example: ten items are priority to be delivered. But suddenly we have more five to do, and after more two or three. Then we have seventeen priority tasks. Its a really and unfinished mess." [P5]

Testing is usually manual [E8]. They have five testers who have performed manual and automated tests. The testers also assist developers on delivering quality. They do automated builds which help regression testing at the end of the sprint. However, a big challenge to the participants is **to have automated test in their complexity legacy systems** [C3]. As said by P2, and P4:

"We have testers in our team, they perform a manual testing. Sometimes, we have automation testing. This automation do not cover the entire systems be-

cause we have or "legacy systems" that I think is a big challenge to start an agile testing here." [P2]

"In our team we have testers, they do acceptance, and white box test manual. We do not have automated testing in our complex system. This is another trouble." [P4]

Also, another challenge is the **conflicts within the team** [C4]. The development team believes the testing team affects the code quality and the refactoring of activities performed by the developers, as P2 and P5 said:

"We defined an agile process, but we do not have refactoring and pair programming in our code process. (...) Our testers dedicate a part of their time identifying code errors. Then a test team began to think: "We will not take the blame for having half that left so many mistakes to go." Then we'll have a process where the developer creates a version. We test and integrate the code after being released and tested, so we can not do refactoring of little things. We commit without refactoring part." [P2]

"(...) Our developer point of view, against the tester's point of view. " [P5]

The defined frequent deliveries characteristic identified in the questionnaire was not included by the participants in this second phase, because they believed that **deliveries are not performed as planned 'on time'** [C5], although they have an environment to support automated build and integration.

In the **feature disclosure category**, the **requirements discovery** [O7] characteristic represents the LSAD team (described by all participants). Non-functional requirements are defined at the beginning of the project by POs, customers, and stakeholders. Functional **requirements are specified at the start of each sprint during the scrum-of-scrums meetings and finalized in the planning poker** [E9]. However, there are ongoing initiatives to refine requirements, which means that the team is not in the requirements quality characteristic. All requirements dependencies are discovering in this phase. Developer P2 described requirements definition:

"(...) PO conducts an overview only spoken and some items without the prototype there. So we'll question for example, how often you have the vision of what the system has to return, that you have to do that part. (...) POs created the user stories during the scrum-of-scrums meeting and also during the planning." [P2]

There is an internal practice used during the planning poker called "10 minutes rule". If PO does not explain the user story in 10 minutes, the user account is removed in the sprint. The developer P3 described:

> *"Also, we (developers and testers), put a rule for planning that the PO can not explain to us. (...) It is 10 minutes, then we removed it (requirement) from the sprint. But, some can not be removed." [P3]*

Also, there is scrum-of-scrums meeting with all POs of each project before the planning poker:

> *"(...) Scrum masters, POs... They use Scrum-of-Scrums that works very well. (...) They gathered the team, including developers, and our team improved quite well. We have contact with other teams. " [P5]*

If there are extra demands, the requirements are already ready to be implemented by the developers. This causes mistrust in the team because many requirements are not well specified by the customers and POs. Thus, participants said that **they do not know if these implemented requirements bring value to the customer** [C6], as said by P5:

"This difficulty we are going through and that in my view we are improving enough but when I got right there, and this is where the red snapper commented here that you received a task and an incredibly simple description to do and she became what exactly do you want it done right? And then you came at the time of planning was told, B, C, D (examples). Then when you were going to do there with the missing one thing that influenced the B, C, and D, and I already was there in D. There for you to sit back, review the requirements, exchange idea." [P5]

In **software product category**, the **awareness of failures** [O8] characteristic was chosen to represent the entire team. There are several flaws and bugs in the code system, thus **they spend time in the sprint correcting bugs, and issues in code building and integration** [E10]. Even existing a team of testers within the LSAD team, testers can not prevent deliveries, sometimes have problems, mainly when a new service is created in a legacy system. **The development process is similar to the waterfall** [E11], the testers only tests the code before being delivered to the customer. So, the team does not have a source of high-level that ensures robustness, efficiency, and quality code. As developer P1 mentioned:

> *"I agree that we are in the category of awareness of failures. We do not reached efficient coding yet. The code is shy." [P1]*

Thus, all participants concluded that the characteristics high-level source code, high-level delivered software, and efficient coding does not correspond to the team, contradicting what was chosen in the questionnaire by themselves. There are no initiatives to guarantee the quality code, as refactoring, pair programming. Although they have an infrastructure to support agility, they do not have unit tests and a few initiatives to Developer Operations (DevOps).

Only a manual testing is realized in each code developed. After, the code goes to continuous integration. They performed three tests: the local test is the first one, in the development environment is the second one and finally in the homologation environment.

Regarding technological environment, the team has an environment to perform continuous integration and automated build with Jenkins. Also, they use Redmine tool to manage the tasks to be performed. And code freeze is a practice for integration of codes a week before finishing the sprint.

Bugs mostly are reported by customers, [C7] and developers also highlight this as a challenge. The participant P5 said that P6 cannot participate in the session because he needs to solve a problem that occurred in the last sprint:

> *"Most failures were proper by customers; they are testers. “For example, P5 are solving now (...) a bug of our last sprint" [P5]*

In the **customer relationship category**, participants agreed that the development team is not represented in no one characteristic. Only PO collaborates with the customer to identify the main requirements of the project. POs do interviews with a customer and receive the demands. Thus, the development team thinks that **PO role is a bridge between the team and the customer** [E12]. As mentioned by P3:

> *”(...) The scrum master, POs, and customer prefer to centralize their customer’s relationship. Thus the PO is responsible for contacting the customer if necessary. The demands and the scope are agreed only between these two parts. The development team learns of the conversation between them.” [P3]*

Also, P5 adds that the main challenge is **to have an intermediary (POs) to understand what the customer needs** [C8]: *”I think that sometimes is (...) we are like in the ping pong. You take here. Then you play there, then we play the ball back, then back again. (...) Now the question we had was not well this then is a confusion where there is an intermediary.”* [P5]. Thus, the communication of developers and customers/stakeholders was weakened after the formality of communication only with the POs: *"(...)I did not choose the "customer awareness the team" because I think about the customer and internal affairs. I think they were very close, but nowadays we do not have this approach anymore."* [P2]

Participants, also a highlight, the **lack of communication with the customer** [C9] as a challenge. Sometimes there is no common understanding between PO and developers on some requirement. Thus, developers waste hours for a customer’s positions.

The last category, **organization support**, participants chose **agile commitment** [O9] as a characteristic related to our organization. The participants follow the organization’s commitment existence in empowering the team and provide an infrastructure for the adoption of agile methods. All participants agreed that "agility" is not a priority neither the business

of such company. This happens because there is a few support to agile transformation, and the company is not recognized for being agile. Thus they understand to change the findings from the questionnaire.

Top management decided to adopt agile in the company, and initiates are official in all IT departments [E13]. There is an infrastructure that supports agility. However, **no training or qualification to the developers is encouraged now to the agile transformation** [C10], as a challenge by P3:

> *"It's just said that for me this is partial, there is no investment in training, coaching, and communication. In the past, they had an investment to adopt the agile method, but not now. (...) For example: it's good to come here, but now you guys already know agile and you will stand on your legs". [P3]*

Also, participants felt that some teams have resistance to change, as infrastructure teams. Company structure (physical and departmental) has changed to support agility, said by P2:

> *"(..) About our resistance, some people had resistance, especially in infrastructure team. Older people, they had a little resistance, they did not change their work process to adopt agile." [P2]*

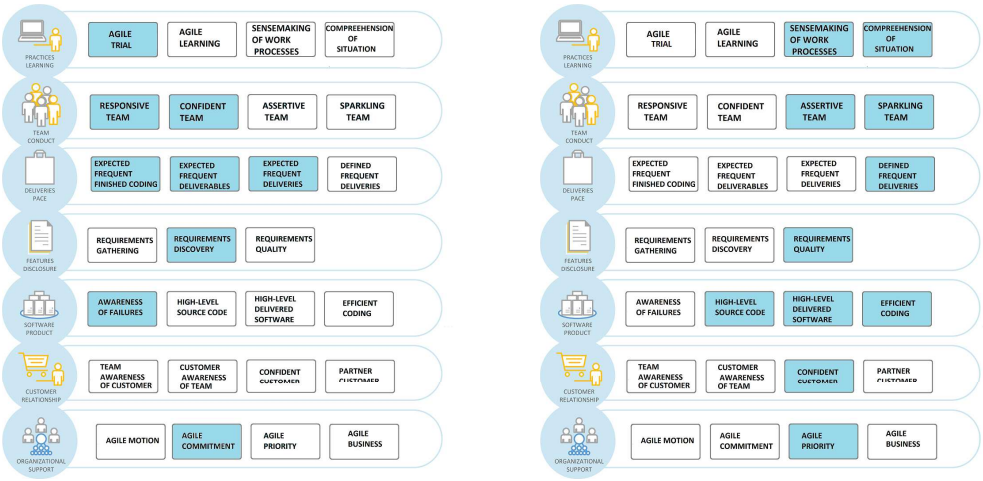
At the end of the session, the moderator asked the participants if they felt as an agile team. According to them (participants) they can not be defined as an agile team, but as a team with agile initiatives. Some participants agreed that the company enjoys the benefits of agile. For P1, P2, P3, and P6 agile is 'to run fast and fail fast.' Also, communication has improvements, as mentioned by P2: *"I think it has benefits because I have worked on projects to stay six months working and when finished taking so put in the corner and do not use. Why it is not what the customer wanted."* [P2].

However P5 does not agree with benefits in adopted agile, he prefers to use traditional methods, to not change so fast the planning, as P5 mentioned:

> *"I think, I came from a company working on a waterfall, and the only thing I saw difference is about the experience I'm having here with an agile method. The issue of code quality and to delivered product late. In waterfall, we delivery with quality, using RUP, much better, the way of how the implementation of the software was made process. But in agile, I saw in the last month's, nothing works, nothing is ok to deliver." [P5]*

6.2.3 Discussion of Results from Company A

The results of Company A shows that the team is in the first characteristics from the Progressive Outcomes Framework. To answer our research question, the results shown in Figure 6.5 indicate that agility is present but still in the 'early stages.' The team is toward to agility, i.e., agility is not entirely followed. To a team be 'agility', mature in agile software development to the Fontana's study (2015) [43] the team must obey the final characteristics of the Framework (shows in Figure 6.5. The blue boxes represent a mature team following agility) that related to the results from the focus group, these characteristics have not been achieved.



(a) Focus Group Results

(b) Agile Compass Results. Source: Fontana et al. (2015) [42]

Figure 6.5: Comparison between the focus group and the progressive outcome framework Company A

Compared to the responses of the two phases of the study, we identified the difference among the results. For instance, all the characteristics of practices learning category have been designated as belonging to the team. After the discussion of the results, the group change their opinion found only agile trial characteristic.

P2 and P5 participants who spoke and contributed mostly in the study. Thus the results may have been influenced by these two developers who contributed mostly to the study. The responses differences were only identified by the moderator at the end of the study, about the agile benefits and if the team feels an agile team. In no time during the team identification in the Framework the participants discussed their views. This indicates that the results were unanimous.

In the agile learning category, P2 does not believe that the team adopts Scrum by the book. However, is a contradiction because they said to implement all ceremonies of Scrum indicating the team follows an agile method. The team may also be in agile learning

level. Thus, as a result of lack of training leave the team without knowing if they implement some agile method. Moreover, they have a lack of confidence in adopting practices related to the code, such as refactoring and code clean. These practices are encouraged in the XP method, but not in Scrum, as the participants said erroneously.

Table 6.4: Results from the presence of agility - Company A

Agile Learning Category	**Evidence**	**Challenge**
[O1] Agile trial	[E1] The team tries adopting Scrum practices by the book as Hands-on learning.	[C1] Lack of training/mentoring in agile practices
Team Conduct Category	**Evidence**	**Challenge**
[O2] Resposive team [O3] Confident team	[E2] The team needs a leadership to perform the activities demands [E3] The team knows each other and are free to communicate	[C2] The development team feels weakened by the autonomy of POs
Deliveries Pace Category	**Evidence**	**Challenge**
[O4] Expected frequent finished coding [O5] Expected frequent deliverables [O6] Expected frequent deliveries	[E4] The deliveries are partials [E5] Technological environment support automated build and continuous integration [E6] Extra demands in the sprint [E7] Sprint size sometimes varies [E8] Testing is usually manual	[C3] The team do not have automated test in their complexity legacy systems [C4] There are conflicts within the team [C5] Deliveries are not performed as planned 'on time'
Feature Disclosure Category	**Evidence**	**Challenge**
[O7] Requirements discovery	[E9] Requirements are specified at the start of each sprint during the scrum-of-scrums meetings and finalized in the planning poker	[C6] Developers do not know if the requirements have value to the customer.
Software Product Category	**Evidence**	**Challenge**
[O8] Awareness of failures	[E10] The team spend time in the sprint correcting bugs, and issues in code building and integration. [E11] The development process is similar to the waterfall.	[C7] Customers report the bugs in the system.
Customer Relationship Category	**Evidence**	**Challenge**
No choice	[E12] The customer does not talk to the whole team, just with POs	[C8] Difficulties in have an intermediary (POs) to understand what the customer needs. [C9] Lack of communication with the customer
Organizational Support Category	**Evidence**	**Challenge**
[O9] Agile Commitment	[E13] Top management decided to adopt agile in the company and initiates are official in all IT departments.	[C10] Lack of training or qualification to the developers in the agile transformation

In the team conduct category, there is an individual autonomy in the team. Only POs and Scrum Master have the responsibility to attend the customer's business. Develop-

ers and testers, in turn, are responsible for ensuring the deliveries. This hierarchy breaks the confidence of the team and is appointed as a challenge. Because if there is one task misinterpretation developers hope the POs talk to the customers to continue their work.

In the deliveries pace category, in each sprint other demands of the customer are welcome. By the way, during the discussion of this study, the moderator perceived that no developer is comfortable in changing or add demands during a sprint. Thus, it is seen that one of the agile principles (responding to change over following a plan) is not followed. Also, developers talked about the testers. The moderator noticed a rivalry between the two parts of the team. Developers blame testers for failed deliveries, and believe that the lack of automated testing makes it difficult to deliver on time.

In the features disclosure category, requirements discovery characteristic was agreed by all participants. The characteristic indicated by the team has raised questions once some activities are not understood by the team during a sprint. This means that the team may do not know what is needed to deliver value to their customer. Also, participants had difficulty to discuss because the requirements are PO responsibility.

In the software product category, participants blame to be in the awareness of failure characteristic due to the lack of a testing process. The testers only test at the end of the deliveries, which is not generated unit tests. Also, there is no automating test. The moderator identified a certain degree of implemented technologies (code integration and management activities) but the lack of training prevent the team from being more efficient.

In the customer relationship category, the customer does not talk to the whole team. It is an internal policy. Just POs and Scrum Master can speak to the customer and to identify the customer needs. This lack of communication with all team hinders security in work being done.

In the organizational support category, the initiatives of agile adoption are top down. But the lack of training is an issue to the agile transformation. Also, the customer seems to prefer the formality more than agility (communication). Furthermore, the team has resistance to using agile methods. When asked about the benefits of agile methods, and they as an agile team, P5 believes waterfall is more structured and fits better with the team profile, by the way, other participants think that agile errs easier and is simpler to use.

In general, a team of company A has few agile adoption initiatives and the presence of agility has many issues to repair. For instance, the lack of informal communication and the customer relationship restricted. Answering the research questions (Q1 and Q2), we summarize the main results of each category in Table 6.4. The evidence and challenges were drawn from the codes in each category answered.

6.3 Focus Group at Company B

We conducted our second focus group study in September 2016. This study was developed at a large telecom company located in the Rio Grande do Sul, Brazil. This company has one of the largest development team in Latin America working on distributed software projects for telecommunications area. Nowadays, the company comprises over 350 developers working directly on the creation of new products. The current project aims to support the network management of telecom. The sprint information and scope of the current project are confidential.

Data were collected from an LSAD team of this company. The LSAD team consists of 08 people, including one product owner, one tester, five Developers, and one scrum master. Only the tester could not participate in the study. Thus, we conducted the focus group study with seven participants. The developers, the product owner, and the scrum master had different levels of Agile Experience. Two developers had less than three years working experience, one had three years' work experience, and four others participants had experienced between five and six years. Table 6.5 presents the participants' details.

Table 6.5: Participants' profile - Company B

Participants	**Job Title**	**Agile Experience**	**Working experience with the team**
P1	Developer	5 years and 1 months	6 months
P2	Developer	6 years	6 months
P3	Developer	2 years	6 months
P4	Developer	1 year and 4 months	6 months
P5	Scrum Master	6 years	6 months
P6	Developer	5 years	6 months
P7	Product Owner	3 years and 8 months	6 months

It is an LSAD team because they work with two other software teams distributed in two sites containing forty people in the project. Some characteristics about the team:

- The project scope is defined by product and project managers following the customer.
- In this company, scrum-of-scrums is a practice uses by POs, scrum masters and technical leaders to communicate. The main contribution of this practice was to align the current demands, to improve communication and integration between teams, to facilitate proofs of concept (POC) and to produce coding standards according to the level of dependence of systems in each team.

6.3.1 Results from the Questionnaire

In Phase 1, first author of this study presented the study goal and distributed a questionnaire to the participants. The results from Phase 1 are shown in Figure 6.6. Each question in the questionnaire represents the seven categories mapped in the framework. In each category, the characteristics were described so the participant could choose which characteristics belong to their large-scale agile team according to each category. To identify the responses of each participant, Figure 6.6 also describes the participant ID: P1, P2, and so on.

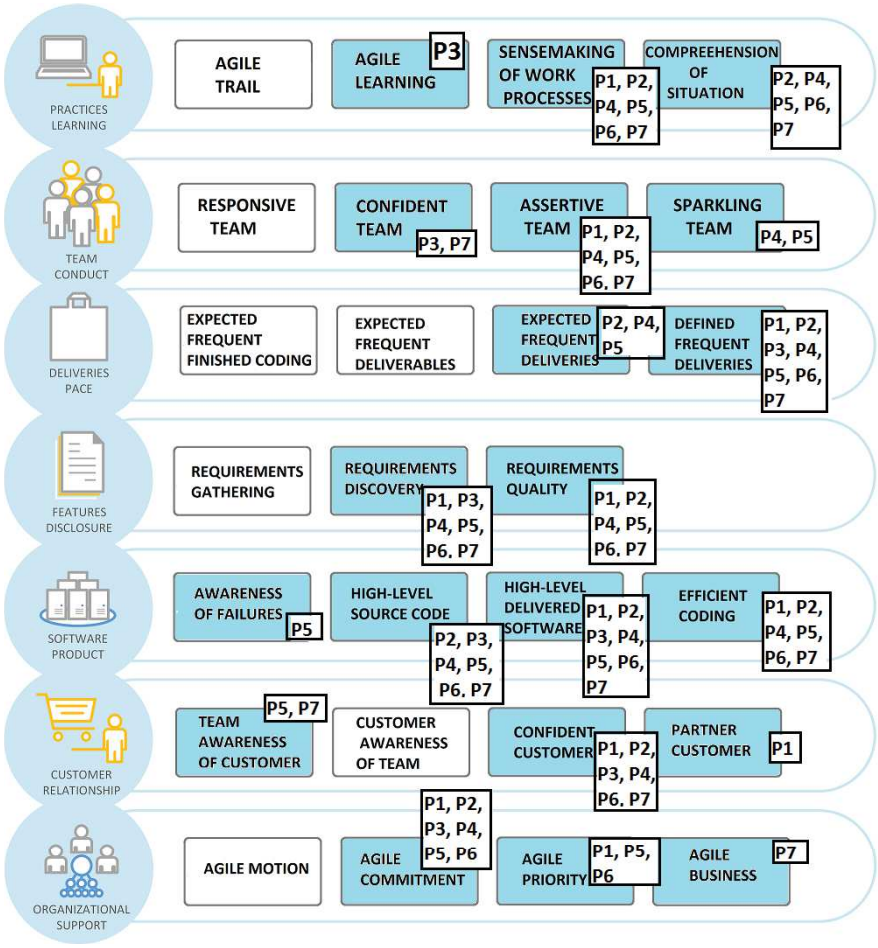


Figure 6.6: Findings from the questionnaire - Company B

In the practices learning category participants believed to be part of three characteristics. Agile learning category appears with one vote (choice of P3) following for comprehension of a situation with four options (choices of P2, P4, P5, P6, P7). The sensemaking of work process characteristic had the majority of options, with five choices (choices of P1, P2, P4, P5, P6, P7). The agile trial had no responses.

In the team conduct category, we found three characteristics that participants believed to belong to the team. The confident team had two choices (from P3, P7), and also

two options from P4, P5 for sparkling team characteristic. Assertive team characteristic had six responses (choices from P1, P2, P4, P5, P6, P7). The responsive team was not indicated by participants.

In the deliveries pace category, all participants selected defined frequent deliveries as a characteristic that represents their deliveries to the customer. Following to expected frequent deliveries with three choices from P2, P4, P5. Other two characteristics: expected frequent finished coding, expected frequent deliverables were not selected.

In the features disclosure category requirements, discovery and requirements quality characteristic had the same number of answers with a total of six choices of each. Requirements gathering were not selected.

In the software product category, all participants chose high-level delivered characteristic. Following to the efficient coding (choices from P1, P2, P4, P5, P6, P7) and high-level source code with six responses (from P2, P3, P4, P5, P6, and P7). Awareness of failure had one choice from P5.

In the customer relationship category, the majority of participants chose confident customer as a characteristic that represents their customer with six responses (choices from P1, P2, P3, P4, P5, and P6). Team awareness of customer characteristic had two choices from P5, P7. Partner customer characteristic had one choice from P1. Customer awareness of team had no answer.

The organizational support category had six choices for agile commitment characteristic (options from P1, P2, P3, P4, P5, and P6) following agile priority characteristic with three responses (choices from P1, P5, and P6 respectively), and agile business with one vote from P7. Agile motion and agile business were not selected by participants.

6.3.2 Results from the Focus Group Session

After the end of the mapping from questionnaire responses to the framework, we started the second phase. In Phase 2, results from Phase 1 were discussed with the entire team to get a deeper conclusion about the results of the first phase and the reasons why each characteristic chosen belongs to the team. The discussion and collective perception means that the results were confirmed by all participants about their characteristics on Progressive Outcomes Framework. The contributions from each participant are shown in Table 6.6. We identified the Participant 6 (P6) had the lowest statement (8,5%), and Participant 7 (P7) had the major statements about the group (20,41%).

At the beginning of the discussion some characteristics were discarded, others were included, and also others had no changes. The following paragraphs detail the group choices of the participants on the characteristics represented by the team. Codes [O], [E],

and [C] presents the outcomes, evidences, and challenges extracted in each category. This codes were merged from the interviews transcriptions. (See in Appendix F - Network of Results).

Table 6.6: Contributions from each participant spoken in the focus group - Company B

Participants	Statements	% Statements	Characters	% Characters
P1	44	11,76	5013	13,28
P2	27	7,22	2262	5,99
P3	38	10,16	2689	7,12
P4	50	13,37	2963	7,85
P5	75	20,05	7228	19,15
P6	21	5,61	3210	8,5
P7	48	12,83	7707	20,41

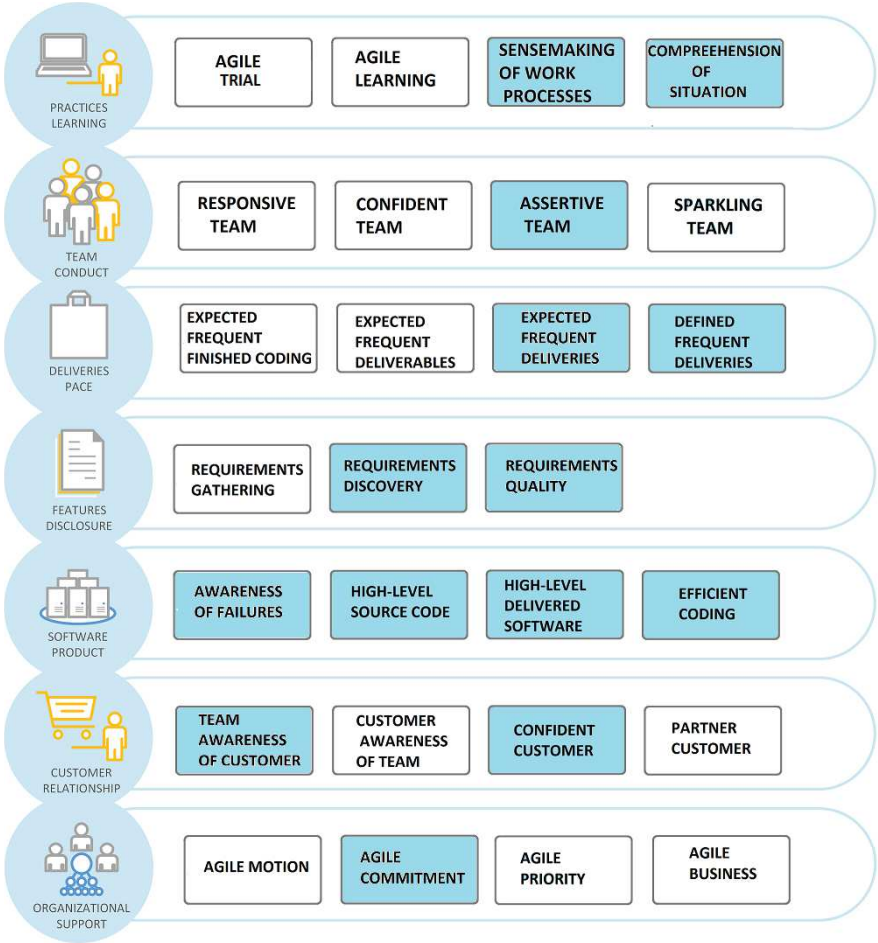


Figure 6.7: Findings from the focus group - Company B

In the **practices learning category** the agile trial and agile learning were characteristics already experienced by a team. In the past, the team hands-on learning tried adopting agile practices without mentoring. Afterward, they received training in for all software development teams. Furthermore, infrastructure was improved to support agility.

Thus, all participants agreed that they were in **sensemaking of work processes** [O1] characteristic. In which, includes **taking the method learned and tailored it to particular needs** [E1]. Currently, the team comprises their "way of agile to be" their values, and they adapt them in the work processes. As Scrum Master (P5) said:

> *"(...)In my view what take after our environment is the C choice. For each team despite having a standard process. Sometimes we have an adaptation. Always the responsive to our environment. Then all the teams do not perform the same processes. This depends on of each scenario, environment. I think we're a little more mature in items A and B. (...) I remember the part of following the tutorial, step by step. I think today. We're already on another level, then yes my opinion is C (...)" [P5]*

Four of them also said that **the team used metrics and many tools to track the process** [E2] as Rally and Jenkins tool to report working status and letting a clean and comprehensive way to work. Thus, the **comprehension of situation** [O2] also is a characteristic which belongs to the team. As mentioned by P4:

> *"We have many indicators of the work done. We always tried to make the information open to everyone." [P4]*

All teams performed Scrum and Kanban methods. Adaptations happen according to the context of each one. For example, planning poker is not a practice adopted by all teams. Other teams prefer to carry out the activities by cadences using a Kanban board. Another example is the sprint without a target and time defined. Some teams vary in size until they find time (effort and size) according to their activities, as mentioned by P5, and P7:

> *"...We do all Scrum ceremonies as a ritual. Some teams put their point of view in agile development processes. Planning poker is an example. Some teams believe is useful, but others do not understand the practice, they don't see a practice with so much benefit." [P5]*

> *"(...) Another example is the sprint period we usually do in three weeks. Since it was started, we tested several time-boxes. I think that is not required to do the same time, always (...) We are flexible. I've worked in teams planning sprints in one week, for example." [P7]*

The agile learning characteristic answered by a single participant (P3) aroused the P5 curiosity:

> *"I have a curiosity, who choice the letter B? I want to know the motivation." [P5]*

Participant 3 who answered agile learning characteristic, believed that there was **some resistance of agile adoption in the past** [C1], but nowadays these symptoms have been dismissed: *We have some resistance before! In the past, this was a challenge! [P3]*

In the **team conduct category**, **assertive team** [O3] characteristic was chosen by the entire team in this second phase. The LSAD team feels **responsible for the project** [E3]. **All team has autonomy to improve their work process** [E4], also the team has autonomy to carry out the activities related to their job and tasks/user stories are estimated and described by the most experienced developers and, POs. But all team members are active voices in the project and the process improvement initiatives. Also, the team does not need to inform senior management about the changes they made.

Participants P1, P2, P4, and P5, believe that the team maturity influenced to be assertive and there is a certain willingness of the team to be a sparkling team. **Team is mature enough to focus on just technical excellence** [C2], as said by P1: *"I think the question of the activities has a lot to do with the team maturity and project experience. It takes the team to be better assertiveness."* [P1]

Also, participants emphasized the presence of the scrum master role as part of the team. This role does not discard the team leadership and autonomy to express their opinions about the decisions and to discuss the demands to be worked. The responsibility of the scrum master is to ensure the ceremonies of the Scrum, unlock impediments and to facilitate good communication within the team. This role is the former responsibility of the project manager, but their activities are related to the agile method. As mentioned by P7:

> *"(...) The scrum master ensures all Scrum ceremonies, we unlocks any impediment related to tasks or things that depend on external teams (...) He also ensures good communication within the team. The scrum master can be a project manager too!" [P7]*

Participants P2, P4 and, P7, consider planning poker an effective practice to improve the team communication. The planning of each sprint is an opportunity for the developer with the lowest experience of work to learn and to discuss ideas with seniors developers. Such practice creates **autonomy and empowerment to the team about their work** [E5]. P4 also claims that once, only senior persons could work in planning activities and construction of user stories, a challenge for younger people on the team learn to create autonomy. Nowadays using planning poker, all members are involved.

Despite the planning poker is considered a good practice by participants P3, P4 and, P7, other participants (P1, P2, and P5) believe that instead of planning poker, the Jenkins tool to cadence activities is a faster and more productive method. And the questions that may arise with the team are in charge to be answered by scrum masters and the POs, as said by P5:

"I think it also depends on the time you will make planning poker at a meeting that you see a lot of very long things, too subjective. I prefer Jenkins instead. Really! Planning Poker is not agile, is a practice to achieve or not agility!" [P5]

In the **deliveries pace category** the team choices were: **expected frequent deliveries** [O4] and **defined frequent deliveries** [O5]. They partially expected frequent deliveries because, sometimes, **the team needs to address extra demands in the sprint** [E6]. **To address extra demands** [C3] is a challenge reported by four participants, P2 and P3 claimed:

"(...)Before, we have a lot of extra demands, but now thank God we don't do anymore. It's a quite rare. Demands can "arise" in the middle of sprint, but we do not stop the sprint or to add things in the middle of the Sprint!" [P2]

"We do not talk about extra demands, anymore. It was very common for about four years ago to deliver late. Annoying thing laughs" [P3]

The PO (P7), reinforces that it is not common this kind of situation happen but, in case it happens, there are work shifts to keep the same effort of activities in sprint: *"Its quite rare (to have extra demands). But sometimes exceptions happen. For example a higher priority item or are some of them that we thought that he was ready to start but was not. Then they usually do not take; we negotiate to change and to keep the same work effort."* [P7]

Another challenge that can hinder the deliveries are **layoffs of support people and the decrease of the teams** [C4]. Thus, extra demands and activities can be included. This challenge can cause high damage in a matter of agility.

About defined frequent deliveries, **the team planned their deliveries and performed them on time** [E7]. Each team has the commitment to deliver on time. In each sprint they defined the deliveries and metrics to make visible the status of work about the whole team.

The sprint size does not change [E8]. Sprints with pending deliveries automatically enter in the release period. One task completed need to be developed, tested, accepted by the PO and committed in the repository, as mentioned by P7:

"yes, We define the sprint period during the planning, and normally we don't change this period. We see the success of the sprint with the amount of deliveries we did. Some sprints have items that are outstanding, (...) we only consider an item delivered even when it is developed, integrated into the repository. This code also is tested and accepted by the PO." [P7]

The team has an excellent technological environment and an infrastructure to support agility. This **environment supports automated build and continuos integration** [E9]. They use Git tool to source versioning, Jenkins tool to run continuous integration and Rally tool to improve agile cycle time — at scale and to align software deliveries to portfolio, and business. However, the **infrastructure dependencies** [C5], is seen as challenge. When an infrastructure problem occurs, teams need to wait for recovery, as mentioned by P2:

> *"Some things need improvements. But we have a lot of things. I think we have a perfect infrastructure. But sometimes, our infrastructure dependency creates some problems. When infrastructure not works, we get limited (...)" [P2]*

Tests are usually manual [E10] and performed by one member of the team. The tester has the responsibility to execute acceptance tests in each user story developed by programmers and shows to PO accepts. Thus, the “user story” is not finished until the flow of acceptance tests is done.

Demands that are not completed during of three weeks of sprint are automatically sent to release, as mentioned by P7: *"(...) We accepted all activities and hit all deliveries in the sprint, but in the release period, we always adjust. If we delayed in the sprint, we got to the release planning. We have this flexibility throughout the project."* [P7]. P5 added about demands priority: *"(...) or just taking the least priority score at the end to finished on time. There is a project negotiation"* [P5]. Also, P6 mentioned about problems during the deliveries not done on time: *"(...) depends primarily on why the delivery was not done, sometimes, very often, it appeared some external problem."* [P6]

In the **features disclosure category**, the team defines broadly requirements during the definition of scope project. Internal customers, stakeholders, and POs participate in this phase. **In the sprints, requirements become user stories** [E11]. This user stories are defined and estimated by the entire team altogether. When deliveries start, PO has the responsibility to ensure that user stories be ready for the developers starts to implement. This process PO (P7) characterizes as **requirements discovery** [O6]:

> *"The project scope it based on internal customer. We do not have an adamant contact with the external customer right; we’re going more for stakeholders who work in other sectors, other areas of the company. Thus we discussed the project scope, and we defined in a macro way the requirements. Thus, when the sprint starts, the PO let all the user stories ready. Then during the development of sprint PO prepares the development of the next stories. That’s how we work." [P7]*

Three of the participants also said that they have flexibility to planning, re-programing or estimating again if a user story is not according to the team expectations.

Another characteristic they pursue is to improve the **quality of the requirements** [O7] to make sure they meet customer expectations. Totally different practices appeared to

accomplish this result, such as using videos to record customer requirements, using systems analysis diagrams, or involving developers in requirements definition. Also to improve the requirements quality, **PO, and intern customer align requirements definition to make sure the both expectations** [E12]. The intern customer has a responsibility to ensure that the objectives of the project are aligned with the final product that will be delivered to the end customer (external customer).

In this category, the participants believe that no specific challenge avoid the process of requirements. The team is satisfied with the requirements elicitation process. Thus, no challenge was coded in this category.

In the **software product category**, all participants agreed in be in all characteristics, **awareness of failures** [O8], **high level source code** [O9], **high-level delivered software** [O10], and **efficient coding** [O11].

When the discussion starts, six of the participants did not agree that bugs happen due to their development process. But the scrum master (P5) said the team also failed deliveries due to the non-existence of an efficient process of automated testing:

> *"I choose awareness of failure because in the process we do not have automated tests coverage all codes or systems (...) what happened is similar to our HS800 (Sprint Codename) that we did not have a certain regression on the codes. (...) Our team doesn't have a good test process, we fail in this, in our development." [P5]*

Thus, all participants were convinced in to have **awareness of failures** [O6] characteristic. The **lack of automated tests in legacy systems** [C6] increases the chances of the team having problems in the development of deliveries, as P3 said:

> *"Our test is basically manual, we do not guarantee that all deliveries are "done", without errors. There is a lack of automated testing in old and legacy systems. It's a real problem, I think, the biggest problem of our team is the automated testing." [P3]*

About the other characteristics, three participants mentioned about their **initiatives and policies to guarantee the quality code.** [E13] They have, pair programming for new members of the team. They also have reviews, refactoring, manual and automated testing (when is possible) as a priority. Also, they concern about the integrity of the code in the repository, continuous integration, and deploy automation. The team has DevOps, who is responsible for ensuring an available and flawless infrastructure to enable efficient coding, as mentioned by P5:

> *"(DevOps) is a person responsible for all of the infrastructure and tools purchases continue working for continuous integration. He is going to set up jobs in Jenkins (...)" [P5]*

In the **customer relationship category** they choose characteristics about the two customers they had. The external and the internal customer. **The external customers is the end users who do not know much about the project** [E14]. This customer is still getting to know the team, but nothing formalized. There are no policies that encourage the external customer to understand how the project happens, due to be an innovation project. This customer only receives the final product for their consumption, without understanding the entire development process. Thus the **team still awareness of customer** [O12], and as a challenge in this characteristic, **the team does not know if the expectations of the external customers are reached** [C7], P3 mentioned:

> *"We have some projects coming right to the external customer. But, I'm suspicious. I don't know if we reached the customers needs. We don't gain feedback from them. We expected to be in a right way (laughs). But I saw this as a challenge. I don't understand that. Even though in a particular and innovative project." [P7]*

The internal customer, on the other hand, is **confident** [O13]. He relies on the work of the team because the **team can meet the customer's expectations and needs**. The internal customer also participates in meetings and retrospectives. Supports and encourages the team, the deliveries occur as expected, without delay. Also, he knows when the deliveries are going to happen and what is going to be delivered, as mentioned by P1, and P6:

> *"(...) Now we improvement the development process to agile and it is clearly now that we gained a moral better. (...) Internal (Customer), so then they have much more confidence now, especially with the dates and sprints values." [P1]*

> *"They (internal customers) participate in the reviews, retrospectives, and feedback, so they know about the timing of things to be done when it will be delivered. Why that's being delivered and why not, all works well." [P6]*

The term 'confidence' fumbled the participant's choice (P1). He believes that the team helps define solutions for the customer's business problems, but the internal customer does not feel like part of the team, as P4 said: *"I put that, but I do not feel that customer is indeed part of the team. This choice is too strong for me, so I cancel my choice."* [P4]

In the **organizational support**, the characteristic chosen was **agile commitment**[O14]. **The company supports agile transformation in all teams** [E16], even in non-software development teams. They also have a technological environment to support agility. Also, they have investments in training, and coaching, as P1 said:

> *"The company paid training to agile methods for five years. Maybe we don't have agile training with often frequency. Also, we have all infrastructure to agility. And Rally too, that is not cheap." [P1]*

However, they do not believe that the company has agile priority characteristic since the initiative was not top-down but **bottom-up** [E17]. The software development teams started the agile transformation, and due to the positive results in the usage of agile, the top management believed the paradigm shift. Participants feel the company is not an agile business because not all teams in all projects use agile methods. As mentioned by Developer P3:

> *"I get commitment, why change roles teams to give support to agility, so I do not know if it is (...)This wasn't top down, or it was a little top down. More bottom-up transformation" [P3]*

In this category, no challenges were extracted since there is no resistance in the software development teams to adopt agile methods and there are also no isolated agile initiatives in their company.

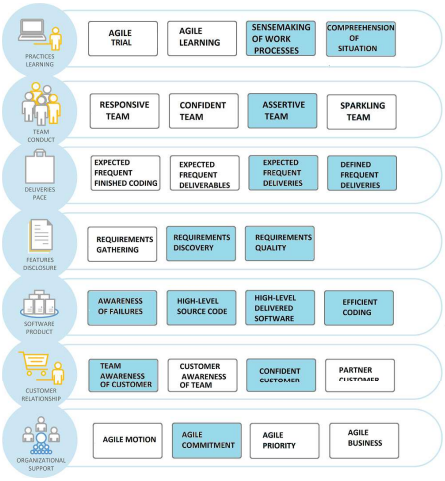
6.3.3 Discussion of Results from Company B

The results presented in the company B show that agility is present in categories: practices learning, software product and team conduct (see in the Figure 6.8). In the disclosure features, the team achieves requirements quality. In the category Software Product, all characteristics were reported by a team. Awareness of failures characteristic present in software development situations which means that agility is not followed by a need to be, thus they partially follow agility in this category. In Customer Relationship category the team also somewhat support agility. The team has an internal customer who belongs on the confident team characteristic, following agility. But they also have an external customer who does not know the team and does not know how the team develops their work, not reaching the presence of agility. The last category of organizational support also does not include agility, and the organization does not prioritize the agile development.

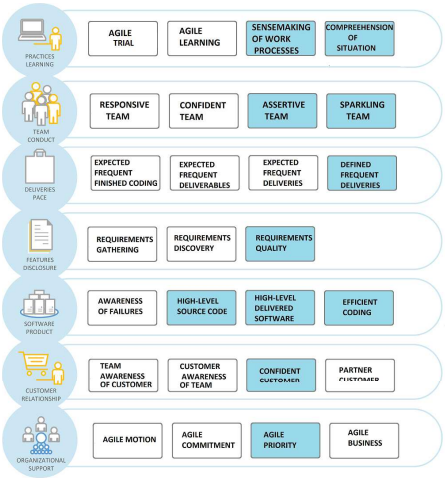
All categories results changed excepted software product. The discussion started all results were aligned with the team. Some characteristics marked in the questionnaire, but not agreed to one participant were discussed to reach a common sense of all. This indicates an unanimous result.

About the focus group session, the main contributions and opinions come from the participants P5 (Scrum Master), P7 (PO). Thus, the results may have been influenced by them. At least one time during the study one of the participants contributed talking about the subject, as seen in Table 6.6.

In the first category, agile learning the team tailor practices to their context. All Scrum ceremonies are adopted. They have simple metrics and indicators that help junior



(a) Focus Group Results.



(b) Agile Compass Results. Agile Compass Results. Source: Fontana et al. (2015) [42]

Figure 6.8: Comparison between the focus group and the progressive outcome framework - Company B

developers to adjust to the environment and understand the demands to be developed during the sprint. This indicates that the team knows, understands and witnesses agility.

In the second category, participants reported the importance of scrum master as part of the team, representing a curator. It helps the team follow agile methods, and he contributes to preventing impediments. The scrum master is characterized as a project manager, and as a curiosity, he does not have autonomy to define what will be done and by whom will be performed. This result shows the need to have a scrum master in the team to follow agility, which may mean a contradiction and dependency. Furthermore, there was a contest of opinions between to performed activities already structured without the involvement of the development team in building user stories or to use Planning Poker. The team has the autonomy to choose the practices that best suit their profile. If there are activities performed by one person, for example, the PO has to build the user stories and developers and testers only 'implement' and 'test,' the team becomes responsive and not assertive. However, if all participants said the team's position as being assertive, our study is not able to judge this result. Thus would require the authors do another empirical study, a case study or ethnography to confirm the results. This will be assigned to the final work of this Master Thesis 7.3.

In the deliveries pace category sprints do not change, and the team can develops the estimated delivery time if not performed they are placed in the release. Tasks not completed at the end of the sprint are addressed to the release. But changing requirements during the sprint are not well seen by the team. Participants were emphatic in saying that they do not like to receive extra demands, and they do not have comfortable to change deliveries during the sprint. This shows that the team does not accept changes during the work.

However, to follow agile principles the team also needed to welcome changing requirements, even late in development.

In the features disclosure category, PO defines the user stories aligning with stakeholders and Internal Customers which will be held during the sprint. When sent to developers, they have autonomy to change the stories. But participants said that they rarely change the requirements during the sprint, causing the non-integration of the activities by the whole team. Thus, the definition and quality of the requirements are the POs responsibility. Also, an external customer does not know the existence of the projects nor in how the process of deliveries is performed.

Participants chose all the characteristics in the software product category. Thus, even though they had a flawed automated test development process. There are efficient code initiatives with the support of infrastructure and agile practices determined for code.

In the customer relationship category, there are two answers for the two types of customers: internal and external customers. The internal customer projects stakeholders who understand how they planned the deliveries process. External are end customers who do not have contact with the team and do not know about the accomplishment of the project.

In the organizational support category, agile commitment was the feature most prominently. Agile business could also have been a choice of the participants, but participants disregarded it because the organization does not have manifested agile commitment before the teams. Not being a top-down initiative. However, agile priority can also be considered, once the company also has a responsibility to make available all technology and training to the teams better empower agile software development.

Answering the research questions (Q1 and Q2), we summarize the main results of each category in Table 6.7. The evidence and challenges were drawn from the analysis with the comments of all participants.

6.4 Threats to Validity

This study includes some limitations described next. To reduce the researcher's bias we sought a theoretical basis for better interpretation of results.

These two studies get reports of respondents about their perspectives, thoughts, and conclusions. However, it is possible that the reports are not reliable with the activities they perform. Because to be tacit respondents believe that facts are not relevant, or fail to report certain information by some fear or to report the activities expected to be performed. Thus, to mitigate this limitation we follow recommendations of Kotio et. al. (2008) [60], and Krueger (1994) [63].

Table 6.7: Results from the presence of agility - Company B

Agile Learning Category	**Evidence**	**Challenge**
[O1] Sensemaking of work processes [O2] Comprehension of situation	[E1]Tailoring practices for their needs. [E2] The team have visibility and simple metrics	[C1] In the past they have some resistance of agile adoption
Team Conduct Category	**Evidence**	**Challenge**
[O3] Assertive Team	[E3] The team feels responsible for the project. [E4] The team have autonomy to improve their work process. [E5] The team promote practices to create autonomy and empowerment to the team about their work.	[C2] Team is mature enough to focus on just technical excellence
Deliveries Pace Category	**Evidence**	**Challenge**
[O4]Expected frequent deliveries [O5] Defined frequent deliveries	[E6] Sometimes, the team needs to address extra demands in the sprint [E7] The team plan their deliveries and performed them on time [E8] The sprint size do not change [E9] Environment supports automated build and continuous integration [E10] Tests are usually manual	[C3] To address extra demands in the sprint [C4] Layoffs of support people and the decrease of the teams [C5] The infrastructure dependencies.
Feature Disclosure Category	**Evidence**	**Challenge**
[O6] Requirements discovery [O7] Requirements quality	[E11] The team detail requirements in the sprint. [E12] PO, and intern customer align requirements definition to make sure the both expectations.	-
Software Product Category	**Evidence**	**Challenge**
[O8] Awareness of failures [O9] High-level source code [O10] High-level delivered software [O11] Efficient coding	[E13] The team has initiatives and policies to guarantee the quality code.	[C6] Lack of automated tests
Customer Relationship Category	**Evidence**	**Challenge**
[O12] Team awareness of customer [O13] Confident customer	[E14] The external customers, is the end users who do not know much about the project. [E15] The team can meet the customer's expectations and needs.	[C7] The team does not know if the expectations of the external customers are reached
Organizational Support Category	**Evidence**	**Challenge**
[O14] Agile Commitment	[E16] The company supports agile transformation in in all teams [E17] Botton-up initiative of agile transformation	-

To ensure that data analysis, we used grounded theory proposed by Strauss, and Corbin (1990) [113] to classify the data, following the cyclic process of collecting, analysis, and reflection.

6.5 Chapter Summary

This chapter presented results from two focus group session in two different LSAD companies. The focus of this study was to investigate the presence of agility in large-scale agile teams. We conducted two studies in different scenarios in which they brought two different results on the agility perspective. The first large-scale agile team has less agility in comparison the second one team. The first team is in the early stages of agility. And the second team is more mature in agile, as they have achieved more that half of the characteristics. These results mean that large-scale agile teams can not be generalized according to their agility. Depending on the context and the maturity of the team, we can identify different results on the presence of agility.

7. FINAL CONSIDERATIONS

This chapter summarizes the research presented in this Master Thesis and presents suggestions and directions for future work. The chapter begins with a review of the research objectives while addressing the research question in Section 7.1. Section 7.2 describes the research limitations of the study. The chapter ends with some suggestions for future work in Section 7.3. Figure 7.1 summarizes how we developed our work and presented throughout the Master Thesis.

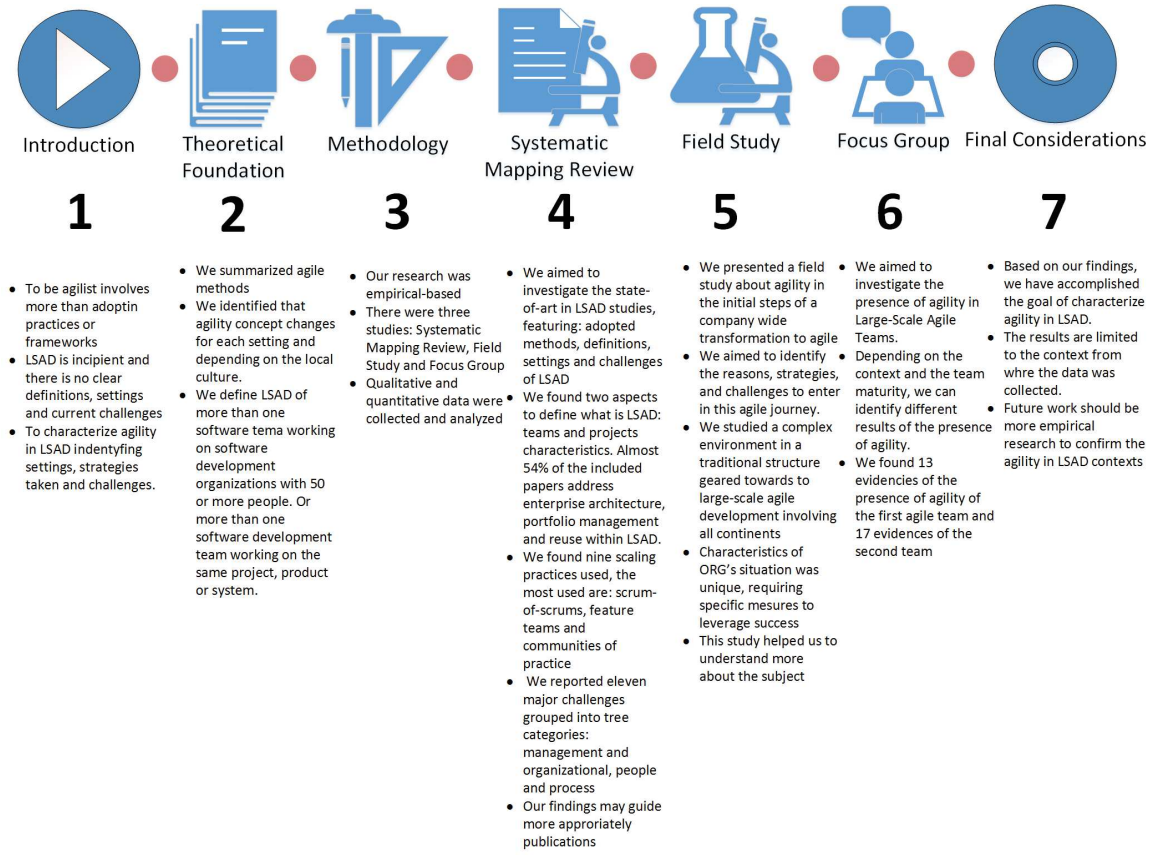


Figure 7.1: Master Thesis summary

7.1 Review of the Research Objectives

From 2001 [96] to 2016 [29], the LSAD term has been published under the condition of large settings adopting agile methods. However, this growing adoption requires more empirical research to investigate the current gaps [29]. Approaching the research problem of Reifer, Maurer and Erdogmus (2003) [96] this Master Thesis addressed the research question: *"To what extent is agility present in large-scale agile software development?"* An-

swered by characterizing agility in LSAD and aiming to identify whether agility is present in large-scale settings.

Thus, the research methodology followed in this study (see Chapter 3) contributed to achieving the research objectives and to answer our research question. First, through an systematic mapping review of the literature (Study 1), after the analysis of a field study (Study 2) and a focus group study (Study 3), publications, and partnerships. This Master Thesis contributes to the agile software development research area, providing a better understanding of LSAD and identifying the presence of agility in this scenario.

In order to achieve our goal, the following objectives were defined as described in Section 1.1:

- **(Obj1).** To identify the state-of-art of LSAD in literature. (Study 1).
- **(Obj2).** To empirically identify how a large-scale agile company goes through the transformation of becoming agile. (Study 2).
- **(Obj3).** To empirically identify whether agility is present in large-scale agile teams and the challenges faced by such teams to be agile in such setting (Study 3).

The Theoretical Foundation presented in Chapter 2 has provided the theoretical knowledge about agile software development, agility, and LSAD concepts. The concepts presented in this chapter contributed to guiding further investigation on the systematic literature review on LSAD. Furthermore, these results contributed to defining the three methods followed in this research.

To achieve the Obj1 we conducted a systematic mapping review in LSAD (Study 1). The systematic mapping study presented in Chapter 4 identified (1) what are the definitions for LSAD, (2) in which settings LSAD takes place (agile practices adopted, project sizes, team sizes, enterprise backround, etc), and (3) which are the challenges faced by agile teams in such settings.

To achieve Obj2 we conducted a field study (Study 2). The field study presented in Chapter 5 allowed to identify how a large-scale agile company goes through the transformation to become agile. We found five reasons for adopting agile software development, six major strategies were taken, and nine concerns about scaling agile were identified from this study. This study ws a significant contribution to this research.

To achieve Obj3 we conducted a focus group study (Study 3). The focus group study presented in Chapter 6 contributed to identifying whether agility is achieved in LSAD teams. This study allowed identifying the presence of agility in LSAD teams and the challenges faced by them.

7.2 Research Limitations

The main threats to the validity of this research are related to the methodological process. Each research method adopted in this Master Thesis followed recommendations from the literature. To ensure the efficacy of this research we systematically executed the research methods using formals protocols to guide each research method adopted in this Master Thesis. Also, we had two others researchers to review the process.

We adopted a systematic mapping review method to get the state-of-art of the LSAD area. As a systematic mapping review, threats to the validity of the process such as study selection, inaccuracy in data extraction, incorrect classification of studies, research methods and types and potential author bias must be considered. To reduce author's bias, at least two researchers reviewed the systematic mapping review protocol and results obtained in this study. In a case of disagreement between reviewers. Disagreements were reviewed until a consensus is achieved.

We performed only one field study in this research. However, the company case is a large company with complex-globally distributed projects. All participants are managers at this company and are either a member of the CIO Committee board, a director or a development manager.

We also adopted a Focus Group Study. During the sessions, some participants may restrain or be concerned about confidentiality. Thus, there is also a possibility that the results can be influenced by personal reasons creating an inaccurate view or making the output biased. Peculiarities and threats to the validity of each research methods were presented in detail previously in Sections 4.3.2, 5.3.2, and 6.4.

To minimize other threats, the research proposal and its preliminary results were presented in workshops and submitted to conferences. The proposal was presented in The Second Latin-American School of Software Engineering (ELA-ES) (as Best Poster Award), and Amazon Advanced School on Software Quality (AASSQ) (as a poster). Initial results were submitted to the WBMA (Brazilian Workshop on Agile Methods) (Appendix 4).

We gain much knowledge from both research visits (field study and focus group sessions) and conferences. This experience contributed to improving all steps from this research, getting a better understanding of results, gaps in the research, and propose the characterization of agility in LSAD.

7.3 Future Work

This Master Thesis presents contributions to the literature previously mentioned in Section 1.2. Since this work addresses the characterization of the presence of agility

in LSAD, several topics have emerged from this research work that needs to be examined further. Based on these results, next we describe directions for future work:

- The need to develop an empirically based terminology and taxonomy for the key concepts in LSAD. This contributes for future researchers, who will publish further empirical work and practitioners, who are interested in published empirical cases;
- To extend the focus group study with other software agile teams with the same project, product, Company or also in different Companies. In different contexts and beyond the IT industry;
- To empirically measure how the benefits of agility impact on teams and LSAD projects;
- To empirically measure new challenges and concerns about the adoption of LSAD;
- To empirically identify the applicability of Scaling Agile Practices;
- Partnering with large technology Companies that adopt agile and research groups that have contact with these Companies to carry out other empirical studies on this topic;
- To extend the Progressive Outcomes Framework to supplement the mechanisms it points out to support large-scale agile teams to evolve.

REFERENCES

[1] Abdullah, E.; Abdelsatir, E.-T. B. “Extreme programming applied in a large-scale distributed system”. In: Proceedings of the International Conference on Computing, Electrical and Electronics Engineering. Khartoum, Sudan, IEEE, 2013, pp. 442–446.

[2] Abrantes, J. F.; Travassos, G. H. “Towards pertinent characteristics of agility and agile practices for software processes”, *CLEI Electronic Journal*, vol. 16–1, 2013, pp. 6–6.

[3] Alzoubi, Y. I.; Gill, A. Q.; Al-Ani, A. “Empirical studies of geographically distributed agile development communication challenges: A systematic review”, *Information and Management*, vol. 53–1, 2016, pp. 22–37.

[4] Ambler, S. W. “Agile software development at scale”. In: *Balancing Agility and Formalism in Software Engineering*, Meyer, B.; Nawrocki, J. R.; Walter, B. (Editors), Springer Berlin Heidelberg, 2008, pp. 1–12.

[5] Ambler, S. W. “IBM agility at scale: Become as agile as you can be”. Source: https://www.ibm.com/developerworks/community/blogs/ambler?lang=en, Accessed in Nov 2015.

[6] Ambler, S. W.; Lines, M. “Disciplined agile delivery: A practitioner’s guide to agile software delivery in the enterprise, First Edition”. Indianapolis, USA: Pearson Education, IBM Press, 2012, 544p.

[7] Anderson, D. J. “Kanban: Successful evolutionary change for your technology business”. Blue Hole Press Sequim, WA, 2010, 278p.

[8] Anderson, D. J. “Lessons in agile management: On the road to Kanban”. Blue Hole Press, 2012, 434p.

[9] Atlas, A. “Accidental adoption: The story of scrum at amazon.com”. In: Proceedings of the Agile Conference, Chicago, USA, 2009, pp. 135–140.

[10] Babinet, E.; Ramanathan, R. “Dependency management in a large agile environment”. In: Proceedings of the Agile Conference, Toronto, Canada, IEEE, 2008, pp. 401–406.

[11] Bass, J. M. “Agile method tailoring in distributed enterprises: Product owner teams”. In: Proceedings of the International Conference on Global Software Engineering, Bari, Italy, IEEE, 2013, pp. 154–163.

[12] Beck, K. “Extreme programming explained: Embrace change”. Addison-Wesley Professional, 2000, 224p.

[13] Beck, K.; colleagues. “Manifesto for agile software development”. Source: http://www.agilemanifesto.org/, Accessed in Nov 2015.

[14] Beck, K.; Fowler, M. “Planning Extreme Programming”. Addison-Wesley Professional, 2001, 160p.

[15] Benefield, G. “Rolling out agile in a large enterprise”. In: Proceedings of the Hawaii International Conference on System Sciences, Big Island, Hawaii, IEEE, 2008, pp. 461–471.

[16] Berger, C.; Eklund, U. “Expectations and challenges from scaling agile in mechatronics-driven companies – A comparative case study”. Cham: Springer International Publishing, 2015, pp. 15–26.

[17] Biolchini, J.; Mian, P. G.; Natali, A. C. C.; Travassos, G. H. “Systematic review in software engineering”, *System Engineering and Computer Science Department COPPE/UFRJ, Technical Report ES*, vol. 679–05, 2005, pp. 45.

[18] Blau, B.; Hildenbrand, T. “Product line engineering in large-scale lean and agile software product development environments - Towards a hybrid approach to decentral control and managed reuse”. In: Proceedings of the International Conference on Availability, Reliability and Security, Vienna, Austria, IEEE, 2011, pp. 404–408.

[19] Budgen, D.; Turner, M.; Brereton, P.; Kitchenham, B. “Using mapping studies in software engineering”. In: Proceedings of the Workshop of Psychology of Programming Interest Group, Lancaster, UK, Lancaster University, 2008, pp. 195–204.

[20] CEO, A. K. “Maxqda - The art of data analysis”. Source: http://www.maxqda.com/, Accessed in Oct 2016.

[21] Cloke, G. “Get your agile freak on! agile adoption at yahoo! music”. In: Proceedings of the Agile Conference, Washington, USA, IEEE, 2007, pp. 240–248.

[22] Cockburn, A. “Agile Software Development”. Boston, USA: Addison-Wesley, 2002, 278p.

[23] Cockburn, A.; Highsmith, J. “Agile software development: The people factor”, *IEEE Computer*, vol. 34–11, 2001, pp. 131–133.

[24] Cohen, D.; Lindvall, M.; Costa, P. “An introduction to agile methods”, *Advances in Computers*, vol. 62, 2004, pp. 1–66.

[25] Conboy, K. “Agility from first principles: Reconstructing the concept of agility in Information Systems Development”, *Information Systems Research*, vol. 20–3, 2009, pp. 329–354.

[26] Corbin, J.; Strauss, A. “Basics of qualitative research: Techniques and procedures for developing Grounded Theory”. Sage Publications, 2014, 400p.

[27] Cristal, M.; Wildt, D.; Prikladnicki, R. “Usage of Scrum practices within a global company”. In: Proceedings of the International Conference on Global Software Engineering, Bangalore, India, 2008, pp. 222–226.

[28] Daneva, M. “Focus group: Cost-effective and methodologically sound ways to get practitioners involved in your empirical RE research”, *REFSQ-JP*, vol. 1342, 2015, pp. 211–216.

[29] Dikert, K.; Paasivaara, M.; Lassenius, C. “Challenges and success factors for large-scale agile transformations: A systematic literature review”, *Journal of Systems and Software*, vol. 119, 2016, pp. 87–108.

[30] Dingsøyr, T.; Moe, N. B. “Research challenges in large-scale agile software development”, *SIGSOFT Software Engineering Notes*, vol. 38–5, Aug 2013, pp. 38–39.

[31] Dingsøyr, T.; Moe, N. B. “Towards principles of large-scale agile development”. In: Proceedings of Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshop, in conjunction with the International Conference of Agile Processes in Software Engineering and Extreme Programming, Rome, Italy, Springer., 2014, pp. 1–8.

[32] Dingsøyr, T.; Nerur, S.; Balijepally, V.; Moe, N. B. “A decade of agile methodologies: Towards explaining agile software development”, *Journal of Systems and Software*, vol. 85–6, 2012, pp. 1213–1221.

[33] Dingsøyr, T.; Fægri, T.; Itkonen, J. “What is large in large-scale? A taxonomy of scale for agile software development”. In: *Product-Focused Software Process Improvement*, Jedlitschka, A.; Kuvaja, P.; Kuhrmann, M.; Männistö, T.; Münch, J.; Raatikainen, M. (Editors), Springer International Publishing, 2014, pp. 273–276.

[34] Doyle, M.; Williams, L.; Cohn, M.; Rubin, K. S. “Agile software development in practice”. In: Proceedings of the Agile Processes in Software Engineering and Extreme Programming, Rome, Italy, Springer, 2014, pp. 32–45.

[35] Dybå, T.; Dingsøyr, T. “Empirical studies of agile software development: A systematic review”, *Information and Software Technology*, vol. 50–9, 2008, pp. 833–859.

[36] Eckstein, J. “Architecture in large scale agile development”. In: *Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshop, in conjunction with International Conference on Agile Processes in*

Software Engineering and Extreme Programming, Rome, Italy,, Springer, 2014, pp. 21–29.

[37] Eklund, U.; Olsson, H. H.; Strøm, N. J. “Industrial challenges of scaling agile in mass-produced embedded systems”. In: *Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshop, in conjunction with International Conference on Agile Processes in Software Engineering and Extreme Programming, Rome, Italy*, Springer, 2014, pp. 30–42.

[38] Elshamy, A.; Elssamadisy, A. “Applying agile to large projects: new agile software development practices for large projects”. In: *Proceedings of the Agile Processes in Software Engineering and Extreme Programming*, Berlin, Germany: Springer, 2007, pp. 46–53.

[39] Erickson, J.; Lyytinen, K.; Siau, K. “Agile modeling, agile software development, and extreme programming: the state of research”, *Journal of Database Management*, vol. 16–4, 2005, pp. 1–13.

[40] Fægri, T. E.; Moe, N. B. “Re-conceptualizing requirements engineering: findings from a large-scale, agile project”. In: Scientific Workshop Proceedings of the XP2015, in conjunction with the International Conference on Agile Processes in Software Engineering and Extreme Programming, Helsinki, Finland, ACM, 2015, pp. 4–5.

[41] Fitzgerald, B.; Stol, K.-J.; O’Sullivan, R.; O’Brien, D. “Scaling agile methods to regulated environments: An industry case study”. In: Proceedings of the International Conference on Software Engineering, San Francisco, CA, USA, IEEE, 2013, pp. 863–872.

[42] Fontana, R.; Meyer, V.; Reinehr, S.; Malucell, A. “Progressive outcomes: A framework for maturing in agile software development”, *Journal of Systems and Software*, vol. 102, 2015, pp. 88–108.

[43] Fontana, R. M.; Reinehr, S.; Malucelli, A. “Agile compass: A tool for identifying maturity in agile software-development teams”, *IEEE Software*, vol. 32–6, 2015, pp. 20–23.

[44] Fry, C.; Greene, S. “Large scale agile transformation in an on-demand world”. In: Proceedings of the Agile Conference, Washington, USA, IEEE, 2007, pp. 136–142.

[45] Gandomani, T. J.; Zulzalil, H.; Nafchi, M. Z. “Agile transformation: What is it about?” In: Proceedings of the Malaysian Software Engineering Conference, Selangor, Malaysia, IEEE, 2014, pp. 240–245.

[46] Gat, I. “How BMC is scaling agile development”. In: Proceedings of the Agile Conference, Minneapolis, USA, IEEE, 2006, pp. 1–6.

[47] Goodman, D.; Elbaz, M. “Italy’s not the pants, italy’s the people in the pants - Learnings from the gap agile transformation what worked, how we did it, and what still puzzles us”. In: Proceedings of the Agile Conference, Toronto, Canada, IEEE, 2008, pp. 112–115.

[48] Greening, D. R. “Enterprise scrum: Scaling scrum to the executive level”. In: Proceedings of the Hawaii International Conference on System Sciences, Kauai, Hawai, IEEE, 2010, pp. 1–10.

[49] Gul, E.; Sekerci, T.; Yuceturk, A.; Yildirim, U. “Using XP in telecommunication software development”. In: Proocedings of the International Conference on Software Engineering Advances, Sliema, Malta, IEEE, 2008, pp. 258–263.

[50] Gunyho, G.; Plaza, J. G. “Evolution of longer-term planning in a large scale agile project–F-secure’s experience”. In: Proceedings of the International Conference of Agile Processes in Software Engineering and Extreme Programming. Madrid, Spain, Springer, 2011, pp. 306–315.

[51] Hallikainen, M. “Experiences on agile seating, facilities and solutions: Multisite environment”. In: Proceedings of the International Conference on Global Software Engineering, Finland, IEEE, 2011, pp. 119–123.

[52] Heikkila, V. T.; Paasivaara, M.; Lassenius, C. “Scrumbut, but does matter? A mixed-method study of the planning process of a multi-team scrum organization”. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement. Baltimore, USA, IEEE, 2013, pp. 85–94.

[53] Heikkilä, V. T.; Paasivaara, M.; Rautiainen, K.; Lassenius, C.; Toivola, T.; Järvinen, J. “Operational release planning in large-scale scrum with multiple stakeholders–a longitudinal case study at f-secure corporation”, *Information and Software Technology*, vol. 57, 2015, pp. 116–140.

[54] Highsmith, J. A. “Agile software development ecosystems”. Addison-Wesley Professional, 2002, 448p.

[55] Ivček, M.; Galinac, T. “Adapting agile practices in globally distributed large scale software development”. In: Proceedings of the International Conference on Telecommunications and Information, Rijeka, Croatia, CiteSeer X, 2009, pp. 1–6.

[56] Kitchenham, B.; Brereton, P. “A systematic review of systematic review process research in software engineering”, *Information and Software Technology*, vol. 55–12, 2013, pp. 2049–2075.

[57] Kitchenham, B.; Charters, S. “Guidelines for performing Systematic Literature Reviews in Software Engineering”, Technical Report, Keele University and Durham University Joint Report, 2007, 65p.

[58] Kitzinger, J. “The methodology of focus groups: The importance of interaction between research participants”, *Sociology of Health & Illness*, vol. 16–1, 1994, pp. 103–121.

[59] Koehnemann, H.; Coats, M. “Experiences applying agile practices to large systems”. In: Proceedings of the Agile Conference, Washington, USA, IEEE, 2009, pp. 295–300.

[60] Kontio, J.; Bragge, J.; Lehtola, L. “The focus group method as an empirical tool in software engineering”. In: *Guide to advanced empirical software engineering*, Shull, F.; Singer, J.; Sjøberg, D. I. K. (Editors), Springer, 2008, pp. 93–116.

[61] Korhonen, K. “Evaluating the impact of an agile transformation: a longitudinal case study in a distributed context”, *Software Quality Journal*, vol. 21–4, 2013, pp. 599–624.

[62] Korhonen, K. “Evaluating the impact of an agile transformation: a longitudinal case study in a distributed context”, *Software Quality Journal*, vol. 21–4, 2013, pp. 599–624.

[63] Krueger, R. A. “Focus Groups: A practical guide for applied research”. Thounsand Oaks, USA: Sage Publications, 1994, 272p.

[64] Ktata, O.; Lévesque, G. “Agile development: Issues and avenues requiring a substantial enhancement of the business perspective in large projects”. In: Proceedings of the Canadian Conference on Computer Science and Software Engineering. Montreal, Canada, ACM, 2009, pp. 59–66.

[65] Laanti, M. “Characteristics and principles of scaled agile”. In: *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*, Dingsøyr, T.; Moe, N.; Tonelli, R.; Counsell, S.; Gencel, C.; Petersen, K. (Editors), Springer International Publishing, 2014, *Lecture Notes in Business Information Processing*, vol. 199, pp. 9–20.

[66] Laanti, M.; Salo, O.; Abrahamsson, P. “Agile methods rapidly replacing traditional methods at nokia: A survey of opinions on agile transformation”, *Information and Software Technology*, vol. 53–3, 2011, pp. 276–290.

[67] Laanti, M.; Sirkiä, R.; Kangas, M. “Agile portfolio management at Finnish broadcasting company Yle”. In: Scientific Workshop Proceedings of the XP2015, in conjunction with the International Conference on Agile Processes in Software Engineering and Extreme Programming, Helsinki, Finland, ACM, 2015, pp. 1–7.

[68] Lagerberg, L.; Skude, T.; Emanuelsson, P.; Sandahl, K.; Stahl, D. “The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two projects at ericsson”. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement, Baltimore, USA, IEEE, 2013, pp. 348–356.

[69] Langford, J.; McDonagh, D. “Focus groups: Supporting effective product development”. CRC Press, 2003, 240p.

[70] Larman, C.; Vodde, B. “Scaling lean and agile development: Thinking and organizational tools for large-scale scrum”. Boston, USA: Addison-Wesley Professional, 2008, 368p.

[71] Larman, C.; Vodde, B. “Practices for Scaling Lean and Agile Development: Large, Multisite, and Offshore Product Development with Large-Scale Scrum, First Edition”. Boston, USA: Addison-Wesley Professional, 2010, 624p.

[72] Larman, C.; Vodde, B. “Large-Scale Scrum: More with LeSS, First Edition”. Boston, MA, USA: Addison-Wesley Professional, 2016, 272p.

[73] LeanKit. “What is kanban board?” Source: https://leankit.com/learn/kanban/kanban-board/, Accessed in Oct 2016.

[74] Leffingwell, D. “Scaling Software Agility: Best practices for large enterprises,”. Boston, USA: Addison-Wesley Professional, 2007, 384p.

[75] Lippert, M.; colleagues. “Developing complex projects using xp with extensions”, *IT Computer*, 2003, pp. 67–73.

[76] Lyytinen, K.; Rose, G. M. “Information system development agility as organizational learning”, *European Journal of Information Systems*, vol. 15–2, 2006, pp. 183–199.

[77] Madden, S. “Versionone agile made easier”. Source: https://www.versionone.com/about/, Accessed in Dec 2015.

[78] Miller, A.; Carter, E. “Agility and the inconceivably large”. In: Proceedings of the Agile Conference, Washington, USA, 2007, pp. 304–308.

[79] Moe, N. B.; Šmite, D.; Šāblis, A.; Börjesson, A.-L.; Andréasson, P. “Networking in a large-scale distributed agile project”. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement. Torino, Italy, ACM, 2014, pp. 12.

[80] Moore, E. “Influence of large-scale organization structures on leadership behaviors”. In: Proceedings of the Agile Conference, Chicago, USA, IEEE, 2009, pp. 309–313.

[81] Morgan, D. L. “Focus groups”, *Annual Review of Sociology*, vol. 22, 1996, pp. 129–152.

[82] Nerur, S.; Mahapatra, R.; Mangalaraj, G. “Challenges of migrating to agile methodologies”, *Communications of the ACM*, vol. 48–5, 2005, pp. 72–78.

[83] Nord, R.; Ozkaya, I.; Kruchten, P. “Agile in distress: Architecture to the rescue”. In: Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshop, in conjunction with the International Conference of Agile Processes in Software Engineering and Extreme Programming, Rome, Italy, 2014, pp. 43–57.

[84] Olsson, H. H.; Bosch, J. “Towards continuous validation of customer value”. In: Scientific Workshop Proceedings of the XP2015, in conjunction with the International Conference of Agile Processes in Software Engineering and Extreme Programming, Helsinki, Finland, ACM, 2015, pp. 3:1–3:4.

[85] Olsson, H. H.; Bosch, J.; Alahyari, H. “Customer-specific teams for agile evolution of large-scale embedded systems”. In: Conference on Software Engineering and Advanced Applications. Santander, Spain, IEEE, 2013, pp. 82–89.

[86] Paasivaara, M.; Behm, B.; Lassenius, C.; Hallikainen, M. “Towards rapid releases in large-scale XaaS development at Ericsson: A case study”. In: International Conference on Global Software Engineering, Shanghai, China, IEEE, 2014, pp. 16–25.

[87] Paasivaara, M.; Heikkilä, V. T.; Lassenius, C. “Experiences in scaling the product owner role in large-scale globally distributed Scrum”. In: International Conference on Global Software Engineering, Porto Alegre, Brazil, IEEE, 2012, pp. 174–178.

[88] Paasivaara, M.; Lassenius, C. “Scaling Scrum in a large distributed project”. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement, Banff, Canada, IEEE, 2011, pp. 363–367.

[89] Paasivaara, M.; Lassenius, C. “Communities of practice in a large distributed agile software development organization–Case Ericsson”, *Information and Software Technology*, vol. 56–12, 2014, pp. 1556–1577.

[90] Paasivaara, M.; Lassenius, C. “Scaling scrum in a large globally distributed organization: A case study”. In: Proceedings of the International Conference on Global Software Engineering, Orange County, USA, IEEE, 2016, pp. 74–83.

[91] Paasivaara, M.; Lassenius, C.; Heikkila, V. T. “Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work?” In: Proceedings of the

International Symposium on Empirical Software Engineering and Measurement. Lund, Sweden, IEEE, 2012, pp. 235–238.

[92] Paasivaara, M.; Väättänen, O.; Hallikainen, M.; Lassenius, C. “Supporting a large-scale lean and agile transformation by defining common values”. In: *Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshops, in conjunction with the International Conference on Agile Processes in Software Engineering and Extreme Programming, Rome, Italy*, Springer, 2014, pp. 73–82.

[93] Papadopoulos, G. “Moving from traditional to agile software development methodologies also on large, distributed projects.”, *Procedia-Social and Behavioral Sciences*, vol. 175, 2015, pp. 455–463.

[94] Petersen, K.; Feldt, R.; Mujtaba, S.; Mattsson, M. “Systematic mapping studies in software engineering”. In: Proceedings of the International Conference on Evaluation and Assessment in Software Engineering. Bari, Italy, ACM, 2008, pp. 1–10.

[95] Power, K. “A model for understanding when scaling agile is appropriate in large organizations”. In: *Proceedings of the International Workshops: Large-Scale Development, Refactoring, Testing, and Estimation, in conjunction with the International Conference on Agile Processes in Software Engineering and Extreme Programming, Rome, Italy*, Springer, 2014, pp. 83–92.

[96] Reifer, D. J.; Maurer, F.; Erdogmus, H. “Scaling agile methods”, *IEEE Software*, vol. 20–4, July 2003, pp. 12–14.

[97] Rolland, K. H. “Desperately seeking research on agile requirements in the context of large-scale agile projects”. In: Proceedings of the Agile Processes, in Software Engineering, and Extreme Programming, Helsinki, Finland, ACM, 2015, pp. 1–6.

[98] Roman, G.; Marczak, S.; Dutra, A.; Prikladnicki, R. “On the agile transformation in a large-complex globally distributed company: Why boarding this journey, steps taken, and main foreseen concerns”. In: Proceedings of the Brazilian Workshop of Agile Methods, Recife, Brazil, 2015, pp. 1–8.

[99] Rubin, K. “Essential Scrum: A practical guide to the most popular agile process”. Pearson Education, 2013, 504p.

[100] Schiel, J. “Enterprise-Scale Agile Software Development Applied Software Engineering Series, First Edition”. New York City, USA: CRC Press, 2009, 382p.

[101] Schwaber, K. “Agile project management with Scrum”. Redmond, WA: Microsoft Press, 2004, 192p.

[102] Schwaber, K.; Beedle, M. “Agile Software Development with Scrum”. Prentice Hall, 2001, 192p.

[103] Seaman, C. B. “Qualitative methods in empirical studies of software engineering”, *IEEE Transactions Software Engineering*, vol. 25–4, Jul 1999, pp. 557–572.

[104] Shafagatova, A. “The importance and presence of agile principles in agile software development”, Master’s Thesis, Faculteit Economie en Bedrijfskunde, Universiteit Gent, Ghent, BE, 2015, 133p.

[105] Shah, V.; Nies, A. “Agile with fragile large legacy applications”. In: Proceedings of the Agile Conference, Toronto, Canada, IEEE, 2008, pp. 490–495.

[106] Shatil, A.; Hazzan, O.; Dubinsky, Y. “Agility in a large-scale system engineering project: A case-study of an advanced communication system project”. In: Proceedings of the International Conference on Software Science, Technology and Engineering, Herzelia, Israel, IEEE, 2010, pp. 47–54.

[107] Shull, F.; Singer, J.; Sjøberg, D. I. “Guide to advanced empirical software engineering”. Springer, 2007, vol. 93, 350p.

[108] Singer, J.; Sim, S. E.; Lethbridge, T. C. “Software Engineering Data Collection for Field Studies”. London: Springer London, 2008, pp. 9–34.

[109] Smits, H.; Rilliet, K. “Agile experience report: Transition and complexity at Cisco voice technology group”. In: Proceedings of the Agile Conference, Salt Lake City, USA, IEEE, 2011, pp. 274–278.

[110] Sommerville, I. “Software Engineering, Ninth Edition”. Boston, USA: Pearson, 2010, 792p.

[111] Spayd, M. K. “Evolving agile in the enterprise: Implementing XP on a grand scale”. In: Proceedings of the Agile Development Conference. Salt Lake City, USA, IEEE, 2003, pp. 60–70.

[112] Stojanov, I.; Turetken, O.; Trienekens, J. J. “A maturity model for scaling agile development”. In: Conference on Software Engineering and Advanced Applications, Madeira, Portugal, IEEE, 2015, pp. 446–453.

[113] Strauss, A. L.; Corbin, J. M. “Basics of qualitative research, Second Edition”. SAGE Publications, 1990, 272p.

[114] Sutherland, J. “Agile can scale: Inventing and reinventing Scrum in five companies”, *Cutter IT Journal*, vol. 14–12, 2001, pp. 5–11.

[115] Sutherland, J.; Schwaber, K. “The scrum guide”. Source: http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf, Accessed in Nov 2015.

[116] Sweetman, R.; O’Dwyer, O.; Conboy, K. “Control in software project portfolios: A complex adaptive systems approach”. In: *Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation Workshop, in conjunction with International Conference on Agile Processes in Software Engineering and Extreme Programming, Rome, Italy*, Springer, 2014, pp. 93–104.

[117] Tabib, R. “Need 4 speed: leverage new metrics to boost your velocity without compromising on quality”. In: Proceedings of the Agile Conference, Nashville, USA, IEEE, 2013, pp. 117–120.

[118] Taylor, G. R. “Integrating quantitative and qualitative methods in research. Second Edition”. University press of America, 2005, 357p.

[119] Turk, D.; France, R.; Rumpe, B. “Limitations of agile software processes”, *arXiv preprint arXiv:1409.6600*, 2014, pp. 43–46.

[120] Valade, R. “The big projects always fail: Taking an enterprise agile”. In: Proceedings of the Agile Conference, Toronto, Canada, IEEE, 2008, pp. 148–153.

[121] Wilby, D. “Roadmap transformation: from obstacle to catalyst”. In: Proceedings of the Agile Conference, Nashville, USA, IEEE, 2009, pp. 229–234.

[122] Wohlin, C. “Writing for synthesis of evidence in empirical software engineering”. In: Proceedings of the International Symposium on Empirical Software Engineering and Measurement. Torino, Italy. ACM, 2014, pp. 46–50.

[123] Yin, R. K. “Case study research: Design and methods, Fifth edition”. Sage Publications, 2013, 312p.

APPENDIX A – PAPERS INCLUDED IN THE SYSTEMATIC MAPPING REVIEW

ID	Reference
P1	Abdullah E. and E. T. B. Abdelsatir, "Extreme programming applied in a large-scale distributed system," In proocedings of the International Conference on Computing, Electrical and Electronics Engineering, Khartoum, 2013, pp. 442-446.
P2	Ambler, S. W. "Agile software development at scale," In the Proceedings of the Balancing Agility and Formalism in Software Engineering, Springer Berlin Heidelberg, 2008. pp.1-12.
P3	Babinet E. and R. Ramanathan, “Dependency Management in a Large Agile Environment”. In Proceedings of the Agile Conference, 2008, Washington, DC, USA, pp. 401-406.
P4	Badampudi D., S. A. Fricker and A. M. Moreno, "Perspectives on Productivity and Delays in Large-Scale Agile Projects", In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer Berlin Heidelberg, 2013 pp 203-210.
P5	Bass J. M., "Agile Method Tailoring in Distributed Enterprises: Product Owner Teams," In proocedings of the International Conference on Global Software Engineering, Bari, 2013, pp. 154-163.
P6	Bass M. J., "How product owner teams scale agile methods to large distributed enterprises," Empirical Software Engineering, v.20, n.6, pp. 1525-1557, 2015.
P7	Berger C. and U., "Expectations and Challenges from Scaling Agile in Mechatronics-Driven Companies–A Comparative Case Study," In Proceedings of the Agile Processes in Software Engineering, and Extreme Programming, Springer International Publishing, 2015, pp. 15-26.
P8	Bjarnason E., K. Wnuk, and B. Regnell, "A case study on benefits and side-effects of agile practices in large-scale requirements engineering", In Proceedings of the Workshop on Agile Requirements Engineering, New York, NY, USA, 2011, Article 3 , 5 pages.
P9	Blau B. and T. Hildenbrand, "Product Line Engineering in Large-Scale Lean and Agile Software Product Development Environments - Towards a Hybrid Approach to Decentral Control and Managed Reuse," In Proceedings of the International Conference on Availability, Reliability and Security , Vienna, 2011, pp. 404-408.
P10	Brown A. W., S. Ambler, W. Royce, "Agility at scale: economic governance, measured improvement, and disciplined delivery", In Proceedings of the International Conference on Software Engineering, San Francisco, CA, 2013, pp. 873-881.
P11	Brown W. A., "A case study in agile-at-scale delivery," In Proceedings of the Agile Processes in Software Engineering and Extreme Programming, Springer Berlin Heidelberg, 2011, pp. 266-281.
P12	Bruegge, B., M. Reiss and J. Schiller, "Agile Principles in Academic Education: A Case Study," In Proceedings of the International Conference on Information Technology: New Generations, Las Vegas, NV, 2009, pp. 1684-1686.
P13	Cao L., K. Mohan, P. Xu, and B. Ramesh. How Extreme Does Extreme Programming Have to Be? Adapting XP Practices to Large-Scale Projects. In Proceedings of the 37th Annual Hawaii International Conference on System Sciences, Washington, DC, USA, 2004 p.3-.

P14	Daneva M., E. V. der Veen, C. Amrit, S. Ghaisas, K. Sikkel, R. Kumar, N. Ajmeri, U. Ramteerthkar, R. Wieringa, "Agile requirements prioritization in large-scale outsourced system projects: An empirical study", Journal of Systems and Software, Vol 86, Nro 5, pp. 1333-1353 May 2013,
P15	Datta S., R. Sindhgatta, and B. Sengupta, "Evolution of developer collaboration on the jazz platform: a study of a large scale agile project," In Proceedings of the India Software Engineering Conference, New York, NY, USA, 2011, pp. 21-30.
P16	Dingsoyr T., T. E. Faegri and J. Itkonen, "What is large in large-scale? A taxonomy of scale for agile software development," In Proceedings of the Product-Focused Software Process Improvement, Springer International Publishing, 2014. pp. 273-276.
P17	Dubinsky Y., David Talby, Orit Hazzan, and Arie Keren, “Agile Metrics at the Israeli Air Force,” In Proceedings of the Agile Conference, Washington, DC, USA, pp. 12-19.
P18	Eckstein J., "Architecture in Large Scale Agile Development," In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2011, pp. 21-29.
P19	Eklund U., H.H. Olsso and N. J. Strom, "Industrial challenges of scaling agile in mass-produced embedded systems," . In Proceedings of the Agile Methods, Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2014, pp. 30-42.
P20	Fægri, T. E.; Moe, N. B. “Re-conceptualizing requirements engineering: findings from a large-scale,agile project”. In: Scientific Workshop Proceedings of the XP2015, in conjunction with the InternationalConference on Agile Processes in Software Engineering and Extreme Programming, Helsinki, Finland,ACM, 2015, pp. 4 -5.
P21	Farmer M., "DecisionSpace infrastructure: Agile development in a large, distributed team," In Proceedings of the Agile Conference, Denver, CO, 2004, pp. 95-99.
P22	Farrow A. and S. Greene, "Fast & Predictable A Lightweight Release Framework Promotes Agility through Rhythm and Flow," In the Proceedings of the Agile Conference, Toronto, ON, 2008, pp. 224-228.
P23	Fitzgerald B., K. J. Stol, R. O'Sullivan and D. O'Brien, "Scaling agile methods to regulated environments: An industry case study," In Proceedings of the International Conference on Software Engineering, San Francisco, CA, 2013, pp. 863-872.
P24	Fry C. and S. Greene, "Large Scale Agile Transformation in an On-Demand World," In Proceedings of the Agile Conference, Washington, DC, 2007, pp. 136-142.
P25	Gat I., "How BMC is scaling agile development," In Proceedings of the Agile Conference, Minneapolis, MN, 2006, pp. 320-326.
P26	Gill A. Q., "Learning Enterprise Agile Software Engineering: Scaling Agility at the Enterprise Level," In Proceedings of the International Conference on Software Engineering, Australasian, Adelaide, SA, 2015, pp. 148-154.
P27	Glas M. and S. Ziemer, "Challenges for agile development of large systems in the aviation industry", In Proceedings of the Conference Companion on Object Oriented Programming Systems Languages and Applications, New York, NY, 2009, USA, pp. 901-908.

P28	Goodman D. and M. Elbaz, ""It's Not the Pants, it's the People in the Pants" Learnings from the Gap Agile Transformation What Worked, How We Did it, and What Still Puzzles Us," In Proceedings of the Agile Conference, Toronto, ON, 2008, pp. 112-115.
P29	Greening D. R., "Enterprise Scrum: Scaling Scrum to the Executive Level," In Proceedings of the Hawaii International Conference on System Sciences, Honolulu, HI, 2010, pp. 1-10.
P30	Grewal H. and F. Maurer, "Scaling Agile Methodologies for Developing a Production Accounting System for the Oil & Gas Industry," In Proceedings of the Agile Conference, Washington, DC, 2007, pp. 309-315.
P31	Gul E., T. Sekerci, A. C. Yücetürk and Ü Yildirim, "Using XP in Telecommunication Software Development," In Proceedings of the International Conference on Software Engineering Advances, Sliema, 2008, pp. 258-263.
P32	Gunyho G. and J. G. Plaza, "Evolution of longer-term planning in a large scale agile project–F-Secure's experience," In Proceedings of the Agile Processes in Software Engineering and Extreme Programming, Springer Berlin Heidelberg, 2011, pp. 306-315.
P33	Hadar I., S. Sherman, E. Hadar and J. J. Harrison, "Less is more: Architecture documentation for agile development," In Proceedings of the International Workshop on Cooperative and Human Aspects of Software Engineering, San Francisco, CA, 2013, pp. 121-124.
P34	Hallikainen, M. "Experiences on Agile Seating, Facilities and Solutions: Multisite Environment," In Proceedings of the International Conference on Global Software Engineering, Helsinki, 2011, pp. 119-123.
P35	Hannay J. E. and H. C. Benestad. "Perceived productivity threats in large agile development projects", In Proceedings of the International Symposium on Empirical Software Engineering and Measurement, New York, NY, USA, 2010, pp. 15 -25.
P36	Heikkilã V. T. et al. "Continuous release planning in a large-scale scrum development organization at Ericsson", In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer Berlin Heidelberg, 2013, pp.195-209
P37	Heikkilä V. T., M. Paasivaara and C. Lassenius, "ScrumBut, But Does it Matter? A Mixed-Method Study of the Planning Process of a Multi-team Scrum Organization," In the Proceedings of the International Symposium on Empirical Software Engineering and Measurement, Baltimore, MD, 2013, pp. 85-94.
P38	Heikkila V., K. Rautiainen and S. Jansen, "A Revelatory Case Study on Scaling Agile Release Planning," In Proceedings of the Conference on Software Engineering and Advanced Applications , Lille, 2010, pp. 289-296.
P39	Heikkilä, V. T., M. Paasivaara, K. Rautiainen, C. Lassenius, T. Toivola, J. Järvinen, "Operational release planning in large-scale Scrum with multiple stakeholders – A longitudinal case study at F-Secure Corporation", Information and Software Technology, Vol. 57, Nro. 12, 2015, pp.116-140.
P40	Isham M.,"Agile Architecture IS Possible You First Have to Believe!," In the Proceedings of the Agile Conference, Toronto, ON, 2008, pp. 484-489.

P41	Johannessen K. L. and G. Ellingsen, "Lightweight methods in heavyweight organizations" In the Proceedings of the Tenth Anniversary Conference on Participatory Design, Indiana University, Indianapolis, IN, USA, 2008, pp.11-20.
P42	Koehnemann H. and M. Coats, "Experiences Applying Agile Practices to Large Systems," In Proocedings of the Agile Conference, Chicago, IL, 2009, pp. 295-300.
P43	Koski A. and T. Mikkonen, "Rolling Out a Mission Critical System in an Agilish Way. Reflections on Building a Large-Scale Dependable Information System for Public Sector," In Proceedings of the International Workshop on Rapid Continuous Software Engineering, Florence, Italy, 2015, pp. 41-44.
P44	Ktata O. and G. Lévesque, "Agile development: issues and avenues requiring a substantial enhancement of the business perspective in large projects," In Proceedings of the Canadian Conference on Computer Science and Software Engineering, New York, NY, USA, pp. 59-66, 2009
P45	Laanti M. , "Implementing Program Model with Agile Principles in a Large Software Development Organization," In Proceedings of the International Conference on Computer Software and Applications,Turku, 2008, pp. 1383-1391.
P46	Laanti M. and M. Kangas, "Is Agile Portfolio Management Following the Principles of Large-Scale Agile? Case Study in Finnish Broadcasting Company Yle," In Proceedings of the Agile Conference, 2015, Washington, USA, 2015, pp. 92-96.
P47	Laanti M., "Characteristics and principles of scaled agile," In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2014, pp. 9-20.
P48	Laanti M., Salo O., Abrahamsson P., "Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation," Information and Software Technology, Vol. 53, Nro. 3, 2011, pp. 276-290.
P49	Lagerberg L., T. Skude, P. Emanuelsson, K. Sandahl and D. Ståhl, "The Impact of Agile Principles and Practices on Large-Scale Software Development Projects: A Multiple-Case Study of Two Projects at Ericsson," In Proceedings of the International Symposium on Empirical Software Engineering and Measurement, Baltimore, MD, 2013, pp. 348-356.
P50	Lippert M. et al., "Developing complex projects using XP with extensions," In Computer, vol. 36, Nro. 6, 2003, pp. 67-73,.
P51	Maples, C. "Enterprise Agile Transformation: The Two-Year Wall," In Proceedings of the Agile Conference, Chicago, IL, 2009, pp. 90-95.
P52	Maranzato R. P., M. Neubert and P. Herculano, "Moving back to scrum and scaling to scrum of scrums in less than one year" In Proceedings of the International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion, New York, NY, 2011, USA, 125-130.
P53	Martini A., L. Pareto; J. Bosch, "Communication factors for speed and reuse in large-scale agile software development," In Proceedings of the International Software Product Line Conference, 2013. pp. 42-51.
P54	Martini A., L. Pareto; J. Bosch, "A multiple case study on the inter-group interaction speed in large, embedded software companies employing agile". Journal of Software: Evolution and Process, vol 28, Nro 1, 2015, pp. 4-26.

P55	Miller A. and E. Carter, "Agility and the Inconceivably Large," In Proceedings of the Agile Conference, Washington, DC, 2007, pp. 304-308.
P56	Moore E. and J. Spens, "Scaling Agile: Finding your Agile Tribe," In Proceedings of the Agile Conference, Toronto, ON, 2008, pp. 121-124.
P57	Moore E., "Influence of Large-Scale Organization Structures on Leadership Behaviors," In Proceedings of the Agile Conference, Chicago, IL, 2009, pp. 309-313.
P58	Nils Brede Moe, Darja Šmite, Aivars Šāblis, Anne-Lie Börjesson, and Pia Andréasson. 2014. Networking in a large-scale distributed agile project. In Proceedings of the International Symposium on Empirical Software Engineering and Measurement, New York, NY, USA, 2014, pp 12 -20.
P59	Nord L. R., I. Ozkaya, P. Kruchten, "Agile in distress: architecture to the rescue" In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2014, pp. 43-57.
P60	Nyfjord J., S. Bathallath and H. Kjellin, "Conventions for Coordinating Large Agile Projects," In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2014, pp. 58-72.
P61	Olsson H. H., J. Bosch and H. Alahyari, "Customer-Specific Teams for Agile Evolution of Large-Scale Embedded Systems," In Proceedings of the Euromicro Conference on Software Engineering and Advanced Applications, Santander, 2013, pp. 82-89.
P62	Olsson H.H. and Jan Bosch, "Towards continuous validation of customer value" In Proceedings of the Scientific Workshop Proceedings of the XP, New York, NY, USA, 2015, 3-7pp.
P63	Onions P. and C. Patel, "Enterprise SoBA: Large-scale implementation of Acceptance Test Driven Story Cards," In Proceedings of the International Conference on Information Reuse & Integration, Las Vegas, NV, 2009, pp. 105-109.
P64	Paasivaara M. and C. Lassenius, "Scaling Scrum in a Large Distributed Project," In Proceedings of the International Symposium on Empirical Software Engineering and Measurement, Banff, AB, 2011, pp. 363-367.
P65	Paasivaara M., B. Behm, C. Lassenius and M. Hallikainen, "Towards Rapid Releases in Large-Scale XaaS Development at Ericsson: A Case Study," 2014 In Proceedings of the 9th International Conference on Global Software Engineering, Shanghai, 2014, pp. 16-25.
P66	Paasivaara M., C. Lassenius, "Communities of practice in a large distributed agile software development organization Case Ericsson", Information and Software Technology, Vol. 56, Nro. 12, 2014, pp. 1556-1577.
P67	Paasivaara M., C. Lassenius, “Deepening our understanding of communities of practice in large-scale agile development,” In Proceedings of the Agile Conference, Orlando, FL, 2014. p.37-40.
P68	Paasivaara M., C. Lassenius, and V. T. Heikkilä. “Inter-team coordination in large-scale globally distributed scrum: do scrum-of-scrums really work?” In Proceedings of the International Symposium on Empirical Software Engineering and Measurement (ESEM '12), New York, USA, pp. 235-238.

P69	Paasivaara M., S. Durasiewicz and C. Lassenius, "Distributed Agile Development: Using Scrum in a Large Project," In Proceedings of the International Conference on Global Software Engineering, Bangalore, 2008, pp. 87-95.
P70	Paasivaara M., V. T. Heikkilä and C. Lassenius, "Experiences in Scaling the Product Owner Role in Large-Scale Globally Distributed Scrum," In Proceedings of the 7th International Conference on Global Software Engineering, Porto Alegre, 2012, pp. 174-178.
P71	Pasivaara M.et al, "Supporting a large-scale lean and agile transformation by defining common values," In Proocedings of the Agile methods. large-scale development, refactoring, testing, and estimation, Springer International Publishing, 2014, pp. 73-82.
P72	Petersen K., C. Wohlin, "A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case, "Journal of Systems and Software, Vol 82, Nro, 9, 2009, pp.1479-1490.
P73	Power K. "A model for understanding when scaling agile is appropriate in large organizations," In Proceedings of the Agile methods. large-scale development, refactoring, testing, and estimation, Springer International Publishing, 2014. pp. 83-92.
P74	Read A. and R. O. Briggs, "The Many Lives of an Agile Story: Design Processes, Design Products, and Understandings in a Large-Scale Agile Development Project," In Proceedings of the Hawaii International Conference on System Science, Maui, HI, 2012, pp. 5319-5328.
P75	Read K., and F. Maurer, "Issues in scaling agile using an architecture-centric approach: A tool-based solution" In Proceedings of the Extreme Programming and Agile Methods-XP/Agile Universe, Springer Berlin Heidelberg, 2003, pp. 142-150.
P76	Richard L., M.Evans. Scaling up pushing scrum out of its comfort zone. In Proocedings of the Agile Conference, 2008, pp. 395-400.
P77	Rolland K. H., "'Desperately' seeking research on agile requirements in the context of large-scale agile projects," In Proceedings of the Scientific Workshop of the XP, New York, NY, USA, 2015, Article 5 , 6pp.
P78	Saddington, P. "Scaling Agile Product Ownership through Team Alignment and Optimization: A Story of Epic Proportions," In Proceedings of the Agile Conference, Dallas, TX, 2012, pp. 123-130.
P79	Scheerer A., T. Hildenbrand and T. Kude, "Coordination in Large-Scale Agile Software Development: A Multiteam Systems Perspective," In Proceedings of the Hawaii International Conference on System Sciences, Waikoloa, HI, 2014, pp. 4780-4788.
P80	Schnitter J. And O. Mackert, "Large-scale agile software development at SAP AG," In Proceedings of the Evaluation of Novel Approaches to Software Engineering, Springer Berlin Heidelberg, 2010, pp. 209-220.
P81	Shatil A., O. Hazzan and Y. Dubinsky, "Agility in a Large-Scale System Engineering Project: A Case-Study of an Advanced Communication System Project," In Proceedings of the International Conference on Software Science, Technology and Engineering, Herzlia, Israel, 2010, pp. 47-54.
P82	Smits H. "The Impact of Scaling on Planning Activities in an Agile Software Development Center," In Proceedings of the Hawaii International Conference on System Sciences, Waikoloa, HI, 2007, pp. 274-274.

P83	Smits H. and K. Rilliet, "Agile Experience Report: Transition and Complexity at Cisco Voice Technology Group," In Proceedings of the Agile Conference, 2011, Salt Lake City, UT, 2011, pp. 274-278.
P84	Snapp M. B. and D. Dagefoerde, "The Accidental Agilists: One Team's Journey from Waterfall to Agile," In Proceedings of the Agile Conference, Toronto, Ont., 2008, pp. 171-175.
P85	Spayd, Michael K. Evolving agile in the enterprise: implementing XP on a grand scale. In Proceedings of the Agile Conference, Salt Lake City, UT, 2003, pp. 60-70.
P86	Stojanov I. , O. Turetken and J. J. M. Trienekens, "A Maturity Model for Scaling Agile Development," In Proceedings of the Euromicro Conference on Software Engineering and Advanced Applications, Funchal, 2015, pp. 446-453.
P87	Sweetman R., O. O'dwyer, K. Conboy, "Control in Software Project Portfolios: A Complex Adaptive Systems Approach," In Proceedings of the Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation, Springer International Publishing, 2014. pp. 93-104.
P88	Tabib, R. "Need 4 Speed: Leverage New Metrics to Boost Your Velocity without Compromising on Quality," In Proceedings of the Agile Conference, Nashville, TN, 2013, pp. 117-120.
P89	Talby D., A. Keren, O. Hazzan and Y. Dubinsky, "Agile software testing in a large-scale project," in IEEE Software, vol. 23, no. 4, pp. 30-37, July-Aug, 2006.
P90	Talby D., O. Hazzan, Y. Dubinsky and A. Keren, "Reflections on reflection in agile software development," In Proceedings of the Agile Conference, Minneapolis, MN, 2006, pp.11-12.
P91	Therrien I. and E. LeBel, "From Anarchy to Sustainable Development: Scrum in Less than Ideal Conditions," In Proceedings of the Agile Conference, Chicago, IL, 2009, pp. 289-294.
P92	Waardenburg G., H. van Vliet, "When agile meets the enterprise", Information and Software Technology, Volume 55, Issue 12, 2013, pp. 2154-2171.
P93	Williams W. and M. Stout, "Colossal, Scattered, and Chaotic (Planning with a Large Distributed Team)," In Proocedings of the Agile Conference, Toronto, ON, 2008, pp. 356-361.
P94	Woodward E. V. et al., "Agile methods for software practice transformation". IBM Journal of Research and Development, v. 54, nro. 2, 2010, pp 161-172.

APPENDIX B – INTERVIEW SCRIPT OF THE FIELD STUDY

Interview Script

Goal of this study: The study is focused on understanding how a large-scale agile transformation is taking place in a company.

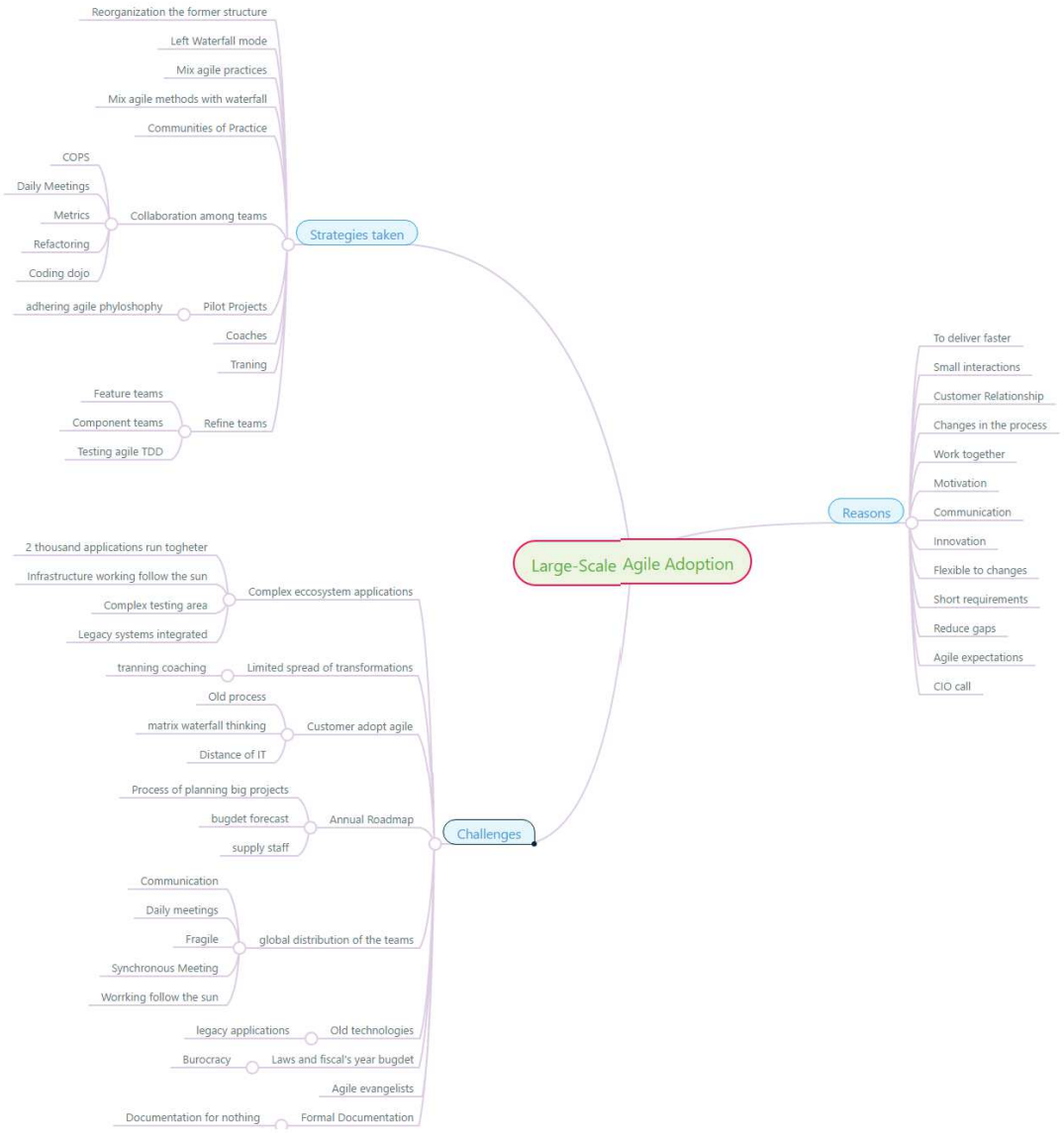
[Background]

1. Full name
2. Function title and brief description of the responsibilities:

[Questions]

ID	QUESTIONS
1	How did the company become agile? When did this happen (on which date)
2	How the new team composition has been defined?
3	Which teams are involved in this transformation?
4	Which areas of the organization are being restructured to agile transformation?
5	How the teams that work with you have reacted to the agile software development?
6	What are the agile practices being adopted by your team?
7	What are the challenges you see ahead during the remaining of the transformation process?

APPENDIX C – FIELD STUDY NETWORK OF RESULTS



APPENDIX D – QUESTIONNAIRE SCRIPT USED IN THE FOCUS GROUP SESSIONS (IN PORTUGUESE)

Parte 1.1– Informação do Participante

1. Nome:
   _____
2. E-mail:
   _____
3. Profissão/Posição:
   _____
4. Há quanto tempo você trabalha com métodos ágeis? (anos e meses)
   _____
5. Há quanto tempo você trabalha com métodos ágeis neste time? (anos e meses)
   _____

Parte 1.2 – Informação da Empresa

6. Área do negócio:
   _____
7. Localização (Cidade e País):
   _____
8. Nome:
   _____

Parte 1.3 – Informação do Projeto

9. Domínio da aplicação:
   _____
10. Tipo do Projeto: ( ) Distribuído ( ) Co-locado
    Número de sítios (se aplicável):
    _____
11. Quantos times trabalham no mesmo projeto que você?
    _____
12. Qual é o tamanho da sua equipe?
    _____
13. Qual(is) é(são) o(s) método(s) ágeis adotados pelo seu time?
    ( ) Scrum ( ) Extreme Programming ( ) Kanban ( ) Lean
    Outros:_____
14. Qual(is) é(são) o(s) Framework(s) Escaláveis adotados pelo seu time?
    ( ) SAFe ( ) Less ( ) Disciplined Agile. ( ) Não se aplica
    Outros:_____

Parte 2 – Questionário

Este trabalho tem por objetivo identificar a agilidade em equipes. As próximas páginas apresentam um conjunto de 7 níveis onde cada um deles representa uma área do domínio ágil. Para cada nível descrito a seguir, marque a(s) alternativa(s) que melhor se identificam com o perfil de sua equipe.

Nível 1: Aprendizado das Práticas

☐ **A. Tentando ser ágil:** A equipe está tentando utilizar algum método ágil. Na maioria das vezes, nem todas as práticas estão implementadas e os benefícios ainda não percebidos.

☐ **B. Aprendendo a ser ágil:** A equipe está aprendendo a ser ágil. Os métodos são usados passo-a-passo, como descrito nos livros e tutoriais. Normalmente a equipe está em treinamento ou *coaching*. Todas as práticas são utilizadas e é importante que a equipe aprenda o valor de cada uma delas para poder adaptá-las ou até abandoná-las mais tarde.

☐ **C. Atuando nos processos de trabalho:** A equipe entende as práticas ágeis e o valor de cada uma. Assim, a equipe decide como adaptar tais práticas ao seu contexto. Os membros da equipe se sentem confiantes com processos de trabalho flexíveis, sem abandonar a agilidade.

☐ **D. Compreendendo a situação:** A equipe tem informação sobre seus processos de trabalho. A informação é simples e normalmente baseada em dados visíveis a todos ou métricas simples. A equipe usa essa informação para tomar decisões e implementar melhorias nos processos. Essas melhorias são, geralmente, pequenas e incrementais.

Nível 2: Conduta da Equipe

☐ **A. Equipe responsiva:** A equipe precisa de uma liderança próxima para que as atividades sejam realizadas. Os membros da equipe não têm (ou não lhes foi dada) autonomia para tomar decisões. A equipe não tem confiança para eleger suas prioridades de trabalho e isso gera, normalmente, trabalhos extras por demandas não planejadas.

☐ **B. Equipe confiante:** A equipe precisa do Scrum Master por perto, mas se sente confiante para tomar pequenas decisões sobre o projeto. Mudanças nos processos de trabalho sugeridas pela equipe precisam de aprovação da gestão.

☐ **C. Equipe assertiva:** A equipe se sente responsável pelo projeto. Seus membros possuem autonomia para mudar os processos de trabalho. Juntos, eles tomam decisões e informam seus gestores.

☐ **D. Equipe Brilhante:** A equipe está focada em excelência técnica. Eles definem ou criam seus próprios processos de trabalho, com o suporte da gestão. Eles precisam de um ambiente que dê suporte ao aprendizado contínuo, para com isso, manterem-se motivados. É possível sentir nos membros dessa equipe, o foco em aprender sempre, o tempo todo.

Nível 3: Ritmo de entregas

☐ **A. Terminando o código:** As entregas (ex. Sprints, Entregas Frequentes, Releases) são planejadas de forma que, ao final, algum código seja finalizado e integrado ao repositório. Na maioria das vezes, a funcionalidade não foi testada ainda. A equipe está aprendendo como entregar valor para o cliente.

☐ **B. Gerando entregáveis:** Esta equipe identifica o valor a ser entregue para o cliente, mas não consegue entregar valor no final do tempo estipulado (ex. não consegue entregar valor ao final de uma Sprint). O ambiente tecnológico não dá o suporte necessário para integração de código e build ágil. O teste geralmente é manual.

☐ **C. Entregando quase no prazo:** As entregas são planejadas e realizadas, mas normalmente atrasadas. Às vezes, demandas extras aparecem durante a execução da entrega e a equipe precisa atender. O ambiente tecnológico dá suporte razoável para integração e build automatizado.

☐ **D. Entregando no prazo:** As entregas são planejadas e realizadas no prazo. Às vezes, até antes do prazo. O tamanho das estórias de usuário é, geralmente, pequeno. O ambiente tecnológico dá suporte completo para integração e build automatizado.

Nível 4: Descoberta das Features

☐ **A. Coletando requisitos:** Os requisitos são definidos no início do projeto e raramente são refinados antes das entregas (Sprints, Entregas Frequentes ou Releases), normalmente com texto ou diagramas. A equipe não se sente confortável com mudanças ou requisitos emergentes durante o projeto.

☐ **B. Descobrindo requisitos:** Os requisitos são vagamente definidos no início do projeto e detalhados assim que as entregas (Sprints, Entregas Frequentes ou Releases) são iniciadas. Mudanças e requisitos emergentes são bem-vindos. Spykes (ou provas de conceitos) podem ser usadas para elicitação de requisitos.

☐ **C. Refinando requisitos:** Existem iniciativas organizadas e continuas para garantir a qualidade dos requisitos, a fim de que eles estejam de acordo com as necessidades dos clientes.

Nível 5: Software

☐ **A. Conhecimento das falhas:** Existem falhas (bugs) nas entregas que precisam ser tratadas o mais rapidamente possível. As falhas acontecem devido a problemas no processo de desenvolvimento.

☐ **B. Código fonte de alto-nível:** A codificação é considerada uma atividade importante e existem iniciativas para garantir a clareza e a robustez do código, utilizando as melhores práticas disponíveis.

☐ **C. Software entregue de alto-nível:** Existe uma preocupação com a qualidade da *feature* que está sendo entregue. Por esta razão, teste é prioridade – automatizado ou não. Também há uma preocupação com a integridade do código no repositório.

☐ **D. Codificação eficiente:** Codificação, integração e testes são realizados em uma infraestrutura tecnológica que permite agilidade. A equipe está preocupada em remover esperas/atrasos desnecessários nos processos de trabalho. Devops é uma infraestrutura comum para permitir codificação eficiente.

Nível 6: Relacionamento com o cliente

☐ **A. Equipe conhecendo o cliente:** A equipe está aprendendo como funciona o negócio do cliente e qual é a dinâmica das demandas. Esta é a razão pela qual a contribuição para o negócio do cliente ainda é principiante.

☐ **B. Cliente conhecendo a equipe:** É o processo de aprendizado do cliente sobre a equipe. O cliente está conhecendo os processos de trabalho da equipe, mas não está completamente acostumado. Às vezes, parece que há falta de confiança na equipe.

☐ **C. Cliente confiante:** O cliente está confiante com a forma de trabalho da equipe. Ele acredita que a equipe está preocupada em entregar o que ele precisa. O cliente tem uma postura mais flexível com a entregas e se sente mais confortável com a repriorização de requisitos. Existe transparência entre a equipe e o cliente.

☐ **D. Cliente parceiro:** O cliente reconhece a parceria com a equipe, enquanto ele sente que a equipe está comprometida com o seu negócio. A equipe sente que o cliente é parte dela.

Nível 7: Suporte Organizacional

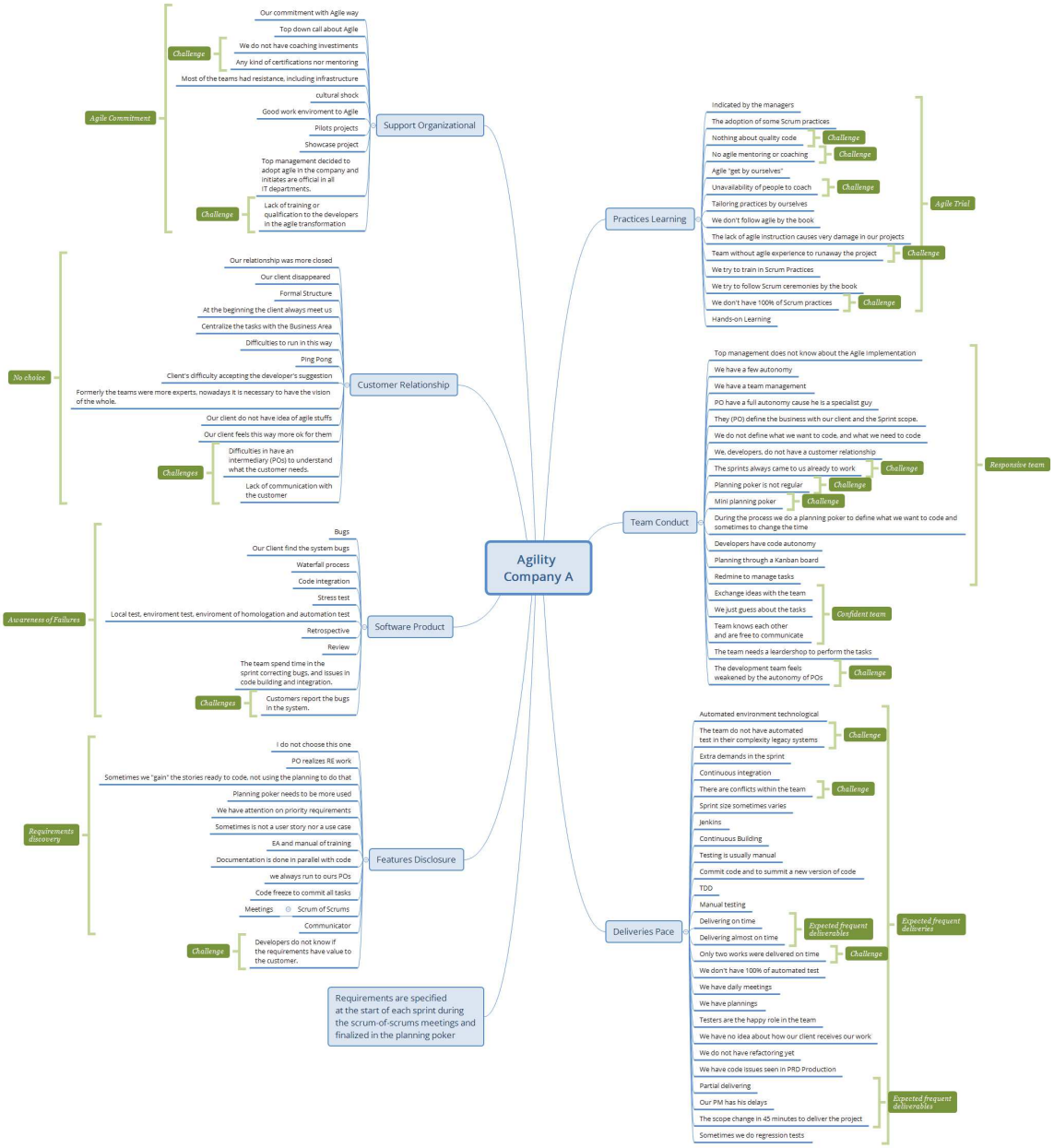
☐ **A. Movimentação ágil:** Algumas equipes isoladas estão iniciando projetos utilizando métodos ágeis. A organização sabe disso, mas não demonstra interesse nos novos processos ou emseus resultados.

☐ **B. Comprometimento ágil:** A alta gestão apoia a implementação de métodos ágeis na empresa e as iniciativas são oficiais. Existem investimentos em treinamentos, coaching, comunicação e infraestrutura tecnológica para a transformação ágil.

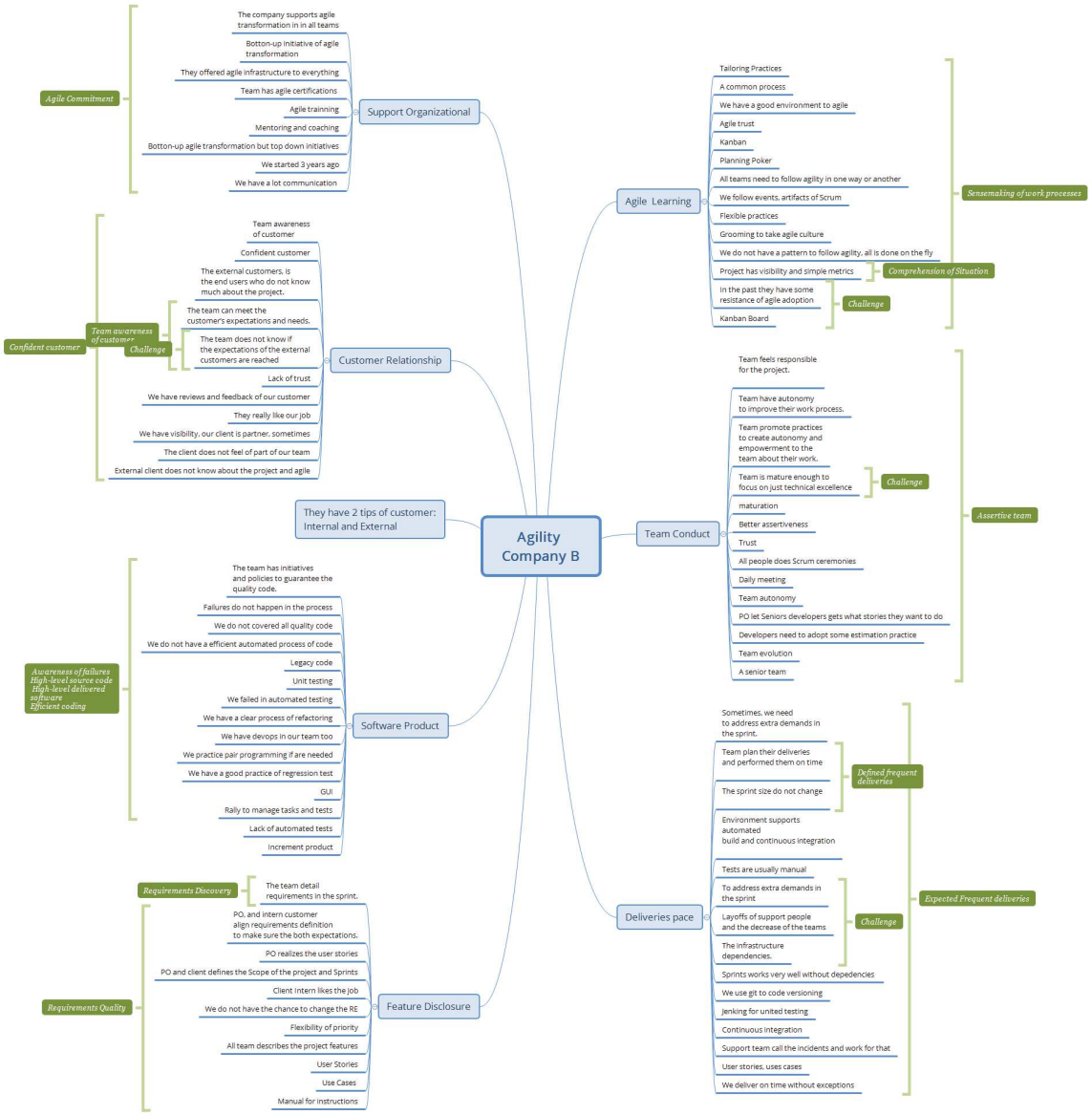
☐ **C. Prioridade ágil:** Existe suporte completo da alta gestão para a transformação ágil. Departamentos, papeis e equipes mudam para dar suporte à agilidade. Quando é uma iniciativa top-down, pode surgir resistência de algumas equipes.

☐ **D. Negócio ágil:** Empresas de desenvolvimento de software criadas com base em métodos ágeis normalmente são “negócios ágeis”. As estratégias de gestão da organização são focadas em pessoas e processos enxutos. Isso não somente no desenvolvimento de software, mas para toda a empresa.

APPENDIX E – FOCUS GROUP NETWORK OF RESULTS - COMPANY A



APPENDIX F – FOCUS GROUP NETWORK OF RESULTS - COMPANY B



APPENDIX G – PUBLICATIONS

Roman, G., Marczak, S., Dutra, A. Prikladnicki, R. **"On the Agile Transformation in a Large-Complex Globally Distributed Company: Why Boarding this Journey, Steps Taken, and Main Foreseen Concerns."** In: Brazilian Workshop of Agile Methods, Porto de Galinhas, PE, 2015. pp. 1–12.

Roman, G., Marczak, S., Dutra, A.**"On the Transformation to Agile in a Large-Complex Globally Distributed Company: A Research Plan to Define Guidelines."** In: Latin-American School on Software Engineering, Porto Alegre, RS, UFRGS, 2015. pp. 130–134.

Roman, G., Marczak, S., Dutra, A. Prikladnicki, R. **"On the Transformation to Agile in a Large-Complex Globally Distributed Company: Identifying How Teams Evolve in Such Setting."** In: Amazon Advanced School on Software Quality, Manaus, AM, 2015. pp. 1–4.

Roman, G., Marczak, S., Kroll J. **"Have you heard about Large-Scale Agile Development? Findings from a mapping literature review"** In: Information and Software Technology. (2017) – (To be submitted).

Roman, G., Marczak, S., Kroll J. **"Characterizing the presence of agility in LSAD teams."** In: International Symposium on Empirical Software Engineering and Measurement. (2017) – (To be submitted).