Using Scrum in Distributed Agile Development: A Multiple Case Study

Conference Paper · July 2009

DOI: 10.1109/ICGSE.2009.27 · Source: DBLP

3 authors, including:

Maria Paasivaara
LUT University
91 PUBLICATIONS **3,381** CITATIONS
SEE PROFILE

Casper Lassenius
Aalto University
134 PUBLICATIONS **5,116** CITATIONS
SEE PROFILE

2009 Fourth IEEE International Conference on Global Software Engineering

Using Scrum in Distributed Agile Development: A Multiple Case Study

Maria Paasivaara, Sandra Durasiewicz and Casper Lassenius
Software Business and Engineering Institute
Helsinki University of Technology
P.O.Box 9210 FIN-02015 TKK, Finland
Firstname.lastname@tkk.fi

Abstract

Distributed agile development (DAD) has received increasing interest both in industry and academia as global software development (GSD) is becoming mainstream. However, agile methods and in particular agile practices have been designed for collocated software development, and are thus not directly applicable to DAD. In this paper, we present findings from a multiple case study on agile practices in two small and one mid-sized distributed Scrum project. Based on an interview study of 19 project team members, we describe how Scrum practices, such as daily scrums, backlogs, and sprints were successfully adopted to distributed development. We also describe supporting GSD practices employed, such as frequent visits and multiple communication modes that the projects used. Finally, we depict the challenges and benefits the case projects reported, as well as lessons learned from applying Scrum in distributed settings.

1. Introduction

Global software development (GSD) including outsourcing, subcontracting and partnerships has become a common business reality [29]. GSD offers many potential benefits, e.g., reduced development costs, but also creates significant challenges with respect to communication, coordination, and control [1]. Several of these challenges have been identified [7][17][21], and solutions have been proposed, i.e., to reduce intensive collaboration between sites [5], and to minimize interdependencies between modules being developed at different sites [9][12].

However, many of the existing solutions are based on the assumption of stable requirements. This makes it possible to develop a clear modular structure and minimize communication. The current dynamic business environment requires projects to work with uncertain requirements and implementation technologies [26]. As a consequence, software development organizations have attempted to apply agile development to GSD [1], as agile methods are particularly suitable for projects facing high uncertainty [6]. Due to the physical separation of development teams in GSD, many of the key assumptions within agile development, with respect to, e.g., customer interaction, team communication, and being face-to-face [35], do not hold. To gain the benefit from agile development, the practices need to be modified when applied to distributed settings. In addition, supporting practices known from GSD might be needed.

Previous case studies [10][11][24][32] have shown that agile methods such as Scrum [31] and XP [2] can be successfully customized to distributed projects. Distributed versions of agile methods like Scrum [33] and XP [18] have also been developed.

Nevertheless, scientific research on pairing agile software development and GSD, also referred to as distributed agile development (DAD), is scarce. There are only a few reported experiences in applying DAD to industrial projects, e.g, [34][36] and even fewer case studies, e.g, [20]. However, in our experience, many companies are interested in taking DAD into use, or have already started to use it.

In this paper, our goal is to shed additional light on DAD by presenting the results of a multiple case study on applying Scrum in distributed settings. Our data is based upon semi-structured interviews in three case organizations. We describe both how agile practices have been tailored for adoption to distributed development, as well as supporting GSD practices employed, and lessons learned.

Following this introduction, we present practices reported in the literature for using agile practices in distributed projects. Then, we describe our case study method and our findings. Finally, we discuss our findings and suggest future research topics.

2. Previous work on distributed agile development

Distributed software projects with volatile requirements and uncertain implementation technologies need effective practices to be organized and managed successfully. We performed a literature review on practices used in distri-

978-0-7695-3710-8/09 $25.00 © 2009 IEEE
DOI 10.1109/ICGSE.2009.27

buted agile software development projects. Based on our findings, we divided the practices into two groups: 1) agile practices, and 2) supporting global software development practices. Agile practices include practices that come from some agile method and are either used as such or adopted for use in distributed projects. Supporting global software development practices include practices that help tackle the distributedness of DAD.

2.1 Agile practices

The literature offers some advice especially on how to use Scrum and XP practices in distributed agile projects. Frequently used practices in Scrum related projects have been summarized in Table 1.

Table 1. Agile practices

Name	Description	References
Distributed daily Scrums	For coordination and communication. Scrum questions should be answered before the meeting via email to overcome language problems. Skype and professional web conferencing tools can be useful.	[3][8] [16][28] [29][32] [33]
Sprint planning meeting	Meeting at the beginning of each iteration to decide what to develop in the iteration. If difficult to schedule due to time-zone differences, only lead-developers can meet. As many issues as possible should be clarified before the meetings to keep them short.	[3][14] [20][32]
Review meeting	The team, including the remote site presents and discusses the progress of the project. Can be centered on the local team and product owner to minimize the amount of remote meetings.	[3]
Demonstrations of working functionality	After each iteration, new functionality is presented usually as part of the sprint review meeting. They might be held using videoconferencing and/or desktop sharing.	[3][8] [11]
Proxy/ Remote customer	A business/software analyst interfaces with the customer on the other site. The proxy customer can make decisions on behalf of the real customer. A remote customer communicates with the team through videoconferencing or email. The customer can also receive prototypes that he can test and comment upon.	[18][20] [24][32]
Distributed Scrum of Scrums	For bigger teams. Only the Scrum masters meet, either in person or via tele- or videoconference. The Scrum masters meet every 2-3 days using tools similar to daily scrum meetings.	[16][34] [33]

2.2 Supporting GSD practices

Besides agile practices, the literature on DAD reports practices that tackle some of the new challenges arising from DAD. Not only agile practices aid teams in DAD development, but known GSD practices can also help a team work effectively in DAD. Challenges caused by distribution include, for example, communication problems [3][18][28][29][32][33], lack of close physical proximity [1][14][36], lack of team cohesion [29], lack of shared context and knowledge [32][36], unavailability of team members [36], and cultural differences [17][21].

The supporting distributed practices strive to provide solutions to these challenges. Commonly used supporting practices are summarized in Table 2.

Table 2 Supporting GSD practices

Name	Description	References
Frequent visits	Used to build and maintain trust and enhance collaboration. *Seeding visits* early in the project aim at building a relationship. *Maintaining visits* are shorter and aim at maintaining the collaboration. Team members should continually rotate between sites and at least one team member at a time should be visiting.	[4][8] [11][29]
Multiple communication modes	Several different kind of communication should be available that can also be applied in parallel, i.e., individual and conference telephone, teleconference, videoconference, email, instant messaging, Wiki and desktop sharing	[4]
Mirroring/ balanced sites	Reduce the dependency between sites. Each role in a team at one site has a counterpart on the other site..	[4][13]
Ambassador/ rotating guru	Experienced engineers are sent to the other site for a longer period of time. Ambassadors report lessons learned and set future directions for the projects. Rotating gurus provide initial training and mentoring to the other site.	[4][13] [24]
Synchronization of work hours	Maximization of overlapping work hours to ensure constant communication. For example, early morning shifts for one site and late evening shifts for the other site.	[29]

3. Research method

The research presented in this paper is a multiple-case study [37] of three globally distributed software development projects using Scrum. We chose a multiple-case study as multiple sources of evidence provide a better validity for the findings [30][37]. We used purposeful sampling [27], looking for globally distributed software development projects that had experience in using some agile method or at least a collection of agile practices. We contacted large Finnish companies that we knew used agile methods in their distributed projects. All the companies were highly

interested in improving their agile processes, and in particular hearing how other companies have applied agile methods in their distributed projects. Company contact persons identified candidate projects, from which we selected one project from each company. Only one company chose not to take part in our study, because their suggested project had only recently started using Scrum and their project manager said that they were still in the early learning phase. All chosen projects had mainly applied practices from Scrum. Thus, even though we were not purposefully searching for projects using Scrum, the fact that all selected projects used the method, made it possible to compare Scrum usage in three different projects, each coming from a different company.

We collected data using semi-structured interviews [27] that were recorded and later transcribed by an outside professional transcription company. Altogether we performed 19 interviews each lasting 1.5-2.5 hours. In each project, we interviewed product owners, scrum masters, developers, and testers. We conducted mostly face-to-face interviews with one researcher asking questions and the other one taking notes. Because of our limited traveling budget we could visit only the Finnish sites, and interviewed offshore personnel during their visits to Finland. In one case project, EnergySoftware, we had the possibility to interview one offshore team member while she was visiting Finland, and two additional offshore team members via SkypeOut calls. One researcher who had participated in the interviews checked each transcription by listening to the tape and at the same time correcting possible transcription errors. In addition, we arranged feedback sessions at PrintCo and PulpCo during which we presented our findings and asked for corrections or further input. No significant new information was discovered during the feedback sessions. At EnergySoftware, the company did not agree to arrange a feedback session.

The purpose of the interviews was to find out which agile practices and how the companies have applied to distributed projects, what have been the benefits and challenges of using these practices in a distributed project, which GSD practices companies have been using in addition to agile practices and what have been the benefits and drawbacks of applying agile practices to a distributed project. An overview of the case projects and the number of persons interviewed is shown in Table 3.

Table 3 Overview of case projects

Case	Type of company & Type of development	Project duration (Scrum usage)	Countries involved (participating personnel)	Teams per site	Interviewees
PrintCo	Service company: Development of a new version of printing service software for internal use in new markets.	2 years (5 months)	Onsite: Finland (7) Offsite: Latvia, two sites (2+1) Germany (1)	People from all sites viewed as one single team	5 : Onsite (4) Offsite (1)
PulpCo	Industrial company: Further development and maintenance of an information management tool for internal use.	3 years (14 months)	Onsite: Finland, main site + one subcontractor consultant working close to onsite (9+1) Offsite: Russia, subcontractor (6)	One team onsite, one team offsite	7: Onsite (4) + subcontractor consultant working close to onsite Offsite (2)
Energy-Software	IT company: Further development and maintenance of a large energy software product that is in use in several companies all over the world.	~10 years (1,5 years)	Onsite: Norway (~20) Offsite: Malaysia (~20)	5-7 teams, often combined across sites, number of teams and persons in each team varies accross iterations	7: Onsite (4) Offsite (1 face-to-face, 2 over Skype)

To analyse the data, each researcher coded the interviews of one case. The grouping and analysis was done by all three researchers using Atlas.ti qualitative data analysis software. In Atlas.ti, we created categories that partly arose from the data and partly were established in the beginning through discussion with all researchers. Then we collected quotations from the data under these groups. Based on these findings the agile and GSD practices, their application, and their benefits and challenges are described in Section 4.

4. Results

In this section we describe the Scrum practices that our case projects used, how these practices were applied to distributed settings, which supporting GSD practices were used, and which benefits and challenges the agile way of development brought according to our interviewees.

4.1 Scrum practices

In this section, we describe how the case projects applied Scrum practices in their distributed projects. Table 4 summarises the collected experiences.

4.1.1 Daily Scrums

The use of daily scrum meetings was clearly the most important Scrum practice used by all case projects. In the meetings, lasting approximately 15 minutes, each team

member answered the three Scrum questions: "What did you do since the last scrum meeting? Do you have any obstacles? What will you do before next meeting?"

In EnergySoftware and in PrintCo the daily scrum meetings were distributed, since their teams were distributed. EnergySoftware used telephone conferencing and web-cameras to arrange the meetings. EnergySoftware had several distributed teams, thus their meetings for different teams were consecutive, and took place in the same meeting room, one after another. In general all daily meetings were distributed, even though some teams were collocated at the offshore site, since the product owner for each team from onsite participated in the daily scrum meeting of his or her team whenever possible. PrintCo had only one team and it was distributed. They used internet relay chat (IRC) for arranging daily scrum meetings. Typically, all team members wrote their answers to the scrum questions prior to the meeting, and the meeting commenced by everybody sending their answers, and reading the others' messages. Subsequently, discussion took place. At the time of the interviews the company had just bought videoconferencing equipment and the team was planning to start using videoconferencing at least once a week for the daily meetings.

PulpCo had one onsite team and one offsite team. Their daily scrum meetings were held separately for each team. The onsite scrum master participated in some daily scrums of the offshore team to check on them. In the beginning, he did it once a week, but later on more seldom. He used Intervoice teleconference to participate in the meetings. He felt that when he joined the scrum meetings, the offsite team reported to him instead of reporting to the other team members, thus he was not sure whether the meetings were always done properly. PulpCo's Finnish one-man subcontractor participated in onsite meetings via teleconference, and once a week in person. He used Office Communicator with integrated Live Meeting. PulpCo was the only project that did not have daily distributed meetings between the two main sites. This might be one of the reasons that the interviewees from this project to complain about not properly knowing the situation on the other site.

In all case projects, discussion took place after the three questions were answered. The meetings always took at most 15 minutes as suggested by Scrum. In PrintCo, the meetings were often shorter, since answers to the questions were written beforehand. In EnergySoftware, the meetings initially lasted only a few minutes, before the teams really learned to discuss and tell their impediments. That was especially difficult for persons coming from an Asian culture. The scrum masters started to encourage everybody to talk and tell more about their tasks and impediments. At the time of the interviews these meetings normally took 15 minutes and were found very useful by all participants.

If problems or need for one-to-one discussion were encountered during daily meetings, teams set up separate meetings after the daily scrums and continued discussions in smaller groups or one-to-one either by teleconference, chat or email. In all the case projects, daily scrum meetings encouraged team members to communicate more also outside the meetings, which was seen as one of the greatest benefits of these meetings. Daily scrum meetings also provided a good way for everybody to get an overview of the project situation. In particular, the interviewees reported that it was easier to monitoring the offshore situation than before. Moreover, problems were identified quickly, since now it was difficult to hide problems over longer time. Even though arranging daily meetings was a heavy process, they were seen as very useful by all the projects. When comparing the projects, that had distributed daily meetings, to the project where they had mainly site-specific daily meetings, a clear difference could be seen. Most of the participants of the distributed meetings mentioned the above benefits: increased transparency to the other site, getting a good overview of what was happening in the project and adding communication across sites. One of our interviewees even stated: *"I think that [daily scrum meetings] was the best thing that happened to these distributed teams."* However, the participants of the site-specific, non-distributed meetings in PulpCo mentioned these things as problems: they did not have enough communication and contacts with the other site, nor did they know enough what was happening at the other site.

4.1.2 Weekly scrum-of-scrums

EnergySoftware was the only project having several distributed scrum teams, thus it was also the only project arranging distributed weekly scrum-of-scrum meetings. These meeting took place once a week. One team member from each team participated in the meeting. The team decided who participates; the participant was not always the same person. In addition to the team representatives, all scrum masters typically also participated in these half-hour long meetings. During the meeting the three scrum questions were answered. Now the questions did not address single persons, but a whole team. Thus, each team representative told what his or her team had been doing since the last meeting, what it was planning to do before the next meeting and what kind of impediments they had. Moreover, they have two additional questions: "Have you put some impediments in the other teams' way?" and "Do you plan to put any impediments in the other teams' way?". The goal of these questions was to ensure successful integration.

Weekly scrum-of-scrums were considered very beneficial, since they distributed information between the teams and revealed possible problems early on. They also opened

discussion channels between the teams and that way encouraged informal communication between the teams.

4.1.3 Sprints

Iterations in Scrum are called sprints. The length of one sprint in Scrum is normally four weeks, but can be also shorter or even longer. Our case projects used both four-week and two-week iterations. In PrintCo the normal iteration length was two weeks, which was regarded as very good in that quite small project. A larger project, EnergySoftware, had synchronized 4-week sprints in the development teams. This meant that all sprints started and ended at the same time. The variation of the end and start dates was at maximum a couple of days. The maintenance team was the only exception to this 4-week sprint cycle, as their sprint cycle was two weeks. The reason was that fixes to customers were released every two weeks. Also in EnergySoftware all the interviewed persons were happy with the current sprint lengths.

In PulpCo the sprint length of the onsite team was four weeks and of the offsite team two weeks. The shorter offsite sprint length made it possible for onsite to monitor the offsite more frequently. The sprints in PulpCo were sometimes lengthened due to the different vacation times of the team members in these two countries. By prolonging the sprints when needed, the teams to could stay synchronized.

Due to short sprints, clear deadlines and goals it was quite clear for all the team members in our case projects what was supposed to be done during the next sprint, as one team member commented: *"Before Scrum I could not really understand when there is a deadline and what should be done until that deadline (...) because there were many different deadlines for customers and development stages (...)"*

Sprints with clear deadlines and goals were seen to increasee the transparency in our case projects. Especially offsites benefited a lot, since often the offsite team members do not have a clear picture of the overall project in a traditional distributed setting. In addition, the frequency of feedback between onsite and offsite was increased. There was no possibility any more to delay the completion of a task because it was "only 95% ready".

In addition to normal sprints, one of our cases, PulpCo, had "release sprints" two to three times a year. The purpose of the release sprints were to complete all items and to move the whole system to system testing. An item was considered ready when it was developed, integration tests were completed, version reports made and user guides updated. The advantage of a release sprint was that onsite managers paid more attention to the product and gave more feedback.

4.1.4 Sprint planning meetings

Before starting each sprint a sprint planning meeting takes place. In PrintCo and EnergySoftware the sprint planning meetings were distributed, since the teams were also distributed. In PulpCo the meetings were separate for both onsite and offsite teams.

In EnergySoftware sprint planning meetings were divided into three phases: distributed meeting, local meeting onsite and local meeting offsite. The distributed meeting was arranged using teleconferencing and Microsoft NetMeeting for application sharing. During the distributed part of the meeting the product owner presented the prioritized items in the backlog, and team asked questions. Because of the time-zone difference this part of the meeting was timeboxed for those three common working hours for both sites. After that the offsite working day ended. The onsite team continued by dividing the backlog items into more detailed tasks, adjusting the estimates made by the product owner and making initial assignments of the tasks to different team members. The offsite continued the work the following morning by discussing and commenting on the draft plan they have received from onsite.

Also PrintCo had distributed sprint planning meetings arranged as teleconference and supported by application sharing. Due to the closer distance between onsite and offsite, a one hour plane trip, especially in the beginning a couple of offsite team members visited onsite during sprint planning, which made these meetings more efficient. Several of our interviewees would have prefered on-site planning to distributed if possible. They commented that in particular effort estimation requires lots of discussion, which is difficult to do efficiently while distributed. However, with a two-week sprint cycle arranging only face-to-face meetings would not have been economically possible.

In PulpCo, onsite and offsite had separate meetings. Only the onsite scrum master participated in the planning meeting of both teams. He was seen by the offshore team as their product owner and a link between the teams. Since PulpCo had the most important roles at onsite, the planning of the offsite sprint started by the onsite scrum master and architect selecting items to the product backlog. The offsite meeting was arranged via teleconference with Live Meeting and desktop sharing. Besides the onsite scrum master, the onsite chief designer and architect participated in these meetings, as needed. During the visits of offsite personnel to onsite, the meetings were arranged face-to-face. During the first part of the meeting the team decided what backlog items would be implemented in the next sprint. Then, the team worked on its own and split the backlog items into 1-2 work day tasks and estimated their effort. The onsite planning meeting had the same agenda. The Finnish subcontractor participated in the onsite meetings face-to-face or by using Office Communicator when the parts he developed were discussed.

Sprint planning meetings were considered very useful, since they provided visibility to the work on both sites and gave team members an opportunity to participate in planning. However, the distributed meetings were quite challenging and tiring, because of the often bad voice quality of teleconference calls. Moreover, it was difficult to always know who was talking when not seeing the persons from the other site. Also, the issues discussed were sometimes difficult to explain when distributed. At the time of the interviews, none of the projects could use videoconferencing and even web cameras were not possible for all the projects because of the narrow bandwidth. All the projects found collocated planning meetings preferable, but they could be arranged quite seldom, even though all the projects had sometimes arranged planning meetings during the visits of offsite personnel to onsite.

4.1.5 Sprint demos

At the end of a sprint, the developed functionality is demonstrated to all interested parties. The meeting is called demo or sprint review meeting. All case projects arranged demos that both onsite and offsite personnel participated in. The demos were normally arranged using teleconference and application sharing. During visits, face-to-face demos were sometimes arranged. In EnergySoftware and PrintCo the demos were team specific, even though other interested parties could participate in addition to the team members. In PulpCo, the offsite and onsite teams had their demos together. That way both sites had the possibility to give and receive more immediate feedback. In PrintCo, the demos were combined with sprint planning meetings. After the demo there was just break before the team continued the meeting by starting to plan the next sprint.

According to our interviewees the demos were beneficial since they increased the visibility of the project in both directions. They also offered one more possibility to monitor the work at offsite. For example, in PulpCo, before starting to use Scrum, the teams were working independently for long periods of time. This usually led to a lot of rework for the offsite team as they often misunderstood the requirements. Short sprints with demos at the end mitigated this problem. The biggest problem with demos was the same as with other distributed meetings: the used technology, teleconferencing and application sharing, did not offer good enough possibilities to communicate efficiently according to our interviewees.

4.1.6 Retrospective meetings

A retrospective meeting normally takes place in the end of a sprint. During that meeting the team discusses three questions: “What has been good during this sprint?”, “What has not been that good?” and “What kind of improvements could we do?”.

In EnergySoftware and PrintCo the retrospective meetings took place directly after the demos as distributed teleconference meetings. In PulpCo neither of the teams had so far regular retrospective meetings. The onsite team had arranged a retrospective meeting once, and that was received very positively. Normally, in that project just the offsite and onsite scrum masters discussed the possible improvements, but rest of the teams did not participate.

4.1.7 Backlogs

Backlogs are lists of items to be developed. In the sprint planning meeting, a product owner with his or her team selects from the product backlog items to be developed during the next sprint to the sprint backlog. Our case projects used different tools to manage their backlogs. EnergySoftware used Jira, with separate backlogs for all the teams. The backlogs were updated by the respective product owners and all the team members had access to their backlog. All the teams were very satisfied with this tool.

PrintCo had their backlog in a Wiki, but they were looking for a better tool, since they felt that the wiki editor was awkward to use and not really designed for this purpose. The good side was that everybody could access the wiki and follow the project progress by looking at the project burndown chat there. However, in this project the responsibilities of the product owner were still quite unclear. Thus, the backlog was sometimes updated by one of the two product owners, sometimes by the scrum master.

PulpCo had their backlog in Team Foundation Server, where it was updated by the onsite product owner, scrum master and chief designer. In practice, everyone could add items into the product backlog, but the offsite team members had never put anything into the backlog yet.

In all our case projects, the backlogs could be accessed by all team members, which was considered beneficial. Only EnergySoftware had a clear process of updating backlogs, the other projects were still practicing the usage of backlogs and having problems both with the tools and responsibilities of different roles.

Table 4. Summary of the distributed Scrum practices, their challenges and benefits

Distributed Scrum practice	How applied to distributed projects?	Challenges	Benefits
Daily scrum	Distributed teams used teleconference and web cameras, or chat.	Cultural differences, e.g. about reporting impediments. It takes time to learn to report information that is useful for others.	Brings transparency to a distributed project. Reveals possible problems early on. Creates contacts and encourages to informal communication especially between the sites. Mentioned as the most useful Scrum practice for distributed projects.
Weekly scrum-of-scrums	Using teleconference and web cameras.	No specific challenges mentioned.	Distributes information between the teams. Reveals possible problems early on. Opens discussion channels and encourages informal communication between the teams.
Sprints	2-4 week long sprints used depending on the project. When several teams, sprints are synchronized.	Different religious or other holidays in different countries cause synchronization challenges.	Provides frequent monitoring opportunities between the sites.
Sprint planning meeting	Distributed meetings can be arranged using teleconferencing and application sharing. The meeting can be divided into several parts: a distributed part and site specific parts. Sometimes collocated during visits.	Time-zone differences cause challenges to arrange, especially longer meetings. Cultural and language differences may cause silence of some participants. Exhausting with long teleconferences. Sound quality not always good enough. Difficult to recognize speakers when not seeing the faces.	Give a possibility for team members from all sites to participate, to ask clarifications, to understand tasks and to commit to common goals. Brings transparency to a distributed project.
Sprint demo	Distributed meetings can be arranged using teleconferencing and application sharing. Can be a combined meeting with sprint planning and retrospective meetings.	The same technical problems as in sprint planning.	Brings transparency to a distributed project. Prevents problems by providing a frequent monitoring opportunity between the sites. Ensure the understanding of the requirements, especially regarding the offsite.
Retrospective meeting	Distributed meetings can be arranged using teleconferencing and application sharing. Can be a combined meeting with sprint planning and demo.	Similar technical problems as in sprint planning.	No specific benefits mentioned.
Backlogs	Access to backlogs by all distributed team members Backlogs in Jira, Wiki or Team Foundation Server.	Updating responsibilities unclear. Wiki editor awkward to use.	All team members can access, pick items and follow the progress.

4.2 Supporting GSD practices

In addition to the above agile practices, our case project used supporting GSD practices as well. These are described next.

4.2.1 Frequent visits

All projects arranged visits between the sites for team members. EnergySoftware and PrintCo did not have a planned schedule for the visit: they were arranged on a need basis. PulpCo, on the other hand, arranged one-week visits of a few offshore team members to onsite regularly every two months. Normally, the offshore scrum master and the chief developer traveled, sometimes accompanied by a developer having some urgent problem. The main reason for these visits was to provide enough information for the offsite until the next visit. If possible, offsite members participated in sprint planning and review meetings during their visits. According to our interviewees, it would have been more productive to have trips every month, but the long travel time (15h by car/direction) made it impractical. Before each visit, an agenda was created and the offshore team collected questions which were more easily discussed face-to-face than by chat or email. During the visit, the offshore team spent most of the time in meetings with onsite domain experts. The Finnish one-man subcontractor visited onsite normally once a week and during the offsite visits, twice a week. Every time the offsite people came for a visit, leisure activities were arranged, such as sauna or dinner. These gave the members of both teams the possibility to get to know each other on a personal level.

Both in EnergySoftware and PrintCo, several visits took place in the beginning, when the projects had just started to apply Scrum, and a bit more seldom later on. At PrintCo, a couple of persons from offsite visited onsite for a couple of days, during which a lot of meetings and informal discussions took place.

At EnergySoftware during the first half a year after starting to apply Scrum, the project had several persons from onsite working full-time at offsite. Subsequently, the team members, especially from offsite, have traveled frequently between the sites on a need basis. The visits nor-

mally lasted between two and four weeks, which made it possible for a team to really work together. Especially during critical phases, it was considered important to collocate the team, e.g. for the last iteration before a release or for the first iteration, when most of the planning takes place. Besides these visits, EnergySoftware had started to arrange annual social gatherings for the whole project for a long weekend. During these gatherings the project members heard and discussed future actions, had team building exercises and social events. Everybody we interviewed saw these as successful events and managers regarded them as investments for the future. At the time of the interviews, approximately half of the project team members had visited the other site and basically all had met face-to-face in the annual gatherings.

All projects arranged frequent visits, which provided good opportunities for getting to know persons from the other site, discuss difficult issues, and get a better picture of the project. Face-to-face meetings also increased trust between team members and encouraged them to continue communication after the visits. It seemed to be important to arrange visits not only in the beginning of the project, but quite frequently during the project, as well. The visits normally lasted from a few days at PrintCo, to a few weeks at EnergySoftware. Thus, the visits were not just short trips to meetings, but instead longer stays during which distributed team members could really work together. Of course, the projects tried to schedule the trips so that the visitors could participate face-to-face in the most important regular meetings, such as sprint planning meetings.

In all the projects it was mainly offsite personnel that travelled. The reason for this was that onsite personnel was mainly experts that did not have time to travel. Offsite personnel, on the other hand, were mainly developers that found it extremely useful to meet the onsite experts face-to-face and ask questions and discuss difficult issues. Some of our interviewees, especially from offsite, hoped that also onsite personnel would travel more often to provide for many more offsite persons an opportunity to meet them and to share the sometimes quite heavy and tiring traveling duties between onsite and offsite. Even though our case projects arranged quite a lot trips, none of the interviewees mentioned any problems of arranging trips due to cost reasons or limitations to travel because of costs. Instead, it seemed that all the projects found the current model of frequent visits as very useful and even more visits were hoped for.

4.2.2 Multiple communication modes

Besides face-to-face visits, onsite and offsite members communicated a lot using different kind of electronic media: email, phone calls, chat, application sharing and teleconferencing. Also tools, like Jira, Wiki and Team Foundation Server, were accessible by all team members regardless of the location. It seemed to be important that there were many different possibilities to communicate between the sites. Tool choice seemed to depend both on the purpose of the communication, e.g. chat was used to ask short questions or for checking whether the other party was available to receive a phone call, and preferences of a user. Some preferred synchronous voice communication, while others with limited language skills preferred written communication. Thus, it seemed to be important to offer several different communication tools for distributed teams. Video-conferencing was not used at the time of the interviews in any of our case projects. EnergySoftware used web cameras, but would have preferred videoconferencing, if just the bandwidth between the sites would have allowed it. PrintCo had just ordered new videoconferencing equipment and was planning to use it instead of teleconferencing. The challenges faced with current communication tools were mainly related to the bad voice quality and difficulties to understand and explain difficult issues when not being able to communicate face-to-face.

4.3 Challenges faced

All our case projects faced challenges when starting to use Scrum in a distributed project. For all the projects Scrum was a totally new method, thus training was needed. Most interviewees thought that they did not receive enough Scrum training in the beginning. In one project, offsite did not even get any training; they were just sent some material about Scrum, which was really not enough according to our interviewees.

Scrum expects open communication, which was difficult in the beginning, especially by offshore team members who were not used to communicate openly. For example, in EnergySoftware, daily scrum meetings were very short in the beginning, only a few minutes, and especially for Asian people it seemed to be difficult to report impediments. However, the situation improved a lot in all the projects later on, when participants learned to follow the Scrum practices and also saw the benefits of frequent and open communication.

Even though it seems to be a good idea to have separate scrum teams at onsite and offsite, the two projects that had distributed teams, PrintCo and EnergySoftware, had better transparency and fewer communication problems than PulpCo, the only project having separate offsite and onsite teams. In PulpCo, especially offsite hoped for more frequent communication between the sites.

4.4 Positive experiences

Even though all case projects faced challenges when taking Scrum into use, the overall experience was very positive. Based on our interviews, all projects were very satis-

fied with the decision to start to use this agile method and were planning to continue its usage also in the future. The agile practices were considered very suitable for distributed projects, especially because they fostered frequent and open communication, provided good visibility to the project and improved trust between distributed developers.

According to our interviewees, Scrum practices provide a frequent and structured way of communication for a distributed project. Daily scrum meetings with the three questions was one of the most liked practices. In a daily scrum meeting you could get quick answers and clarifications, monitor what was happening at the other site, and get access to the right persons, as one of our interviewees said: *"Everyone is there [in daily scrum meeting], domain experts are supposed to be there, all members are there and you can actually talk and get what you want."*

Since Scrum emphasizes fast problem solving, it encouraged distributed team members to talk about issues more openly than before starting to use Scrum. That was considered especially important when collaborating with cultures where problems are not used to be discussed openly.

The frequent and structured communication forced by Scrum also encouraged distributed team members to informal one-to-one communication between sites, e.g. after daily scrum meetings. The regular communication was seen to lower the threshold to talk and phone to foreign colleagues.

Scrum practices with short iterations and daily meetings provide frequent check-points for monitoring the distant sites. The practices ensure that, e.g. misunderstood requirements are revealed early on or that no-one can work for a long time alone with a problem.

5. Discussion and conclusions

In this paper we presented a multiple-case study on the application of Scrum practices to three globally distributed projects. We also discussed the challenges and benefits related to the application of these practices and Scrum in general to distributed projects. The contribution of this paper was twofold. First, we believe that companies planning to take Scrum into use in their distributed projects will find the collected experiences useful. Second, little empirical literature on the usage of agile practices, especially Scrum practices, in distributed projects exist. Therefore, our multiple-case study provides an addition to the scientific knowledge.

5.1 Lessons learned

Based upon the case studies, we extract the following lessons learned:

- Do not distribute your project if you do not need to! A distributed project brings additional challenges, more effort and can take longer than a collocated project.
- Proper Scrum training is needed in the beginning if the team has not used Scrum before. Besides classroom trainings, for example, a visiting engineer or an outside expert could help the team to apply the Scrum practices during the first iteration. Just sending material about Scrum is not enough!
- Scrum is based on frequent and open communication, which is also its main benefit when applied to a distributed project according to our results. Remember to encourage and ensure open communication, otherwise the benefits of Scrum cannot be realized.
- Frequent visits of development personnel from collaborating sites are needed especially in the beginning of a project, and in critical project phases, such as testing, or planning of new functionality. Visits should preferably be at least a few weeks long, so that persons get to know each other well and really learn to work together. Plan visits and a visiting schedule properly. Do not save in travelling budget.
- Arrange an access to multiple communication tools – videoconferencing for regular meetings; and internet telephones, desktop sharing and chat for unofficial meetings. Tools that are easy to access and use lower the threshold to communicate outside official meetings.

5.2 Limitations

As a multiple case study, the generalizability of the results is limited.

Though we interviewed several people in different roles from each project, additional interviews might have provided additional insight. Due to resource limitations and confidentiality issues, we were not able to triangulate our findings by, e.g. document analysis or observation. Furthermore, data was validated only in two cases using feedback sessions, since the third company declined to arrange for such a possibility.

5.3 Future research

In the future, we hope to augment this study by additional case projects on distributed agile development. We also hope to be able to collect and analyze communication data from several sources to get a more complete understanding of communication in distributed agile development.

Acknowledgements

The authors gratefully acknowledge the financial support of the Academy of Finland (Grant No. SA 107636, 2005-2007) and the Finnish Funding Agency for Technology and Innovation (MaPIT project). We also thank our case

companies and all the interviewees for sharing their experiences.

References

[1] Agerfalk, P. and Fitzgerald, B. Introduction. *Communications of ACM*, vol. 49, no. 10, pp. 26-34, 2006.

[2] Beck, K. Extreme Programming Explained: Embrace Change. Addison Wesley, New York, 2000.

[3] Berczuk, S., Back to basics: The role of agile principles in success with an distributed scrum team. *Proceedings of AGILE*, 2007, pp. 382-388.

[4] Braithwaite, K. and Joyce, T. XP expanded: Distributed extreme programming. *Proceedings of 6th International Conference on Extreme Programming and Agile Processes in Software Engineering*, XP 2005. Springer, 2005, pp. 180-188

[5] Carmel, E. and Agarwal, R. (2001). Tactical approaches for alleviating distance in global software development. IEEE Software, March/April, p. 22-29.

[6] Cockburn, A and Highsmith, J. Agile software development: The people factor. *Computer,* vol. 34, no. 11, pp. 131-133, 2001.

[7] Damian, D. Global software development: growing opportunities, ongoing challenges. Software Process: Improvement and Practice, vol. 8, no. 4, pp. 179-182 2003.

[8] Danait, A. Agile offshore techniques—a case study. Proceedings in Agile Conference, 24-29 July 2005, pp. 214-217.

[9] Ebert, C. and De Neve, P. Surviving Global Software Development. *IEEE Software*, (March/April) 2001, 62-69.

[10] Farmer, M. (2004) DecisionSpace Infrastructure: Agile Development in a Large, Distributed Team. *Proceedings of the Agile Development Conference* 2004.

[11] Fowler, M. Using an agile software process with offshore development. 2006. http://martinfowler.com/artcles/agileOffshore.html. Referenced: 19.12.2007.

[12] Herbsleb, J.and Grinter, R. Architectures, coordination, and distance: Conway's law and beyond.*IEEE Software*, 16, 5, (Sept.-Oct. 1999), 63 –70.

[13] Hogan, B. Lessons learned from an extremely distributed project. *Proceeding of the Agile Conference*, 23-28 July 2006.

[14] Holmstrom, H., Conchùir, E.Ò, Agerfalk, P.J., Fitzgerald, B. (2006). Global Software Development Challenges: A Case Study on Temporal, Geographical and Socio-Cultural Distance. *ICGSE-06. International Conference on Global Software Engineering*, (pp. 3-11). Florianopolis.

[15] Jain, N. Offshore agile maintenance. Proceedings of Agile Conference, 2006

[16] Jensen, B. and Zilmer, A. Cross-continent development using Scrum and XP. *Proceedings of XP*. Springer Berlin, 2003, pp. 146-153.

[17] Krishna, S., Sahay, S., and Walsham, G. Managing cross-cultural issues in global software outsourcing. *Communications of the ACM*, Vol. 47, No. 4,pp. 62-66,2004.

[18] Kircher, M., Jain, P., Corsaro, A. and Levine, D. Distributed Extreme Programming. Proceedings of the International Conference on eXtreme Programmingand Flexible Processes in Software Engineering, Sardinia, Italy, May 20 - 23,2001.

[19] Kussmaul, C., Jack, R., and Sponsler, B. Outsourcing and offshoring with agility: A case study. p. 147-154, 2004. http://www.springerlink.com/content/nr9xt91c6jxuf751

[20] Layman, L., Williams, L., Damian, D. and Bures, H. Essential communication practices for extreme programming in a global software development team. *Information and Software Technology,* vol. 48, no. 9, pp. 781-794, 2006.

[21] MacGregor, E., Hsieh Y. and Kruchten, P, "The impact of intercultural factors on global software development", Canadian Conference on Electrical and Computer Engineering, 2005, pp. 920-926, 2005.

[22] Mockus, A. and Herbsleb, J. Challenges of Global Software Development. *Proceedings of the Seventh International Software Metrics Symposium*, (METRICS 2001, IEEE), 182-184.

[23] Ngo-The, A., Hoang, K., Nguyen, T., and Mai, N. Extreme programming in distributed software development: A case study. *Proceedings of International Workshop on Distributed Software Development*, August 2005.

[24] Nisar, M. and Hameed, T. Agile Methods Handling Offshore Software Development Issues. *Proceedings of INMIC 2004, 8th International Multitopic Conference,* Dec.2004. 417-422.

[25] Paasivaara, M. and Lassenius, C. Synchronizing Global Software Development Using Incremental Development and Frequent Deliveries. *Proceedings of the ICSE Internaltional Workshop on Global Software Development*, Edinburgh, May 2004.

[26] Paasivaara, M. and Lassenius, C. Could global software development benefit from agile methods?" Proceedings of the International Conference on Global Software Engineering, ICGSE '06., Oct. 2006, pp. 109-113.

[27] Patton, M. Q. *Qualitative Research and Evaluation Methods*. Newbury Park, CA: Sage Publications. 1990.

[28] Poole, C. J. Distributed product development using extreme programming. pp. 60-67, 2004. (Available: http://www.springerlink.com/content/kkfhajn8a4jbclau)

[29] Ramesh, B. Lan-Cao, Mohan, K. and Peng-Xu, Can distributed software development be agile? *Communications of the ACM*, vol. 49(10), pp. 41-46, 2006.

[30] Robson, C. Real World Research. A Resource for Social Scientists and Practitioner-Researchers. Blackwell. 1997.

[31] Schwaber, K. and Beedle, M. Agile Software Development with Scrum. NewYork, United States: Prentice Hall, 2001.

[32] Simons, M. Internationally Agile. *InformIT*, March 15th, 2002.

[33] Sutherland, J., Viktorov, A., Blount, J. and Puntikov, N. Distributed scrum: Agile project management with outsourced development teams. Proceedings of 40th Annual Hawaii International Conference on System Sciences, HICSS 2007, pp. 274a-274a.

[34] Smits, H. and Pshigoda, G. Implementing scrum in a distributed software development organization. Proceedings of AGILE 2007, pp. 371-375.

[35] D. Turk, R. France, and B. Rumpe, "Assumptions Underlying Agile Software-Development Processes", *Journal of Database Management*, vol. 16, no. 4, pp. 62–87, 2005.

[36] Yap, M. Follow the sun: distributed extreme programming development. Proceedings of Agile Conference, 24th-29th July 2005, pp. 218-224.

[37] Yin, R.K. *Case Study Research, Designs and Methods*. Thousand Oaks, California: Sage Publications, 1994.