Challenges in Transitioning to an Agile Way of Working

Hekkala, Riitta; Stein, Mari-Klara; Rossi, Matti; Smolander, Kari

Document Version
Final published version

Published in:
Proceedings of the 50th Hawaii International Conference on System Sciences (HICSS) 2017

DOI:
10.24251/HICSS.2017.707

Publication date:
2017

Challenges in Transitioning to an Agile Way of Working

Riitta Hekkala
School of Business
Aalto University,
Helsinki, Finland
riitta.hekkala@aalto.fi

Mari-Klara Stein
Copenhagen Business School
Department of IT management,
Frederiksberg, Denmark
mst.itm@cbs.dk

Matti Rossi
School of Business
Aalto University,
Helsinki, Finland
matti.rossi@aalto.fi

Kari Smolander
Department of Computer Science
Aalto University,
Helsinki, Finland
kari.smolander@aalto.fi

Abstract

This longitudinal study examined how an information systems development team transitioned to an agile way of working. We describe the main events of a large, inter-organizational project, where agile methods and practices were applied for the first time. The organizations involved had a long tradition in heavy, waterfall style projects, and many of those past projects had severe challenges. We examine how the agile way of working was understood by particular groups (project team, management and suppliers), as well as how these understandings changed over time. The lack of experience with agile development, no common view on 'agility' and its key principles and practices were obvious challenges for the transition. Our study suggests that complex agile projects need to have very clear goals and management has to be able to communicate these, while preserving the autonomy of teams and individual team members.

1. Introduction

During the past decade a number of studies have been conducted on agile software development [e.g. 1, 2, 3]. Extant literature in the information systems (IS) field focuses particularly on three different perspectives on agility: 1) as empirically validated software development methods and practices; 2) as an organizational capability to learn, to explore and exploit knowledge; and 3) as 'collective agility' which is seen as a performance of daily practices by social actors (cf. [4]). This paper is positioned within the first perspective, focusing on the "specific needs of organizations and human nature [that] inevitably lead to diverse interpretations and implementations of a method, which in turn lead to different, sometimes surprising, effects and consequences of use of agile methods and associated practices" [5].

There have been calls for studies that investigate the influence of organizational culture and environmental constraints on agile development [e.g. 5], as well as how and why organizations select agile approaches for managing and delivering IS projects [6]. In this paper we describe events in a large, inter-organizational project, where agile methods and practices are being applied for the first time. Our goal is to identify the challenges of transitioning to an agile way of working through a longitudinal study of a case project. We identify management challenges that inevitably arise, when an organization with a long tradition in waterfall-style development wants to develop new systems in a more dynamic way. We look especially at choices made by the management to introduce the new approach and how these resonate with the people working in the project and their different backgrounds. As the project has had challenges in the transition, we identify issues and conflicts that appear during the transition.

In sum, this qualitative case study research is guided by the following questions: *How do information systems development (ISD) teams transition to an agile way of working? What are the organizational and managerial challenges of this transition?*

This study contributes to calls for better understanding of the influence of organizational culture and environmental constraints on changing development methods, and the role of organizational-level implementation of 'agility' in ISD environment. Our longitudinal study is particularly suitable for this as it enables us to investigate the trajectory of a project and the groups (project team, management and suppliers) within it, and shows how a project organization learns to work in an agile way.

The rest of this paper is organized as follows. In the next section, we present the definition of agility and the basic principles, processes and challenges of agile software development. The following three sections present the research case, the research method and our

URI: http://hdl.handle.net/10125/41871
ISBN: 978-0-9981331-0-2

findings. In the final sections, we discuss our findings, and conclude the paper.

2. Theoretical Background

In the following we consider how agility is defined and describe the key aspects of agile development methods.

2.1. Definition of Agility

Seventeen software developers [7] published the Agile Manifesto in 2001 – in a nutshell the idea was to present "better ways of developing software by doing it and helping others do it". The Agile Manifesto includes four values: individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan. These values are manifested in twelve principles: 1) Highest priority is customer satisfaction; 2) Welcome changing requirements; 3) Frequent delivery of software; 4) Business people and developers cooperate daily; 5) Build projects around motivated people; 6) Face-to-face conversation is best; 7) Progress is measured by working software; 8) Sustainable development pace; 9) Continuous attention to technical excellence; 10) Simplicity; 11) Self-organizing team, and 12) Regular reflection and adaptation.

Since 2001, agility has been a multidimensional concept, and interpretations of it abound (e.g. [5, 8, 9]). Often highlighted key aspects of agility include quickness, nimbleness (e.g. [10]), lightness, responsiveness to changes [11], and keeping the code short [12]. However, while agility implies speed – being fast does not imply being agile (e.g. [13]). In the context of ISD, agility can be defined as organization's ability not only to sense, but to respond swiftly [10] and flexibly [12, 14] to technical changes, new business opportunities and unexpected environmental changes. While there is some agreement with regard to the conceptual principles of agile development, there is much room for interpretation when applying these principles in practice [5], as considered next.

2.2. Agile Software Development: Principles, Processes and Challenges

In order to respond swiftly to turbulent business environments and technical changes (e.g. [10, 15-17]), an agile ISD project typically requires the following three principles and processes to function properly: (1) self-organizing teams, distributed leadership and decision-making, (2) incremental and iterative development, and (3) a supporting organizational culture. Table 1 summarizes each of these.

While these principles are useful and widely followed in practice, they come with many challenges. For example, 'just enough planning', lack of upfront commitment to scope, cost and schedule (key aspects of iterative and incremental development) may pose very challenging demands on managers who are responsible for making funding decisions [15].

Table 1. Key Principles and Processes of Agile

Key Principles & Processes of Agile	Description
Self-organizing teams; distributed leadership & decision-making	1) Rather than being guided by others outside the team, a self-organizing team aims to choose the best way to accomplish their duties and work; 2) Leadership is divided between team-members, who should be able to make decisions collectively. Leadership is usually given to the person who has the key knowledge and skills for specific issue(s); 3) The role of project manager is to ensure that people with key knowledge are able to affect decisions (cf. [18, 19])
Incremental and iterative development (planning as you go along)	Self-organizing, agile teams (in this case, Scrum[1] teams) develop software in increments done in iterations, called "sprints", to optimize predictability. In addition to the Scrum team(s) and their associated roles (Scrum master, product owner, development team), the Scrum framework consists of specific events (e.g. planning, review, and backlog). The product owners have the power to decide which backlog items should be developed in the following sprint (e.g. [14]).
Supporting organizational	Organizational culture should support the key principles and

[1] We describe the incremental and iterative development principles of Scrum, which is one of the many agile development frameworks (and the one adopted in the project we studied).

culture (responding to change over following a plan)	values of agile ISD (especially the shift of power from management towards the team), because agile ISD requires a very different organizational culture than traditional, so-called plan-driven methods (e.g. [5, 20]).

Self-organizing teams and iterative development often work well within small teams and when "championed by a small number of highly effective people". However, many teams find it difficult to "implement agile beyond their own boundaries. As a result, they are constrained by many of the functions they are dependent on to get work done" [5]. In sum, while the agile way of working has the potential to improve software development outcomes [21], there is a lack of in-depth empirical studies of ISD projects transitioning to an agile way of working that are able to shed light on how the theory of agile works in practice (cf. [2, 5]). In addition, there is a lack of studies that give evidence-based guidelines to project managers on, for example, how to maximize the benefits from an agile way of working during an IS project [2].

3. Methodology

We studied the development of a planned records management system (RMS) in four public sector organizations (Alpha, Beta, Gamma and Delta) in Northern Europe. The goal of the new RMS was to provide a centralized means of collecting customer information. In addition, the system should also facilitate the dissemination of certain information back to the customers, as well as offer web-based self-service capabilities. All four organizations were from the same sector, so their requirements for the system were fundamentally the same. Alpha, Beta and Gamma decided to develop the new system together because of budgetary constraints.

Because of the different financial situations of the organizations, the project expenses were not divided equally: Alpha was expected to cover 30%, Beta 50%, and Gamma 20% of the costs. The current project organization was established in early 2013. Delta joined the project in 2015. The project organization consisted of the project group, steering group and management group, each including representatives of the four organizations. Later on, several external software houses also joined the project. The transition to an agile way of working in the project was largely triggered by cost considerations – the management group believed that agile was a natural fit with time-and-material-basis contracts, allowing consultants hired by the project to be paid based on the hours they spent on the project, rather than a fixed amount. However, the IT managers also highlighted that "extremely agile is difficult (e.g., it is difficult to manage work when programmers have the freedom in choosing their tools)", especially in a situation where the project members are not familiar or experienced with an agile way of working.

3.1. Data Collection and Analysis

Our data consists of 42 qualitative interviews, collected in three phases: (1) March 2013 - April 2013, (2) May 2014 - June 2014, and (3) May 2015 - August 2015. The project is forecasted to end in 2017/2018.. All interviews were recorded and fully transcribed. For the purposes of this paper, we have only included interviews with the management and project groups as well as various suppliers, as these were the key actors involved in the transition to agile. Overview of the interviews conducted is given in Table 2.

We began our data analysis by identifying all descriptions related to an agile way of working in the interviews. After we had coded the data for 'agility', we then focused on the key principles and processes of agile (Table 1) and followed the transition to agility along the trajectory of the IS project.

In order to do so, we considered how the agile way of working was understood by particular groups (project team, management and suppliers), and how these understandings changed over time (from 2013 to 2015 or covering the requirements specification, design and implementation phases identified by the project manager Alex). The labels of the project phases (as described by Alex) reflect the significant influence of the waterfall method (familiar to the team from previous projects), particularly in the beginning of the project. In addition, we identified the key events that influenced the transition. As a final step, we coded for the different challenges encountered during the transition.

Table 2. Overview of Data Collection

Group	Role	Member	# of Interviews
MANA-GEMENT GROUP (16 interviews altogether)	The members of the management group decide on all personnel and budgeting issues. They guide the project group and define general policies. It is also the duty of the management group to take a stand on issues, which the project group is not able to solve.	Lily (Beta)	3
		Kelly (Alpha)	1 (left the project in June 2013)
		Leslie (Alpha)	1 (started in June 2013, substituting for Kelly)
		Leon (Gamma)	3
		Ewan (Alpha)	3 (Left the project at the end of 2015)
		Ben (Beta)	3
		Sean (Gamma)	2 (Left the project in 2014)
PROJECT GROUP (19 interviews altogether)	The aim of the project group is to find possible technical solutions for the new system and to make sure that the processes are defined and done by people who know the substance well. The group has software developers (SD) and representatives of users. Alex is the overall project manager. He was hired externally to run the project, but is now paid by Alpha.	Alex (Alpha)	3
		Isaac (Gamma)	2
		Carol (Alpha)	3
		Jacob (Beta)	1 (left the project in Sept. 2013)
		Amber (Beta)	2 (left the project in Sept. 2013)
		Nathan (Beta)	1
		Chloe (Alpha)	3
		Nicole (Beta)	2 (was on longer leave in 2015)
		Wendy (Beta)	1 (Substituting Nicole)
		Philip (Delta)	1 (Started in 2015)
SUPPLIERS (7 interviews altogether)	*Omicron* is an agile software house founded in 2005 (consists of about 20 people). Omicron has an agreement with Beta.	Robert (SD) Tom (SD)	2 (Started in April- May, 2014) 1 (Started in 2014)
	Midén develops digital business solutions. The staff consists of about 50 people. Midén has an agreement with Alpha.	Samuel (SD) Justin (SD, scrummaster)	1 (Started in March 2015; left in March 2016) 1 (Started in May 2015)
	Déka is a global design firm founded in 1999. The staff consists of about 150 people. Déka has an agreement with Beta.	Amanda (user interface designer)	1 (Sept. 2014, on longer leave from June 2015)
	Ekatón is a software architecture company (consists of about 100 people). Ekatón has an agreement with Alpha.	Tobias (user interface designer)	1 (Started in August 2014)

4. Findings

In the following we present our findings through the analysis of the key aspects of transitioning to an agile way of working: transitioning to self-organizing teams and distributed leadership; to incremental development, and to an agile organizational culture.

4.1. Learning How to Guide Self-Guided Teams

Because of the lack of experience with agile methods (training only took place in 2014), the members of the project faced difficult problems right from the initiation of the project in 2013. For example, project team members expected clear and precise instructions, while the manager wanted them to self-organize: *"Nicole is the person who needs exact guidelines for work. She belongs to the 'old school', where the boss gives the exact tasks, and says that you'll do this today and these ones tomorrow…*" (Alex, project manager). Meanwhile, the management group was worried that people in the project did not only lack experience with agile, but experience with successfully completing projects at all: *"The challenge is that many people in this organization haven't done a systematic, target-oriented project work, they don't have experience. When I came here three years ago, and I of course discussed with people and asked from one person, whose title is a project manager, 'When did you finish the latest project and what kind of project was it?' And the response was that 'I haven't done the project, or s/he wasn't able to show any project that had ended, so s/he wasn't able to tell how they succeeded'*." (Ewan, IT manager).

Even if the people in the project had a long history of working on what they saw as projects, even in managerial roles, they had been working more in a process that develops single software features at a time. Furthermore, several key decisions, such as architectural and technology choices, that would have allowed agile practices to be followed, were not fixed early enough in the project. In particular, there were radically different opinions about key technological responsibilities and choices – the IT managers (management group) and the software designers (project group) did not agree on which technology was the best one for the project, nor whose responsibility it was to make such choices.

As a result, the software designers worked on technical issues for several months, while the IT managers went ahead and commissioned a solution from Omicron (an external software house). The software designers were so insulted by this that they left the project altogether (in Sept. 2013). "*When I go to the seminars of this project group, people are talking about the agile way of working… but it is totally different to say on PowerPoint that we are working the agile way, and we trust on experts, if we are not doing so in practice …* […] *I looked at the minutes of the management group meeting, and they have drawn some architecture pictures at the meeting. I do not think that it is the duty of the management group to do them, but it is more the duty of experts. The management group should do bigger strategic alignments…*" (Amber, project group).

The transition to the design and implementation phase (from 2014 forward) was accompanied with changes in the project organization (new personnel in the project group, a new supplier). People also received training on agile methods (Scrum) to avoid further conflicts: "*When the row was over, there was training about agile methods for the other project members. […] Anyway, it convinced us that the agile way is the right way…*" (Alex, project manager). After the training, the project group and software developers of Omicron established Scrum roles as defined in the Scrum guide (product owner, development team, scrummaster). Some project members still lamented the lack of daily leadership and someone clearly telling them how to proceed: "*In one team meeting Alex said that his duty is not to take care of daily leadership … well, I could criticize that he hasn't hired a person who tells us how we should go ahead… I guess that different product owners think differently about this […] it is stated that the project is self-guided [laughing]. It's tragicomic at times. I don't feel that I'm in safe hands*" (Nicole, project group).

At the same time other project members were happy with the new way of doing things, but found it difficult to work together with people with different interpretations of agile. "*We [Alpha people], we think that the developers can decide within frames that have been given to us [project members]. So there are clear frames in an agile project as well, and there is freedom to do issues within these specific frames… but still we face situations with Beta people that they think an agile project is a project where nothing can be decided beforehand… that we can't guide them, for example if we [me and Chloe] discuss with Nicole, the discussion always ended in that user interface designers will decide specific issues and software developers decide specific issues, and product owners [like me and Chloe], we are just twiddling our thumbs beside them…*" (Carol, project member).

As the project progressed (2015), tensions between wanting to go agile and wanting to have a clearly controlled frame became increasingly discussed in the management group: "*It's important that we don't do*

things for two months and then check if something works or not. I also think that although we are working in an agile way, we need to anyway have some scope, roadmap and goals. Some project members thought that it is like a blue-sky way, and you just start … well, there is a need to have a clear frame" (Lily, management group). This quote again highlights the importance of scope and architecture to be in place to allow for concentration on daily and weekly tasks without individual project team members being concerned about the progress in global development.

By this point, both Alpha and Beta had hired their own external software houses (Midén and Omicron), but this unfortunately created a situation of two self-organizing teams, with the idea that they will work in parallel on different issues. The developers from Midén started in the project later and were not satisfied with the work that Omicron had done, suggesting to start over with a clean slate. Of course Omicron defended their work and the importance of continuity in the project. The standoff ended with one developer from Midén being fired from the project, however, the overall problem of parallel development (without much coordination) has yet to be resolved.

The transition to an agile way of working changed the control relationships between managers, supplier and project team members. While the project team members were controlled by two IT managers from their respective organizations [Alpha and Beta] at the beginning [2013], the transition to the "design and implementation phase" [2014] changed this situation. The scrum master from Omicron was perceived as the new controller of the project team members: "*Robert, our scrum master has taken a role of daily leader, he is bossing us and saying that you should think about this kind of issue now…*" (Nicole, Beta).

4.2. Learning How to Make Small Increments

In the beginning of the project (2013), the project management and the team lacked experience with agile methods. Thus it is not surprising that they were not following the agile principles for incremental development at all in this phase. While on paper the project was framed as agile, in practice this had yet to manifest. Given the level of conflict that had developed between the software developers and managers, the developers started to work in secret: "*I started to work on things in secret… and then when I've finished something, the project members have said that, 'Well, it's nice', and I myself thought that, 'Oh surprise, why didn't you think of it earlier'. I felt that I had to fight about everything, it's really frustrating…*" (Jacob, Beta). As already noted, two key software designers (Amber and Jacob) left the project altogether in September 2013.

The training received in 2014 was intended to help the project not only in terms of transitioning to a new way of self-organizing and distributed leadership, but also to a new way of doing incremental development. In the training on Scrum the project team members learned about sprints, backlogs, product owners, and the like. External software developers already familiar with Scrum (and other agile frameworks) joined the project. Various processes, techniques and tools (e.g. Kanban & Jira) became increasingly employed in the project in this phase. As practical working experience with incremental development grew (2015), the daily challenges the project team faced became more nuanced. Issues, such as the misuse of sprints to make it appear that development is progressing faster than it actually is, and lack of visibility (of what other members are doing) emerged:

"*I would like to see how fast we are able to solve problems. I think that the product owners look very much at Jira [a bug and issue tracking as well as a project management software]. But it has been hard to get information for example about what my team members [software developers] are doing although we are sitting in the same room*." (Justin, scrummaster, Midén).

"*The challenge is that in Scrum there are these story points, which tell how much you have achieved in two weeks. I think that it would be more sensible to use this in a way that we would not measure how much we achieved, but as an evaluation tool of how much we are able to do in the next sprint. The idea of the sprint has turned out to be more of a negative for our project. You can 'manipulate' things in Scrum so that it looks buoyant … there is often a situation that people think that it is sensible to add as many issues as possible into one sprint, but the problem is that they will not just be done by magic…*" (Robert, scrummaster, Omicron). Philip (Delta) summarized the challenge as "cherry picking": the software developers did not necessarily use Scrum as it was originally planned for agile projects. Instead they were just using the parts of Scrum that served this specific operational environment.

4.3. Hierarchy Prevails: Difficulties of Cultural Change

From the beginning of the project, the management group was aware that despite the desire to follow an agile way of working, the project would face challenges stemming from public procurement regulations. The procurement law and the EU directive dictate that in public procurement tendering documents

must fully specify the artefact to be procured, yet this is seldom (if ever) true for tailored IT solutions, especially when agile methodologies are used to find the best solution for the customer. In addition to external constraints, the project was set up in a highly hierarchical fashion (management, steering and project groups), creating the expectation among the project group that they will be led by the project manager (Alex) and the management group. Yet, for some project team members the experienced leadership style was neither clearly directive nor facilitative of self-management: "*It was one of the big reasons that I wanted to leave the project, because the style of leadership of Alex was so odd… he is not leading the team but rather serving the management group, and the communication, he doesn't say 'That we could do so or what do you think', but he says that 'The management group says that we need to do it this way…', and then when we criticized some issue like 'What? Why this way?...', he says that, 'Well, I don't know, but the management group said so…*' "(Amber, project group). Furthermore, while the project organization brought together people from different organizations, it failed to create a feeling and culture of a new collective. Project team members from Alpha and Beta kept to themselves, and communicated with the management group via managers from their home organizations. Tensions along the organizational lines were exacerbated when the first external software house was hired, but with a contract only with Beta. This was obviously not very good for communication and Scrum practices in general.

As the project progressed (2014), such tensions continued, leading to a lack of transparency in communication and the suspicion of hidden agendas: "*We don't have our own staff, but we have people from three different organizations, who try to work together. And because all plans are not transparent, the consequences are that the content of planning is sometimes poorly seen… so it leads to the feeling that people have hidden agendas… and the most probable reason is that there has not been time for it [to talk about issues], or it just hasn't come to mind that people should know about these things …*" (Alex, project manager). Several more software houses joined the project at this stage; each of them having an agreement with a specific user organization, rather than the project. By 2015, it was increasingly clear to all working on the project that agile principles were challenging to implement when the operational environment (user- and project organization) was not facilitative of working in the agile way. There was and still is a continued tension between 'conservative' user organizations and the 'agile' project: "*Our own organization is very conservative and slow, and we always have to have very clear plans. When people in our organization are asking about this project, I have to say that I don't know what is going to be ready next spring, what is working and how – it is very difficult…*" (Wendy, Beta). This quote highlights the issue of "definition of done" in agile projects: how to know when the release is ready? We will consider the implications of our findings, and future research avenues next.

5. Discussion

The aim of this study was to explore how ISD teams transition to an agile way of working. We described the main events of a large project, where agile methods and practices were applied for the first time. The organizations participating in the project had a ong tradition in waterfall style projects, and many of those past projects had had severe challenges. Thus, these organizations were lured by 'agility' because of the promise of better results [21] that, for example, self-organizing teams, distributed leadership, incremental development, and the assumed 'better' contracts with suppliers would deliver.

However, an agile way of working has in this case also turned out to be the Achilles' heel of the project instead of a silver bullet. We claim that the diverse interpretations of what agility means (cf. [5]), and the lack of strong vision led to an unstructured approach [3]. We have summarized the key milestones and events in this project's transition to an agile way of working in Appendix 1 (Please find it at the end of this document). The findings reveal insights with regard to the key managerial and organizational challenges in the transition. Table 3 summarizes the issues that we identified in this project, but that we believe to be far more common than one would expect, especially when adopting an agile way of working. In the following we look at the key challenges that we claim to be nearly all cultural and managerial in nature (rather than, for example, technical or specific to the domain of public sector).

Table 3. Challenges in transitioning to agile development

Issue type	Issue
Learning	Agile practices were misunderstood and misused
Managerial	Self-organized teams were not able to proceed with consensus
Managerial/ Cultural	More leadership and guidance was expected by the developers
Cultural	Hierarchical organizational

	structure was not suitable for agile
Managerial	Conflicts were avoided by working in secret
Managerial/ Cultural	Old organizational borders prevailed after adoption of agile
Managerial/ Legal	Agreements were made between individual partners - overall project objectives were not clear
Legal/Domain -specific	Procurement laws were not suitable for agile

To most project members, this was the first agile project in their work history. The lack of experience with 'agility' was an obvious problem and challenge for the transition. Supporters of agile frameworks highlight that changes and learning must take place throughout the project (cf. [10, 22]). Our study suggests that such learning does not happen through training alone. While adopting iterative and incremental development processes and principles was the least problematic aspect of this transition, the surrounding elements of team-work, leadership and culture were much more challenging to address.

This was made worse by the organizational issues: there were three organizations working separately and with very little common training in the new way of working. This led to poor communication and repeated failures in coordinating work. There were severe managerial challenges already at the beginning; the choices made by the management group did not resonate with the ideas of project members working in the project and their different backgrounds. In addition to this, the IT managers did not support the more group intensive approach, as they wanted to control, for example, technological choices. The management group and software developers, thus, had very different conceptions about what self-organizing teams and distributed leadership meant in practice. Some people on the project level made assumptions that agile means anything goes, whereas the management group still needs status reports, even within sprints. At the same time, for some project group members the idea of 'self-guided' work was an uncomfortable experience.

Open communication and meetings that discuss current issues are important for, on the one hand, knowing the status on any working issues and, on the other hand, for building trust among the team members. Building trust is essential for creating a common understanding among the project members on how things are progressing and confidence in that others are working in the same pace and with the same goals in mind. This idea was violated seriously in this case with secret sub-projects and parallel developments that were not coordinated in any way.

This lack of coordination was made worse by two further managerial issues: first, individual partners procured work from different external software houses (without coordination); and, second, lack of clear central project objectives and architecture, within which self-organizing teams could thrive. These two issues, together with lack of clear leadership, can be seen as the main causes of the seemingly chaotic work and unsatisfactory outcomes that we have observed here. Our impression is that these managerial issues are especially challenging in large-scale agile adoption. Many simultaneous autonomous teams require skilled coordination and cross-cutting concerns (such as architecture) require careful governance, which may be seen as "non-agile".

6. Conclusions

As is evident from our findings, most of the challenges seem to stem from an organizational conflict between the assumptions made and beliefs held by the management and the developers. This is in no way made easier by outsourcing to several partners. We believe that these kinds of challenges are quite typical for modern software development. In addition to external constraints, the project itself was set up in a highly hierarchical fashion (management, steering and project groups). One challenge stemming from the hierarchical organization was the fact that negotiations of contracts took a long time and software houses joined the project gradually, each time creating the need to re-organize teams and tasks. In sum, our data suggests that large, complex agile projects need (1) very clear high-level objectives, and (2) architecture and management controls derived from those. A challenge for management is to be able to communicate high-level objectives and overall architecture, while preserving the autonomy of the teams and individual team members. Further research is needed to better understand how the transition to an agile way of working changes the dynamics of control and power relations, and the kinds of consequences this has (cf. [22]).

References

1 Fitzgerald, B., Hartnett, G., and Conboy, K.: 'Customising agile methods to software practices at Intel Shannon', European Journal of Information Systems, 2006, 15, (2), pp. 200-213

2 Maruping, L.M., Venkatesh, V., and Agarwal, R.: 'A control theory perspective on agile methodology use and changing user requirements', Information Systems Research, 2009, 20, (3), pp. 377-399

3 Robinson, H., and Sharp, H.: 'The characteristics of XP teams': 'Extreme programming and agile processes in software engineering' (Springer, 2004), pp. 139-147
4 Zheng, Y., Venters, W., and Cornford, T.: 'Collective agility, paradox and organizational improvisation: the development of a particle physics grid', Information Systems Journal, 2011, 21, (4), pp. 303-333
5 Abrahamsson, P., Conboy, K., and Wang, X.: '"Lots done, more to do": the current state of agile systems development research', 2009
6 Wells, H., Dalcher, D., and Smyth, H.: 'The adoption of agile management practices in a traditional project environment: An IT/IS Case Study', in: 'Book The adoption of agile management practices in a traditional project environment: An IT/IS Case Study' (IEEE, 2015, edn.), pp. 4446-4453
7 Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., and Jeffries, R.: 'Manifesto for agile software development', 2001
8 Holmström, H., Fitzgerald, B., Ågerfalk, P.J., and Conchúir, E.Ó.: 'Agile practices reduce distance in global software development', Information Systems Management, 2006, 23, (3), pp. 7-18
9 Sarker, S., and Sarker, S.: 'Exploring agility in distributed information systems development teams: an interpretive study in an offshoring context', Information Systems Research, 2009, 20, (3), pp. 440-461
10 Lyytinen, K., and Rose, G.M.: 'Information system development agility as organizational learning', European Journal of Information Systems, 2006, 15, (2), pp. 183-199
11 Highsmith, J.A.: 'Agile software development ecosystems' (Addison-Wesley Professional, 2002. 2002)
12 Anderson, D.J.: 'Kanban' (Blue Hole Press, 2010. 2010)
13 Börjesson, A., Martinsson, F., and Timmerås, M.: 'Agile improvement practices in software organizations', European Journal of Information Systems, 2006, 15, (2), pp. 169-182
14 Dybå, T., and Dingsøyr, T.: 'Empirical studies of agile software development: A systematic review', Information and software technology, 2008, 50, (9), pp. 833-859
15 Cao, L., Mohan, K., Ramesh, B., and Sarkar, S.: 'Adapting funding processes for agile IT projects: an empirical investigation', European Journal of Information Systems, 2013, 22, (2), pp. 191-205
16 Cao, L., Mohan, K., Xu, P., and Ramesh, B.: 'A framework for adapting agile development methodologies', European Journal of Information Systems, 2009, 18, (4), pp. 332-343
17 Vidgen, R., and Wang, X.: 'Coevolving systems and the organization of agile software development', Information Systems Research, 2009, 20, (3), pp. 355-376
18 McAvoy, J., and Butler, T.: 'The role of project management in ineffective decision making within Agile software development projects', European Journal of Information Systems, 2009, 18, (4), pp. 372-383
19 Moe, N.B., Dingsyr, T., and Kvangardsnes, O.: 'Understanding shared leadership in agile development: A case study', in: 'Book Understanding shared leadership in agile development: A case study' (IEEE, 2009, edn.), pp. 1-10
20 Hummel, M., and Epp, A.: 'Success Factors of Agile Information Systems Development: A Qualitative Study', in: 'Book Success Factors of Agile Information Systems Development: A Qualitative Study' (IEEE, 2015, edn.), pp. 5045-5054
21 Hastie, S., and Wojewoda, S.: 'Standish Group 2015 Chaos Report-Q&A with Jennifer Lynch', Retrieved, 2015, 1, (15), pp. 2016
22 Mahadevan, L., Kettinger, W.J., and Meservy, T.O.: 'Running on Hybrid: Control Changes when Introducing an Agile Methodology in a Traditional "Waterfall" System Development Environment', Communications of the Association for Information Systems, 2015, 36, (1), pp. 5

Appendix 1. Overview of Project's Transition to an Agile Way of Working

SELF-ORGANIZING TEAMS; DISTRIBUTED LEADERSHIP & DECISION-MAKING	- Leadership was entirely centralized in the hands of IT managers from the management group (even though on the surface the developers were given the task of finding the right technology)	- People received training on agile and different ideas on what self-organizing teams and distributed leadership mean emerged; establishment of Scrum roles (product owner, development team, and Scrum masters). Project members of user organizations started to call themselves product owners	- Agile principles are increasingly discussed by the management group and there are continued tensions between wanting to do agile and wanting to have a clear, controlled frame for the project. - The benefits and disadvantages of distributed leadership start to emerge - Power relationships change in many ways
INCREMENTAL DEVELOPMENT	-Development is happening in an incremental way in theory, not in practice - No common idea what incremental development is.	- New software developers and user interface designers join the project (from many software houses). People start to learn and follow Scrum framework, and employ various processes and techniques (e.g. Kanban & Jira)	-Project members follow Scrum framework: Scrum events: the Sprint (planning, reviews, retrospectives) + Scrum artifacts (product backlog/sprint backlog) - The benefits and disadvantages of Scrum framework emerge in daily project work
SUPPORTING ORGANIZATIONAL CULTURE	- Public procurement regulations, EU directive and hierarchical organizational culture in user organizations set challenges to following an agile development style. - A software house joins; has an agreement only with Beta.	- Project culture: The communication between managers and project members, and even inside organizations [e.g. Beta] is not open - Plans are not transparent ('hidden agendas') - Several software houses join – each of them has an agreement with a specific user organization (not with the collective project)	- Conservative and 'slow' user organizations (from software house perspective). - New software houses join the project also in 2015; a new user organization join in 2015 – this keeps 'resetting' whatever project culture has been established. - A key manager [Ewan] of the whole project leaves the project –>instability
	Requirements phase (2013); Data collected: March, April 2013.	**Design and Implementation phase (2014); Data collected: May, June 2014.**	**Design and Implementation phase (2015); Data collected: May, June, August 2015.**