Challenges and Success Factors for Large-Scale Agile Transformations: A Systematic Literature Review

Kim Dikert[a], Maria Paasivaara[a,b], Casper Lassenius[b,a]

[a]*Aalto University, School of Science, Department of Computer Science and Engineering*
[b]*Massachusetts Institute of Technology, Sloan School of Management*

Abstract

Agile methods have become an appealing alternative for large companies striving to improve their performance, but the methods were originally designed for small and individual teams. This creates unique challenges when introducing agile to large development organizations, where development teams cannot be fully autonomous and there is a need to interface with other organizational units. In this paper we present a systematic literature review on how large organizations have adopted agile and lean software development, focusing on experienced challenges and reported success factors. We conducted a systematic literature review of industrial large-scale agile transformations. Our keyword search found 1875 papers. We we included 52 publications describing 42 industrial cases presenting the process of taking agile methods into use in large organizations. We identified 35 challenges grouped into eight categories, and 29 success factors, grouped into eleven categories. The most salient success factor categories were management support, choosing and customizing the agile model, training and coaching, and mindset and alignment.

Keywords:
agile software development, organizational transformation, large-scale agile, adopting agile software development, challenges, success factors, systematic literature review

1. Introduction

Increasing pressure to reduce cycle time, improve quality, and swiftly react to changes in customer needs are driving companies to adopt agile software development [47]. Agile methods can improve efficiency and quality [31], and enable shorter lead times and a stronger focus on customer needs [42].

Email addresses: kim-karol.dikert@aalto.fi (Kim Dikert), maria.paasivaara@aalto.fi (Maria Paasivaara), casperl@mit.edu (Casper Lassenius)

Agile methods were originally designed for use in small, single-team projects [5]. However, their shown and potential benefits have made them attractive for large companies and projects as well, despite the fact that they are more difficult to implement in large organizations [14]. Compared to small organizations ideal for agile development, large organizations are characterized by the need for additional coordination. A particular problem in applying agile to large software organizations is how to handle inter-team coordination. Large organizations also tend to have additional organizational units that development must interface with, such as human resources, marketing and sales, and product management. Still further, large organizations may cause users and other stakeholders to become distant from development teams. Despite these known problems, there is an industry trend towards adopting agile methodologies in-the-large [47, 41, 39, 12].

While the research literature contains several experience reports and some case studies on the adoption of agile methodologies in large organizations, a systematic overview and synthesis of this growing body of research is still missing. Freudenberg and Sharp [19] asked the industrial practitioners at the XP2010 conference to create a backlog of topics they think should be studied. The practitioners voted "Agile and large projects" as the top burning research question. Moreover, among the top ten items three focused on distributed agile development, which is common especially in large organizations as they are often geographically distributed. In two recent workshops on large-scale agile development organized in XP2013 and XP2014 conferences, adoption of agile methods was one of the highlighted themes needing more research [10, 12].

While research on agile software development is accumulating and maturing, and has provided a basis for conducting systematic literature reviews [13, 25, 46, 26], the area of large-scale agile development has not yet been studied through secondary studies. In this paper we start filling in this gap by presenting a systematic literature review of agile transformations in large organizations.

The rest of the paper is structured as follows: in the next section we present background for this work. Section 3 discusses the research method used in this study, and Section 4 presents our findings. In Section 5 we discuss our results and the final section concludes the paper.

2. Background

2.1. Agile software development

Agile software development is a collection of methods developed as an alternative to so called traditional software development methods. From the point of view of agile methods traditional methods strive to minimize change during the development process and attempt to fully define requirements before development. However, adaptability during the development life cycle and the ability to respond to conditions unforeseen at start are nowadays seen as critical factors for the success of a software development project. Embracing change during development, people centricity, and pursuing high quality from the start are fundamental themes in agile development. [23, 7]

Agile methods have been both criticized and advocated, and research has shown that accommodating change may be a factor in both success and failure [4]. It has been shown that agile methods have improved satisfaction of both customers and developers, but on the other hand there is evidence that agile methods may not be a good fit for large undertakings [14]. A proposed solution is for each organization to find its own balance of agile and plan driven methods [4] .

Two of the most popular agile methods are Extreme Programming (XP) and Scrum [22]. Scrum is a method focusing on the project management viewpoint of agile development [44], prescribing timeboxing, continuous tracking of project progress, and customer centricity. The XP development method is a collection of practices for enabling efficient incremental development [1]. In practice, many agile development implementations combine the two in some way [17].

2.2. Adopting agile methods in large organizations

Introducing agile methods in large organizations is more difficult than in small organizations [13]. The difficulty is partly related to size bringing higher organizational inertia which slows down organizational change [31]. Agile development is not founded on the use of individual tools or practices, but rather on a holistic way of thinking. Adopting agile often requires change of the entire organizational culture [36].

One significant difference between small and large scale adoptions is that large organizations have more dependencies between projects and teams. This increases the need for formal documentation and thus reduces agility [30]. In addition to inter-team coordination, development teams must interact with other organizational units, which are often non-agile in nature. For instance, human resources unit may demand individuals to have strictly specified roles in projects [5], or a change control board may inhibit the use of continuous integration or refactoring [30]. All units affected by the agile transformation need to be informed and consulted, and the agile process must be adjusted according to their needs [30, 8, 5].

Agile methods also affect management and business related functions. A key challenge is that management must move away from life-cycle models and towards iterative and feature centric models [38], which requires a change of mindset. The focus must be shifted from long-term planning to shorter term project planning [36], as agile methods emphasize that planning is only meaningful for the near future [8]. However, the lack of planning can be a concern as business and customer relationships often build on long term roadmapping. Enabling operation with shorter term planning requires educating stakeholders and reviewing contracting practices [5].

2.3. Definition of large scale agile

A brief literature search [11] identified previous interpretations of what large-scale agile is. Large-scale had been regarded in terms of size in persons or teams, project budget, code base size, and project duration. The examples of cases that

were called "large-scale" included 40 people and 7 teams [40], project cost of over £10 million and a team size of over 50 people [2], a code base size of over 5 million lines of code [42], and a project time of 2 years with a project scope of 60-80 features [3]. Based on their findings Dingsøyr et al. [11] ended up measuring large-scale by the number of collaborating and coordinating teams: they categorized as large-scale 2-9 collaborating teams and as very large-scale over ten collaborating teams.

We identified a number of additional studies discussing large-scale agile software development and their interpretations of large-scale. All of these referred to the number of people involved. In early work on agile, Fowler considers the Crystal methodology to be suitable for up to 50 people [18]. The same number has been reported as seen by practitioners and researchers as the size of the largest organization suitable for agile [48]. Other studies have referred to agile projects including up to 50 people as small [29], and considered a development project large if it had a staff between 50 and 100 people, including all project personnel [15]. The largest numbers were 300 people across 3 sites [37]. Participants of the XP2014 large-scale agile workshop gave very varying definitions for large-scale agile development [12], showing that what is seen as large-scale depends very much on the context and the person defining it.

Based on these findings, we defined large-scale to denote *software development organizations with 50 or more people or at least six teams.* All people do not need to be developers, but must belong to the same software development organization developing a common product or project, and thus have a need to collaborate. For instance, Scrum masters and software architects are counted when assessing the organizational size. As some studies present the number of teams rather than the number of people, we correspondingly defined large-scale to denote development efforts involving at least six teams. Having six teams with an average size of six to seven people, plus a number of supporting staff, can reasonably be considered to form an organization of 50 people.

3. Research method

3.1. Research questions

In this paper, we report a systematic literature review [28] aimed at answering the following research questions:

- RQ1: What challenges have been reported for large-scale agile transformations?
- RQ2: What success factors have been reported for large-scale agile transformations?

3.2. Research process

The research process consisted of four main stages depicted in Figure 1. The selection of primary studies was done in two stages, first using keyword-based database searches to identify potentially relevant sources, and then manually

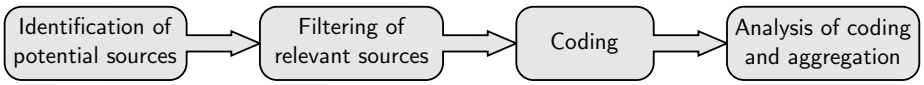

Figure 1: Outline of the research process

Table 1: Inclusion criteria

Facet	**Relevant topics**	**Examples of non-relevant topics**
Agile software development	The organization develops software; the introduced method is agile	Agile manufacturing; Scrum in management boards
Organizational transformation	Presents insights about the transformation process	Comparison of before and after; how agile is being used in large scale
Large scale	Agile practiced by a development organization of at least 50 people or 6 teams	Scaling up from small; a single agile team in a large setting
Empirical	Case studies, experience reports	Textbooks, student experiments, theory papers

filtering the search result. The manual filtering process was executed independently by the two first authors. Data extraction was done by qualitative coding of the selected primary studies by the first author. Finally, the results were elicited by aggregating and analyzing the coding of the primary documents. The entire process was audited and mentored by the third author.

3.2.1. Inclusion criteria

Based on the research questions and focus of the research, we defined four facets to guide our inclusion/exclusion decisions: *agile software development, organizational transformation, large-scale*, and *empirical*. Table 1 lists the facets and gives examples on matching topics and non-relevant topics. For a study to be included it needed to be relevant on all facets.

Examples of topics excluded by the facet of *agile software development* are agile manufacturing and the application of Scrum outside software engineering, e.g., in management boards. In addition, we required that the organizational transformation was aimed at introducing agile methods, which excluded other development methods [43], and the use of agile methods in other contexts than software development [24].

The facet of *organizational transformation* was interpreted so that the primary study had to present insights on the process of an agile transformation. Examples of excluded topics include comparing the original and agile development methods [42], discussing the use of agile in a large organization but not describing how the new methods were introduced [35], and merely presenting agile tools in large-scale use [27].

Transformation and "scaling up" of agile practices in use are very closely related concepts, and in some cases they are one and the same. For instance, if transformation begins with a pilot and then is gradually rolled out in the organization, the process could well be seen as a "scaling up" journey. These kinds of journeys are included in our study. However, we excluded cases where an initially small agile organization grew organically [33], and discussions focusing on processes or tools without describing organizational change [32].

The facet of *large scale* was interpreted as discussed in Section 2.3: a single software development organization with at least 50 persons or six teams. In some cases, the source presented very vague indicators on size. For instance, one case [6] was included, as there were indications of large-scale considerations, although the size remained unclear. If there were no indications of size, the paper was excluded, e.g. [34].

To complicate matters, some papers talked about "the team" in singular when referring to the organization [24], making it nontrivial to judge whether an organization was large based on the choice of words of the author.

Further examples on exclusion by the *large scale* facet were cases with large organizations but only a single team adopting agile [21]. We considered cases of single teams (although in large organizations) unrelated to this research as our focus is on transformation of the entire (development) organization.

Also piloting cases that reported only single teams using agile, even though the whole organization was large, were excluded, e.g. [45]. Finally, cases where the organization was growing to large scale, but did not meet the size criteria at the start of the transformation, were excluded.

According to the facet of *empirical* we excluded studies that did not discuss a distinguishable real world case. We excluded textbooks, studies merely presenting theories, as well as studies that did not include any case organization. Moreover, we excluded studies on the benefits or limitations of agile in general. Lastly, we excluded student experiments as it is implausible to simulate the dynamics of a large organization in an experiment.

3.2.2. Preliminary searches

Before proceeding with identifying the primary studies a number of preliminary searches were performed. The purpose of the preliminary searches was to develop and evaluate different search strings. In addition, we used the searches to identify a set of relevant papers that should be matched by the actual search. We started by examining top ranked hits by trivial keywords that the more complex final search string might miss. Initial searches were made using keywords that were as general as possible, including "agile transformation" and "large scale agile". Based upon these preliminary searches, we selected 117 papers that seemed relevant based upon the title. We used this set as a "sanity check" when developing the actual database search.

3.2.3. Identification of primary studies

The gathering of potential primary studies was based on a search in the online databases listed in Table 2. We constructed a search string based on the

Table 2: Databases included in search, and number of matched articles

Database	URL	# of matches
IEEExplore	http://ieeexplore.ieee.org	745
ACM	http://dl.acm.org	168
Scopus	http://www.scopus.com/home.url	1596
Web of Knowledge	http://apps.webofknowledge.com	786

Table 3: Facets and related search terms used

Facet	Keywords
Agile methods (before & after)	`agile, scrum, "extreme programming", waterfall, "plan-driven", RUP`
Organizational transformation	`transform*, transiti*, migrat*, journey, adopt*, deploy, introduc*, "roll-out", rollout`
Only software related articles	`(software OR (conference="agile, xp, icgse, icse")) AND NOT (title+abs="manufacturing" OR conference="agile manufacturing")`

previousy discussed facets. However, our preliminary searches showed that picking keywords with good precision was difficult. In particular, we did not succeed in representing the facets *large-scale* and *empirical* with precise words. Therefore, we included only the facets *agile software development* and *organizational transformation* in the keyword search, with the consequence of increasing the manual filtering effort in the subsequent steps. The used keywords are shown in Table 3.

3.2.4. Study selection

The set of potential primary studies identified by the database search was refined in two steps, first by filtering based on abstracts and finally based on the full text. The study selection process is outlined in Figure 2.

The database keyword search matched 1875 unique papers. The abstracts of these papers were categorized independently by the two first authors into three categories: include, exclude, and uncertain. There were 1578 exclusions and 62 inclusions that both researchers agreed on. The inclusion decisions for the 235 abstracts with disagreement or uncertainty were resolved through discussion. At this stage papers were excluded only if both researchers deemed it clearly irrelevant, including any uncertain cases for full text filtering. As a result 170 papers were selected for full text filtering.

Full text filtering was performed by evaluating the text of each article against the four facets of the inclusion criteria. Filtering was done in two steps. In the first step, the first author extracted data relevant to the four facets. Based on the extracted data, 76 papers could be immediately deemed as included or excluded. The remaining 94 papers were evaluated against the inclusion criteria by the two first authors, and a decision was made after discussing each paper separately. In difficult cases, the third author was consulted to reach a decision. As a result of the full text filtering, 47 papers were selected for inclusion.

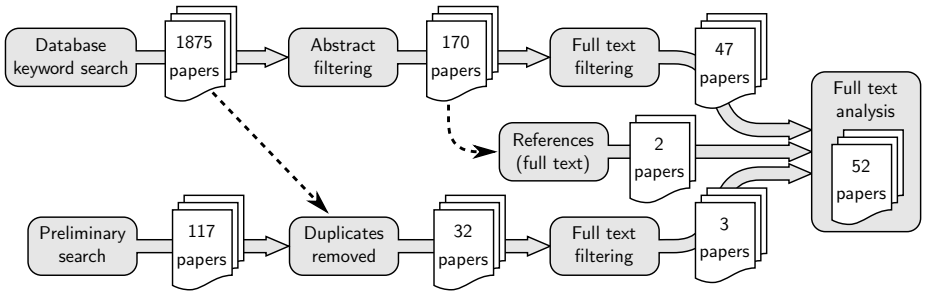

Figure 2: The study selection process

We evaluated the result of the database keyword search against the benchmark created in the preliminary search step. We concluded that 75 of the 117 preliminarily selected papers were matched by the database keyword search. The missed 32 preliminary papers were examined, resulting in including three additional papers as primary sources.

In parallel with the full text filtering step the references of all 170 papers selected for full-text filtering were also examined for relevance. Most papers used references very scarcely, typically referencing well known descriptions of agile methods. This step led us to include two additional papers in our full text analysis.

As a result, we selected a total of 52 papers for inclusion in the analysis stage.

3.2.5. Handling of duplicate reports on a single case

In several cases, more than one primary study discussed the transformation of the same organization. Duplicate descriptions typically focused on different aspects. For example, one paper would highlight the viewpoint of the developers [20], and another would consider the transformation from the user experience designers' point of view [16].

Even if the transformation of a single organization was described in many studies, all sources that passed the inclusion criteria were included. Studies presenting the same organization were treated as one unit so that we could gain as much insight on each organization as possible. Conversely, there were also a few papers that presented multiple case studies, and in those cases we treated each studied organization individually.

Instead of using the most complete paper as suggested in the guidelines for systematic literature reviews [28], we combined the results presented in each paper and considered the case as a single unit in our analysis. Keeping in mind the potential bias caused by duplicate publications, we think that including all papers enabled us to get a more in-depth understanding of the individual cases.

As a result, we identified 42 unique organizations in the primary studies. We use the term *study* to refer to the primary publications, and the term *case*

to refer to an individual case organization that may be described in several different studies.

3.2.6. Study quality assessment

The primary studies for this literature review are almost exclusively industry experience reports. We identified only six case studies with a clearly defined research method, and observations on transformation were presented only as a minor part in these studies. Based on this finding we conclude that case studies presenting insights into large-scale agile organizational transformations are very scarce. We deemed that the results would be distorted heavily and many valuable studies would be left out if a strict quality assessment would be part of the inclusion criteria. As a result we decided to include all experience reports, regardless of the potential problems of author and publication bias.

3.2.7. Coding of primary studies

We coded the primary studies using an integrated deductive and inductive approach, coding both a contextual and a findings part [9]. We established a list of codes for contextual information, which included the codes agile method used, business area, organization size, time of transformation, research process used, geographical location, definition of large-scale used in the paper, and multisite case. Codes related to the research questions were created by an inductive process to avoid having our previous assumptions affect the choice of codes.

To guide inductive coding, we established seven different code families a priori, as shown in Table 4. A description defining the scope and containing examples of codes were defined for each family. The example codes would not necessarily be used as actual codes, and would not represent the entire or final scope of the families. Table 4 also shows the total number of codes and quotations created in the coding process. The total number of quotations is less than the sum of quotations in the categories because a single quotation may have multiple codes.

3.2.8. Synthesis of primary studies based on coding

We synthesized our findings by creating an initial organization of codes into high level categories based on the code labels. Each code was classified into a single category.

After assigning each code to a unique category, the content of the categories was studied. Each category was studied by reading through each quotation of each code included. Typically the quotations were displayed and examined in their original context, considering the surrounding paragraphs of the quotation. Notes were made on each quotation presenting noteworthy observations. Based on the notes, an accurate description emerged for each code.

As an accurate description had been created for each code, we refined the categories. Some codes were re-categorized, and the definitions of a few categories were revisited, until a final ordering was reached. The results are presented according to the final categorization.

Table 4: Code families, codes and quotations.

Code family	**Description**	**Examples**	**Codes**	**Quotations**
Reason to change	Reasons to start the transformation	demand for faster delivery	30	123
Transformation process	Statements describing the transformation process	top-down, big bang, step-wise	16	580
Challenges	Statements presenting challenges in the transformation	change resistance	40	323
Success factors	Statements presenting success factors in the transformation	management support	44	260
Investing in change	Factors presenting how the organization is investing in the transformation	training, consultants, tools	5	137
Practices	Practices used or established during the transformation, but described as neutral wrt. to successes and challenges	coaching, piloting, continuous integration, communities of practice	11	170
Contextual	Contextual codes defined in Table **??**.	agile method, organization size, large-scale definition	8	215
		Total	154	1575

4. Results

4.1. Overview of the primary studies

4.1.1. Study types

The results include findings from 52 primary studies discussing how 42 different large software organizations introduced agile methods. Most studies were experience reports (45 studies), and in many cases it was evident that the author had been personally involved in the transformation. Only six of the included studies had a research method explicitly stated. In addition, one of the studies was an interview article. The publication forums of the primary studies were distributed so that 47 sources were conference proceedings, four sources were journal articles, and one source was a technical report.

All included studies and transformations were dated after year 2000. There were peaks in transformation studies in 2008 and 2011. The studies were typically published two years after the start of the transformation.

4.1.2. Case organizations

The size of the organizations varied from the minimum included size of 50 to 18,000 people. The median size was 300 people. In seven studies the size was presented in terms of teams, ranging from the minimum of six teams to 150 teams. The median was 10 teams. In eight studies there was no direct indication about the size but the issues discussed revealed that the organization was of a large size according to our definition.

Different business areas were represented quite evenly. Online services was the largest group, including companies providing software as a service solutions for businesses, online media players for consumers, online services for consumers, and communication software for businesses. The second largest group was telecommunications, including companies such as British Telecom, Cisco, Ericsson, and Nokia Siemens Networks. The third largest group was enterprise management solution providers with products for business process management, project portfolio management, and facility management.

4.1.3. Agile methods used

The most prevalent agile method used in the transformed organizations was Scrum, which was the sole agile method mentioned in 25 cases. The second most mentioned agile method was Extreme Programming (in 4 cases), which was often combined with Scrum (in 5 cases). Lean software development was mentioned in 6 studies, although in all cases in combination with Scrum. Other agile methods mentioned were Unified Process[1], ADM, and Rapid Application Development. In one case the agile method was not named.

It was quite common that organizations sought to combine agile methods. Especially Scrum, XP, and Lean software development were used together. In

[1]We acknowledge that it can be debated whether Unified Process is an agile method. However, since the author listed it as such, we list it here.

addition, many cases mentioned use of XP practices, such as test driven development and continuous integration, without explicitly stating XP as the process being used. Combining and customizing agile practices was also evident as many organizations viewed the agile method as continuously evolving. Organizations evolved the agile methods for instance through retrospectives and continuous improvement.

4.2. Transformation challenges

Any organizational transformation that involves numerous individuals will face challenges. In this section we answer our first research question, RQ1: *What challenges have been reported for large-scale agile transformations?*

We organized the found 35 challenges, each mentioned by several sources, into eight categories. The categories are summarized in Table 5.

4.2.1. Change resistance

General resistance to change. People are not willing to change unless there are good reasons that they understand, and the change is perceived easy enough. It is difficult to attain a buy-in for a change, and even organizations that have a flexible culture will face resistance [P10]. It should be expected that not everyone will be willing to change, even so that some employees will never adapt to the new model [P6]. During a transformation period, objections to change may become a major reason for loss in time and productivity [P29].

Numerous reasons for change resistance were reported. For instance, Fecarotta [P14] described the organization of Boeing as risk-averse and cautious. Any change was further hindered by a deeply rooted status quo and high employee retention [P14]. In some cases it was reported that people worried about new roles and responsibilities that the agile model might bring [P44]. For instance, testers were worried that they would need to take on cross-functional tasks, which would be outside their area of expertise [P18]. Yet another reason for change resistance was the move from individual offices to team areas [P22]. People felt that they were being monitored more because of the increased level of interaction within the team and between the various project stakeholders [P45].

In these cases, change resistance was mitigated, for instance, by education [P45] and one-on-one discussions [P22]. Resistance was also softened by first engaging only teams willing to try agile, so that insights could be gathered to ease the continued transformation [P45].

Skepticism towards new model. Skepticism and distrust in agile development in general were other common problems. Seffernick describes how management did on the one hand acknowledge the benefits of agility, but on the other hand opposed its introduction due to contract reasons, the current matrix organization, and other organizational practices [P46]. Skepticism was often created by misconceptions, including that agile does not work for complex products [P5, P36], agile needs to be implemented in a prescriptive by-the-book way [P9], that frequent meetings will cause overhead [P18], and that agile equals avoiding governance and working without a plan [P8].

Table 5: Challenges experienced in large-scale agile transformations

Challenge type	Primary sources	Challenge type	Primary sources
Change resistance		**Challenges specific to large-scale**	
General resistance to change	P14, P18, P22, P44, P45	Middle managers' role in agile unclear	P2, P11, P27, P30, P38, P49, P52
Skepticism towards new model	P5, P8, P9, P18, P36, P46	Old and new models side by side	P1, P8, P10, P26, P29
Top down mandate creates resistance	P2, P12, P37, P49	Management in waterfall mode	P3, P6, P35, P38, P45, P46
Management unwilling to change	P2, P3, P6, P49	Keeping the old bureaucracy	P20, P48
		Interpretation of agile differs between teams	P8, P9, P38, P43, P48
		Internal silos kept	P10, P11
Lack of investment		**Requirements engineering challenges**	
Lack of training	P11, P20, P45	High-level requirements management missing in agile	P24, P31, P39, P45, P51
Lack of coaching	P1, P6, P23, P45, P47, P49	Requirement refinement challenging	P1, P17, P33, P48
Too high workload	P37, P42, P49	Creating and estimating user stories hard	P5, P11, P27, P30, P33, P37
Old commitments kept	P11, P21	Gap between long and short term planning	P9, P10, P13, P26, P31, P38
Rearranging physical spaces	P6, P14, P29, P36		

Top down mandate creates resistance. The way the transformation is initiated affects how change resistance will show. In several cases, the change initiative came from management, and when presented in a bad way, became a mandate that people were not receptive for. Spayd [P49] summarizes this as: *"Organizations do not change merely because the boss says so, at least not in the way that is intended"*. In that organization the introduction of CMM had been carried out consistently by a mandate, but the collaborative nature of agile did not mix well with such a mindset. A top-down mandate may also dilute the understanding of the reasons for starting the transformation and the understanding of agile development overall [P2]. O'Connor describes how team members became skeptical towards agile when implementing the mandated changes did not bring any visible benefits [P37]. A problem relating to a top-down mandate is that if management does not define a clear goal for using agile, developers may feel that the agile methods may be replaced by something else at any time [P12].

Management unwilling to change. Change resistance amongst managers can also create problems. In cases where the change was emerging bottom-up, management became reluctant to change, making significant organizational change above the team level impossible. For instance, Atlas [P3] describes how executive support would have been required to extend the agile process to involve the product management office and separate quality assurance groups. In this case, when managers were not involved in the transformation, the agile model could not spread beyond the development teams [P3]. Lack of middle management support for change and a disinclination to change management culture were seen as some of the most serious problems in the transformation [P2, P49]. Middle management is in a position to undermine the entire transformation, and may do so if they do not participate in, and thus understand, the agile method. Agile brings changes to some management roles [P2], and lack of understanding of agile development will leave managers feeling left out [P6].

4.2.2. Lack of investment.

Lack of training. Training and coaching are direct investments in transformation and their lack is an evident problem. Not providing enough funding for these activities can create difficulties in the transformation [P11]. Hajjdiab [P20] describes how reluctance of management to invest in training caused teams to be ill prepared for the transformation effort, eventually resulting in ending the use of agile methods. A less dramatic outcome of too little training was lower motivation [P45].

Lack of coaching. Arranging proper coaching was a problem in many organizations. It is critical to coach teams in their real work environment as a proper change in mindset is difficult to achieve only by attending training sessions. [P9, P23]

In large organizations, where numerous teams needed to be coached the demand often exceeded the capacity of available coaches [P1, P6, P23, P49]. Lack of coaching was also attributed as a reason why the success of pilot teams could not be repeated when agile was adopted more widely [P9, P45]. Silva and

Doss [P47] described that when it was hard to fill the coaching positions, people less seasoned in agile were appointed as coaches, which created the risk that agile practices would not be taught correctly.

Too high workload. Even though the case organizations were in a state of transformation, the workload of the teams was not always adjusted to facilitate the change process. Some organizations started their agile journey in a state where people were over-committed, and later realized that overloaded people will not be able to change their behavior and learn new ways of working [P37, P42].

Old commitments kept. Sometimes all old commitments were kept despite the transformation. In one case, management forced people to remain committed to firm deadlines, even midst in the organizational transformation [P21]. The engagement in old commitments and tasks resulted in ignoring new agile practices, and eventually the agile method broke down [P21]. One case [P11] describes how time was allocated for tending to old commitments. However, even though they had prepared for extra work, people with specific expertise became overloaded. It was realized that over-commitment must be reduced by shifting excess work to a later time [P11].

Another case describes how senior management pressed teams to deliver what the customer had been promised, regardless of what the current velocity of development predicted. The pressure to deliver interfered with the transformation, but also forced a change in the old way of working [P49].

Rearranging physical spaces. In some cases, the old way of working had people seated physically distributed, in opposition to what agile methods suggest. Office spaces needed to be arranged so that the entire team could work in a single room, but it took time and effort, and it was not always possible [P6, P14, P36, P38]. Lewis and Neher describe how dedicated rooms for teams could not be arranged, and team events such as daily stand-ups required booking a conference room [P29]. In another case, the facilities organization shut down the teams' attempts to modify their working spaces [P49].

4.2.3. Agile difficult to implement

A common challenge was that implementing the agile model turned out to be difficult. An experienced software team may do well in training, but when the time comes to apply agile techniques in practice, the team may get lost [P21].

Misunderstanding agile concepts. There were many examples of problems caused by misconceptions of what agile software development is. On a general level, Bang [P4] describes how the values of the agile manifesto were not understood, and agile practices were carried out without understanding their purpose. In one case, management saw the purpose of agile simply being faster product delivery [P9].

Examples on the team level include how agile was seen as the freedom to hack without documentation [P26], that development could be done without design tasks [P37], and that agile meant that everyone should become a generalist [P42]. Schatz and Abdelshafi describe how teams presented unfinished work at reviews and ignored the principle of showing only completed increments, which

resulted in a backlog of bugs [P44]. In one case, team members had perceived the introduction of agile as a means to squeeze out more efficiency [P45]. In addition, the flatter organization was seen as fewer career opportunities, pushing team members to compete for visibility [P45].

A further misunderstanding of agile was due to viewing it only through the tools used, such as management software [P48]. Superficially focusing only on the tools themselves and not the reasons behind their use led to frustration [P42].

In some cases the misunderstanding of the agile model led to *"doing mini waterfalls"* [P6]. Managers used Scrum terminology, but had the teams commit to unreasonable workloads [P6]. In another case, a waterfall nature was evident as task estimates were given as hours left, and task breakdowns became disconnected from what was really being done [P27].

As a final example of misconceptions, some organizations considered agile being a solution to all problems [P9]. Success was declared prematurely [P49] but expectations created by successful early experiments were not fulfilled [P38].

Lack of guidance from literature. Several cases describe how an agile model was hard to learn from the literature [P21, P27], especially when it comes to implementing it on a large scale [P13]. As Beavers [P5] writes: *"There simply was not a manual or document where we could find easy answers on how to do things"*. Schnitter and Mackert [P45] report that theoretical considerations on how to scale up the agile methods were not good enough, and that product managers and architects struggled when several Scrum teams were working concurrently. Another case concluded that all practices suggested by XP did not fit enterprise scale development, and some practices required customization [P49].

Benefield [P6] describes how it was difficult to find a balance between prescribing a by-the-book implementation which may put people off, and giving too much freedom in the agile methods, which may weaken core practices.

The reality where the practices needed to be applied was described as messy in comparison to the ideal circumstances presented in the literature [P10]. Thus, it was difficult to choose a model to start with and gain buy-in for [P10]. Difficulties in choosing an initial model were also due to variances perceived in the agile approaches suggested in literature [P30].

Agile customized badly. Furthermore, the difficulty of and misunderstandings related to agile were evident in cases where the methods were customized badly. As a by-the-book implementation often was not feasible, organizations attempted to tailor the agile method to suit their specific needs. However, in some cases this simply meant skipping practices, which led to problems. In one case, certain individuals were allowed to ignore core elements of Scrum, which turned the teams' decision making into a variant of command and control [P9].

In one case [P29], there was a temptation to strip some agile practices and enhance others. Previous attempts had proven that one of the reasons for agile implementations to fail was deviations from the process, because of which the agile mindset did not take root [P29]. A bad customization may lead teams to adopt only practices that reflect their current needs, thus failing to achieve any

real change in process and mindset [P9, P46].

A further case of bad customization included replacing the agile vocabulary with familiar terminology from the old model [P44]. An important benefit of introducing new vocabulary is that it stimulates new ways of thinking [P44].

Reverting to old model. In several cases, challenges in the transformation resulted in people reverting to the old model of working. In some cases it was only a temporary struggle while learning agile practices, but in other cases the old model displaced agile. Development work has to go on during the transformation but there will be new things to learn for the team. Stress caused by the combination of schedule pressure and much change at once can pull people back to the old way of working [P11, P18, P38].

Even subtle trouble may put the transformation at risk, as people will always look for reasons to revert to familiar behavior [P44]. Teams without adequate training were struggling with applying agile practices correctly, and the challenge the new practices posed made people abandon them and return to the ways they know [P6, P49].

In some cases agile was seen as an overhead, and was therefore abandoned. For instance, as new practices were being introduced there was a decrease in performance, and when teams realized that the benefits are not immediate, they started reverting to the old model [P21].

Evans [P12] describes that new senior appointments made management less favorable towards agile, and a waterfall development model started to return.

Excessive enthusiasm. A phenomena that troubled some organizations was over-enthusiasm towards agile methods. Several cases contained reports of change leaders or teams becoming agile zealots. For instance, some members of the agile community could become too evangelic, making people take sides for or against agile development [P12]. Also, while starting out with an overly eager attitude, team members' interest faded, and they reverted to old ways of working when the new approach did not deliver immediate benefits [P20].

Attempting to implement agile too strictly may cause conflicts, and especially in a large organization, the change leaders need to maintain a collaborative attitude towards various groups in the surrounding organization [P6]. Introducing agile methods does not guarantee success, and therefore they should not be followed blindly [P4].

4.2.4. Coordination challenges in multi-team environments

Interfacing between teams difficult. One of the most prominent transformation challenges was the difficulty in coordinating the work of several agile teams.

Challenges arose when teams needed to work with other teams, and as parts of the larger surrounding organization [P13, P17, P24]. While introducing agile had created flexibility at the team level, the surrounding organization was not responsive enough [P26]. The roll-out of agile had not removed dependencies, and the dependencies made managing development difficult [P9, P10].

Many studies presented solutions for the inter-team coordination challenges, which also highlight the nature of the challenges themselves. The general model

of solving inter-team problems was introducing an additional layer of coordination, such as a Scrum-of-Scrums team [P43], a product team [P45], or an entire customized release framework [P13].

Autonomous team model challenging. Some organizations initially created models in which teams operate autonomously — well in line with agile principles. However, a number of issues arose from this independence. For instance, teams needed to strike a balance between the their own goals and the broader goals of the organization, but often chose to focus only on their own goals [P7]. Coordination was obstructed by independent teams that did not respect the larger context [P13]. Coordination was also difficult when teams were working independently for different customers, but the applications being built were interdependent [P29]. One case reported that even allowing teams to have different Sprint durations created delays in delivery [P51].

Global distribution challenges. Further coordination problems were encountered when scaling up agile over many geographically distributed sites. Distribution had negative effects, such as missing kick-off meetings, reduced feelings of proximity when telecommunication is necessary, and difficulty in arranging frequent meetings due to time zone differences [P45]. Agile project management was also problematic. In a waterfall model, separate parts of projects could be isolated to different sites, but the agile model does not allow such strict slicing of projects [P29]. In one case, it was simply admitted that a distributed organization will impose additional burden on communication and require additional care in the process, but distribution and agile would still be used together [P48].

Achieving technical consistency. Technical problems relating to inter-team coordination were reported as well. Integrating the products of teams was seen as a problem in some cases [P5, P50], and the lack of standardized build scripts was mentioned as a problem in one case [P27]. There were also problems in synchronizing definitions of software interfaces between teams, and dependencies in code caused problems in larger features [P26]. One case reported that the strong focus on individual teams in the agile model created a fragile architecture, divergence in coding style, and even distrust between teams [P24].

One case describes the progression of the above mentioned coordination problems. At first, an attempt was made to reduce cross-team dependencies, but it became evident that the dependencies were an inherent part of the project. A traditional approach to managing dependencies created excess work, reduced independence and flexibility of teams, and created contract-based and adversarial relationships within the organization. There was some success when up to five teams were coordinated using Scrum-of-Scrums, but that model could not be scaled to a global level. The main problem were teams that were thinking too much inside the team walls. When the new practices were scaled up the communication channels narrowed and created communication breakdowns. Problems with collaboration, dependencies, and integration were solved in the presence of individuals having a proactive and open mindset towards working over team boundaries. It was concluded that a balance is needed between completing new stories from the team backlog and maintaining overall stability of the application. [P34]

4.2.5. Challenges specific to large-scale

Middle managers' new role in agile unclear.

The need for additional management positions to coordinate a large organization may pose problems to agile processes that emphasize self-organization. Especially the role of middle management was unclear in agile methods [P2]. This is problematic, as an agile transformation requires a cultural change particularly on the middle management level [P2]. Managers were reported to need to resist the tendency to command and control and allow room for self-organization, but the change in mindset was difficult to achieve for the people involved [P11, P52]. One case [P30] describes how the project management group had previously worked through big up front plans and competed for resources, but those ways would need to dramatically change.

Several other problems related to management roles were also presented. For instance, [P49] describes how managers were left outside the roles offered by the new agile model. In another case, when managers were appointed as Scrum masters, developers felt being micromanaged [P27]. This was partially because of a poor understanding of the agile method [P27]. It was also described how mixing the role of project manager and Scrum master created a conflict of interest, and turned the Scrum manager's role into one of policing instead of supporting the team [P30]. In some cases, problems with manager positions were solved by phasing out roles relating to the old model, and replacing them with new positions more suitable for agile development [P38, P52].

Using old and new models side by side. In most cases, the organizational transformation proceeded gradually, and during the process it was possible that the new agile model was used in parallel with the old methods. Using the old and new models side by side was generally seen as problematic [P1, P26], causing tensions on all organizational levels [P10]. Ongoing projects had been set up with old development methods, and the agile methods needed to be arranged to fit in with them [P8, P29]. A particular problem in collaboration between waterfall and agile projects was that in agile, the software design was fleshed out over time as sprints progressed, but the waterfall model required a detailed design to be frozen upfront [P29].

Management in waterfall mode. Even after adopting agile, there were cases where management continued to work according to the old waterfall model. One case [P38] described management as *"focused on meetings and big up-front project plans"*, despite having adopted agile. In another case, management was losing confidence in agile because reports on costs and progress were not produced in the same format as before [P35]. As Scrum teams did not commit to fixed schedules, they were considered unreliable [P45].

Some cases reported that development efforts were still controlled on the top level by a project management office (PMO). The PMO was described as entrenched in a waterfall paradigm, and the top-level rigidity was causing friction in the agile adoption [P3, P6]. The PMO was seen as a hub and bottleneck controlling all aspects of projects, and its central structure had to be dismantled for the organization to become agile [P46].

Keeping the old bureaucracy. There were also problems with duplicating bureaucracy when two models were in effect. For instance, agile teams were required to comply with current procedures producing excess documentation and stepping through approval gates [P20, P21]. The bureaucracy of the previous traditional model was still enforced, and management was not willing to lighten the process. Another case describes that after introducing the agile method teams were required to fill in templates of two processes [P48].

Interpretation of agile differs between teams. When many teams implement agile without consistent guidance, friction and fragmentation may emerge [P9]. The organization may require moving people between teams from time to time, and therefore it is desirable that the agile cultures of different teams are not too different. Divergence in process creates increased costs when relocating people [P38]. Further, forecasting and benchmarking teams become difficult [P38, P48]. To overcome problems with divergence in agile models some organizations defined standards [P8, P43].

Internal silos kept. In some cases, the initial organization had internal boundaries, or specialized knowledge in silos, causing problems in the agile implementation. Cloke [P10] describes how the use of Scrum revealed an internal segmentation where teams operated with differing priorities and agendas. This hampered the initial effort to use Scrum in a larger context [P10]. Cowan [P11] describes how projects needed specialized skills that were scarce and people were often relocated to match the needs of the moment. Too much reorganization made it difficult for teams to plan ahead [P11].

4.2.6. Requirements engineering challenges

High-level requirements management missing in agile. In agile methods, requirements engineering is informal and closely tied to the development team. However, large development projects demand additional high-level requirements management. This is apparent in cases where product complexity requires additional layers of product management [P45], requirements are created by several stakeholder groups and development teams cannot be in contact with all of them [P39], or do not have access to stakeholders due to distribution [P24].

Agile methods need to be extended to include a process for getting large requirements into team backlogs [P31].

Requirement refinement challenging. In some cases, requirements were initially defined at a high level in marketing requirements documents [P1, P31] or functional specifications [P33]. These high level requirements needed to be elaborated to be useful by agile development teams. The requirements refinement in itself [P1, P33], when to do it, and to which level of detail [P17] were all reported as challenges.

Creating and estimating user stories hard. Product managers and business analysts struggled with creating high level requirements [P33, P37, P51], and teams struggled breaking them down to an estimatable size [P33]. One study [P48] describes how high level product management delivered requirements in

large chunks, resulting in development teams spending huge amounts of time defining suitably sized stories.

Several cases highlighted that much learning was needed to master the new process of creating user stories, both on product management and development levels [P27, P30, P33, P37]. There were problems such as ambiguity in requirements [P5, P11], and effort estimation for stories [P27, P37].

Gap between long and short term planning. High level requirements work and estimation was problematic as there was a gap between short and long term planning [P9, P10]. Typical agile backlogs give only short-term visibility, creating a need for additional practices for long-term planning [P26, P31].

Several cases described that care had to be taken to avoid long term planning becoming a prescriptive practice. In order to preserve the agility of development, a schedule-driven approach was avoided [P10], sensitivity was used when considering setting milestones [P13], and striving for more accurate estimates was not allowed to become an excuse for gathering requirements up front [P38].

4.2.7. Quality assurance challenges

Similarly to requirements engineering, the agile model may need to be extended to accommodate additional testing activities [P31]. Organizations may have existing separate QA teams that must be coordinated to work with development teams [P17, P30, P48].

Accommodating non-functional testing. Full quality assurance of a system might require special testing such as performance, load, and memory testing, but agile methods lack a focus on them [P17]. These testing activities can not be done within the boundaries of user stories, and require more resources than teams can spare [P31]. When testing tasks overlap team boundaries, it is sensible to have separate teams for the tasks, but coordination between specialized QA teams and development teams must be defined [P28, P48].

Lack of automated testing. Several cases reported problems with lack of automated testing. For a large system, the lack of automated tests caused excess testing work and late discovery of defects [P10, P11, P17].

Requirements ambiguity affects QA. Another challenge in QA was the relationship to requirements engineering. The QA functions were struggling if there were problems such as ambiguity or ineffective breakdown of large requirements [P48]. In the waterfall model, an extended time was reserved for QA at the end of projects, which allowed retroactively resolving ambiguities in requirements [P5]. However, ambiguity became an impediment to QA when development worked in short increments and there was no extended period for stabilization [P5].

4.2.8. Integrating non-development functions in the transformation

Several cases described how introducing agile started from the development teams [P2, P17, P31, P38]. However, development operates in a larger context and needs to interface with other organizational functions, causing challenges.

Other functions unwilling to change. Challenges were faced when exposing other parts of the organization to the agile model, as functions distanced

from development were resistant to change [P2, P18, P31]. Tension grew between development and other roles in the organization [P1]. The full benefits of the transformation could not be attained unless the entire organization was set to work along the same paradigm [P18, P42].

Our primary sources listed various functions beyond development that interface with development and are affected by the agile transformation. Marketing was particularly frequently mentioned [P5, P17, P28, P31, P38, P45, P48]. Other groups included sales [P17, P31], infrastructure and operations [P10, P45, P46, P50], user experience and design [P9, P10, P28], documentation [P28], legal [P10], security [P10], customer services [P48], and finance [P48].

Challenges in adjusting to incremental delivery pace. The iterative nature and the fact that agile affects timings in the software delivery life cycle, often caused problems in the interface between development and other functions.

Iterativeness changed the pace of delivery [P17], and forced design to focus on a shorter scope [P15]. Various groups feeding and supporting development had earlier committed once to a long term plan, but in the new arrangement they needed to adjust to incremental deliveries at a faster pace [P48].

Especially user experience (UX) and interaction designers did not welcome an incremental model [P6], and were struggling with maintaining the big picture in design while working in iterations [P9, P10]. The infrastructure and operations teams fell behind due to the increased speed of development teams, and were forced to update their way of working [P46, P50].

Federoff and Courage [P15] elaborate on the user experience team's problems with the short time horizons of Scrum. Previously, the UX team had supplied development with designs within a long release cycle, but Scrum required completing features within weeks. The problem was that the UX work required a holistic view instead of an incremental one, and the design process did not fit well into the time frame of a sprint. The UX team was overloaded and discontent, but the situation was fixed by refining the schedule for interactions with the development teams. [P15, P28]

Challenges in adjusting product launch activities. A second type of timing problem was that certain release activities inevitably have long lead times, and required the commitment to a set of functionality early on. Agile's emphasis on short time horizons and flexibility in prioritization did not mix well with such activities. For instance, preparing printed marketing material and press releases requires information to accurately present the upcoming features of the product [P38].

One case describes that marketing needed three months for preparing the product launch, but the agile process allowed development to keep changing the content of the product. As a result, marketing struggled, creating material and campaigns without knowing the exact product features. Further, the processes of acquiring export clearances and licensing authority required that the feature set of the product was known well in advance. Development felt that these requests slowed them down. The issues were solved by agreeing that marketing could at any time request a release in 90 days, and development would be able to commit to a fixed feature set to be delivered [P31].

Rewarding model not teamwork centric. In several cases, human resources (HR) was mentioned as working against the agile adoption. In order to gain the full benefits of going agile, the entire organization should to be aligned, including HR [P38, P42, P49]. Especially the rewarding practices set by HR were seen as problematic, as rewards were tied to personal performance, acting against the team-centric thinking and the agile model in general [P3, P6, P49]. One case even reported a practice of tracing failures down to individuals [P40].

4.3. Success factors in transformation

In this section answer our second research question RQ2: *What success factors have been reported for large-scale agile transformations?* We organized the 29 success factors, each identified in several primary studies, into eleven categories, which are elaborated in this section and summarized in Table 6.

4.3.1. Management support

Ensure management support. Numerous cases made it clear that management support is an absolute necessity [P8, P18, P27, P31, P33, P36, P43, P47, P49]. This was reflected in statements such as *"Adopting agile, or implementing any significant change, requires an executive's sincere support."* [P44], *"Executive commitment was crucial to implementing massive change."* [P16], and *"Having upper management engaged, supportive and visible is critical for wholesale organization involvement with Scrum."* [P11].

Managers were seen to be in a key role in making the change stick, as they had the authority and power to remove impediments [P18]. Seffernick [P46] describes how a number of people attempted to explain away the applicability of agile methods, but the objections were overruled by the director's commitment to make agile work. Management support was similarly needed when tight release schedules had to be flexed in order to give room for the adoption process [P16, P27]. Favorable management decisions were also critical when additional resources were allocated to training and coaching [P47].

Make management support visible. Visible involvement of management was reported to motivate and encourage employees to adopt the new model [P40]. For instance, the CTO organized training sessions personally [P24], and the engineering VP frequently visited sprint demos [P11]. When the corporate level support for the agile initiative was showing teams adopted agile methods even spontaneously [P9].

Educate management on agile. In order to gain support for agile several cases highlighted the need to educate management [P8, P47].

Without education managers might make harmful interventions in the process, or just stick to the old model [P16, P35, P46]. Managers not understanding the principles of agile felt left out with the introduction of self-organizing teams, which sometimes resulted in backlashes [P6]. Providing proper training corrected the situation, and even created strong agile supporters in management [P6, P9]. Training cleared misconceptions [P29], and helped create a consistent implementation of the agile model across the organization [P11].

Table 6: Success factors for large-scale agile transformations

Success factors	Primary sources
Management support	
Ensure management support	P8, P11, P16, P18, P27, P31, P33, P36, P43, P44, P46, P47, P49
Make management support visible	P9, P11, P24, P40
Educate management on agile	P6, P8, P9, P11, P16, P29, P35, P46, P47
Commitment to change	
Communicate that change is non-negotiable	P11, P22, P48, P49
Show strong commitment	P8, P10, P11, P43
Leadership	
Recognize the importance of change leaders	P3, P4, P11, P16, P18, P31, P33, P42
Engage change leaders without baggage of the past	P11, P16
Choosing and customizing the agile model	
Customize the agile model carefully	P8, P10, P11, P29, P31, P40, P41, P43, P45, P49, P51
Conform to a single model	P8, P11, P12, P16, P32, P43
Map to old model to ease adaptation	P3, P14, P16, P29, P44, P46, P48
Keep it simple	P5, P10, P16, P40
Piloting	
Start with a pilot to gain acceptance	P3, P9, P14, P17, P31, P36, P38, P39, P47
Gather insights from a pilot	P8, P22, P27, P40, P45
Training and coaching	
Provide training on agile methods	P3, P6, P11, P14, P16, P30, P37, P46
Coach teams as they learn by doing	P3, P6, P8, P9, P16, P17, P18, P29, P30, P33, P42, P44, P47, P48
Engaging people	
Start with agile supporters	P6, P29, P34
Include persons with previous agile experience	P6, P10, P37
Engage everyone	P4, P6, P16, P22, P31
Communication and transparency	
Communicate the change intensively	P16, P22, P33, P40, P41, P43, P48
Make the change transparent	P6, P11, P16, P42, P43
Create and communicate positive experiences in the beginning	P6, P32, P29, P40, P42, P46
Mindset and Alignment	
Concentrate on agile values	P1, P16, P42, P46, P48
Arrange social events	P11, P23, P27, P32, P40, P41
Cherish agile communities	P3, P12, P41, P42, P47
Align the organization	P8, P18, P31, P42, P43, P46
Team autonomy	
Allow teams to self-organize	P5, P16, P18, P27, P30, P34, P41, P45
Allow grass roots level empowerment	P3, P6, P41
Requirements management	
Recognize the importance of the Product Owner role	P12, P14, P16, P24, P30, P33, P39, P46
Invest in learning to refine the requirements	P5, P11, P17, P33, P48

There were a few suggestions on how to better involve management. Foremost, a successful experience will likely raise the interest of management [P36]. Mencke [P33] suggests giving executives tasks related to the agile process, so that they have better visibility of the new way of working.

4.3.2. Commitment to change

Communicate that change is non-negotiable. Feedback from the personnel should certainly be listened to, but in the end it must be made clear that the old model can not be returned to [P22].

For a case where the organization was changed all at once it was reported that the magnitude of change helped to create an impression that the change was non-negotiable [P11]. Firstly, as it was evident that a large change was to happen people felt encouragement and permission to move on from the old way, and secondly, the urgency eliminated wasteful discussions on whether Scrum was a good model or not [P11].

When the organization culture is ingrown, the change vision must constantly be reminded of, and each step towards the new model considered as a victory [P48].

Transformation can be facilitated by setting up mechanisms that force change. Articulating urgency can help getting the change under way [P49]. Spayd [P49] describes that senior management pushed hard on a mandate to have deliveries every 90 days. The mandate made change necessary, and while the strong pressure did not promote agile practices in every case, the drive for change was perceived good in general [P49].

Show strong commitment. A large change will inevitably require strong commitment, but problems in the transformation can put the commitment to test. The agile model is introduced because of problems in the old model, and therefore there will be organizational issues uncovered during the transformation [P10]. People must not be demoralized when facing challenges, and the determination to change must be maintained [P8, P10]. Problems might not be due to the agile model — in some cases it can expose existing problems in the organization [P10].

Ryan and Scudiere [P43] describe that management commitment was showing as a strong focus on certain agile practices that had been chosen as non-negotiable, and constant assessment that the practices work. The expectations were clearly communicated to the teams, and constant assessment made it clear that change was desired [P43]. A strong commitment from management assures teams that the change is the right thing to do [P11].

4.3.3. Leadership

Recognize the importance of change leaders. Transforming the way of working of a large group requires coordination and leadership [P31, P42]. In addition to the leadership provided by coaches, specific change leaders were mentioned. Cases indicated the importance of having spokespersons for the change [P3, P4, P31]. Cowan [P11] describes how one person was strongly driving the transformation, and made an indisputable contribution for transforming the

organization. Maples [P31] describes how the change and agile adoption was guided by one "counselor" [P31]. Other cases described that the change was led by a competent roll-out team, which had representatives from all parts of the organization [P16, P33]. Finally, the responsibility of line and project managers to act as change leaders was highlighted [P18].

Engage change leaders without baggage of the past. A few cases discussed the benefit of having change leaders without baggage from the past. Cowan [P11] describes how the newly hired director was able to circumvent territorial battles that existed from before, and therefore focus fully on getting the agile model implemented. In another case, external coaches were described to better spot places for correction in the agile model, and their advice was received better because they were considered impartial when being external to the organization [P16].

4.3.4. Choosing and customizing the agile model

Customize the agile model carefully. Customizing the agile model and practices was often seen as a necessary step in the agile implementation. As each organization will have its own challenges in adopting agile, it should be carefully considered which organization specific areas to focus on when choosing the agile practices to implement [P8, P43]. Lewis and Nehrer [P29] recommended choosing the agile model according to the current corporate model in order to avoid interference.

Cases reported that customization was part of a successful agile implementation. For instance, teams were let to customize their practices individually, to fit the needs of each team [P41]. Spayd [P49] writes that teams who modified agile practices to fit the large and distributed environment ended up doing better than teams that did not.

To allow teams to become innovative and perform well, the agile model should be customized in a pragmatic way instead of following a strict textbook interpretation [P10]. In order to draw the most out of agile, teams should innovate and find their own practices that really work for their case [P40, P41]

Especially in a large organization it is not feasible to use the same process for all projects [P49]. An individual application of the agile process is needed for different types of development, depending on, e.g., the type of software being developed or the project size [P45]. Applying agile at scale will require to deviate from some of the typical recommendations [P31]. However, it is essential to remember the agile principles when doing customizations, and watch out for customization that would contradict the principles [P31].

The customization of the agile model was also seen as an evolutionary process. Cowan [P11] suggested that in a big bang transformation it might be necessary to selectively compromise some parts of the agile implementation so that the core practices can be set in place. The agile transformation is a constantly ongoing process, and it is recommended to regularly challenge teams to refine the agile implementation to reflect the current needs of the organization [P41, P43].

Conform to a single model. Studies reported that conformity to a single model should be considered when implementing agile. A consistent common vocabulary was seen to benefit the organization and support the change [P8, P16, P43]. Other benefits of using a single model were the possibility to compare work between teams, easier relocation of people, and predictable progress for the stakeholders [P43]. Consistency and common understanding was created in discipline meetings and public events, or by peer coaching [P11, P12, P32].

Map to old model to ease adaptation. Mapping the agile model to the old model was seen as necessary in a few cases. Although a general mapping to the old model may not be good [P44, P46], some cases needed to preserve high-level management practices [P14, P29, P48]. By allowing management to remain unchanged it was possible to introduce agile step-wise on the team level, which helped in getting management buy-in for the process [P14, P29, P48].

In one case, Scrum was considered as a wrapper for existing practices, and could therefore easily fit in the organization [P44]. The agile methodology was considered complementing the existing culture, which eased management buy-in for the new method [P44].

A few organizations had structures in place that were similar to agile, making the transformation easier. For instance, a previous organizational model based on small and autonomous teams strongly aided the adoption [P3]. Another case described the original on-demand delivery model a natural fit for agile, and the transformation was presented as a return to core values instead of a remodeling [P16].

Keep it simple. One piece of advice given by a few cases was to keep the organization and process simple [P10, P16]. Beavers [P5] describes that attempting to operate in a complex and global setting complicated even simple agile practices. The organizational chart was simplified, which was welcomed by both employees and managers [P5]. Rather than focusing on details in process, communication practices, and tools the focus should be on engaging the teams [P40].

4.3.5. Piloting

Start with a pilot to gain acceptance. Having a pilot project was reported as a significant success factor in several cases [P3, P9, P38]. The pilot project helped create confidence that the agile model would be suitable for the organization and the general acceptance of agile was increased [P3, P17]. The pilot also cleared disbeliefs of the suitability of agile for large organizations, and created acceptance both in development and management [P9, P31, P36].

Pilots were especially important for gaining management acceptance. Cases reported that management gave approval for agile methods only after seeing successful pilots [P14, P47]. Prokhorenko [P39] describes that seeing the example of successful pilots will make managers from other departments eager to try the new method, and thus help to spread its use.

Gather insights from a pilot. A benefit in piloting was that it provided knowledge on how to fit the agile method to the particular organization. A pilot project enabled the organization to get feedback on how teams and managers

are best introduced to the new methods [P8, P22]. Organizations met challenges in the pilot projects. However, the pilot projects served as valuable learning experiments providing insights on how to mitigate problems when the rest of the organization is transforming [P27, P40, P45].

4.3.6. Training and coaching

Provide training on agile methods. Several studies stated that training improved the chances of succeeding in the transformation [P3, P16, P30]. It was even highlighted that the change would have failed without training [P14], as Benefield [P6] claimed: *"we saw again and again how training could make the difference between success and failure for a team"*. Training also helped people to become more positively inclined towards the new model [P16, P37], and at best training made people enthusiastic to change [P3, P46]. A few cases reported having some disinclination to change that they thought training could have helped overcome [P11, P14].

Coach teams as they learn by doing. Agile methods avoid prescribing exact ways of working, and rather emphasize a mindset of adapting to the situation. It is difficult to explain by theory how such a mindset should be applied, and the agile practices are best learned by doing. Coaching teams while they apply the agile methods in practice was seen as an important factor in change [P3, P16, P17, P30, P47]. A few cases stated even that coaching was essential for success in transformation [P8, P18, P29, P33]. Also, without coaching, teams can do damage to the agile transformation if techniques are applied improperly [P9].

There were a number of statements on the benefits of coaching. A coach can watch for and correct problems when they arise [P16, P42]. Coaches helped draw attention away from a focus on tools towards understanding the principles of agile [P42, P48]. There were also benefits in using both internal and external coaches. External coaches were able to have an objective view of the organization [P16, P44], whereas internal coaches were more accessible and knew the specifics of the organization [P3, P6].

4.3.7. Engaging people

Start with agile supporters. Choosing people with a disposition towards agile methods was seen as a requirement for the change to take the right track. Teams are staffed usually based on technical competences, but also considering personality aspects was emphasized for agile teams [P29, P34]. The personality of people was seen as a key aspect for achieving change [P6]. There is a need for collaborative and understanding persons who are prepared to discard preconceptions and willing to try new untested approaches [P6, P29].

Include persons with previous agile experience. A few cases mentioned the importance people with previous agile experience had at the start of the transformation [P6]. For instance, it helped the product management office to implement agile over the entire enterprise when the staffing was updated to include people with agile experience [P10]. Staffing teams partially with devel-

opers familiar with agile helped the rest of the team get a good hold of agile development [P37].

Engage everyone. Many cases highlighted the importance of engaging people broadly in the organization. One of the lessons learned presented was that it is important to involve all stakeholders to gain acceptance of the transformation [P4, P31]. The transition team made an effort to engage people by both inviting them to give feedback online and by holding an extensive number of feedback meetings [P6, P16, P22]. Being inclusive towards everyone was seen as one of the key success factors in the transformation [P16]. Inclusiveness will encourage people to participate and use the new model visibly [P16].

4.3.8. Communication and transparency

Communicate the change intensively. Intensive communication was emphasized in a number of studies. It is important to reach as many people throughout the organization as possible, as without communication the new model will not take root [P16, P43]. It was recommended that use of the new model is made highly visible on many communication channels and even over-communicated [P16, P33, P41]. Mencke [P33] summarizes the viewpoint on over-communication: *"Even if you think that your teams understand a new method or process, repeat your communication to be sure."* Workshops, coaching sessions, online discussions, and newsletters are examples on different communication formats used [P41]. Another approach in communicating the change was that managers explained and encouraged change in an extensive series of one-on-one discussions [P22].

Some studies emphasized communicating the goals of the transformation. Having a clear message of expectations helped remove confusion and make people understand the purpose of the transformation [P43, P48]. The motivation to change can be increased by having a simple proposal on how to reach the desired outcomes [P40]. McDowell and Dourambeis [P32] describe how the organization launched a series of communication events especially designed to emphasize the reasons behind the agile practices.

Make the change transparent. Enabling transparency during the transformation was reported as important, and even highlighted as a critical factor for success [P11, P16]. Fry and Greene [P16] underline the importance of transparency and sharing of information: *"This bias to sharing information with everyone was critical in our ability to adapt on a daily basis to ensure our success."* Transparency was achieved by showing both successes and challenges [P6], actively reaching out for feedback [P6], using project management tools to display project status publicly [P11], by rearranging physical spaces [P42], and by holding the meetings of the roll-out team in public [P16]. By sharing experiences and status of the transformation the organization was aligned and everyone was moving in the same direction [P42, P43].

Create and communicate positive experiences in the beginning. It is clear that any positive results will facilitate an organizational transformation. The move towards agile is assisted by making any benefits publicly visible and celebrating even the small victories [P40, P42].

Several cases highlighted that the agile transformation spread effectively through positive word-of-mouth [P6, P40]. When good results were shown by a team it created interest in others, and enthusiasm to try the new model will spread [P39, P46]. Some companies used agile and waterfall methods side by side. This setting made comparison possible, bringing up the benefits of the agile model [P19, P36].

Although a recipe to fabricate positive experiences can not be given, many cases described how they had succeeded. Benefield [P6] describes how a survey was done to objectively measure satisfaction in agile, and the positive results helped teams make the decision to change their way of working. Further, management requested proof on better performance as a response to requests to increase coaching and training budget. After conducting inquiries, the coaching team was able to present an estimate of a 30% increase in performance, which made management highly convinced [P6].

4.3.9. Mindset and Alignment

Concentrate on agile values. In a number of studies it was described that the principles of agile should be emphasized over practices and simple mechanics [P46, P48]. When people understand the agile values they will also understand why the change is being done and feel motivated [P1, P16, P42].

Some cases described that inexperienced coaches and Scrum masters made mistakes in focusing too much on the implementation of practices [P23, P30].

Arrange social events. Social events were reported to benefit the transformation by helping to build an agile mindset. A few cases even described the transformation being driven by a series of events where people received information and had the possibility to participate in shaping the new model of working [P32, P41]. One case described how the corporate agile awareness and teaching event was designed to be fun and engaging, which made people more enthusiastic in applying the agile practices [P32].

Various social activities were presented as valuable tools for improving bonding within teams [P11, P27, P40]. The importance of creating personal bonding was also highlighted for change leading entities such as management and coaching teams [P11, P23].

Cherish agile communities. The formation and influence of agile communities was reported to have a significant impact on transformation. The emergence of an agile community was reported as a success factor in the transformation [P3]. A community was also seen as indispensable as it has power to overcome impediments that would be too large for individuals to affect [P47]. The agile communities raised awareness of agile methods in the organization [P41, P42, P47], and spawned eager followers who spread the word even further [P12].

Align the organization. A factor in achieving change was creating alignment towards the common goal of introducing new development methods. It was seen as an important factor that all levels in the organization speak the same language and accept the change [P18, P31]. Alignment was built by promoting success stories and gathering lists of problems to tackle [P42, P46]. In

one case, a complete alignment between teams and within management was considered as necessary, to use the agile model in a large context [P43]. Another case highlighted creating and applying a structured roll-out process to uniformly introduce agile to a large number of teams [P8].

4.3.10. Team autonomy

Allow teams to self-organize. The agile principle of giving teams the power to decide over themselves was seen as an important factor in advancing the transformation. In some cases, management initially attempted to prescribe how the new practices should be implemented [P5, P18, P41]. However, it was learned that only when full control was given to teams the agile methods could be properly established [P5, P16, P41]. Allowing self-organization creates commitment to the change and motivation to continued use [P16, P18]. It allows teams to take ownership of the development model and voluntarily improve it even further [P41].

The acceptance of agile methods was easy when teams gained the authority to decide on development speed and quality [P45]. Favoring empowerment over prescribing the new practices was also reported to improve performance [P27]. Giving teams the authority to decide over work items increased productivity and morale [P30, P34].

Roche and Vasquez-McCall [P41] describe how prescribing the agile methods to use led only so-far in the transformation. The effectiveness of teaching and communicating the transformation started to decline. To enable the transformation to proceed further, an organization-wide challenge was created where teams would independently develop and showcase the best agile practices. As a result the ownership of the methods was transferred to teams, and the new model gained a secure foothold. [P41]

Allow grass roots level empowerment. An interesting success factor was the absence of a top-down mandate. Atlas [P3] describes that the use of Scrum was spreading because teams were both allowed to use and enabled to learn the method. The transformation itself was agile when people felt empowerment in making the decision to adopt. The incentive to change was created when teams learned about agile methods, and perceived that a change would be beneficial [P3]. The absence of mandate granted genuine support for the model on the grass roots level, which was a necessity for the process to work [P6]. It was also thought that proceeding with a top-down mandate would have made teams conform to a single process [P41]. This would have been sub-optimal, as it was important that each individual team and project tailored their practices [P41].

4.3.11. Requirements Management

Recognize the importance of the Product Owner role. A particularly important role was the Product Owner. Many cases reported problems if the Product Owner did not perform adequately, and it was seen as critical to have a dedicated person in that role [P12, P14, P30, P39]. Successes or problems emerged respectively depending on how well the Product Owners were performing.

In one case, a well-implemented Product Owner role was understood to be a key success factor when using agile methods [P16, P33]. It was reported that when the Product Owner role was implemented correctly, the team performed better and the work products were correct [P24]. Projects started off better when the Product Owners were engaged early on [P46]. Some Product Owners were even described to become agile enthusiasts when they learned how business and technology can collaborate in the agile model [P46].

Several cases endorsed training and coaching for the Product Owner role [P16, P39, P46]. Product Owners should receive training on agile principles, backlog management, user story breakdown, and agile planning [P16]. Also tool support for backlog management and two way communication between Product Owners and teams should be worked on [P39].

Invest in learning to refine the requirements. Some cases reported challenges in requirements management, highlighting the difficulty in managing the gap between high-level requirements and user stories handled by teams. It is important that the requirements are concise and small enough for the teams to handle [P5, P11]. It was recommended to invest in teaching skills in breaking down and writing user stories [P11, P33, P48]. Gat [P17] describes how the gap between development and requirements management was bridged by having a dedicated "requirements architect" to help with requirements refinement.

5. Discussion

In the past decade numerous publications have discussed the use of agile in large-scale organizations. We identified 52 papers presenting 42 industry cases describing large-scale agile transformations. The primary studies identified were almost exclusively experience reports. We could find only six papers that we classified as case studies. Thus, a main finding of this study is that despite the relevance of the topic for practitioners, research is lagging behind, and there is in particular a need for rigorous case studies and summarizing research. In this work, we provide a compilation and summarizing analysis of existing publications on large-scale agile adoption.

5.1. Lessons Learned for Practitioners

In this section we summarize the most important success factors for large-scale agile transformations, which we believe are helpful for practitioners planning and implementing transformations. These success factors are based on the 11 categories we presented earlier.

1. **Ensure management support:** Numerous cases showed that when the leaders of the organization are committed and supportive, the change is much more likely to happen. When management support is visible, it encourages and motivates the employees to adapt the new way of working. However, managers need to be enlightened, understanding and adhering to the agile principles, e.g. regarding empowerment. If this is not the case, they might make decisions or take action that harm the transformation. Thus, training and coaching of management is strongly recommended.

2. **Commit to change:** To make a large organization change, the commitment to change must be made clear to everyone involved. Communicate to the organization that the change in non-negotiable and returning to the old model is not an option. Problems in the transformation can put the commitment to test, and the agile model may even expose problems in the organization. Strong commitment is needed, and a determination to change must be maintained despite any problems that might surface.
3. **Engage change leaders:** Organizing a large number of people requires management and leadership. Both internal and external coaches can have an important role in leading the change. An individual, as well as a task force, e.g., a roll-out team, or a community can take the role of change leader and evangelist. Sometimes an external person, like a new director, is needed.
4. **Choose and customize the agile model carefully:** The cases emphasized that conforming to a single agile model in the whole large organization is beneficial. Customize the agile model to your own organization carefully. Keep both the organization and the process simple.
5. **Start with a pilot:** Initially piloting agile on a smaller scale gives a possibility to test it before full-scale rollout. A successful pilot provides proof that agile can work in the organization. Management approval for an agile transformation is easier to get when they see a successful pilot. Gather knowledge from the pilot on how to adapt the agile model to your organization.
6. **Invest in training and coaching:** In order to get a good start in the transformation, the ideas of the new methods need to be explained thoroughly. Training sessions are good starting points, but teams should be provided coaching for a longer time to ensure that the new practices are learned and applied correctly in practice.
7. **Engage people:** The choice of people whom to initially involve in the transformation was reported to make a difference. Interest or previous experience of agile methods were valuable characteristics to look for. Start with agile supporters. If available, engage persons that have previous agile experience. Finally, involve all stakeholders to gain acceptance in the transformation. Being inclusive of everyone was one of the reported key success factors.
8. **Communicate the change intensively:** Reach as many people throughout the organization as possible, as without communication the new model will not take root. Communicate the new model and the goals of the transformation using, e.g., workshops, coaching sessions, online discussions, news letters and one-to-one discussions. Using many channels and even over-communicating is encouraged. Keep the transformation transparent by showing both successes and challenges, sharing experiences and displaying the status of the transformation publicly.
9. **Build an agile mindset and align the organization:** Creating an agile mindset among all participants of the transformation and aligning the

whole organization towards a common goal are important success factors. Concentrate on agile values over practices and simple mechanics. When people understand the values, they will also understand why the change is being done and feel motivated. Arrange social events to both inform and engage people. Cherish agile communities, as they, e.g., build awareness of agile in the organization and help to remove impediments.

10. **Give teams autonomy:** Allow teams autonomy to self-organize and decide for themselves on how they execute their tasks and tailor their practices. Allowing self-organization creates commitment to the change and motivation to continued use. It allows teams to take ownership of the development model and voluntarily improve it even further. Instead of a top-down mandate, allow grass-roots level empowerment.
11. **Put effort on requirements management:** The Product Owner is the key role in handling requirements. Thus, a dedicated person or persons are needed for this role. This role has a huge impact on how the agile teams will perform, thus implementing it carefully is highly important. The Product Owner should be trained well on agile, e.g., on backlog management, user story breakdown, agile planning, as well as communication with the teams. The gap between high-level requirements and user stories handled by teams can be challenging, thus investing in building skills to learn how to refine requirements into user stories is needed.

5.2. Discrepancies and Open Issues

The papers included in our review provided several pieces of good advice in the form of challenges and success factors. However, looking at the challenges and success factors, we can notice discrepancies, and even controversial advice, highlighting aspects of agile transformation that would require deeper study to gain more insight.

Dealing with the old process. It is unclear how and when to completely drop the old process and its related concepts and practices. The literature mentions challenges when keeping the old model, such as using the "Old and new models side by side", "Management in waterfall mode", and "Integrating non-development functions in the transformation" indicating that the coexistence of models is difficult. On the other hand, most transformations were done using a stepwise approach, meaning that this situation will exist at least for some while in the organization, and using the old model as a reference for the transformation can also be beneficial, as the good practice "Map to old model to ease adaptation" suggests.

Drawing the line between "one single model" and allowing self-organization. On the one hand, to successfully perform an agile transformation, it seems important to use a single model as a starting point, as the success factor: "Conform to a single model" indicates. This allows the organization to have a consistent vocabulary, which was seen to benefit the organization and support the change [P8, P16, P43]. A single model makes it possible to compare work between teams and makes it easier for people to relocate and have

predictable progress [P43]. If such a model is lacking, challenges like "interpretation of agile differs between teams", becomes exacerbated, making it difficult to collaborate between teams or change team members between the teams. On the other hand, some primary sources suggest that teams should be allowed to self-organize and customize their practices individually to fit the needs of each team, and full control should be given to teams. It was also mentioned that in a large organization it is not feasible to use the same process for all projects [P49], but individual application of the agile process is needed for different types of development, depending, e.g., on the type of the software being developed or the project size [P45].

Providing management support without suppressing grass-root level empowerment. On the one hand, having management support is a necessary condition for successful transformation, as shown by the related success factors. However, forcing the transformation from the top can create problems, as some organizations reported "Top-down mandate creates resistance" as a challenge. Of the reported transformations, about half were led top-down, and about one forth bottom-up, and one fourth using a hybrid model.

5.3. Limitations

Researcher bias might have influenced the selection of primary studies, as well as data extraction. The selection of the primary studies may have been distorted by interpreting the inclusion criteria falsely. This risk was mitigated by using three researchers in designing the inclusion criteria. When the inclusion criteria was subsequently applied, the abstract filtering was performed independently by two researchers, and unclear cases were resolved by case-by-case discussions between two or three researchers. In the full-text filtering phase, one researcher did the initial filtering making section regarding the most clear cases and the rest, over half of the papers, were read by two researchers and all unclear cases discussed and resolved by two or three researchers together.

The second part of the research that may have been affected by subjective bias was the elicitation of results by coding and analysis. Our tools for mitigating this threat were limited, as the work stages in question are particularly laborious. For resource constraints the process could not be duplicated. We made an attempt to prevent subjective bias in analysis by making the results as traceable as possible, by supplying references to each claim presented in the results.

A particular problem in this systematic literature review was the limitations of Boolean keyword searches in online databases. A keyword search cannot easily identify the facets *large-scale* and *empirical*, and also the facet *transformation* is difficult to capture with keywords. For these reasons we did not include the facets *large-scale* and *empirical* in the keyword search, but instead did a manual walkthrough of all the selected papers in the filtering phase to determine the size of the organization and whether the paper contained empirical material. This added manual work, but mitigated the threat.

Regarding the facet *transformation*, we selected as many synonyms for it as keywords as possible. However, it is possible that some studies were missed by the keyword search, potentially causing bias.

Only six of the primary studies presented a clearly defined research method. Most of the papers were experience reports, where the author was a member of the organization discussed, creating author bias. However, due to the low number of high-quality studies, we deemed that the results would be distorted heavily and many valuable studies would be left out if a strict quality assessment would be part of the inclusion criteria. As a result we decided to include all experience reports, regardless of the perceived objectivity.

Many primary studies were openly pro-agile, without giving a solid motivation for the standpoint. The tendency to publish only positive results is another particular problem for this literature review. The primary studies typically reported the transformations as successful, a sign of publication bias. However, most paper did bring up several challenges experienced during the transformation as well as perceived success factors. Due to the current state of research it is necessary to include studies with varying strength of evidence in order to relevantly aggregate evidence on large-scale transformations.

Due to the author and publication bias of the primary studies and qualitative nature of transformation descriptions, we decided not to make quantitative interpretations in the results, instead use only qualitative analysis. Therefore, based on this data we could not, e.g., rank the challenges and success factors and say which ones are the most important ones. However, by looking at the number of cases that mentioned some success factor or challenge we can see at least some direction.

6. Conclusions

We presented a systematic literature review of empirical studies and experience reports on large-scale agile transformations. We analyzed 52 papers describing 42 different organizations, presenting qualitative findings describing challenges and success factors for large-scale agile transformations.

Challenges included change resistance, lack of investment in the transformation, finding agile difficult to implement in general, coordination challenges between teams, and friction caused by the fact that a large organization cannot transform all at once. Important success factors were management support, choosing and customizing the agile model to fit the organization, the use of training and coaching, and creating an agile mindset and aligning the organization.

Adopting agile methods in large organizations is closely related to a generic organizational change in knowledge-based organizations. Future research should look into what the software engineering field can learn from existing theories on organizational change. Furthermore, longitudinal studies studying the maturation and long-term impact of the usage of agile in the large organizations would be valuable.

Acknowledgements

This study was funded by TEKES as part of the Need for Speed (N4S) SHOK program.

References

[1] K. Beck. Embracing change with extreme programming. *Computer*, 32(10):70–77, 1999.

[2] H. Berger and P. Beynon-Davies. The utility of rapid application development in large-scale, complex projects. *Information Systems Journal*, 19(6):549–570, 2009.

[3] E. Bjarnason, 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 1st Workshop on Agile Requirements Engineering*, AREW '11, pages 3:1–3:5, 2011.

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

[5] B. Boehm and R. Turner. Management challenges to implementing agile processes in traditional development organizations. *Software, IEEE*, 22(5):30–39, 2005.

[6] G. Cloke. Get your agile freak on! agile adoption at yahoo! music. In *Agile Conference (AGILE), 2007*, pages 240 –248, aug. 2007.

[7] A. Cockburn and J. Highsmith. Agile software development, the people factor. *Computer*, 34(11):131–133, Nov 2001.

[8] M. Cohn and D. Ford. Introducing an agile process to an organization [software development]. *Computer*, 36(6):74–78, 2003.

[9] D.S. Cruzes and T. Dyba. Recommended steps for thematic synthesis in software engineering. In *Proceedings of the 2011 International Symposium on Empirical Software Engineering and Measurement*, ESEM '11, pages 275–284, Washington, DC, USA, 2011. IEEE Computer Society.

[10] T. Dingsøyr and N.B. Moe. Research challenges in large-scale agile software development. *SIGSOFT Softw. Eng. Notes*, 38(5):38–39, August 2013.

[11] Torgeir Dingsøyr, TorErlend Fægri, and Juha Itkonen. What is large in large-scale? a taxonomy of scale for agile software development. In Andreas Jedlitschka, Pasi Kuvaja, Marco Kuhrmann, Tomi Männistö, Jürgen Münch, and Mikko Raatikainen, editors, *Product-Focused Software Process Improvement*, volume 8892 of *Lecture Notes in Computer Science*, pages 273–276. Springer International Publishing, 2014.

[12] Torgeir Dingsøyr and NilsBrede Moe. Towards principles of large-scale agile development. In Torgeir Dingsøyr, NilsBrede Moe, Roberto Tonelli, Steve Counsell, Cigdem Gencel, and Kai Petersen, editors, *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation*, volume 199 of *Lecture Notes in Business Information Processing*, pages 1–8. Springer International Publishing, 2014.

[13] T. Dybå and T. Dingsøyr. Empirical studies of agile software development: A systematic review. *Information and Software Technology*, 50(9-10):833–859, 2008.

[14] T. Dybå and T. Dingsøyr. What do we know about agile software development? *Software, IEEE*, 26(5):6–9, 2009.

[15] A. Elshamy and A. Elssamadisy. Divide after you conquer: An agile software development practice for large projects. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4044 LNCS:164–168, 2006.

[16] M. Federoff and C. Courage. Successful user experience in an agile enterprise environment. In *Procceedings of HCI International 2009, San Diego, CA, USA, July 19-24, Part I*, volume 5617 of *LNCS*, pages 233–242, 2009.

[17] Brian Fitzgerald, Gerard Hartnett, and Kieran Conboy. Customising agile methods to software practices at intel shannon. *European Journal of Information Systems*, 15(2):200–213, 2006.

[18] M. Fowler. Put your process on a diet. *Software Development*, 8(12):32–36, 2000.

[19] Sallyann Freudenberg and Helen Sharp. The top 10 burning research questions from practitioners. *Software, IEEE*, 27(5):8–9, Sept 2010.

[20] C. Fry and S. Greene. Large scale agile transformation in an on-demand world. In *Agile Conference (AGILE), 2007*, pages 136 –142, aug. 2007.

[21] C. Fulgham, J. Johnson, M. Crandall, L. Jackson, and N. Burrows. The fbi gets agile. *IT Professional*, 13(5):57–59, 2011.

[22] AM.M. Hamed and H. Abushama. Popular agile approaches in software development: Review and analysis. In *Computing, Electrical and Electronics Engineering (ICCEEE), 2013 International Conference on*, pages 160–166, Aug 2013.

[23] J. Highsmith and A. Cockburn. Agile software development: the business of innovation. *Computer*, 34(9):120–127, 2001.

[24] P. Hodgkins and L. Hohmann. Agile program management: Lessons learned from the verisign managed security services team. In *Agile Conference (AGILE), 2007*, pages 194 –199, aug. 2007.

[25] S. Jalali and C. Wohlin. Global software engineering and agile practices: a systematic review. *Journal of Software: Evolution and Process*, 24(6):643–659, 2012.

[26] M. Kaisti, V. Rantala, T. Mujunen, S. Hyrynsalmi, K. Konnola, T. Makila, and T. Lehtonen. Agile methods for embedded systems development - a literature review and a mapping study. *EURASIP Journal on Embedded Systems*, 2013(1):15, 2013.

[27] Eunha Kim and Seokmoon Ryoo. Agile adoption story from nhn. In *Computer Software and Applications Conference (COMPSAC), 2012 IEEE 36th Annual*, pages 476 –481, july 2012.

[28] B. A. Kitchenham. Guidelines for performing systematic literature reviews in software engineering. Technical report EBSE-2007-01, Keele University Technical Report, 2007.

[29] H. Koehnemann and M. Coats. Experiences applying agile practices to large systems. In *Agile Conference, 2009. AGILE '09.*, pages 295–300, Aug 2009.

[30] M. Lindvall, D. Muthig, A. Dagnino, C. Wallin, M. Stupperich, D. Kiefer, J. May, and T. Kahkonen. Agile software development in large organizations. *Computer*, 37(12):26–34, 2004.

[31] J. A. Livermore. Factors that significantly impact the implementation of an agile software development methodology. *Journal of Software*, 3(4):31–36, 2008.

[32] R. Lyon and M. Evans. Scaling up – pushing scrum out of its comfort zone. In *Agile, 2008. AGILE '08. Conference*, pages 395 –400, aug. 2008.

[33] R.P. Maranzato, M. Neubert, and P. Herculano. Scaling scrum step by step: The mega framework. In *Agile Conference (AGILE), 2012*, pages 79 –85, aug. 2012.

[34] J.R. Miller and H.M. Haddad. Challenges faced while simultaneously implementing cmmi and scrum: A case study in the tax preparation software industry. In *Information Technology: New Generations (ITNG), 2012 Ninth International Conference on*, pages 314–318, April 2012.

[35] D. Mishra and A. Mishra. Complex software project development: Agile methods adoption. *Journal of Software Maintenance and Evolution*, 23(8):549–564, 2011.

[36] S. C. Misra, V. Kumar, and U. Kumar. Identifying some critical changes required in adopting agile practices in traditional software development projects. *International Journal of Quality and Reliability Management*, 27(4):451–474, 2010.

[37] E. Moore and J. Spens. Scaling agile: Finding your agile tribe. In *Agile, 2008. AGILE '08. Conference*, pages 121–124, 2008.

[38] S. Nerur, R. Mahapatra, and G. Mangalaraj. Challenges of migrating to agile methodologies. *Communications of the ACM*, 48(5):72–78, 2005.

[39] M. Paasivaara, B. Behm, C. Lassenius, and M. Hallikainen. Towards rapid releases in large-scale xaas development at ericsson: A case study. In *Global Software Engineering (ICGSE), 2014 IEEE 9th International Conference on*, pages 16–25, Aug 2014.

[40] M. Paasivaara, S. Durasiewicz, and C. Lassenius. Using scrum in a globally distributed project: A case study. *Software Process Improvement and Practice*, 13(6):527–544, 2008.

[41] M. Paasivaara, C. Lassenius, V.T. Heikkila, K. Dikert, and C. Engblom. Integrating global sites into the lean and agile transformation at ericsson. In *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on*, pages 134–143, Aug 2013.

[42] K. Petersen and C. Wohlin. The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering*, 15(6):654–693, DEC 2010.

[43] K. Sagesser, B. Joseph, and R. Grau. Introducing an iterative life-cycle model at credit suisse it switzerland. *Software, IEEE*, 30(2):68–73, 2013.

[44] K. Schwaber and M. Beedle. *Agile software development with scrum*. Series in agile software development. Prentice Hall, 2002.

[45] J. Scott, R. Johnson, and M. McCullough. Executing agile in a structured organization: Government. In *Agile, 2008. AGILE '08. Conference*, pages 166–170, Aug 2008.

[46] M. Senapathi and A. Srinivasan. Sustained agile usage: A systematic literature review. In *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering*, EASE '13, pages 119–124. ACM, 2013.

[47] VersionOne, Inc. 7th annual "state of agile development" survey, 2013.

[48] L. Williams and A. Cockburn. Agile software development: it's about feedback and change. *Computer*, 36(6):39–43, June 2003.

Primary Sources

[P1] J. Abdelnour-Nocera and H. Sharp. Adopting agile in a large organization: balancing the old with the new. Technical report, The Open University, Faculty of Mathematics and Computing, Department of Computing, 2007.

[P2] J. Abdelnour-Nocera and H. Sharp. Adopting agile in a large organisation. In *Agile Processes in Software Engineering and Extreme Programming, Proceedings*, volume 9 of *LNBIP*, pages 42–52, 2008.

[P3] A. Atlas. Accidental adoption: The story of scrum at amazon.com. In *Agile Conference, 2009. AGILE '09.*, pages 135 –140, aug. 2009.

[P4] T.J. Bang. Introducing agile methods into a project organisation. In *Agile Processes in Software Engineering and Extreme Programming, Proceedings*, volume 4536 of *LNCS*, pages 203–207, 2007.

[P5] P.A. Beavers. Managing a large "agile" software engineering organization. In *Agile Conference (AGILE), 2007*, pages 296 –303, aug. 2007.

[P6] G. Benefield. Rolling out agile in a large enterprise. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, page 461, jan. 2008.

[P7] S. Berczuk and Yi Lv. We're all in this together. *Software, IEEE*, 27(6):12 –15, nov.-dec. 2010.

[P8] A.W. Brown. A case study in agile-at-scale delivery. In *Agile Processes in Software Engineering and Extreme Programming, Proceedings*, volume 77 of *LNBIP*, pages 266–281, 2011.

[P9] Mun-Wai Chung and B. Drummond. Agile at yahoo! from the trenches. In *Agile Conference, 2009. AGILE '09.*, pages 113 –118, aug. 2009.

[P10] G. Cloke. Get your agile freak on! agile adoption at yahoo! music. In *Agile Conference (AGILE), 2007*, pages 240 –248, aug. 2007.

[P11] C.L. Cowan. When the vp is a scrum master, you hit the ground running. In *Agile Conference (AGILE), 2011*, pages 279 –283, aug. 2011.

[P12] M. Evans. The fragile organisation. In *Agile, 2008. AGILE '08. Conference*, pages 181–185, 2008.

[P13] A. Farrow and S. Greene. Fast & predictable - a lightweight release framework promotes agility through rhythm and flow. In *Agile, 2008. AGILE '08. Conference*, pages 224–228, 2008.

[P14] J. Fecarotta. Myboeingfleet and agile software development. In *Agile, 2008. AGILE '08. Conference*, pages 135 –139, aug. 2008.

[P15] M. Federoff and C. Courage. Successful user experience in an agile enterprise environment. In *Procceedings of HCI International 2009, San Diego, CA, USA, July 19-24, Part I*, volume 5617 of *LNCS*, pages 233–242, 2009.

[P16] C. Fry and S. Greene. Large scale agile transformation in an on-demand world. In *Agile Conference (AGILE), 2007*, pages 136 –142, aug. 2007.

[P17] I. Gat. How bmc is scaling agile development. In *Agile Conference, 2006*, pages 6 pp.–320, 2006.

[P18] J. Goos and A. Melisse. An ericsson example of enterprise class agility. In *Agile, 2008. AGILE '08. Conference*, pages 154 –159, aug. 2008.

[P19] D.R. Greening. Release duration and enterprise agility. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pages 4835–4841, 2013.

[P20] H. Hajjdiab and A.S. Taleb. Agile adoption experience: A case study in the u.a.e. In *Software Engineering and Service Science (ICSESS), 2011 IEEE 2nd International Conference on*, pages 31 –34, july 2011.

[P21] H. Hajjdiab, A.S. Taleb, and J. Ali. An industrial case study for scrum adoption. *Journal of Software*, 7(1):237–242, 2012.

[P22] M. Hallikainen. Experiences on agile seating, facilities and solutions: Multisite environment. In *Global Software Engineering (ICGSE), 2011 6th IEEE International Conference on*, pages 119 –123, aug. 2011.

[P23] S. Hanly, L. Wai, L. Meadows, and R. Leaton. Agile coaching in british telecom: making strawberry jam. In *Agile Conference, 2006*, pages 9 pp. –202, july 2006.

[P24] M.T. Hansen and H. Baggesen. From cmmi and isolation to scrum, agile, lean and collaboration. In *Agile Conference, 2009. AGILE '09.*, pages 283 –288, aug. 2009.

[P25] K. Korhonen. Evaluating the impact of an agile transformation: a longitudinal case study in a distributed context. *Software Quality Journal*, 21(4):599–624, 2012.

[P26] M. Laanti. Implementing program model with agile principles in a large software development organization. In *Computer Software and Applications, 2008. COMPSAC '08. 32nd Annual IEEE International*, pages 1383–1391, 2008.

[P27] E.C. Lee. Forming to performing: Transitioning large-scale project into agile. In *Agile, 2008. AGILE '08. Conference*, pages 106 –111, aug. 2008.

[P28] A. Leszek and C. Courage. The doctor is "in" - using the office hours concept to make limited resources most effective. In *Agile, 2008. AGILE '08. Conference*, pages 196–201, 2008.

[P29] J. Lewis and K. Neher. Over the waterfall in a barrel - msit adventures in scrum. In *Agile Conference (AGILE), 2007*, pages 389 –394, aug. 2007.

[P30] K. Long and D. Starr. Agile supports improved culture and quality for healthwise. In *Agile, 2008. AGILE '08. Conference*, pages 160 –165, aug. 2008.

[P31] C. Maples. Enterprise agile transformation: The two-year wall. In *Agile Conference, 2009. AGILE '09.*, pages 90 –95, aug. 2009.

[P32] S. McDowell and N. Dourambeis. British telecom experience report: Agile intervention - bt's joining the dots events for organizational change. In *Agile Processes in Software Engineering and Extreme Programming, Proceedings*, volume 4536 of *LNCS*, pages 17–23, 2007.

[P33] R. Mencke. A product manager's guide to surviving the big bang approach to agile transitions. In *Agile, 2008. AGILE '08. Conference*, pages 407–412, 2008.

[P34] E. Moore and J. Spens. Scaling agile: Finding your agile tribe. In *Agile, 2008. AGILE '08. Conference*, pages 121–124, 2008.

[P35] P. Murphy and B. Donnellan. Lesson learnt from an agile implementation project. In *Agile Processes in Software Engineering and Extreme Programming*, volume 31 of *LNBIP*, pages 136–141, 2009.

[P36] J. Nielsen and D. McMunn. The agile journey - adopting xp in a large financial services organization. In *Extreme programming and agile processes in software engineering, Proceedings*, volume 3556 of *LNCS*, pages 28–37, 2005.

[P37] C.P. O'Connor. Letters from the edge of an agile transition. In *Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion*, SPLASH '10, pages 79–84, 2010.

[P38] C.P. O'Connor. Anatomy and physiology of an agile transition. In *Agile Conference (AGILE), 2011*, pages 302 –306, aug. 2011.

[P39] S. Prokhorenko. Skiing and boxing: Coaching product and enterprise teams. In *Agile Conference (AGILE), 2012*, pages 191 –196, aug. 2012.

[P40] P. Ranganath. Elevating teams from 'doing' agile to 'being' and 'living' agile. In *Agile Conference (AGILE), 2011*, pages 187 –194, aug. 2011.

[P41] G. Roche and B. Vasquez-McCall. The amazing team race a team based agile adoption. In *Agile Conference, 2009. AGILE '09.*, pages 141 –146, aug. 2009.

[P42] P. Rodríguez, K. Mikkonen, P. Kuvaja, M. Oivo, and J. Garbajosa. Building lean thinking in a telecom software development organization: strengths and challenges. In *Proceedings of the 2013 International Conference on Software and System Process, ICSSP'13*, pages 98–107, 2013.

[P43] J.J. Ryan and R. Scudiere. The price of agile is eternal vigilance. In *Agile, 2008. AGILE '08. Conference*, pages 125–128, 2008.

[P44] B. Schatz and I. Abdelshafi. Primavera gets agile: a successful transition to agile development. *Software, IEEE*, 22(3):36 – 42, may-june 2005.

[P45] J. Schnitter and O. Mackert. Large-scale agile software development at sap ag. In *Evaluation of Novel Approaches to Software Engineering, 5th International Conference, ENASE 2010*, volume 230 of *Communications in Computer and Information Science*, pages 209–220, 2011.

[P46] T.R. Seffernick. Enabling agile in a large organization our journey down the yellow brick road. In *Agile Conference (AGILE), 2007*, pages 200 –206, aug. 2007.

[P47] K. Silva and C. Doss. The growth of an agile coach community at a fortune 200 company. In *Agile Conference (AGILE), 2007*, pages 225 –228, aug. 2007.

[P48] H. Smits and K. Rilliet. Agile experience report: Transition and complexity at cisco voice technology group. In *Agile Conference (AGILE), 2011*, pages 274 –278, aug. 2011.

[P49] M.K. Spayd. Evolving agile in the enterprise: implementing xp on a grand scale. In *Agile Development Conference, 2003. ADC 2003. Proceedings of the*, pages 60–70, 2003.

[P50] J. Sutherland and R. Frohman. Hitting the wall: What to do when high performing scrum teams overwhelm operations and infrastructure. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1 –6, jan. 2011.

[P51] K.a Vlaanderen, P.a Van Stijn, S.a Brinkkemper, and I.b Van De Weerd. Growing into agility: Process implementation paths for scrum. In *Product-Focused Software Process Improvement, Proceedings*, volume 7343 of *LNCS*, pages 116–130, 2012.

[P52] Lv Yi. Manager as scrum master. In *Agile Conference (AGILE), 2011*, pages 151 –153, aug. 2011.